THE UNIVERSITY of EDINBURGH

Edinburgh Research Explorer

A near-stationary subspace for ridge approximation

OPEN ACCESS

A near-stationary subspace for ridge approximation

Paul G. Constantine[a], Armin Eftekhari[b], Rachel A. Ward[b]

[a]*Department of Applied Mathematics and Statistics, Colorado School of Mines, Golden, CO 80211*
[b]*Department of Mathematics and Institute for Computational Engineering and Sciences, University of Texas at Austin, Austin, TX 78712*

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 many models. We propose *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 develop essential theory for ridge approximation—e.g., the best mean-squared approximation and an optimal low-dimensional subspace—and we show that the gradient-based active subspace is near-stationary for the least-squares problem that defines an optimal subspace. We propose practical computational heuristics motivated by the theory including an alternating minimization heuristic that estimates an optimal ridge approximation. We show a simple example where the heuristic fails, which reveals a type of function for which the proposed approach is inappropriate. And we demonstrate a successful example with an airfoil model of drag as a function of its 18 shape parameters.

Keywords: active subspaces, ridge functions, projection pursuit regression

Email address: `paul.constantine@mines.edu` (Paul G. Constantine)

1. Introduction

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

When the number of input parameters is more than a handful, the cost of constructing an accurate response surface increases exponentially as the dimension of the input space increases; in approximation, this is the tractability problem [5], though it is more colloquially referred to as the *curse of dimensionality* [6, Section 5.16]. Several techniques attempt to alleviate this curse—each with advantages and drawbacks for certain classes of problems; see [7] for an extensive survey. One such idea is to approximate the input/output map by a ridge function, which is a function of a few linear combinations of the inputs. If one can identify the few most important linear combinations of inputs for a given model, then she may fit a response surface of only those linear combinations, which allows a higher degree of accuracy along important directions in the model's input space.

We define a *ridge function* to be a function of 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* [8, Chapter 1], though Keiper uses the qualifier *generalized* for a model where $\boldsymbol{U}$ depends on $\mathbf{x}$ [9]. If $\boldsymbol{U}$ is given, 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 weight function $\rho : \mathbb{R}^m \rightarrow \mathbb{R}_+$. The ridge approximation problem may 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. We show that, under certain conditions, this $\boldsymbol{U}$ 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 computational heuristic for fitting a ridge approximation given pairs $\{(\mathbf{x}_i, f(\mathbf{x}_i))\}$. In Section 4, we show (i) a simple bivariate example that exposes the limitations of the heuristic and (ii) an 18-dimensional example from an airfoil shape optimization problem where the heuristic succeeds.

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 [10] proposed *projection pursuit regression* with a ridge function model:

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

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. 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 $\{g_k\}$. Chapter 11 of Hastie, Tibshirani, and Friedman [11] links projection pursuit regression to neural networks, which use 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

3

from the approximation context, since there is no inherent randomness in the approximation problem.

1.1.2. Gaussian processes with low-rank covariance models

In Gaussian process regression, the conditional mean of the Gaussian process model given data (e.g., $\{y_i\}$ as in (1)) 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 covariance. Vivarelli and Williams [12] proposed a covariance model of the form

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

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. [13], use a similar approach from a Bayesian perspective in the context 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 [14, 15, 16]. 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 [14]) 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 $\rho : \mathbb{R}^m \to \mathbb{R}_+$,

$$\int f(\mathbf{x})^2 \, \rho(\mathbf{x}) \, d\mathbf{x} \ < \ \infty, \tag{3}$$

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,

$$\left\| f(\mathbf{x}) - g(\boldsymbol{U}^T \mathbf{x}) \right\|_{L^2(\rho)} = \left(\int (f(\mathbf{x}) - g(\boldsymbol{U}^T \mathbf{x}))^2 \, \rho(\mathbf{x}) \, d\mathbf{x} \right)^{\frac{1}{2}}. \tag{4}$$

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 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 $\text{span}(\boldsymbol{U})$ in $\mathbb{R}^m$, where $\text{span}(\boldsymbol{U})$ 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 $\text{span}(\boldsymbol{U})$ and $\text{span}(\boldsymbol{V})$ as follows. For $\mathbf{y} \in \mathbb{R}^n$ and $\mathbf{z} \in \mathbb{R}^{m-n}$, define the following:

$$\begin{aligned} \pi(\mathbf{y}, \mathbf{z}) &= \rho(\boldsymbol{U}\mathbf{y} + \boldsymbol{V}\mathbf{z}) &&\text{(joint density)} \\ \pi(\mathbf{y}) &= \int \pi(\mathbf{y}, \mathbf{z}) \, d\mathbf{z} &&\text{(marginal density)} \\ \pi(\mathbf{z}|\mathbf{y}) &= \pi(\mathbf{y}, \mathbf{z})/\pi(\mathbf{y}) &&\text{(conditional density)} \end{aligned} \tag{5}$$

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

$$\mu = \mu(\mathbf{y}, \boldsymbol{U}) = \int f(\boldsymbol{U}\mathbf{y} + \boldsymbol{V}\mathbf{z}) \, \pi(\mathbf{z}|\mathbf{y}) \, d\mathbf{z}. \tag{6}$$

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

$$\int (\mu(\mathbf{y}) - f(\boldsymbol{U}\mathbf{y} + \boldsymbol{V}\mathbf{z})) \, \pi(\mathbf{z}|\mathbf{y}) \, d\mathbf{z} = 0, \tag{7}$$

for all $\mathbf{y}$ such that $\pi(\mathbf{y}) > 0$. As a consequence of Pinkus's Theorem 8.3 [8], for fixed $\boldsymbol{U}$, (7) 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.

5

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

$$
\begin{aligned}
\pi(\mathbf{z}|\mathbf{y} = \boldsymbol{Q}^T \boldsymbol{U}^T \mathbf{x}) &= \frac{\rho(\boldsymbol{U}\boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{U}^T\mathbf{x} + \boldsymbol{V}\mathbf{z})}{\int \rho(\boldsymbol{U}\boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{U}^T\mathbf{x} + \boldsymbol{V}\mathbf{z})\, d\mathbf{z}} \\
&= \frac{\rho(\boldsymbol{U}\boldsymbol{U}^T\mathbf{x} + \boldsymbol{V}\mathbf{z})}{\int \rho(\boldsymbol{U}\boldsymbol{U}^T\mathbf{x} + \boldsymbol{V}\mathbf{z})\, d\mathbf{z}} \\
&= \pi(\mathbf{z}|\mathbf{y} = \boldsymbol{U}^T\mathbf{x}).
\end{aligned}
\tag{8}
$$

Next examine the definition of μ,

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

This implies that μ only depends on the subspace $\mathrm{span}(\boldsymbol{U})$ 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 $\mathrm{span}(\boldsymbol{U})$ in $\mathbb{R}^m$.

To characterize the optimal $\boldsymbol{U}$, we derive a differentiable cost function from (4). Define $R = R(\boldsymbol{U})$ as

$$
R(\boldsymbol{U}) \;=\; \frac{1}{2} \left\| f(\mathbf{x}) - \mu(\boldsymbol{U}^T\mathbf{x}, \boldsymbol{U}) \right\|_{L^2(\rho)}^2.
\tag{10}
$$

Note that, similar to μ, R only depends on $\mathrm{span}(\boldsymbol{U})$ 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 program:

$$
\begin{aligned}
\underset{\boldsymbol{U}}{\text{minimize}} \quad & R(\boldsymbol{U}), \\
\text{subject to} \quad & \boldsymbol{U} \in \mathbb{G}(n, m).
\end{aligned}
\tag{11}
$$

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 $\boldsymbol{U}_*$.

6

It is convenient to reformulate the optimization (11) in terms of the complement subspace span($\boldsymbol{V}$). The conditional average μ in (6) is the average of $f(\mathbf{x})$ over the affine subspace $S(\mathbf{x})$ defined as

$$S(\mathbf{x}) \;=\; \{\, \mathbf{x}' \in \mathbb{R}^m \mid \mathbf{x}' = \boldsymbol{U}\boldsymbol{U}^T\mathbf{x} + \boldsymbol{V}\mathbf{z}, \, \mathbf{z} \in \mathbb{R}^{m-n} \,\}. \tag{12}$$

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

$$\boldsymbol{U}\boldsymbol{U}^T\mathbf{x} \;=\; (\boldsymbol{I} - \boldsymbol{V}\boldsymbol{V}^T)\,\mathbf{x}, \tag{13}$$

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 (6) as

$$\mu \;=\; \mu(\mathbf{x}, \boldsymbol{V}) \;=\; \int f\left((\boldsymbol{I} - \boldsymbol{V}\boldsymbol{V}^T)\mathbf{x} + \boldsymbol{V}\mathbf{z}\right) \, \pi(\mathbf{z}|\mathbf{y}) \, d\mathbf{z}. \tag{14}$$

Similarly, we rewrite R from (10) as

$$R(\boldsymbol{V}) \;=\; \frac{1}{2} \, \|f(\mathbf{x}) - \mu(\mathbf{x}, \boldsymbol{V})\|^2_{L^2(\rho)}. \tag{15}$$

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

$$\begin{aligned}
\underset{\boldsymbol{V}}{\text{minimize}} \quad & R(\boldsymbol{V}), \\
\text{subject to} \quad & \boldsymbol{V} \in \mathbb{G}(m-n, m).
\end{aligned} \tag{16}$$

An optimal $\boldsymbol{U}_*$ that solves (11) 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. [17, 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

$$\bar{\nabla} R(\boldsymbol{V}) \;=\; \frac{\partial}{\partial \boldsymbol{V}} R(\boldsymbol{V}) - \boldsymbol{V}\boldsymbol{V}^T \frac{\partial}{\partial \boldsymbol{V}} R(\boldsymbol{V}) \;=\; \boldsymbol{U}\boldsymbol{U}^T \frac{\partial}{\partial \boldsymbol{V}} R(\boldsymbol{V}), \tag{17}$$

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

$$\left(\frac{\partial}{\partial \boldsymbol{V}} R\right)_{ij} \;=\; \frac{\partial R}{\partial v_{ij}}, \qquad i = 1, \ldots, m, \quad j = 1, \ldots, m-n, \tag{18}$$

7

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 (16) 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*

$$\left\| \bar{\nabla} R(\boldsymbol{V}_*) \right\|_F \ \leq \ A, \tag{19}$$

where $\| \cdot \|_F$ is the Frobenius norm, and $\bar{\nabla} R$ is the gradient on the Grassmann manifold of R from (16).

In what follows, we show that one particular subspace built from f's derivatives is near-stationary. The subspace is the eigenspace of a particular matrix, and the bound A from Definition 1 is related to the matrix's eigenvalues. In statistical regression, Samarov [19] 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 known, 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 $f(\mathbf{x})$.

---

[1] Recent work suggests that the probability of termininating at a stationary point that is not a local minimizer is zero [18].

Assumption 1. *Given the probability density $\rho : \mathbb{R}^m \to \mathbb{R}_+$, assume $f \in L^2(\rho)$ is differentiable and its partial derivatives are square-integrable with respect to ρ,*

$$\int \left(\frac{\partial f}{\partial x_i}(\mathbf{x}) \right)^2 \rho(\mathbf{x})\,d\mathbf{x} \; < \; \infty. \tag{20}$$

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

$$\boldsymbol{C} \;=\; \int \nabla f(\mathbf{x})\,\nabla f(\mathbf{x})^T \rho(\mathbf{x})\,d\mathbf{x} \;=\; \boldsymbol{W}\Lambda\boldsymbol{W}^T, \tag{21}$$

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

$$\Lambda = \begin{bmatrix} \Lambda_1 & \\ & \Lambda_2 \end{bmatrix}, \qquad \boldsymbol{W} = \begin{bmatrix} \boldsymbol{W}_1 & \boldsymbol{W}_2 \end{bmatrix}, \tag{22}$$

where Λ_1 contains the first n eigenvalues, and $\boldsymbol{W}_1$ contains the first n eigenvectors. The gradient-based subspace—also called the *active subspace* [20]—is the span of the columns of $\boldsymbol{W}_1$. The eigenvalues reveal if 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 $n < m$. The eigenvalues $\lambda_{n+1}, \ldots, \lambda_m$ are zero if and only if $f(\mathbf{x})$ is constant along* span$(\boldsymbol{W}_2)$.

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 [21], bounds its $L^2(\rho)$ approximation error.

Theorem 2. *For f that satisfies Assumption 1, define μ as in* (6). *Then*

$$\left\| f(\mathbf{x}) - \mu(\boldsymbol{W}_1^T\mathbf{x}, \boldsymbol{W}_1) \right\|_{L^2(\rho)} \;\leq\; C\,(\lambda_{n+1} + \cdots + \lambda_m)^{\frac{1}{2}}, \tag{23}$$

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 [21] and Section 4.2 of [20]. Edmunds and Opic [22] provide details of the Poincaré constant with weight function ρ. Note that if we write μ as $\mu(\mathbf{x}, \boldsymbol{V})$, as in (14), then the 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 (16),

$$\left\| \bar{\nabla} R(\boldsymbol{W}_2) \right\|_F \;\leq\; L \left(2m^{\frac{1}{2}} + (m-n)^{\frac{1}{2}} \right) (\lambda_{n+1} + \cdots + \lambda_m)^{\frac{1}{2}}, \qquad (24)$$

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 (11). We expect that the Gaussian assumption on on ρ can be relaxed at the cost of a more complicated bound in (24) 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 [23]), consider the steps in Algorithm 1.

Our previous work studies a Monte Carlo method for estimating $\boldsymbol{C}$ and its eigendecomposition [24] 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

Algorithm 1 Exploiting the active subspace for ridge approximation

1. Estimate the matrix $\boldsymbol{C}$ from (21) 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 (11).

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.

4.1. An 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 second eigenvector—though a stationary point—is far from the global minimizer of the objective function. 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

$$f(x_1, x_2) = 5x_1 + \sin(10\pi x_2). \tag{25}$$

This function has a Lipschitz constant L that is bounded by 32. The matrix $\boldsymbol{C}$ from (21) is (to 4 significant digits)

$$\boldsymbol{C} = \begin{bmatrix} 25.00 & 0 \\ 0 & 526.4 \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}}_{\boldsymbol{W}} \underbrace{\begin{bmatrix} 526.4 & 0 \\ 0 & 25.00 \end{bmatrix}}_{\Lambda} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}^T. \tag{26}$$

We estimate $\boldsymbol{C}$ with a tensor product Gauss-Hermite quadrature rule with 101 points per dimension (10201 total points), which was sufficient for four digits of accuracy.

11

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. Figure 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 (90601 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.
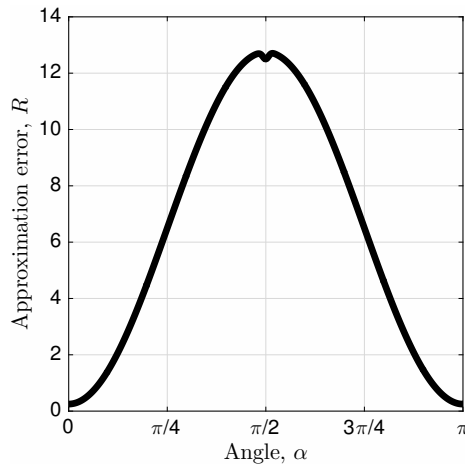

Figure 1: The $L^2(\rho)$ error (10) as a function of subspace angle α for the ridge approximation of $f(x_1, x_2) = 5x_1 + \sin(10\pi x_2)$. The active subspace is span$([0, 1]^T)$—corresponding to $\alpha = \pi/2$—which is a poor initial guess for a gradient-based optimizer.

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 approxi-

12

mation is more accurate, in the mean-squared sense, along another direction.

4.2. An example where the heuristic succeeds

We offer a computational heuristic for estimating the minimizer of (11) based on a polynomial model and alternating minimization; this corresponds to step 3 of Algorithm 1. The polynomial model plays the role of μ in the approximation error (10). 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 $\binom{N+n}{n}$. This model is more general than the projection pursuit regression model (1), since it includes

products of powers of the n linear combinations.

Algorithm 2 Polynomial-based alternating minimization scheme for (11)

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,

$$\underset{\theta}{\text{minimize}} \quad \sum_{i=1}^M \left(f_i - p_N(\mathbf{y}_i, \theta) \right)^2. \tag{27}$$

3. Compute $\boldsymbol{U}_*$ as the solution to the Grassmann manifold-constrained least-squares problem,

$$\begin{aligned} \underset{\boldsymbol{U}}{\text{minimize}} \quad & \sum_{i=1}^M \left(f_i - p_N(\boldsymbol{U}^T \mathbf{x}_i, \theta_*) \right)^2, \\ \text{subject to} \quad & \boldsymbol{U} \in \mathbb{G}(n, m). \end{aligned} \tag{28}$$

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

13

schemes are known to stall and/or converge very slowly relative to gradient-based approaches using all variables [25, 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 $P = 10$, 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 (28) is much easier to implement than R from (11) or (16), since $p_N(\mathbf{y}, \theta_*)$ is independent of $\boldsymbol{U}$—unlike $\mu = \mu(\mathbf{y}, \boldsymbol{U})$ from (6). 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 [26] to solve the Grassmann manifold-constrained least-squares problem in (28) with the a 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 (27), 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 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 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 [20, 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. Given the airfoil geometry,

the drag coefficient is computed with the Stanford University Unstructured

225 compressible Euler solver [27]. This software also solves the continuous adjoint equation for the Euler equations, which enables the computation of the gradient of the drag coefficient as a function of 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 $\mathbf{x}$.

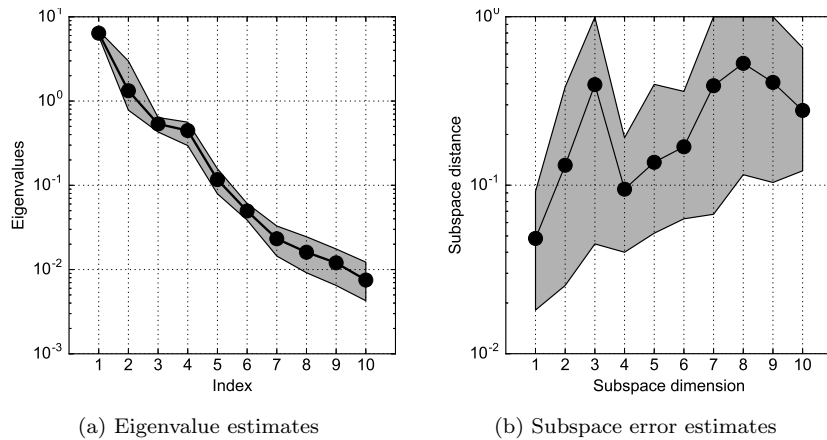(a) Eigenvalue estimates   (b) Subspace error estimates

Figure 2: Eigenvalue estimates and subspace error estimates for a Monte Carlo approximation of $\boldsymbol{C}$ from (21) using 1000 independent samples. The gray regions are uncertainty estimates from a nonparametric bootstrap; more details can be found in [24].

230 We estimate $\boldsymbol{C}$ from (21) with Monte Carlo using 1000 independent samples $\mathbf{x}_i$ drawn uniformly from 18-dimensional hypercube. Figure 2a plots the first 10 of 18 eigenvalue estimates along with uncertainty bars estimated with a nonparametric bootstrap. Observe the rapid decay in the eigenvalues, which suggests that subspaces derived from $\boldsymbol{C}$'s estimated eigenvectors may be a good

235 starting point for an optimization routine to fit a ridge approximation. Figure 2b shows bootstrap-based estimates of the subspace error—i.e., the error in estimating the subspaces with Monte Carlo—as the subspace dimension increases. For details of the computations for Figures 2a and 2b, see [24].

15

The 1000 simulations that produce gradient evaluations used to estimate C also yield the $M = 1000$ pairs $(\mathbf{x}_i, f_i)$, with $i = 1, \ldots, 1000$, used to fit the ridge approximation with the alternating minimization heuristic. Figure 4 compares the number of terms in the polynomial approximation, for increasing degree N, to the number of available runs (1000) for different numbers (n) of linear combinations; the horizontal black line shows $M = 1000$. The topmost line (cyan) shows how many terms are in a polynomial in $m = 18$ variables. Notice that in 18 variables, one cannot fit a polynomial of degree greater than $N = 2$ with the available 1000 runs.

Recall the ridge approximation is most beneficial with a good initial guess for the subspace. Figure 3 shows the results of an experiment comparing different starting points 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 C. Each subfigure in Figure 3 shows the value of the residual (28) 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. The black connected dots show the results using the identity matrix starting point; the blue dashed lines show results from 10 different random starting points; and the red connected dots show results using the first n eigenvectors of C. 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 C provide a superior starting point for the alternating heuristic, and the advantage increases as N and n increase.

The experiment represented by Figure 3 was run on two nodes of the Colorado School of Mines Mio cluster. Each experiment used 8 cores, which accelerated the numpy operations. Figure 5 shows average wall clock times for the experiments; note that the codes were not optimized for performance. The averages are over 10 repeated trials to reduce the effect of the operating system abnormalities. For the random starting points, the averages are also over the 10 independently generated starting points. Each group of bars shows the times

for the different values of n (the number of linear combinations). The labels on the horizontal axis are RN for random starting points, ID for identity matrix starting point, and AS for active subspace starting point—i.e., the first n eigenvectors of $\boldsymbol{C}$. The different subfigures show the times for different polynomial degrees (N). Ten iterations of the alternating scheme was much faster using the n eigenvectors of $\boldsymbol{C}$. This was due to the Grassmann-constrained optimization converging much faster at each iteration of Algorithm 2. Note that the case of $n = 1$ completed in less than 1 second for all starting points and all polynomial degrees, so its bars do not appear on the bar charts.

The results of this experiment show that the first n eigenvectors of $\boldsymbol{C}$ from (21) provide an excellent 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 with a fixed budget of $M = 1000$ model runs.

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 show 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 to fit 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

17

consistently along another. Our second numerical example shows this heuristic succeed with a polynomial-based alternating scheme to fit a ridge approximation applied to an aerospace design model with 18 parameters and a fixed budget of 1000 simulations. 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.

Appendix   A. Proof of Theorem 1

Let C, W_1, W_2, Λ_1, and Λ_2 be defined as in Section 3. Note that

$$
\begin{aligned}
\int \left\| W_2^T \nabla f(\mathbf{x}) \right\|^2 \rho(\mathbf{x}) \, d\mathbf{x} &= \int \nabla f(\mathbf{x})^T W_2 W_2^T \nabla f(\mathbf{x}) \, \rho(\mathbf{x}) \, d\mathbf{x} \\
&= \int \mathrm{tr} \left(W_2^T \nabla f(\mathbf{x}) \nabla f(\mathbf{x})^T W_2 \right) \rho(\mathbf{x}) \, d\mathbf{x} \\
&= \mathrm{tr} \left(W_2^T \left(\int \nabla f(\mathbf{x}) \nabla f(\mathbf{x})^T \rho(\mathbf{x}) \, d\mathbf{x} \right) W_2 \right) \quad \text{(A.1)} \\
&= \mathrm{tr} \left(W_2^T C W_2 \right) \\
&= \mathrm{tr} \left(\Lambda_2 \right) \\
&= \lambda_{n+1} + \cdots + \lambda_m.
\end{aligned}
$$

Assume that $f(\mathbf{x})$ is constant along directions corresponding to the columns of W_2. Then the directional derivatives $W_2^T \nabla f(\mathbf{x})$ are zero for all $\mathbf{x}$. Then by (A.1), $\lambda_{n+1} = \cdots = \lambda_m = 0$.

Next, assume $\lambda_{n+1} = \cdots = \lambda_m = 0$. Since the integrand in (A.1) is a norm and $f(\mathbf{x})$ is differentiable, $W_2^T \nabla f(\mathbf{x})$ is zero for all $\mathbf{x}$. Therefore, $f(\mathbf{x})$ is constant along directions corresponding to the columns of W_2.

18

Appendix B. Proof of Theorem 3

For $R = R(\boldsymbol{V})$ from (16), 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

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

Then

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

Recall Edelman's formula for the Grassmann gradient [17, 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 $C = 1$ [28]. Then combining (B.16) with Theorem 2 achieves the desired result.

Acknowledgment

21

References

₃₂₅ **References**

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

₃₃₀ [2] 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.
URL `http://doi.acm.org/10.1145/167293.167352`

[3] J. Sacks, W. Welch, T. Mitchell, H. Wynn, Design and analysis of computer ₃₃₅ experiments, Statistical Science 4 (4) (1989) 409–423.
URL `http://www.jstor.org/stable/2245858`

[4] M. Kennedy, A. O'Hagan, Predicting the output from a complex computer code when fast approximations are available, Biometrika 87 (1) (2000) 1–13.
₃₄₀ URL `http://biomet.oxfordjournals.org/content/87/1/1.abstract`

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

[6] R. Bellman, Adaptive Control Processes: A Guided Tour, Princeton University Press, Princeton, 1961.
₃₄₅

[7] S. Shan, G. Wang, Survey of modeling and optimization strategies to solve high-dimensional design problems with computationally-expensive black-box functions, Structural and Multidisciplinary Optimization 41 (2) (2010)

22

219–241.

350   URL http://dx.doi.org/10.1007/s00158-009-0420-2

[8] A. Pinkus, Ridge Functions, Cambridge University Press, Cambridge, 2015.

[9] S. Keiper, Analysis of generalized ridge functions in high dimensions, in: 2015 International Conference onSampling Theory and Applications (SampTA), 2015, pp. 259–263.

355   URL http://dx.doi.org/10.1109/SAMPTA.2015.7148892

[10] J. H. Friedman, W. Stuetzle, Projection pursuit regression, Journal of the American Statistical Association 76 (376) (1981) 817–823.
URL http://www.jstor.org/stable/2287576

[11] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning, 2nd Edition, Springer, New York, 2009.

360

[12] 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.

365 [13] I. Bilionis, R. Tripathy, M. Gonzalez, Gaussian processes with built-in dimensionality reduction: Applications in high-dimensional uncertainty propagation, arXiv:1602.04550v1.

[14] M. Fornasier, K. Schnass, J. Vybiral, Learning functions of few arbitrary linear parameters in high dimensions, Foundations of Computational Mathematics 12 (2012) 229–262.

370   URL http://dx.doi.org/10.1007/s10208-012-9115-y

[15] A. Cohen, I. Daubechies, R. DeVore, G. Kerkyacharian, D. Picard, Capturing ridge functions in high dimensions from point queries, Constructive Approximation 35 (2) (2012) 225–243.

375   URL http://dx.doi.org/10.1007/s00365-011-9147-6

[16] H. Tyagi, V. Cevher, Learning non-parametric basis independent models from point queries via low-rank methods, Applied and Computational Harmonic Analysis 37 (3) (2014) 389–412.
URL http://dx.doi.org/10.1016/j.acha.2014.01.002

[17] A. Edelman, T. A. Arias, S. T. Smith, The geometry of algorithms with orthogonality constraints, SIAM Journal on Matrix Analysis and Applications 20 (2) (1998) 303–353.
URL http://dx.doi.org/10.1137/S0895479895290954

[18] J. Lee, M. Simchowitz, M. Jordan, B. Recht, Gradient descent converges to minimizers, arXiv:1602.04915v2.

[19] A. Samarov, Exploring regression structure using nonparametric functional estimation, Journal of the American Statistical Association 88 (423) (1993) 836–847.
URL http://dx.doi.org/10.1080/01621459.1993.10476348

[20] P. G. Constantine, Active Subspaces: Emerging Ideas for Dimension Reduction in Parameter Studies, SIAM, Philadelphia, 2015.

[21] P. Constantine, E. Dow, Q. Wang, Active subspace methods in theory and practice: Applications to kriging surfaces, SIAM Journal on Scientific Computing 36 (4) (2014) A1500–A1524.
URL http://dx.doi.org/10.1137/130916138

[22] D. E. Edmunds, B. Opic, Weighted Poincaré and Friedrichs inequalities, Journal of the London Mathematical Society s2-47 (1) (1993) 79–96. doi:10.1112/jlms/s2-47.1.79.
URL http://jlms.oxfordjournals.org/content/s2-47/1/79.short

[23] A. Griewank, Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation, SIAM, Philadelphia, 2000.

[24] P. Constantine, D. Gleich, Computing active subspaces with Monte Carlo, arXiv:1408.0545v2.

[25] J. Nocedal, S. J. Wright, Numerical Optimization, 2nd Edition, Springer, New York, 2006.

[26] J. Townsend, N. Koep, S. Weichwald, Pymanopt: A Python toolbox for manifold optimization using automatic differentiation, arXiv:1603.03236.

[27] F. Palacios, J. Alonso, K. Duraisamy, M. Colonno, J. Hicken, A. Aranake, A. Campos, S. Copeland, T. Economon, A. Lonkar, T. Lukaczyk, T. Taylor, Stanford University Unstructured (SU^2): An open-source integrated computational environment for multi-physics simulation and design, 51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition.
URL http://dx.doi.org/10.2514/6.2013-287

[28] L. H. Chen, An inequality for the multivariate normal distribution, Journal of Multivariate Analysis 12 (2) (1982) 306–315.
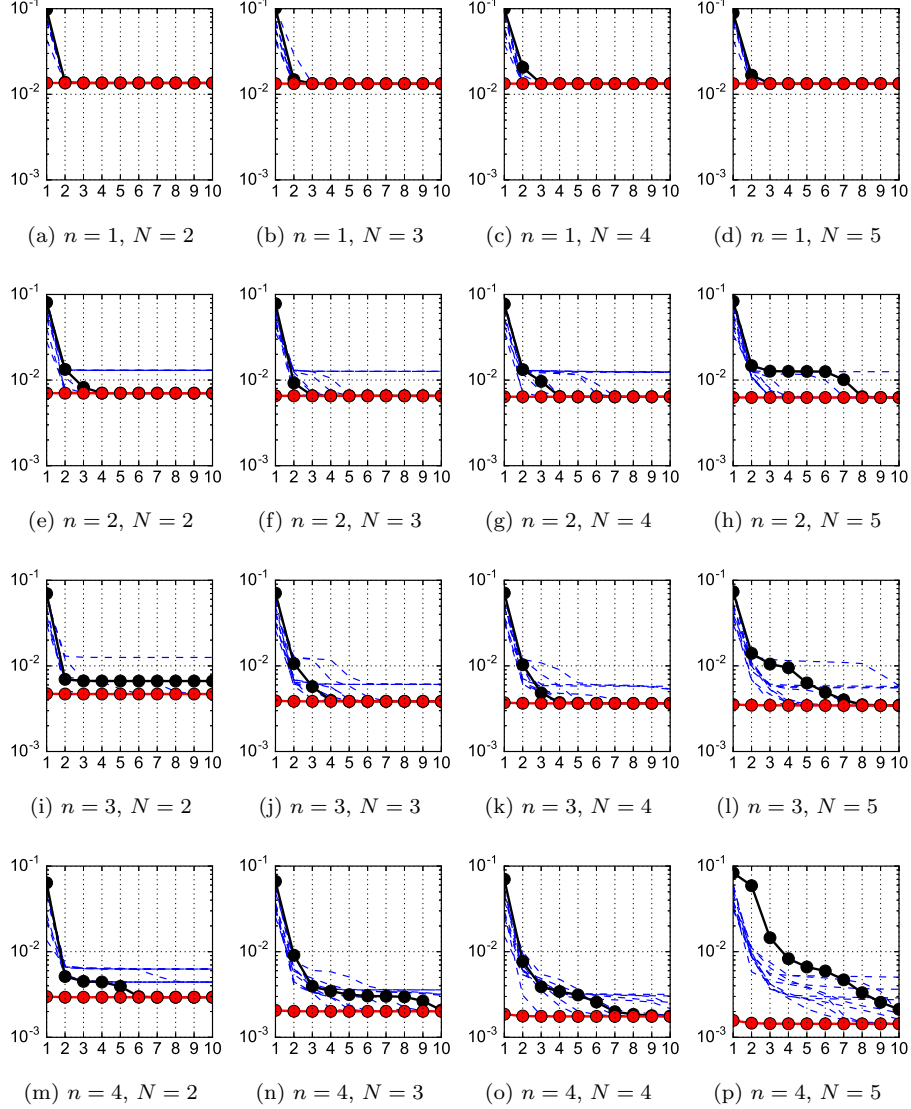URL http://dx.doi.org/10.1016/0047-259X(82)90022-7

25

Figure 3: Each subfigure shows the residual (28) 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 $\boldsymbol{C}$. 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 $\boldsymbol{C}$ provide a superior starting point.
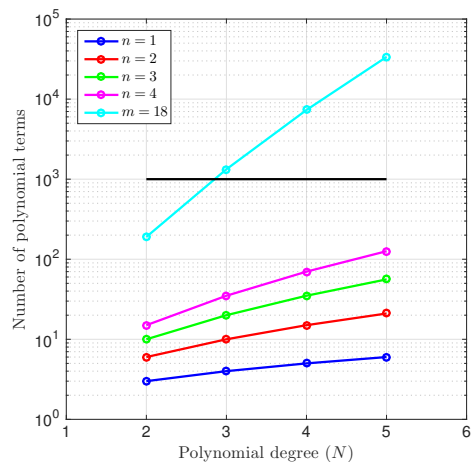
26

Figure 4: Number of terms in the polynomial approximation as a function of degree N for varying number of linear combinations, $n = 1, \ldots, 4$. The black horizontal line shows the budget of $M = 1000$ available simulations. Note that in all $m = 18$ variables, it is not possible to fit a polynomial of degree greater than $N = 2$.

(a) $N = 2$

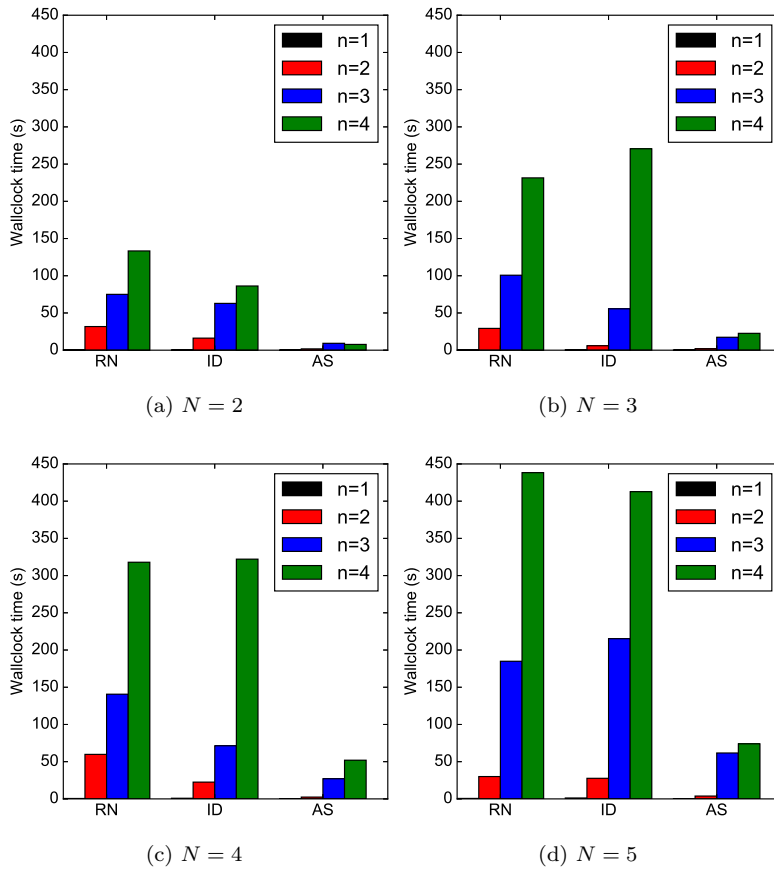(b) $N = 3$

(c) $N = 4$

(d) $N = 5$

Figure 5: Average wall clock times for the experiments from Figure 3; the codes were not optimized for performance. Each group of bars varies the number n of linear combinations. The labels on the horizontal axis are RN for random starting point, ID for n columns of the identity matrix, and AS for the first n eigenvectors of C. Each subfigure varies the degree N of polynomial. In all cases, the first n eigenvectors of C lead to a faster completion of 10 iterations.