Available online at www.sciencedirect.com
ScienceDirect
Comput. Methods Appl. Mech. Engrg. 326 (2017) 402–421
Computer methods in applied mechanics and engineering
www.elsevier.com/locate/cma

A near-stationary subspace for ridge approximation

Paul G. Constantine[a,*], Armin Eftekhari[b], Jeffrey Hokanson[a], Rachel A. Ward[c]

[a] *University of Colorado at Boulder, Department of Computer Science, 80309 Boulder, CO, United States*
[b] *Alan Turing Institute, British Library, 96 Euston Road, London, NW1 2DB, United Kingdom*
[c] *Department of Mathematics and Institute for Computational Engineering and Sciences, University of Texas at Austin, Austin, TX 78712, United States*

Received 6 June 2016; received in revised form 7 June 2017; accepted 26 July 2017
Available online 24 August 2017

Highlights

- We present ridge approximation as a method for constructing response surfaces of expensive computer simulations.
- We show that the gradient-based active subspace is near-stationary for optimal ridge approximation.
- We offer computational heuristics for fitting ridge approximations, and we demonstrate them on an airfoil design problem.

Abstract

Response surfaces are common surrogates for expensive computer simulations in engineering analysis. However, the cost of fitting an accurate response surface increases exponentially as the number of model inputs increases, which leaves response surface construction intractable for high-dimensional, nonlinear models. We describe *ridge approximation* for fitting response surfaces in several variables. A ridge function is constant along several directions in its domain, so fitting occurs on the coordinates of a low-dimensional subspace of the input space. We review essential theory for ridge approximation – e.g., the best mean-squared approximation and an optimal low-dimensional subspace – and we prove that the gradient-based *active subspace* is near-stationary for the least-squares problem that defines an optimal subspace. Motivated by the theory, we propose a computational heuristic that uses an estimated active subspace as an initial guess for a ridge approximation fitting problem. We show a simple example where the heuristic fails, which reveals a type of function for which the proposed approach is inappropriate. We then propose a simple alternating heuristic for fitting a ridge function, and we demonstrate the effectiveness of the active subspace initial guess applied to an airfoil model of drag as a function of its 18 shape parameters.

Keywords: Active subspaces; Ridge functions; Projection pursuit

1. Introduction

Engineering computations often employ cheap response surfaces that mimic the input/output relationship between an expensive computer model's parameters and its predictions. The essential idea is to use a few expensive model runs

* Corresponding author.
E-mail address: paul.constantine@colorado.edu (P.G. Constantine).

http://dx.doi.org/10.1016/j.cma.2017.07.038
0045-7825/

at particular parameter values (i.e., a *design of experiments* [1]) to fit or train a response surface, where the surface may be a polynomial, spline, or radial basis approximation [2,3]. The same scenario motivates statistical tools for design and analysis of computer experiments [1,4,5], which use Gaussian processes to model uncertainty in the surrogate's predictions.

The cost of constructing an accurate response surface increases exponentially as the dimension of the input space increases; in approximation theory, this is the tractability problem [6,7], though it is colloquially referred to as the *curse of dimensionality* [8, Section 5.16]. Several techniques attempt to alleviate this curse—each with advantages and drawbacks for certain classes of problems; see [9] for an extensive survey. One idea is to identify unimportant input variables with global sensitivity metrics [10] and fix them at nominal values, which effectively reduces the dimension for response surface construction; Sobol' et al. studied the effects of such coordinate-based dimension reduction on the approximation [11]. A generalization of coordinate-based dimension reduction is to identify unimportant directions—not necessarily coordinate aligned. If the scientist can identify a few important linear combinations of inputs, then she may fit a response surface of only those linear combinations, which allows a higher degree of accuracy along important directions in the input space.

A *ridge function* [12] is a function of a few linear combinations of inputs that takes the form $g(\boldsymbol{U}^T\mathbf{x})$, where $\mathbf{x} \in \mathbb{R}^m$, $\boldsymbol{U} \in \mathbb{R}^{m\times n}$ with $n < m$, and $g : \mathbb{R}^n \to \mathbb{R}$. The term *ridge function* is more commonly used when $\boldsymbol{U}$ is a single vector ($n = 1$). Pinkus calls our definition a *generalized ridge function* [12, Chapter 1], though Keiper uses the qualifier *generalized* for a model where $\boldsymbol{U}$ depends on $\mathbf{x}$ [13]. A ridge function is constant along directions in its domain that are orthogonal to $\boldsymbol{U}$'s columns. To see this, let $\mathbf{v} \in \mathbb{R}^m$ be orthogonal to $\boldsymbol{U}$'s columns; then

$$g(\boldsymbol{U}^T(\mathbf{x}+\mathbf{v})) = g(\boldsymbol{U}^T\mathbf{x} + \underbrace{\boldsymbol{U}^T\mathbf{v}}_{=0}) = g(\boldsymbol{U}^T\mathbf{x}). \tag{1}$$

If $\boldsymbol{U}$ is known, then one need only construct g, which is a function of $n < m$ variables. Thus, constructing g may require exponentially fewer model evaluations than constructing a comparably accurate response surface on all m variables.

Let $f : \mathbb{R}^m \to \mathbb{R}$ represent the simulation model's input/output map to approximate, and let its domain be equipped with a probability function $\rho : \mathbb{R}^m \to \mathbb{R}_+$. The function ρ may model uncertainty in the simulation's input parameters, which is a common modeling choice in *uncertainty quantification* [14,15]. The ridge approximation problem may be stated as: given f and ρ, find g and $\boldsymbol{U}$ that minimize the approximation error. After a brief survey of related concepts, we define a specific ridge approximation problem in Section 2. We then study a particular $\boldsymbol{U}$ derived from f's gradient known as the *active subspace* [16]. We show that, under certain conditions, the active subspace is nearly stationary—i.e., that the gradient of the objective function defining the approximation problem is bounded; see Section 3. This result motivates a heuristic for the initial subspace when fitting a ridge approximation given pairs $\{(\mathbf{x}_i, f(\mathbf{x}_i))\}$. In Section 4, we show a simple bivariate example that exposes the limitations of the heuristic. We then study an 18-dimensional example from an airfoil shape optimization problem where the heuristic succeeds; in particular, we demonstrate a numerical procedure for estimating the active subspace using samples of the gradient, and we show how the estimated active subspace is a superior starting point for a numerical optimization heuristic for fitting the ridge approximation.

1.1. Related concepts

There are many concepts across subfields that relate to ridge approximation. In what follows, we briefly review three of these subfields with citations that point interested readers to representative works.

1.1.1. Projection pursuit regression

In the context of statistical regression, Friedman and Stuetzle [17] proposed *projection pursuit regression* with a ridge function model:

$$y_i = \sum_{k=1}^{r} g_k(\mathbf{u}_k^T\mathbf{x}_i) + \varepsilon_i, \tag{2}$$

where $\mathbf{x}_i$'s are samples of the predictors, y_i's are the associated responses, and ε_i's model random noise—all standard elements of statistical regression [18]. The g_k's are smooth univariate functions (e.g., splines), and the $\mathbf{u}_k$'s are

the directions of the ridge approximation. To fit the projection pursuit regression model, one minimizes the mean-squared error over the directions $\{\mathbf{u}_k\}$ and the parameters of $\{g_k\}$. Motivated by the projection pursuit regression model, Diaconis and Shahshahani [19] studied the approximation properties of nonlinear functions (g_k in (2)) of linear combinations of the variables ($\mathbf{u}_k^T\mathbf{x}$ in (2)). Huber [20] surveyed a wide class of projection pursuit approaches across an array of multivariate problems; by his terminology, *ridge approximation* could be called *projection pursuit approximation*. Chapter 11 of Hastie, Tibshirani, and Friedman [21] links projection pursuit regression to neural networks, which uses ridge functions with particular choices for the g_k's (e.g., the sigmoid function). Although algorithm implementations may be similar, the statistical regression context is different from the approximation context, since there is no inherent randomness in the approximation problem.

1.1.2. Gaussian processes with low-rank correlation models

In Gaussian process regression [22], the conditional mean of the Gaussian process model given data (e.g., $\{y_i\}$ as in (2)) is the model's prediction. This conditional mean is a linear combination of radial basis functions with centers at a set of points $\{\mathbf{x}_i\}$, where the form of the basis function is related to the Gaussian process' assumed correlation. Vivarelli and Williams [23] proposed a correlation model of the form

$$C(\mathbf{x},\mathbf{x}') \propto \exp\left[-\frac{1}{2}(\mathbf{x}-\mathbf{x}')^T\boldsymbol{U}\boldsymbol{U}^T(\mathbf{x}-\mathbf{x}')\right], \tag{3}$$

where $\boldsymbol{U}$ is a tall matrix. In effect, the resulting conditional mean is a function of linear combinations of the predictors, $\boldsymbol{U}^T\mathbf{x}$—i.e., a ridge function. A maximum likelihood estimate of $\boldsymbol{U}$ is the minimizer of an optimization similar to the one we define for ridge approximation; see Section 2. Bilionis et al. [24], use a similar approach from a Bayesian perspective in the context of uncertainty quantification, where the subspace defined by $\boldsymbol{U}$ enables powerful dimension reduction.

1.1.3. Ridge function recovery

Recent work in constructive approximation seeks to recover the parameters of a ridge function from point queries [25–27]. In other words, assume $f(\mathbf{x}) = g(\boldsymbol{U}^T\mathbf{x})$ is a ridge function; using pairs $\{\mathbf{x}_i, f(\mathbf{x}_i)\}$, one wishes to recover the components of $\boldsymbol{U}$. Algorithms for determining $\boldsymbol{U}$ (e.g., Algorithm 2 in [25]) are quite different than optimizing a ridge approximation over $\boldsymbol{U}$. However, the recovery problem is similar in spirit to the ridge approximation problem.

2. Optimal ridge approximation

Consider a function $f : \mathbb{R}^m \to \mathbb{R}$ that is square-integrable with respect to a given probability density function $\rho : \mathbb{R}^m \to \mathbb{R}_+$,

$$\int f(\mathbf{x})^2\,\rho(\mathbf{x})\,d\mathbf{x} < \infty, \tag{4}$$

where we assume the domain of f is the support of ρ. Given $\boldsymbol{U} \in \mathbb{R}^{m\times n}$ with $n < m$ and $g : \mathbb{R}^n \to \mathbb{R}$, we measure the error in the ridge approximation with the $L^2(\rho)$ norm,

$$\left\| f(\mathbf{x}) - g(\boldsymbol{U}^T\mathbf{x})\right\|_{L^2(\rho)} = \left(\int (f(\mathbf{x}) - g(\boldsymbol{U}^T\mathbf{x}))^2\,\rho(\mathbf{x})\,d\mathbf{x}\right)^{\frac{1}{2}}. \tag{5}$$

We restrict attention to matrices $\boldsymbol{U}$ with orthonormal columns, $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}$, where $\boldsymbol{I}$ is the $n \times n$ identity matrix. For a more general matrix with full column rank, we can transform to the orthonormal column case with a thin QR factorization, where the $\boldsymbol{R}$ factor represents an invertible change of variables in g's domain.

Given $\boldsymbol{U}$ with orthonormal columns, let $\boldsymbol{V}$ be an orthogonal basis for the complement of span($\boldsymbol{U}$) in $\mathbb{R}^m$, where span($\boldsymbol{U}$) denotes the span of $\boldsymbol{U}$'s columns. The density function $\rho(\mathbf{x})$ induces joint, marginal, and conditional densities on the subspace coordinates of span($\boldsymbol{U}$) and span($\boldsymbol{V}$) as follows. For $\mathbf{y} \in \mathbb{R}^n$ and $\mathbf{z} \in \mathbb{R}^{m-n}$, define the following:

$$\begin{aligned}
\pi(\mathbf{y},\mathbf{z}) &= \rho(\boldsymbol{U}\mathbf{y} + \boldsymbol{V}\mathbf{z}) \quad \text{(joint density)}\\
\pi(\mathbf{y}) &= \int \pi(\mathbf{y},\mathbf{z})\,d\mathbf{z} \quad \text{(marginal density)}\\
\pi(\mathbf{z}|\mathbf{y}) &= \pi(\mathbf{y},\mathbf{z})/\pi(\mathbf{y}) \quad \text{(conditional density)}.
\end{aligned} \tag{6}$$

The conditional density enables construction of a particularly useful ridge approximation. Define the conditional average of f given subspace coordinates $\mathbf{y}$, denoted μ, as

$$\mu = \mu(\mathbf{y}, \boldsymbol{U}) = \int f(\boldsymbol{U}\mathbf{y} + \boldsymbol{V}\mathbf{z})\, \pi(\mathbf{z}|\mathbf{y})\, d\mathbf{z}. \tag{7}$$

Consider the ridge function $\mu(\boldsymbol{U}^T\mathbf{x}, \boldsymbol{U})$. By construction,

$$\int \left(\mu(\mathbf{y}) - f(\boldsymbol{U}\mathbf{y} + \boldsymbol{V}\,\mathbf{z})\right)\, \pi(\mathbf{z}|\mathbf{y})\, d\mathbf{z} = 0, \tag{8}$$

for all $\mathbf{y}$ such that $\pi(\mathbf{y}) > 0$. As a consequence of Pinkus' Theorem 8.3 [12], for fixed $\boldsymbol{U}$, (8) implies that $\mu(\boldsymbol{U}^T\mathbf{x}, \boldsymbol{U})$ is the unique best ridge approximation in the $L^2(\rho)$ norm; see the discussion immediately following the theorem's statement.

The particular choice of basis $\boldsymbol{U}$ does not affect the ridge approximation μ. In other words, we can replace $\boldsymbol{U}$ by $\boldsymbol{UQ}$, where $\boldsymbol{Q}$ is an $n \times n$ orthogonal rotation matrix, and μ does not change. To see this, first examine the conditional density,

$$\begin{aligned}
\pi(\mathbf{z}|\mathbf{y} = \boldsymbol{Q}^T\boldsymbol{U}^T\mathbf{x}) &= \frac{\rho(\boldsymbol{UQQ}^T\boldsymbol{U}^T\mathbf{x} + \boldsymbol{V}\,\mathbf{z})}{\int \rho(\boldsymbol{UQQ}^T\boldsymbol{U}^T\mathbf{x} + \boldsymbol{V}\,\mathbf{z})\, d\mathbf{z}} \\
&= \frac{\rho(\boldsymbol{UU}^T\mathbf{x} + \boldsymbol{V}\,\mathbf{z})}{\int \rho(\boldsymbol{UU}^T\mathbf{x} + \boldsymbol{V}\,\mathbf{z})\, d\mathbf{z}} \\
&= \pi(\mathbf{z}|\mathbf{y} = \boldsymbol{U}^T\mathbf{x}).
\end{aligned} \tag{9}$$

Next examine the definition of μ,

$$\begin{aligned}
\mu(\boldsymbol{Q}^T\boldsymbol{U}^T\mathbf{x};\ \boldsymbol{UQ}) &= \int f(\boldsymbol{UQQ}^T\boldsymbol{U}^T\mathbf{x} + \boldsymbol{V}\,\mathbf{z})\, \pi(\mathbf{z}|\mathbf{y} = \boldsymbol{Q}^T\boldsymbol{U}^T\mathbf{x})\, d\mathbf{z} \\
&= \int f(\boldsymbol{UU}^T\mathbf{x} + \boldsymbol{V}\,\mathbf{z})\, \pi(\mathbf{z}|\mathbf{y} = \boldsymbol{U}^T\mathbf{x})\, d\mathbf{z} \\
&= \mu(\boldsymbol{U}^T\mathbf{x};\ \boldsymbol{U}).
\end{aligned} \tag{10}$$

This implies that μ only depends on the subspace span($\boldsymbol{U}$) as opposed to the particular basis. For the rest of this section, the notation $\boldsymbol{U}$ denotes an equivalence class of matrices whose columns span the same subspace. Similarly, we use $\boldsymbol{V}$ to represent an equivalence class of matrices whose columns span the orthogonal complement of span($\boldsymbol{U}$) in $\mathbb{R}^m$.

To characterize the optimal $\boldsymbol{U}$, we derive a differentiable cost function from (5). Define $R = R(\boldsymbol{U})$ as

$$R(\boldsymbol{U}) = \frac{1}{2}\, \left\| f(\mathbf{x}) - \mu(\boldsymbol{U}^T\mathbf{x}, \boldsymbol{U}) \right\|^2_{L^2(\rho)}. \tag{11}$$

Note that, similar to μ, R only depends on span($\boldsymbol{U}$) as opposed to the choice of basis. Therefore, the appropriate manifold for optimization is the Grassmann manifold—i.e., the space of n-dimensional subspaces of $\mathbb{R}^m$, denoted $\mathbb{G}(n, m)$. Let $\boldsymbol{U}_*$ be a solution to the following optimization problem:

$$\begin{aligned}
&\underset{\boldsymbol{U}}{\text{minimize}}\ \ R(\boldsymbol{U}), \\
&\text{subject to}\ \ \boldsymbol{U} \in \mathbb{G}(n, m).
\end{aligned} \tag{12}$$

We call $\boldsymbol{U}_*$ an *optimal subspace*. In general, the objective function is not a convex function of $\boldsymbol{U}$, so its minimizer may not be unique. In practice, we use numerical methods to estimate $\boldsymbol{U}_*$.

It is convenient to reformulate the optimization (12) in terms of the complement subspace span($\boldsymbol{V}$). The conditional average μ in (7) is the average of $f(\mathbf{x})$ over the affine subspace $S(\mathbf{x})$ defined as

$$S(\mathbf{x}) = \{\, \mathbf{x}' \in \mathbb{R}^m \ \mid\ \mathbf{x}' = \boldsymbol{UU}^T\mathbf{x} + \boldsymbol{V}\,\mathbf{z},\ \mathbf{z} \in \mathbb{R}^{m-n} \,\}. \tag{13}$$

This space depends only on the shift $\boldsymbol{UU}^T\mathbf{x}$ and span($\boldsymbol{V}$)—not the choice of basis for span($\boldsymbol{V}$). We can write the shift as

$$\boldsymbol{UU}^T\mathbf{x} = (\boldsymbol{I} - \boldsymbol{VV}^T)\,\mathbf{x}, \tag{14}$$

where $\boldsymbol{I}$ is the $m \times m$ identity matrix. Again, this shift does not depend on the choice of basis—only the subspace span($\boldsymbol{V}$). Therefore, we can write μ from (7) as

$$\mu = \mu(\mathbf{x}, \boldsymbol{V}) = \int f\left((\boldsymbol{I} - \boldsymbol{V}\boldsymbol{V}^T)\mathbf{x} + \boldsymbol{V}\,\mathbf{z}\right)\,\pi(\mathbf{z}|\mathbf{y})\,d\mathbf{z}. \tag{15}$$

Similarly, we rewrite R from (11) as

$$R(\boldsymbol{V}) = \frac{1}{2}\,\|f(\mathbf{x}) - \mu(\mathbf{x}, \boldsymbol{V})\|^2_{L^2(\rho)}. \tag{16}$$

Let $\boldsymbol{V}_*$ be an $(m-n)$-dimensional subspace that satisfies

$$\begin{aligned} &\underset{\boldsymbol{V}}{\text{minimize}} \;\; R(\boldsymbol{V}), \\ &\text{subject to} \;\; \boldsymbol{V} \in \mathbb{G}(m-n, m). \end{aligned} \tag{17}$$

An optimal $\boldsymbol{U}_*$ that solves (12) is the orthogonal complement of a particular $\boldsymbol{V}_*$.

Reformulating R as a function of $\boldsymbol{V}$ is convenient for studying its gradient. Edelman et al. [28, Section 2.5.3] derive a formula for the gradient of R on the Grassmann manifold in terms of the partial derivatives on the ambient Euclidean space $\mathbb{R}^{m\times(m-n)}$. Denote the gradient on the Grassmann by $\bar{\nabla}$. Then

$$\bar{\nabla} R(\boldsymbol{V}) = \frac{\partial}{\partial \boldsymbol{V}} R(\boldsymbol{V}) - \boldsymbol{V}\boldsymbol{V}^T \frac{\partial}{\partial \boldsymbol{V}} R(\boldsymbol{V}) = \boldsymbol{U}\boldsymbol{U}^T \frac{\partial}{\partial \boldsymbol{V}} R(\boldsymbol{V}), \tag{18}$$

where $\frac{\partial}{\partial \boldsymbol{V}} R$ is the $m \times (m-n)$ matrix of partial derivatives

$$\left(\frac{\partial}{\partial \boldsymbol{V}} R\right)_{ij} = \frac{\partial R}{\partial v_{ij}}, \qquad i = 1, \ldots, m, \quad j = 1, \ldots, m-n, \tag{19}$$

where v_{ij} is the (i, j) element of $\boldsymbol{V}$. This formula can be implemented and passed to a gradient-based nonlinear optimization routine, e.g., steepest descent or a quasi-Newton method.

3. A near-stationary subspace

The objective function $R(\boldsymbol{V})$ in (17) is, in general, not a convex function of $\boldsymbol{V}$, so a gradient-based optimization algorithm is only guaranteed to converge to a stationary point.[1] Additionally, the cost of reaching a stationary point may depend heavily on the initial guess. We call a subspace *near-stationary* if we can bound the norm of the objective's gradient at that subspace.

Definition 1. A subspace $\boldsymbol{V}_* \in \mathbb{G}(m-n, m)$ is near-stationary if there is a constant $A = A(f, \rho)$ such that

$$\left\|\bar{\nabla} R(\boldsymbol{V}_*)\right\|_F \;\leq\; A, \tag{20}$$

where $\|\cdot\|_F$ is the Frobenius norm, and $\bar{\nabla} R$ is the gradient on the Grassmann manifold of R from (17).

In what follows, we show that the active subspace derived from f's derivatives is near-stationary. The active subspace is the eigenspace of a particular symmetric, positive semidefinite matrix, and the bound A from Definition 1 is related to the matrix's eigenvalues. In statistical regression, Samarov [30] studied related matrices built from derivatives of the regression's link function, which he termed *average derivative functionals*; Samarov's $\boldsymbol{T}_1$ is similar to the matrix we study. The regression case contrasts ours since we assume f and its derivatives are given, whereas the link function in regression depends on parameters to be estimated from data.

To ensure that all necessary quantities exist, we make the following assumption on $f(\mathbf{x})$.

Assumption 1. Given the probability density $\rho : \mathbb{R}^m \to \mathbb{R}_+$, assume (i) $f \in L^2(\rho)$ is continuous and differentiable for all $\mathbf{x}$ in the support of ρ and (ii) the partial derivatives of f are continuous and square-integrable with respect to ρ,

$$\int \left(\frac{\partial f}{\partial x_i}(\mathbf{x})\right)^2 \rho(\mathbf{x})\,d\mathbf{x} \;<\; \infty. \tag{21}$$

[1] Recent work suggests that the probability of terminating at a stationary point that is not a local minimizer is zero [29].

For f that satisfies Assumption 1, define the $m \times m$ symmetric positive semidefinite matrix $\boldsymbol{C} = \boldsymbol{C}(f, \rho)$ as

$$\boldsymbol{C} = \int \nabla f(\mathbf{x}) \nabla f(\mathbf{x})^T \rho(\mathbf{x}) \, d\mathbf{x} = \boldsymbol{W} \Lambda \boldsymbol{W}^T, \tag{22}$$

where Λ is the diagonal matrix of nonnegative eigenvalues $\lambda_1, \ldots, \lambda_m$ in decreasing order, and $\boldsymbol{W}$ is the orthogonal matrix of corresponding eigenvectors. Assume also that $\lambda_n > \lambda_{n+1}$ for some $n < m$, and consider the partition

$$\Lambda = \begin{bmatrix} \Lambda_1 & \\ & \Lambda_2 \end{bmatrix}, \qquad \boldsymbol{W} = \begin{bmatrix} \boldsymbol{W}_1 & \boldsymbol{W}_2 \end{bmatrix}, \tag{23}$$

where Λ_1 contains the first n eigenvalues, and $\boldsymbol{W}_1$ contains the first n eigenvectors. The active subspace [16,31] is the span of the columns of $\boldsymbol{W}_1$. The eigenvalues reveal whether f is a ridge function, as seen in the following theorem.

Theorem 1. *For f that satisfies Assumption 1, assume that $\lambda_n > \lambda_{n+1}$ for some $n < m$. A vector $\mathbf{w}$ is in the null space of $\boldsymbol{C}$ from (22) if and only if $f(\mathbf{x})$ is constant along $\mathbf{w}$ for all $\mathbf{x}$ in the support of ρ.*

The proof of Theorem 1 is in Appendix A. We next consider a ridge approximation constructed with the subspace span($\boldsymbol{W}_1$); the next theorem, which is Theorem 3.1 from [31], bounds the $L^2(\rho)$ approximation error. The key to the following theorem is a weighted Poincaré inequality for the weight function ρ. Edmunds and Opic [32] provide details of the Poincaré constant with weight function ρ, and Bebendorf [33] proves a Poincaré inequality for convex domains. Chen [34] shows that the Poincaré constant is 1 when ρ is a standard normal density, and he proves a version of the Poincaré inequality for a correlated normal density. The following theorem assumes that ρ admits a Poincaré inequality.

Theorem 2. *For f that satisfies Assumption 1 and ρ that admits a Poincaré inequality, define μ as in (7). Then*

$$\left\| f(\mathbf{x}) - \mu(\boldsymbol{W}_1^T \mathbf{x}, \boldsymbol{W}_1) \right\|_{L^2(\rho)} \leq C \left(\lambda_{n+1} + \cdots + \lambda_m\right)^{\frac{1}{2}}, \tag{24}$$

where $C = C(\rho)$ is the Poincaré constant associated with the probability density function ρ.

The proof of Theorem 2 is in Section 3 of [31] and Section 4.2 of [16]. Note that if we write μ as $\mu(\mathbf{x}, \boldsymbol{V})$, as in (15), then Theorem 2 holds for $\mu(\mathbf{x}, \boldsymbol{W}_2)$. The next theorem shows that the active subspace is near-stationary when $\rho(\mathbf{x})$ is a standard Gaussian density and $f(\mathbf{x})$ is Lipschitz continuous.

Theorem 3. *Let ρ be a standard Gaussian density on $\mathbb{R}^m$, and assume that f satisfies Assumption 1 with ρ a Gaussian density. Additionally, assume that*

(i) *f is Lipschitz continuous with constant L,*
(ii) *$\lambda_n > \lambda_{n+1}$ for some $n < m$.*

Then for R as in (17) and $\boldsymbol{W}_2$ from (23),

$$\left\| \bar{\nabla} R(\boldsymbol{W}_2) \right\|_F \leq L \left(2m^{\frac{1}{2}} + (m-n)^{\frac{1}{2}}\right) \left(\lambda_{n+1} + \cdots + \lambda_m\right)^{\frac{1}{2}}, \tag{25}$$

where $\bar{\nabla}$ denotes the gradient on $\mathbb{G}(m-n, m)$, and $\|\cdot\|_F$ is the Frobenius norm.

The proof of Theorem 3 is in Appendix B. The bound's dependence on the eigenvalues implies that if f is a ridge function of n variables, then span($\boldsymbol{W}_1$) is a stationary point for the minimization (12). We expect that the Gaussian assumption on ρ can be relaxed at the cost of a more complicated bound in (25) involving the gradient of ρ. Such an extension is beyond the scope of this manuscript.

4. Computational examples

Theorems 1 and 3 suggest a computational heuristic for fitting a ridge approximation. Assuming the gradient $\nabla f(\mathbf{x})$ can be evaluated as a subroutine (e.g., via algorithmic differentiation [35]), consider the steps in Algorithm 1.

Constantine and Gleich [36] analyze a Monte Carlo method for estimating $\boldsymbol{C}$ and its eigendecomposition as in step 1 of Algorithm 1. If the estimated eigenvalues do not decay appropriately to choose n in step 2, then the given $f(\mathbf{x})$ may not be a good candidate for ridge approximation. It is easy to construct functions that are not amenable to ridge approximation, e.g., $f(\mathbf{x}) = \|\mathbf{x}\|^2$ or any radially symmetric function; such structure would manifest as little-to-no decay in the eigenvalues. In Section 4.2, we offer a computational heuristic for step 3 of Algorithm 1 based on alternating minimization.

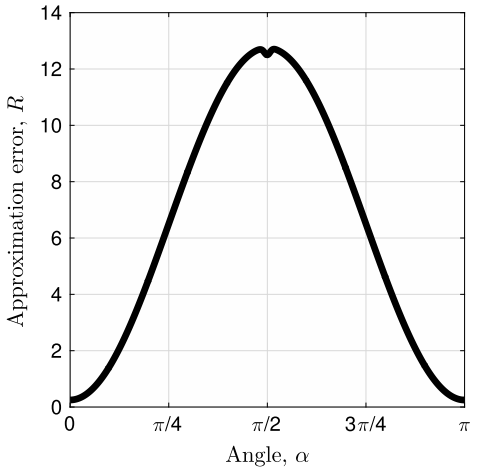


Fig. 1. The $L^2(\rho)$ error (11) as a function of subspace angle α for the ridge approximation of $f(x_1, x_2) = 5x_1 + \sin(10\pi x_2)$. The active subspace is span$([0, 1]^T)$ – corresponding to $\alpha = \pi/2$ – which is a poor initial guess for a gradient-based optimizer.

Algorithm 1 Exploiting the active subspace for ridge approximation

1. Estimate the matrix $\boldsymbol{C}$ from (22) with a numerical integration rule, and compute its eigendecomposition.
2. Choose n such that $\lambda_n > \lambda_{n+1}$ and $\lambda_{n+1}, \ldots, \lambda_m$ are relatively small.
3. Use the first n estimated eigenvectors as an initial guess for numerical optimization of (12).

4.1. A simple example where the heuristic fails

The heuristic in Algorithm 1 relies on $\boldsymbol{C}$'s eigenvalues to measure the suitability of the associated eigenvectors for an initial guess when fitting a ridge approximation. We show a bivariate example where there is a large gap between the first and second eigenvalues, but the first eigenvector – though a stationary point – is far from the global minimizer of the objective function (11). In the bivariate case, we can parameterize the rotation in the two-dimensional domain by one angle $\alpha \in [0, \pi]$.

Let $\rho(x_1, x_2)$ be a standard bivariate Gaussian density, and consider the bivariate function

$$f(x_1, x_2) = 5x_1 + \sin(10\pi x_2). \tag{26}$$

This function has a Lipschitz constant L that is bounded by 32. The matrix $\boldsymbol{C}$ from (22) is (to 4 significant digits)

$$\boldsymbol{C} = \begin{bmatrix} 25.00 & 0 \\ 0 & 526.4 \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}}_{\boldsymbol{W}} \underbrace{\begin{bmatrix} 526.4 & 0 \\ 0 & 25.00 \end{bmatrix}}_{\Lambda} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}^T. \tag{27}$$

We estimate $\boldsymbol{C}$ with a tensor product Gauss–Hermite quadrature rule with 101 points per dimension (10 201 total points), which was sufficient for four digits of accuracy.

According to the heuristic in Algorithm 1, the eigenvalues Λ suggest that the vector $[0, 1]^T$, which corresponds to $\alpha = \pi/2$, would be a good starting point for a numerical optimization. Fig. 1 plots the error R as a function of the subspace angle α for 500 values of $\alpha \in [0, \pi]$. Each R is computed with Gauss–Hermite quadrature rule with 301 points in each dimension (90 601 total points), which is sufficient for four digits of accuracy. The figure shows that $[0, 1]^T$ (i.e., $\alpha = \pi/2$) is actually a local minimizer of R with $R([0, 1]^T) = 12.5$. A gradient-based optimization routine starting at $[0, 1]^T$ is unlikely to escape the local minimum. In contrast, $R([1, 0]^T) = 0.25$, where span$([1, 0]^T)$ (corresponding to $\alpha = 0$) is the orthogonal complement of the active subspace. In other words, the eigenvector associated with the smaller eigenvalue is both a stationary point and a minimizer.

This example suggests a type of function for which the heuristic is not well suited, namely, functions that oscillate rapidly along one direction and vary slowly but consistently along another. The derivative-based metrics choose the direction of oscillation as the important direction even when a ridge approximation is more accurate, in the mean-squared sense, along another direction.

4.2. A ridge approximation response surface for drag in a transonic airfoil

The following numerical example demonstrates the suitability of the heuristic in Algorithm 1 applied to a model of drag coefficient as a function of airfoil shape in a standard transonic test problem; more details on the model follow. We emphasize that the goal of the study is to assess the quality of the gradient-based active subspaces as an initial guess for a numerical optimization method for the ridge approximation problem. It is not our goal to assess the quality of the ridge approximation; however, we do present metrics on the approximation quality for completeness.

First, we propose a computational heuristic for estimating the minimizer of (12) based on a polynomial model and alternating minimization [37]; this corresponds to step 3 of Algorithm 1. The polynomial model plays the role of μ in the approximation error (11). Let $p_N(\mathbf{y}, \theta)$ be a polynomial of degree N in n variables (i.e., $\mathbf{y} \in \mathbb{R}^n$), where θ is the vector of the polynomial's parameters (i.e., coefficients). The dimension of θ depends on the number of terms in the polynomial model, which depends on N and n. In our experiments, we use a multivariate polynomial model of total degree N, so the number of terms is $\binom{N+n}{n}$ [38]. This model is more general than the projection pursuit regression model (2), since it includes products of powers of the n linear combinations.

Algorithm 2 Polynomial-based alternating minimization scheme for (12)

Given M input/output pairs $(\mathbf{x}_i, f(\mathbf{x}_i))$, $\boldsymbol{U}_0 \in \mathbb{R}^{m\times n}$ with orthogonal columns, polynomial degree N, and number of iterations P.

For i from 1 to P, do

1. Compute $\mathbf{y}_i = \boldsymbol{U}_0^T \mathbf{x}_i$ for $i = 1, \ldots, M$.
2. Compute θ_* as the solution to the least-squares problem,
$$\underset{\theta}{\text{minimize}} \quad \sum_{i=1}^{M} \left(f_i - p_N(\mathbf{y}_i, \theta)\right)^2. \tag{28}$$
3. Compute $\boldsymbol{U}_*$ as the solution to the Grassmann manifold-constrained least-squares problem,
$$\begin{aligned} &\underset{\boldsymbol{U}}{\text{minimize}} && \sum_{i=1}^{M} \left(f_i - p_N(\boldsymbol{U}^T\mathbf{x}_i, \theta_*)\right)^2, \\ &\text{subject to} && \boldsymbol{U} \in \mathbb{G}(n, m). \end{aligned} \tag{29}$$
4. Set $\boldsymbol{U}_0 = \boldsymbol{U}_*$.

Algorithm 2 warrants several comments. We use an alternating scheme over the two sets of variables, θ and $\boldsymbol{U}$, because of its simplicity. Alternating schemes are known to stall and/or converge very slowly relative to gradient-based approaches using all variables [39, Section 9.3]. For this reason, we do not offer specific stopping criteria in Algorithm 2. Instead, we opt for a user-defined number P of iterations, which requires more intervention from the user; in the experiment below, we use $P = 20$, which was sufficient to demonstrate the efficiency of the active subspace as a starting point $\boldsymbol{U}_0$. Also, the gradient of the objective function in (29) is much easier to implement than R from (12) or (17), since $p_N(\mathbf{y}, \theta_*)$ is independent of $\boldsymbol{U}$—unlike $\mu = \mu(\mathbf{y}, \boldsymbol{U})$ from (7). However, analyzing the ridge approximation with p_N is much more difficult. The Python codes that implement Algorithm 2 can be found at bitbucket.org/paulcon/near-stationary-subspace. We use the package Pymanopt [40] to estimate the Grassmann manifold-constrained least-squares problem in (29) with the gradient-based steepest descent method.

Relative to standard polynomial-based response surfaces, the ridge approximation can – for the same number M of function evaluations $(\mathbf{x}_i, f_i)$ – fit a higher degree polynomial along the directions that $f(\mathbf{x})$ varies. In other words, with M fixed in (28), the degree N can be much larger in n variables than in $m > n$ variables. However, if several iterations

of the alternating heuristic are needed to achieve some stopping criteria, then fitting the ridge approximation may itself be costly due to the relatively expensive Grassmann-constrained minimization step. Therefore, a good initial subspace can be very advantageous, as seen in the following example.

We apply the alternating minimization heuristic to build a ridge approximation response surface for an aerospace engineering model of a transonic airfoil's drag coefficient as a function of its shape; details on this model are in [16, Section 5.3]. The baseline airfoil is the NACA0012 (a standard transonic test case for computational fluid dynamics), and perturbations to the baseline shape are parameterized by $m = 18$ Hicks–Henne bump functions. Each of the 18 parameters is constrained to the interval $[-0.01, 0.01]$ to ensure valid airfoil geometries, and we choose $\rho(\mathbf{x})$ to be a uniform density on the hypercube $[-0.01, 0.01]^{18}$. Note that this density does not satisfy the Gaussian assumption of Theorem 3, but this does not impede us from numerically testing the heuristic in Algorithm 1. Given the airfoil geometry, the drag coefficient is computed with the Stanford University Unstructured (SU2) compressible Euler solver [41]. This software also solves the continuous adjoint equation for the Euler equations, which enables the computation of the gradient of the drag coefficient with respect to the 18 shape parameters. To summarize, $f(\mathbf{x})$ is the airfoil's drag coefficient as a function of the shape parameters, and a computer model returns f and ∇f given $\mathbf{x}$. Preliminary tests with the model (e.g., standard grid convergence tests and parameter sweeps) indicate that, using our chosen mesh and solver tolerances in SU2, we can expect at least four digits of accuracy in each evaluation of drag and its partial derivatives over the parameter space, which was sufficient for our experiments. One evaluation of both drag and its gradient takes approximately 2 minutes on one core; in total, we ran 20 000 simulations for the following experiments. Given the data set of 20 000 simulations, all computations for the following experiments were run on two nodes of the Colorado School of Mines Mio cluster. Each experiment used 4 cores, which accelerated the Numpy operations.

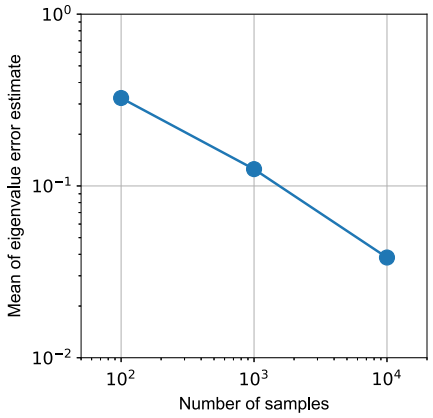
We estimate $\boldsymbol{C}$ from (22) with Latin hypercube sampling [42]. Constantine and Gleich [36] analyzed a simple Monte Carlo method for estimating active subspaces. We first generate 20 000 Latin hypercube samples $\{\mathbf{x}_i\}$, and we compute drag $f(\mathbf{x}_i)$ and its gradient vector $\nabla f(\mathbf{x}_i)$ (computed via the adjoint equations in SU2) for each sample. We use all 20 000 samples to compute reference values $\boldsymbol{C}_{\text{ref}} = \boldsymbol{W}_{\text{ref}} \Lambda_{\text{ref}} \boldsymbol{W}_{\text{ref}}^T$. We then select the first 100, 1000, and 10 000 samples from the Latin hypercube design and estimate $\boldsymbol{C}$ with the associated subsets of $\{\nabla f(\mathbf{x}_i)\}$; note that the subsets of $\{\mathbf{x}_i\}$ do not satisfy the Latin property of the original Latin hypercube design [43]. To verify the expected $\mathcal{O}(N^{-1/2})$ of the Monte Carlo scheme, we compute the following two error metrics:

$$\begin{aligned} \text{err}_{\Lambda,N} &= \frac{1}{m} \sum_{k=1}^{m} \frac{|\lambda_{\text{ref},k} - \lambda_{N,k}|}{|\lambda_{\text{ref},k}|}, \\ \text{err}_{\boldsymbol{W},N} &= \frac{1}{m-1} \sum_{k=1}^{m-1} \left\| \boldsymbol{W}_{\text{ref},k} \boldsymbol{W}_{\text{ref},k}^T - \boldsymbol{W}_{N,k} \boldsymbol{W}_{N,k}^T \right\|_2, \end{aligned} \tag{30}$$

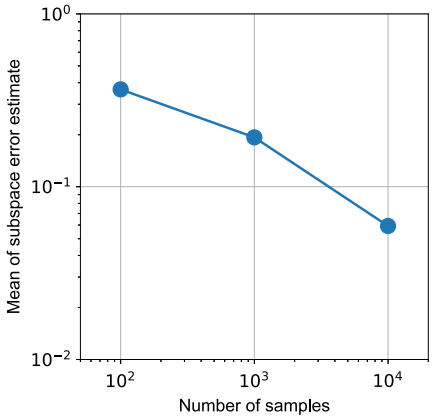
where a subscript k on λ indicates the kth eigenvalue (ordered in descending order), a subscript k on $\boldsymbol{W}$ indicates the first k eigenvectors, a subscript ref indicates the reference value, and the subscript N indicates using the first N samples, where $N \in \{100, 1000, 10000\}$. The error $\text{err}_{\Lambda,N}$ is the average relative error across eigenvalue estimates. Fig. 2(a) shows this error as a function of N. The convergence rate of $\mathcal{O}(N^{-1/2})$ appears as expected, which verifies the Monte Carlo implementation. The error $\text{err}_{\boldsymbol{W},N}$ is the average subspace error over subspaces of dimension 1 through $m-1$. Each term in the sum is the matrix 2-norm difference between projector matrices onto their respective subspaces, which is also the sine of the principal angle between subspaces; this is a common measure of subspace distance [44, Chapter 2]. Fig. 2(b) shows this error as a function N, and it shows the expected convergence rate, which verifies the Monte Carlo implementation.

For a fixed number of samples, Constantine and Gleich [36] propose a bootstrap-based heuristic for estimating the variability in the Monte Carlo estimates of the eigenvalues and subspaces, which is similar to the bootstrap-based eigenvalue standard error estimates from Efron and Tibshirani [45, Chapter 7.1]. Since the eigenvalue and subspace estimates are nonlinear functions of the data, a central limit theorem-based standard error is not possible. For a data set with N elements, the bootstrap method draws N samples with replacement from the data; computing the desired quantities from this data set (e.g., the eigenvalues of $\boldsymbol{C}$'s estimate) yields a *bootstrap replicate*. Repeating this procedure many times produces a set of bootstrap replicates – eigenvalues and subspaces, in our case – for the same original data set of N elements.

Fig. 3(a) shows (i) the first 10 eigenvalue estimates as black dots and (ii) the bootstrap ranges (min/max over the replicates) in the shaded region for the first $N = 1000$ samples from the 20 000-sample Latin hypercube design. The

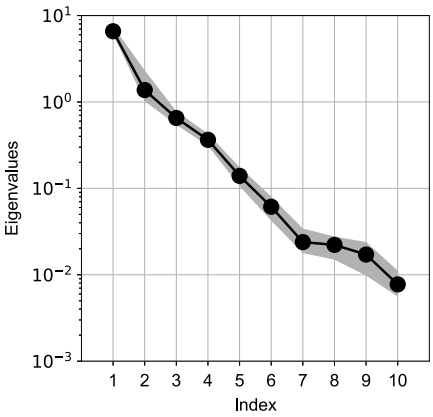


(a) Eigenvalue error estimates.

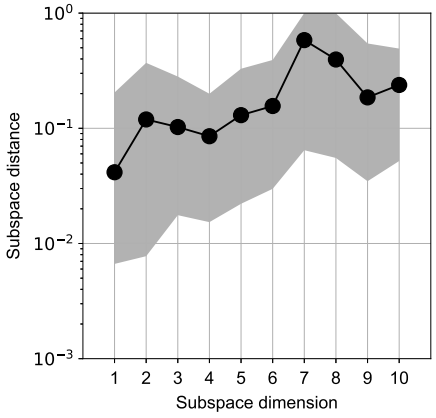


(b) Subspace error estimates.

Fig. 2. Eigenvalue and subspace errors (30) with respect to reference values computed from a 20 000-sample Latin hypercube design.



(a) Eigenvalue estimates.



(b) Subspace distance estimates.

Fig. 3. Bootstrap-based variability estimates for eigenvalues and subspaces using the first $N = 1000$ samples from the 20 000-sample Latin hypercube design.

tight bootstrap ranges suggest that there is not much variability in the eigenvalue estimates under perturbations to the data. For each bootstrap replicate of the first k eigenvectors, $\boldsymbol{W}^*_{N,k}$, the k-dimensional subspace distance is computed as

$$\text{err}^*_{N,k} = \left\| \boldsymbol{W}_{N,k}\boldsymbol{W}_{N,k}^T - \left(\boldsymbol{W}^*_{N,k}\right)\left(\boldsymbol{W}^*_{N,k}\right)^T \right\|_2 . \tag{31}$$

Fig. 3(b) shows the mean (black dots) and min/max range (shaded region) over these subspace distance replicates as a function of subspace dimension. We emphasize that calculations represented by Figs. 3(a) and 3(b) do not use reference values; they only include a subset of $N = 1000$ samples from the 20 000-sample Latin hypercube design.

The bootstrap-based subspace variability estimates behave consistently with the error analysis from Gleich and Constantine [36]. In particular, a small subspace error for subspace dimension n corresponds to a large gap between eigenvalues λ_n and λ_{n+1}; this is consistent with well known perturbation results for invariant subspaces [46]. Constantine et al. [31] showed that errors in the active subspace have substantial impact on errors in the ridge approximation $\mu(\boldsymbol{W}_1^T\mathbf{x}, \boldsymbol{W}_1)$ from Theorem 3. However, for an optimization-based ridge approximation from Algorithm 2, since the numerically estimated active subspace is used only as an initial guess, we expect errors in the subspace to matter little; this is supported by the following experiment.

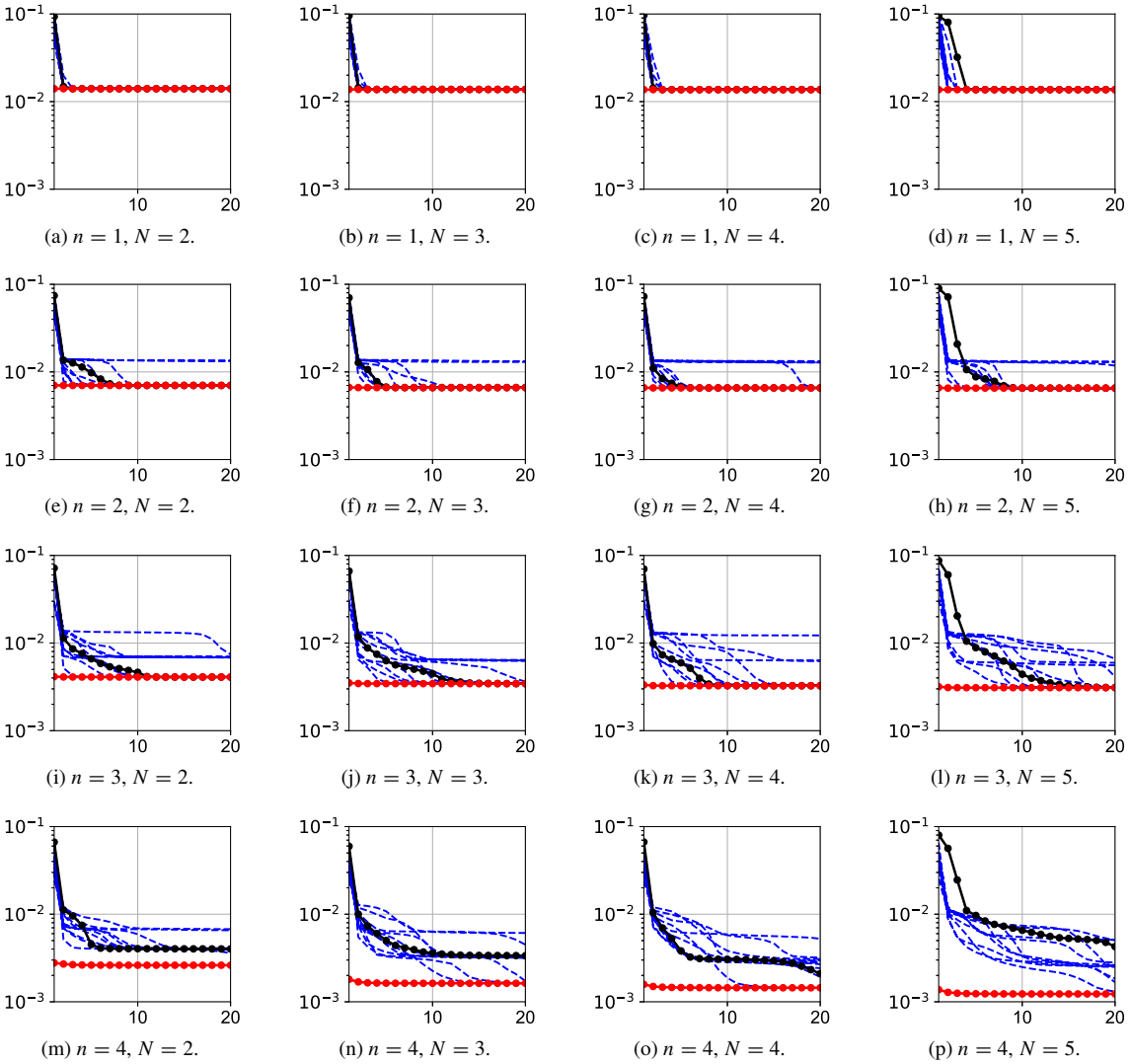


Fig. 4. Each subfigure shows the residual (29) – using the first 1000 samples from the 20000-sample Latin hypercube design – as a function of the iteration in the alternating minimization heuristic Algorithm 2. The black connected dots use the first n columns of the $m \times m$ identity matrix as $\boldsymbol{U}_0$. The blue dashed lines show results using 10 random starting points for $\boldsymbol{U}_0$. The red connected dots use the first n eigenvectors of $\boldsymbol{C}_{\text{ref}}$. The subfigures vary the number n of linear combinations from 1 to 4 (top to bottom) and the degree N of the polynomial approximation from 2 to 5 (left to right). In all cases, the n eigenvectors of the numerically estimated $\boldsymbol{C}$ from (22) provide a superior starting point. (Color figures appear in the web version of this article.)

Fig. 4 shows the results of an experiment comparing different starting points $\boldsymbol{U}_0$ for the alternating heuristic in Algorithm 2: (i) a random $m \times n$ matrix with orthogonal columns, (ii) the first n columns of the $m \times m$ identity matrix, and (iii) the first n estimated eigenvectors from the 20 000-sample reference estimate $\boldsymbol{C}_{\text{ref}}$ of $\boldsymbol{C}$ from (22). The data in each case is the first $M = 1000$ pairs $(\mathbf{x}_i, f(\mathbf{x}_i))$ – i.e., shape parameters and associated drag – from the 20 000-sample Latin hypercube design of experiments. We ran similar experiments for the first 100, 500, 5000, and 10 000 pairs—all results were qualitatively similar. Each subfigure in Fig. 4 shows the residual (29) as a function of the iteration count; the residual value is on the vertical axis, and the number of iterations is on the horizontal axis. For the Grassmann manifold optimization in (29), we set the maximum number of steepest descent steps to 10, which made each iteration considerably faster than converging to a specified tolerance on the norm of the residual gradient on the Grassmann. The black connected dots show the results using the identity matrix starting point; the blue dashed lines show results

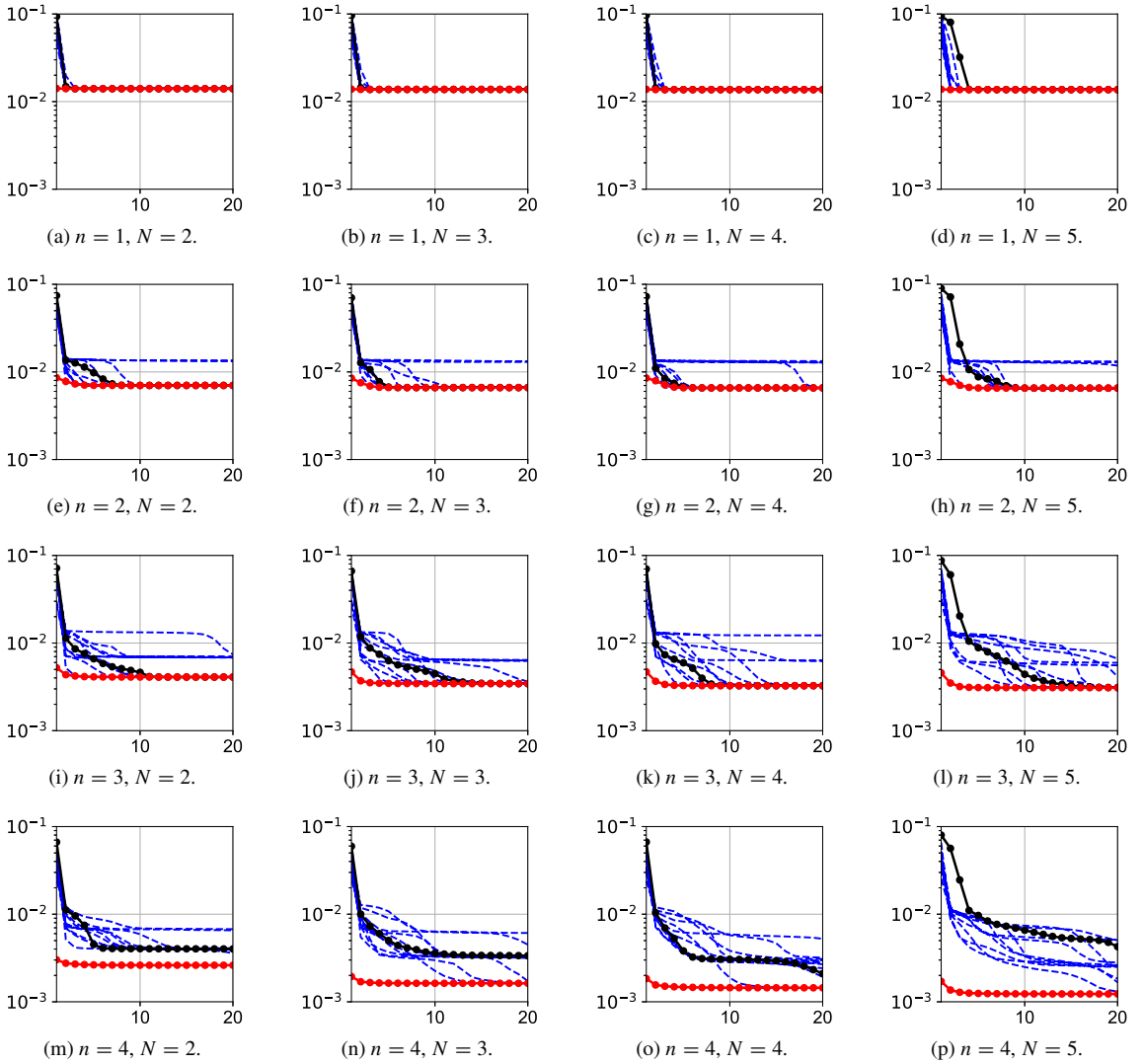


Fig. 5. Identical to Fig. 4, except the active subspace initial guesses are computed with only 10 gradient samples. Notice the slightly greater decrease in the first few iterations, compared to Fig. 4, that results from the lower accuracy approximation.

from 10 different random starting points; and the red connected dots show results using the first n eigenvectors of $\boldsymbol{C}_{\text{ref}}$. The subfigures vary the polynomial degree N from 2 to 5 (left to right) and the number n of linear combinations from 1 to 4 (top to bottom). For every case, the first n eigenvectors of $\boldsymbol{C}_{\text{ref}}$ provide a superior starting point for the alternating heuristic, and the advantage increases as N and n increase.

Computing the active subspace may take significant computational effort, whereas generating a random starting point or a few columns of the identity matrix takes relatively no computational effort. The experiment from Fig. 4 uses estimated eigenvectors from the very expensive 20 000-sample reference value $\boldsymbol{C}_{\text{ref}}$. In Fig. 5, we repeat the experiment using a 10-sample estimate of $\boldsymbol{C}$'s eigenvectors; the results are remarkably similar. There is a marginally greater decrease in the residual from the initial cheap, 10-sample estimate of the active subspace relative to the 20 000-sample reference value. This suggests that a cheap estimate of the active subspace may still be a better starting point than a random starting point for the ridge approximation heuristic in Algorithm 2. For this particular data set of $M = 1000$ runs, a marginal cost of 10 runs (with adjoint-based gradients) to compute a good initial subspace for ridge

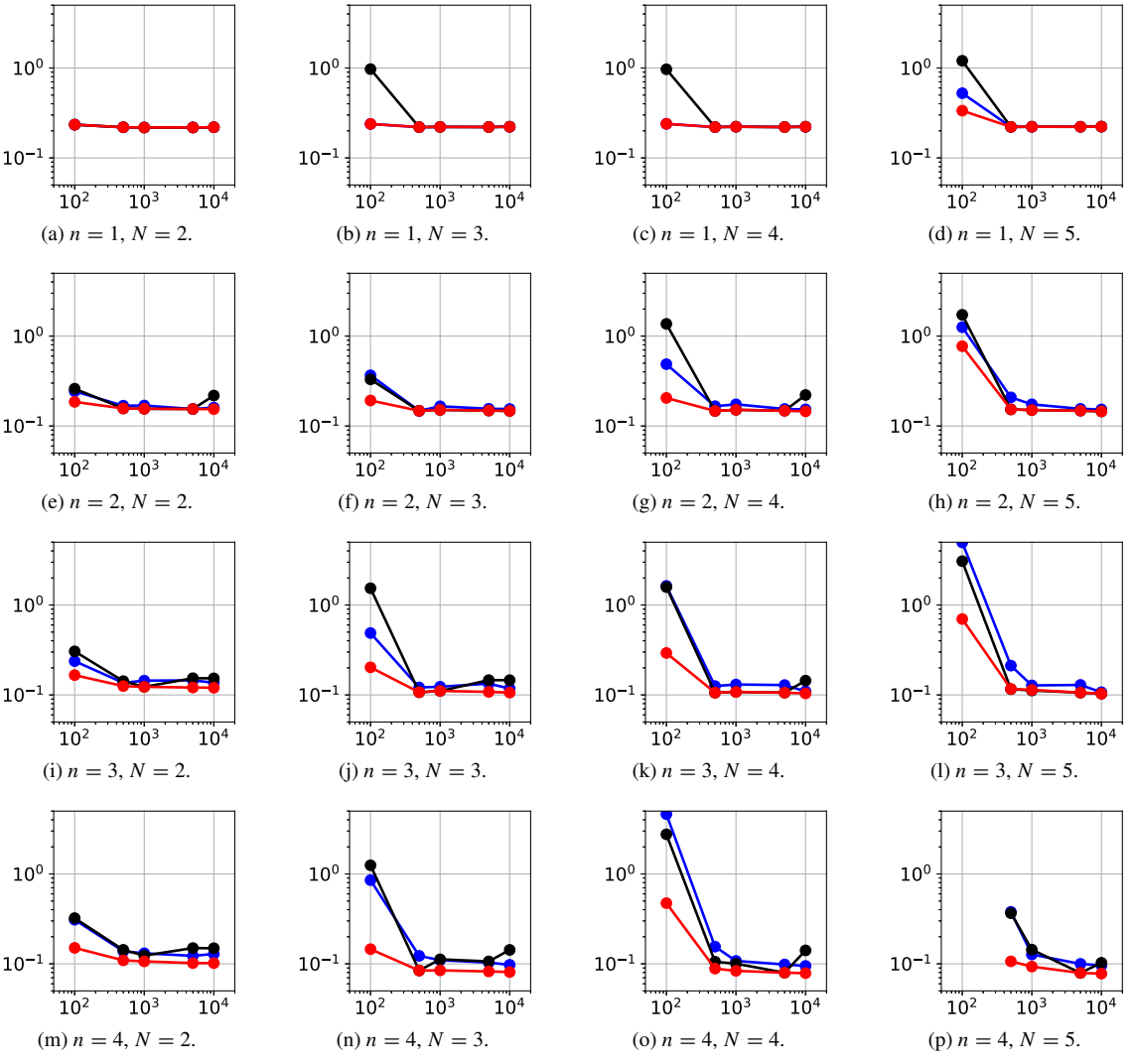


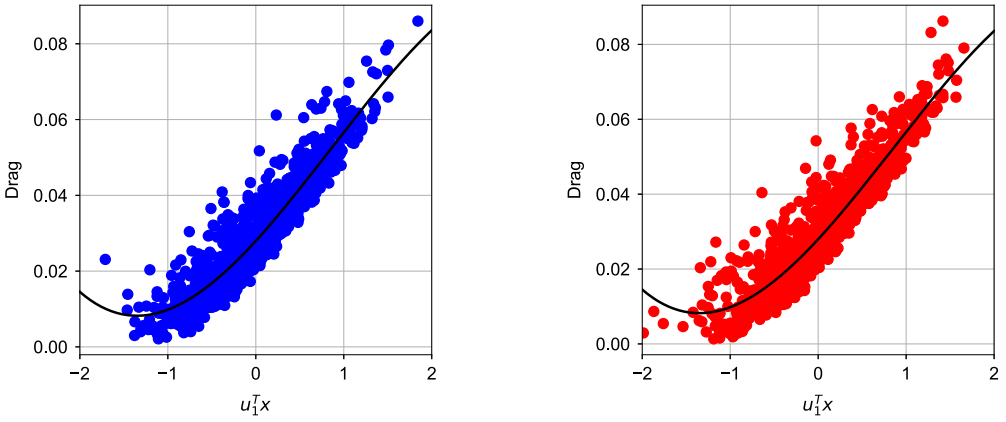
Fig. 6. Average pointwise relative errors in the polynomial ridge approximation – fitted with 20 iterations of the alternating scheme in Algorithm 2 – on a testing set of 10 000 samples as a function of the number of training samples. The colors correspond to a ridge approximation with different initial subspaces: black is uses the first n columns of the identity matrix, blue is the average testing error over 10 random initial subspaces, and red uses the n-dimensional active subspace estimate. (Color figures appear in the web version of this article.)

approximation is quite reasonable. The specific trade-offs for other models and data sets will depend on the relative cost of estimating the eigenvectors of $\boldsymbol{C}$ versus fitting the ridge approximation.

For completeness, we study the error in the fitted ridge approximation as the number of training samples increases, $M \in \{100, 500, 1000, 5000, 10000\}$. We emphasize that it is not our primary goal to show that the ridge approximation model is ideal for the drag quantity of interest. To estimate error, we split the 20 000-sample Latin hypercube design into a training set of maximum size 10 000 and a testing set with 10 000 samples. The testing error is computed as the average relative pointwise error over the testing set,

$$\frac{1}{M_{\text{test}}} \sum_{j=1}^{M_{\text{test}}} \frac{|f(\mathbf{x}_j) - p_N(\boldsymbol{U}^T\mathbf{x}_j, \theta)|}{|f(\mathbf{x}_j)|}, \tag{32}$$

where $\boldsymbol{U}$ and θ are fitted with 20 iterations of the alternating heuristic from Algorithm 2. Fig. 6 shows the testing error as a function of the number M of training samples for each combination of N (polynomial degree) and n (number



(a) Train. (b) Test.

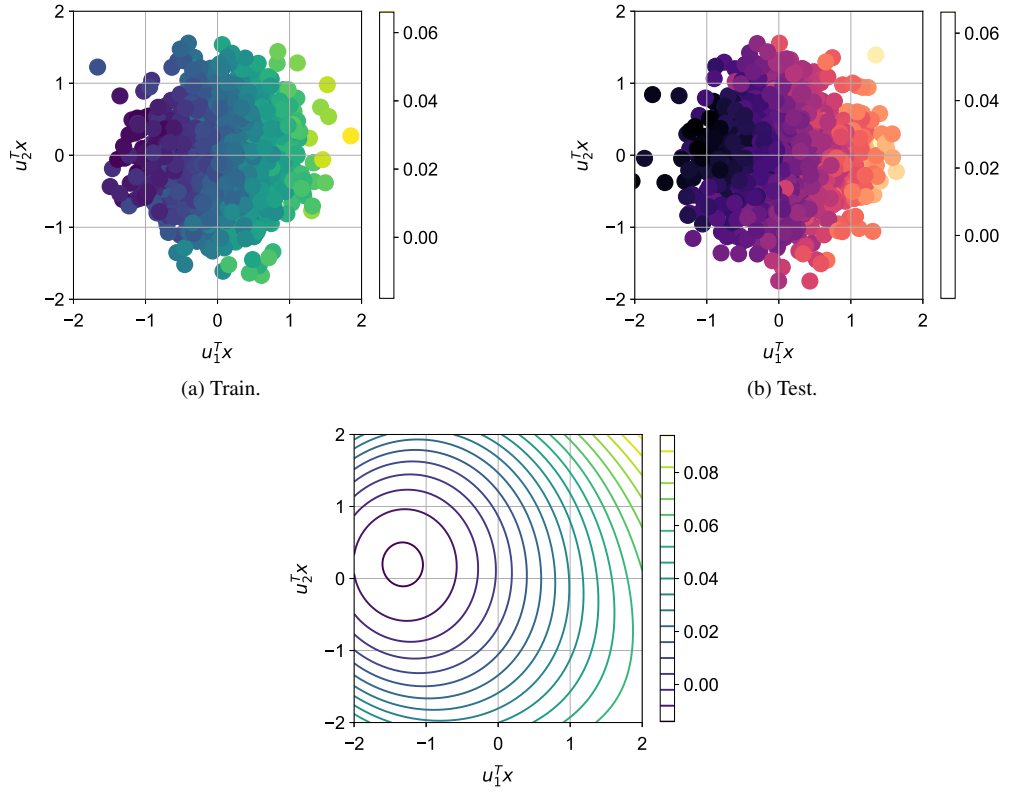
Fig. 7. One-dimensional shadow plots on a training (left) and testing (right) set of 1000 input/output pairs. The black line is the fitted univariate polynomial of degree $N = 5$ used in the ridge approximation. The globally monotonic structure may yield insight for specific design problems.

of linear combinations). Note that 100 samples are too few to fit a polynomial of degree $N = 5$ in $n = 4$ variables, so the bottom right subfigure is missing one data point relative to the other subfigures. The colors indicate the initial subspace guess in Algorithm 2: black is the first n columns of the identity matrix, blue is the average error over 10 random n-dimensional subspaces, and red is the active subspace. In most cases, the error is slightly smaller with the active subspace initial guess. However, all initial guesses lead to roughly the same testing error, which flattens after 500 training samples. Also, increasing the polynomial degree beyond $N = 2$ has hardly any impact on the testing error. Increasing the number of linear combinations has a minor effect—decreasing the testing error by approximately 50% from $n = 1$ to $n = 4$.

One advantage of the ridge approximation with $n = 1$ or $n = 2$ linear combinations is the possibility of visualizing the input/output relationship with simple scatter plots. More precisely, for the data set of input/output pairs $(\mathbf{x}_i, f(\mathbf{x}_i))$, if $\boldsymbol{U}$ has 1 or 2 columns, then we can create scatter plots of the pairs $(\boldsymbol{U}^T\mathbf{x}_i, f(\mathbf{x}_i))$. In regression problems where the data set consists of predictor/response pairs from an unknown joint density, such plots are called *sufficient summary plots* [47]. The name *sufficient* is used to imply that the plots are statistically sufficient for the data set, which is a technical condition on the linear combination weights $\boldsymbol{U}$; for details, see Cook [47]. Since our data sets are not random in the sense of regression data (i.e., the function $f(\mathbf{x})$ is deterministic), we call these plots *shadow plots*, since they resemble a shadow of a high-dimensional surface projected on a wall. For the $n = 1$ case, in-to and out-of the page represents $m - 1$-dimensional subspaces in the input space.

Fig. 7 shows the one-dimensional shadow plots of 1000 training samples and 1000 testing samples when $n = 1$ along with the one-dimensional polynomial ridge approximation with degree $N = 5$ (black line). The similarity between the plots suggests that the ridge direction $\boldsymbol{U}$ would be similar if computed from the testing samples. The plot can be used to assess the quality of the one-dimensional ridge approximation. By visual inspection, the error in the one-dimensional ridge approximation may be up to roughly 25% for this problem, and this is consistent with the top row of plots in Fig. 6. Although this error may be unacceptably large for pointwise approximation purposes (e.g., response surface-based optimization or uncertainty quantification), the plot reveals a globally monotonic trend in drag as a function of the shape parameters over the parameter space. Such insight may be exploited when designing an airfoil for drag. Moreover, the plot may assist in determining the set of shapes that produce sufficiently small drag for a constrained optimization; Constantine et al. [48] use a similar approach for quantifying uncertainty in a scramjet model. Fig. 8 shows the two-dimensional shadow plots, where the color is the value of drag. The contour plot shows the contours of the bivariate polynomial defining the two-dimensional ridge approximation. The quadratic behavior is apparent in the bivariate polynomial; again, this insight may be valuable to designers.

The results of these experiments provide numerical support that the first n eigenvectors of $\boldsymbol{C}$ from (22)– even a cheap estimate – provide a good initial subspace $\boldsymbol{U}_0$ for the alternating heuristic for ridge approximation of this aerospace model. This enables us to fit a ridge approximation with a relatively high polynomial degree along important directions in the model's input space.



(a) Train. (b) Test.

(c) Contour.

Fig. 8. Two-dimensional shadow plots on a training (top left) and testing (top right) set of 1000 input/output pairs. The contour plot (bottom) is the bivariate polynomial of degree $N = 5$ used in the ridge approximation. (Color figures appear in the web version of this article.)

5. Summary and conclusions

Motivated by response surface construction for expensive computational models with several input parameters, we study ridge approximation for functions of several variables. A ridge function is constant along a set of directions in its domain, and the approximation problem is to find (i) optimal directions and (ii) an optimal function of the linear combinations of variables. For a fixed set of directions, the best approximation in the mean-squared sense is a particular conditional average. We define an optimal subspace as one that minimizes a mean-squared cost function over the Grassmann manifold of subspaces. We then prove that a particular subspace – the active subspace defined by the function's gradient – is a near-stationary point for an optimization defining the optimal subspace. We offer a heuristic to exploit this fact when fitting a ridge approximation. Our first numerical example shows a simple case where this heuristic fails; this case reveals a type of function for which the heuristic is ill-suited, namely, functions that oscillate rapidly along one direction while varying slowly but consistently along another. Our second numerical example demonstrates this heuristic's success with a polynomial-based alternating scheme to fit a ridge approximation applied to an aerospace design model with 18 parameters. The alternating scheme with the active subspace as the initial guess outperforms the same scheme with random initial subspaces. Given the prevalence of anisotropic parameter dependence in most complex physical simulations, we expect that ridge functions are appropriate forms for response surfaces approximations. The analyses and heuristics we present advance the state-of-the-art in ridge approximations.

Acknowledgments

The first author's work is supported by the U.S. Department of Energy Office of Science, Office of Advanced Scientific Computing Research, Applied Mathematics program under Award Number DE-SC-0011077. The first and

third authors' work is supported by Department of Defense, Defense Advanced Research Project Agency's program Enabling Quantification of Uncertainty in Physical Systems. The second author's work is supported by the Alan Turing Institute under the EPSRC grant EP/N510129/1. The fourth author's work is partially funded by NSF CAREER grant #1255631.

Appendix A. Proof of Theorem 1

Let $\mathbf{w} \in \mathbb{R}^m$ be in the null space of $\boldsymbol{C}$ from (22), i.e., $\boldsymbol{C}\mathbf{w} = \mathbf{0}$. Then

$$0 = \mathbf{w}^T \boldsymbol{C}\, \mathbf{w} = \int \left(\mathbf{w}^T \nabla f(\mathbf{x})\right)^2 \rho(\mathbf{x})\, d\mathbf{x}. \tag{A.1}$$

The integrand is the squared directional derivative of f along $\mathbf{w}$. Since the squared quantity is non-negative, its average equaling zero – combined with continuity of ∇f from Assumption 1– implies that the directional derivative $\mathbf{w}^T \nabla f(\mathbf{x})$ is zero for all $\mathbf{x}$ in the support of ρ. Therefore, f is constant along $\mathbf{w}$.

Now assume that f is constant along $\mathbf{w}$ in the support of ρ. Then $\mathbf{w}^T \nabla f(\mathbf{x}) = 0$ for all such $\mathbf{x}$. Then

$$0 = \int \left(\mathbf{w}^T \nabla f(\mathbf{x})\right)^2 \rho(\mathbf{x})\, d\mathbf{x} = \mathbf{w}^T\, \boldsymbol{C}\, \mathbf{w}, \tag{A.2}$$

which implies that $\mathbf{w}$ is in the null space of $\boldsymbol{C}$, since $\boldsymbol{C}$ is positive semidefinite.

Appendix B. Proof of Theorem 3

For $R = R(\boldsymbol{V})$ from (17), consider the gradient of R on the Grassmann manifold $\mathbb{G}(m-n, n)$.

$$\begin{aligned}
\bar{\nabla} R(\boldsymbol{V}) &= \bar{\nabla}\left(\frac{1}{2}\int (f(\mathbf{x}) - \mu(\mathbf{x}, \boldsymbol{V}))^2\, \rho(\mathbf{x})\, d\mathbf{x}\right) \\
&= \frac{1}{2}\int \bar{\nabla}(f(\mathbf{x}) - \mu(\mathbf{x}, \boldsymbol{V}))^2\, \rho(\mathbf{x})\, d\mathbf{x} \\
&= \int (f(\mathbf{x}) - \mu(\mathbf{x}, \boldsymbol{V}))\, \bar{\nabla}(f(\mathbf{x}) - \mu(\mathbf{x}, \boldsymbol{V}))\, \rho(\mathbf{x})\, d\mathbf{x} \\
&= \int (f(\mathbf{x}) - \mu(\mathbf{x}, \boldsymbol{V}))\, (\underbrace{\bar{\nabla} f(\mathbf{x})}_{=0} - \bar{\nabla}\mu(\mathbf{x}, \boldsymbol{V}))\, \rho(\mathbf{x})\, d\mathbf{x} \\
&= \int (\mu(\mathbf{x}, \boldsymbol{V}) - f(\mathbf{x}))\, \bar{\nabla}\mu(\mathbf{x}, \boldsymbol{V})\, \rho(\mathbf{x})\, d\mathbf{x}.
\end{aligned} \tag{B.1}$$

Let μ'_{ij} be the (i, j) element of $\bar{\nabla}\mu$, with $i = 1, \ldots, m$ and $j = 1, \ldots, m-n$. Using Cauchy–Schwarz, we can bound

$$\int (\mu - f)\, \mu'_{ij}\, \rho\, d\mathbf{x} \;\le\; \left(\int (\mu - f)^2\, \rho\, d\mathbf{x}\right)^{\frac{1}{2}} \left(\int (\mu'_{ij})^2\, \rho\, d\mathbf{x}\right)^{\frac{1}{2}}. \tag{B.2}$$

Then

$$\begin{aligned}
\|\bar{\nabla} R(\boldsymbol{V})\|_F^2 &= \sum_{i=1}^{m}\sum_{j=1}^{m-n}\left(\int (\mu - f)\, \mu'_{ij}\, \rho\, d\mathbf{x}\right)^2 \\
&\le \sum_{i=1}^{m}\sum_{j=1}^{m-n}\left(\int (\mu - f)^2\, \rho\, d\mathbf{x}\right)\left(\int (\mu'_{ij})^2\, \rho\, d\mathbf{x}\right) \\
&= \left(\int (\mu - f)^2\, \rho\, d\mathbf{x}\right)\left(\int \sum_{i=1}^{m}\sum_{j=1}^{m-n}(\mu'_{ij})^2\, \rho\, d\mathbf{x}\right) \\
&= \left(\int (\mu - f)^2\, \rho\, d\mathbf{x}\right)\left(\int \|\bar{\nabla}\mu\|_F^2\, \rho\, d\mathbf{x}\right).
\end{aligned} \tag{B.3}$$

Recall Edelman's formula for the Grassmann gradient [28, Section 2.5.3],

$$\bar{\nabla}\mu(\mathbf{x}, \boldsymbol{V}) = (\boldsymbol{I} - \boldsymbol{V}\,\boldsymbol{V}^T)\,\frac{\partial}{\partial \boldsymbol{V}}\mu(\mathbf{x}, \boldsymbol{V}) = \boldsymbol{U}\boldsymbol{U}^T\,\frac{\partial}{\partial \boldsymbol{V}}\mu(\mathbf{x}, \boldsymbol{V}), \tag{B.4}$$

where $\frac{\partial}{\partial \boldsymbol{V}}\mu$ is the $m \times (m-n)$ matrix of partial derivatives of μ with respect to the elements of $\boldsymbol{V}$. For Gaussian ρ, the conditional density

$$\pi(\mathbf{z}|\mathbf{y}) = \pi(\mathbf{z}) \ \propto \ \exp\left(\frac{-\mathbf{z}^T\mathbf{z}}{2}\right) \tag{B.5}$$

is independent of $\boldsymbol{V}$. Therefore,

$$\begin{aligned}\frac{\partial}{\partial \boldsymbol{V}}\mu(\mathbf{x}, \boldsymbol{V}) &= \frac{\partial}{\partial \boldsymbol{V}} \int f((\boldsymbol{I} - \boldsymbol{V}\boldsymbol{V}^T)\mathbf{x} + \boldsymbol{V}\mathbf{z})\,\pi(\mathbf{z})\,d\mathbf{z} \\ &= \int \frac{\partial}{\partial \boldsymbol{V}} f((\boldsymbol{I} - \boldsymbol{V}\boldsymbol{V}^T)\mathbf{x} + \boldsymbol{V}\mathbf{z})\,\pi(\mathbf{z})\,d\mathbf{z}.\end{aligned} \tag{B.6}$$

Next we examine the gradient of f with respect to the elements of $\boldsymbol{V}$. For notation, define $\mathbf{s}$ as

$$\mathbf{s} = \mathbf{s}(\mathbf{x}, \mathbf{z}, \boldsymbol{V}) = (\boldsymbol{I} - \boldsymbol{V}\boldsymbol{V}^T)\mathbf{x} + \boldsymbol{V}\,\mathbf{z}. \tag{B.7}$$

Let v_{ij} be the (i, j) element of $\boldsymbol{V}$, and compute the derivative,

$$\frac{\partial}{\partial v_{ij}} f(\mathbf{s}) = \nabla f(\mathbf{s})^T \left(\frac{\partial}{\partial v_{ij}}\mathbf{s}\right). \tag{B.8}$$

The derivative of $\mathbf{s}$ is

$$\begin{aligned}\frac{\partial}{\partial v_{ij}}\mathbf{s} &= \frac{\partial}{\partial v_{ij}} \left((\boldsymbol{I} - \boldsymbol{V}\boldsymbol{V}^T)\mathbf{x} + \boldsymbol{V}\,\mathbf{z}\right) \\ &= \mathbf{e}_i\,\mathbf{v}_j^T\mathbf{x} + x_i\,\mathbf{v}_j + \mathbf{e}_i\,z_j,\end{aligned} \tag{B.9}$$

where $\mathbf{e}_i$ is the ith column of the $m \times m$ identity matrix, $\mathbf{v}_j$ is the jth column of $\boldsymbol{V}$, x_i is the ith component of $\mathbf{x}$, and z_j is the jth component of $\mathbf{z}$. Then

$$\begin{aligned}\frac{\partial}{\partial v_{ij}} f(\mathbf{s}) &= f_i(\mathbf{s})\,\mathbf{v}_j^T\mathbf{x} + x_i\,\nabla f(\mathbf{s})^T\mathbf{v}_j + f_i(\mathbf{s})\,z_j \\ &= (f_i(\mathbf{s})\,\mathbf{x}^T + x_i\,\nabla f(\mathbf{s})^T)\mathbf{v}_j + f_i(\mathbf{s})\,z_j,\end{aligned} \tag{B.10}$$

where f_i is the ith component of the gradient vector ∇f. Putting i's and j's together,

$$\frac{\partial}{\partial \boldsymbol{V}} f(\mathbf{s}) = \left(\nabla f(\mathbf{s})\,\mathbf{x}^T + \mathbf{x}\,\nabla f(\mathbf{s})^T\right)\,\boldsymbol{V} + \nabla f(\mathbf{s})\,\mathbf{z}^T. \tag{B.11}$$

Then, for $f = f(\mathbf{s})$, $\pi = \pi(\mathbf{z})$, and $\nabla f = \nabla f(\mathbf{s})$,

$$\begin{aligned}\int \frac{\partial}{\partial \boldsymbol{V}} f\,\pi\,d\mathbf{z} &= \int \left(\nabla f\,\mathbf{x}^T + \mathbf{x}\,\nabla f^T\right)\,\boldsymbol{V} + \nabla f\,\mathbf{z}^T\,\pi\,d\mathbf{z} \\ &= \left(\mathbf{g}\,\mathbf{x}^T + \mathbf{x}\,\mathbf{g}^T\right)\,\boldsymbol{V} \ + \int \nabla f\,\mathbf{z}^T\,\pi\,d\mathbf{z},\end{aligned} \tag{B.12}$$

where

$$\mathbf{g} = \mathbf{g}(\mathbf{x}) = \int \nabla f((\boldsymbol{I} - \boldsymbol{V}\boldsymbol{V}^T)\mathbf{x} + \boldsymbol{V}\mathbf{z})\,\pi(\mathbf{z})\,d\mathbf{z}. \tag{B.13}$$

By the Lipschitz continuity of f, $\|\mathbf{g}\| \leq L$. Then we can bound the norm of Grassmann gradient of μ as

$$\begin{aligned}
\|\bar{\nabla}\mu\|_F &= \left\| \boldsymbol{U}\boldsymbol{U}^T \frac{\partial}{\partial \boldsymbol{V}} \mu \right\|_F \\
&\leq \left\| \frac{\partial}{\partial \boldsymbol{V}} \mu \right\|_F \\
&= \left\| \int \frac{\partial}{\partial \boldsymbol{V}} f \, \pi \, d\mathbf{z} \right\|_F \\
&= \left\| \left(\mathbf{g}\,\mathbf{x}^T + \mathbf{x}\,\mathbf{g}^T \right) \boldsymbol{V} + \int \nabla f \, \mathbf{z}^T \, \pi \, d\mathbf{z} \right\|_F \\
&\leq \left\| \left(\mathbf{g}\,\mathbf{x}^T + \mathbf{x}\,\mathbf{g}^T \right) \boldsymbol{V} \right\|_F + \left\| \int \nabla f \, \mathbf{z}^T \, \pi \, d\mathbf{z} \right\|_F \\
&\leq \left\| \mathbf{g}\,\mathbf{x}^T + \mathbf{x}\,\mathbf{g}^T \right\|_F + \left(\int \|\nabla f \, \mathbf{z}^T\|_F^2 \, \pi \, d\mathbf{z} \right)^{\frac{1}{2}} \\
&\leq 2 \, \|\mathbf{g}\| \, \|\mathbf{x}\| + \left(\int \|\nabla f\|^2 \|\mathbf{z}\|^2 \, \pi \, d\mathbf{z} \right)^{\frac{1}{2}} \\
&\leq 2 \, L \, \|\mathbf{x}\| + \left(L^2 \int \|\mathbf{z}\|^2 \, \pi \, d\mathbf{z} \right)^{\frac{1}{2}} \\
&= L \left(2 \, \|\mathbf{x}\| + (m-n)^{\frac{1}{2}} \right).
\end{aligned} \tag{B.14}$$

Therefore,

$$\begin{aligned}
\int \|\bar{\nabla}\mu\|_F^2 \, \rho \, d\mathbf{x} &\leq L^2 \int \left(2 \, \|\mathbf{x}\| + (m-n)^{\frac{1}{2}} \right)^2 \rho \, d\mathbf{x} \\
&\leq L^2 \left(2m^{\frac{1}{2}} + (m-n)^{\frac{1}{2}} \right)^2.
\end{aligned} \tag{B.15}$$

Combining this with (B.3), we have

$$\|\bar{\nabla} R(\boldsymbol{V})\|_F \;\leq\; L \left(2m^{\frac{1}{2}} + (m-n)^{\frac{1}{2}} \right) \left(\int (\mu(\mathbf{x}, \boldsymbol{V}) - f(\mathbf{x}))^2 \, \rho(\mathbf{x}) \, d\mathbf{x} \right)^{\frac{1}{2}}. \tag{B.16}$$

Note that the Poincaré constant for the Gaussian density is $C = 1$ [34]. Then combining (B.16) with Theorem 2 achieves the desired result.

References

[1] T.J. Santner, B.J. Williams, W.I. Notz, The Design and Analysis of Computer Experiments, Springer, New York, 2003. http://dx.doi.org/10.1007/978-1-4757-3799-8.

[2] D. Jones, A taxonomy of global optimization methods based on response surfaces, J. Global Optim. 21 (4) (2001) 345–383. http://dx.doi.org/10.1023/A:1012771025575.

[3] R. Barton, Metamodels for simulation input-output relations, in: Proceedings of the 24th Conference on Winter Simulation, WSC '92, ACM, New York, USA, 1992, pp. 289–299. http://dx.doi.org/10.1145/167293.167352.

[4] J. Sacks, W.J. Welch, T.J. Mitchell, H.P. Wynn, Design and analysis of computer experiments, Statist. Sci. 4 (4) (1989) 409–423. http://www.jstor.org/stable/2245858.

[5] M. Kennedy, A. O'Hagan, Predicting the output from a complex computer code when fast approximations are available, Biometrika 87 (1) (2000) 1–13. http://dx.doi.org/10.1093/biomet/87.1.1.

[6] H. Woźniakowski, Tractability and strong tractability of linear multivariate problems, J. Complexity 10 (1) (1994) 96–128. http://dx.doi.org/10.1006/jcom.1994.1004.

[7] J.F. Traub, A.G. Werschulz, Complexity and Information, Cambridge University Press, Cambridge, 1998.

[8] R. Bellman, Adaptive Control Processes: A Guided Tour, Princeton University Press, Princeton, 1961.

[9] S. Shan, G.G. Wang, Survey of modeling and optimization strategies to solve high-dimensional design problems with computationally-expensive black-box functions, Struct. Multidiscip. Optim. 41 (2) (2010) 219–241. http://dx.doi.org/10.1007/s00158-009-0420-2.

[10] A. Saltelli, M. Ratto, T. Andres, F. Campolongo, J. Cariboni, D. Gatelli, M. Saisana, S. Tarantola, Global Sensitivity Analysis: The Primer, John Wiley & Sons, Ltd, 2008. http://dx.doi.org/10.1002/9780470725184.
[11] I. Sobol', S. Tarantola, D. Gatelli, S. Kucherenko, W. Mauntz, Estimating the approximation error when fixing unessential factors in global sensitivity analysis, Reliab. Eng. Syst. Saf. 92 (7) (2007) 957–960. http://dx.doi.org/10.1016/j.ress.2006.07.001.
[12] A. Pinkus, Ridge Functions, Cambridge University Press, Cambridge, 2015.
[13] S. Keiper, Analysis of generalized ridge functions in high dimensions, in: 2015 International Conference on Sampling Theory and Applications, SampTA, 2015, pp. 259–263. http://dx.doi.org/10.1109/SAMPTA.2015.7148892.
[14] R.C. Smith, Uncertainty Quantification: Theory, Implementation, and Applications, SIAM, Philadelphia, 2013.
[15] T.J. Sullivan, Introduction to Uncertainty Quantification, Springer, New York, 2015.
[16] P.G. Constantine, Active Subspaces: Emerging Ideas for Dimension Reduction in Parameter Studies, SIAM, Philadelphia, 2015. http://dx.doi.org/10.1137/1.9781611973860.
[17] J.H. Friedman, W. Stuetzle, Projection pursuit regression, J. Amer. Statist. Assoc. 76 (376) (1981) 817–823. http://www.jstor.org/stable/2287576.
[18] S. Weisberg, Applied Linear Regression, third ed., John Wiley & Sons, Inc., Hoboken, 2005. http://dx.doi.org/10.1002/0471704091.
[19] P. Diaconis, M. Shahshahani, On nonlinear functions of linear combinations, SIAM J. Sci. Stat. Comput. 5 (1) (1984) 175–191. http://dx.doi.org/10.1137/0905013.
[20] P.J. Huber, Projection pursuit, Ann. Statist. 13 (2) (1985) 435–475. http://www.jstor.org/stable/2241175.
[21] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning, second ed., Springer, New York, 2009.
[22] C.E. Rasmussen, C.K.I. Williams, Gaussian Processes for Machine Learning, The MIT Press, Boston, 2006. http://www.gaussianprocess.org/gpml/.
[23] F. Vivarelli, C.K.I. Williams, Discovering hidden features with Gaussian processes regression, in: M.S. Kearns, S.A. Solla, D.A. Cohn (Eds.), Advances in Neural Information Processing Systems, Vol. 11, MIT Press, Cambridge, 1999.
[24] R. Tripathy, I. Bilionis, M. Gonzalez, Gaussian processes with built-in dimensionality reduction: Applications to high-dimensional uncertainty propagation, J. Comput. Phys. 321 (2016) 191–223. http://dx.doi.org/10.1016/j.jcp.2016.05.039.
[25] M. Fornasier, K. Schnass, J. Vybiral, Learning functions of few arbitrary linear parameters in high dimensions, Found. Comput. Math. 12 (2012) 229–262. http://dx.doi.org/10.1007/s10208-012-9115-y.
[26] A. Cohen, I. Daubechies, R. DeVore, G. Kerkyacharian, D. Picard, Capturing ridge functions in high dimensions from point queries, Constr. Approx. 35 (2) (2012) 225–243. http://dx.doi.org/10.1007/s00365-011-9147-6.
[27] H. Tyagi, V. Cevher, Learning non-parametric basis independent models from point queries via low-rank methods, Appl. Comput. Harmon. Anal. 37 (3) (2014) 389–412. http://dx.doi.org/10.1016/j.acha.2014.01.002.
[28] A. Edelman, T.A. Arias, S.T. Smith, The geometry of algorithms with orthogonality constraints, SIAM J. Matrix Anal. Appl. 20 (2) (1998) 303–353. http://dx.doi.org/10.1137/S0895479895290954.
[29] J. Lee, M. Simchowitz, M. Jordan, B. Recht, Gradient descent converges to minimizers, in: 29th Annual Conference on Learning Theory, 2016, pp. 1246–1257. http://www.jmlr.org/proceedings/papers/v49/lee16.html.
[30] A.M. Samarov, Exploring regression structure using nonparametric functional estimation, J. Amer. Statist. Assoc. 88 (423) (1993) 836–847. http://dx.doi.org/10.1080/01621459.1993.10476348.
[31] P. Constantine, E. Dow, Q. Wang, Active subspace methods in theory and practice: Applications to kriging surfaces, SIAM J. Sci. Comput. 36 (4) (2014) A1500–A1524. http://dx.doi.org/10.1137/130916138.
[32] D.E. Edmunds, B. Opic, Weighted Poincaré and Friedrichs inequalities, J. Lond. Math. Soc. s2-47 (1) (1993) 79–96. http://dx.doi.org/10.1112/jlms/s2-47.1.79.
[33] M. Bebendorf, A note on the Poincaré inequality for convex domains, Z. Anal. Anwend. (22) (2003) 751–756.
[34] L.H. Chen, An inequality for the multivariate normal distribution, J. Multivariate Anal. 12 (2) (1982) 306–315. http://dx.doi.org/10.1016/0047-259X(82)90022-7.
[35] A. Griewank, Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation, SIAM, Philadelphia, 2000. http://dx.doi.org/10.1137/1.9780898717761.
[36] P. Constantine, D. Gleich, Computing active subspaces with Monte Carlo, 2015, ArXiv:1408.0545v2, https://arxiv.org/abs/1408.0545.
[37] A. Ruhe, P. Åke Wedin, Algorithms for separable nonlinear least squares problems, SIAM Rev. 22 (3) (1980) 318–337. http://dx.doi.org/10.1137/1022057.
[38] C.F. Dunkl, Y. Xu, Orthogonal Polynomials of Several Variables, second ed., Cambridge University Press, Cambridge, 2014. http://dx.doi.org/10.1017/CBO9781107786134.
[39] J. Nocedal, S.J. Wright, Numerical Optimization, second ed., Springer, New York, 2006.
[40] J. Townsend, N. Koep, S. Weichwald, Pymanopt: a python toolbox for optimization on manifolds using automatic differentiation, J. Mach. Learn. Res. 17 (137) (2016) 1–5. http://jmlr.org/papers/v17/16-177.html.
[41] T.D. Economon, F. Palacios, S.R. Copeland, T.W. Lukaczyk, J.J. Alonso, SU2: An open-source suite for multiphysics simulation and design, AIAA J. 54 (3) (2016) 828–846. http://dx.doi.org/10.2514/1.J053813.
[42] M.D. McKay, R.J. Beckman, W.J. Conover, Comparison of three methods for selecting values of input variables in the analysis of output from a computer code, Technometrics 21 (2) (1979) 239–245. http://dx.doi.org/10.1080/00401706.1979.10489755.
[43] P.Z.G. Qian, Nested Latin hypercube designs, Biometrika 96 (4) (2009) 957–970. http://www.jstor.org/stable/27798879.
[44] G.H. Golub, C.F.V. Loan, Matrix Computations, fourth ed., The Johns Hopkins University Press, 2013.

[45] B. Efron, R.J. Tibshirani, An Introduction to the Bootstrap, Chapman & Hall/CRC, Boca Raton, 1994.
[46] G.W. Stewart, Error and perturbation bounds for subspaces associated with certain eigenvalue problems, SIAM Rev. 15 (4) (1973) 727–764. http://dx.doi.org/10.1137/1015095.
[47] R.D. Cook, Regression Graphics: Ideas for Studying Regressions through Graphics, John Wiley & Sons, Inc., Hoboken, 1998. http://dx.doi.org/10.1002/9780470316931.
[48] P.G. Constantine, M. Emory, J. Larsson, G. Iaccarino, Exploiting active subspaces to quantify uncertainty in the numerical simulation of the HyShot II scramjet, J. Comput. Phys. 302 (2015) 1–20. http://dx.doi.org/10.1016/j.jcp.2015.09.001.