A LANCZOS-STIELTJES METHOD FOR ONE-DIMENSIONAL RIDGE FUNCTION APPROXIMATION AND INTEGRATION

ANDREW GLAWS† AND PAUL G. CONSTANTINE‡

Abstract. High-fidelity computational science models are useful research tools but are often very costly to evaluate. Computationally inexpensive surrogates (e.g., polynomial expansions) approximate these models and can enable studies that require repeated sampling, such as optimization or uncertainty quantification. However, the construction of such surrogates suffers from the curse of dimensionality—an exponential increase in computational cost with increased input dimension. To combat this curse, we turn to tools for dimension reduction, such as active subspaces or sufficient dimension reduction, which seek out low-dimensional linear transformations of the input space that describe most or all of the variation in the model output. Functions that exhibit this type of low-dimensional structure are known as ridge functions. The linear transformation of the input space can make approximation and integration of these functions difficult. In this paper, we introduce a method for building polynomial surrogates of one-dimensional ridge functions. We use the Lanczos method as a discrete approximation to the Stieltjes procedure to obtain orthogonal polynomials and Gauss-Christoffel quadrature rules with respect to arbitrary density functions. This enables cheap and accurate approximation or integration of the one-dimensional ridge function. We also provide a heuristic approach to extend the methodology to functions that are well-approximated by one-dimensional ridge functions.

Key words. orthogonal polynomials, Gauss-Christoffel quadrature, dimension reduction

AMS subject classifications. 42C05, 33C50, 41A55

1. Introduction. Computational models that simulate various physical phenomena are important to the progress of scientific and engineering research. We represent these models mathematically by a deterministic function that maps a vector of physical inputs to some outputs of interest. As computational power improves, we can build higher fidelity models that better represent reality and consider more independent variables (i.e., function inputs). Due to this increased complexity, these high-fidelity models can be very expensive to evaluate.

In the statistics community, studies of these computational models are known as computer experiments [35, 25, 33]. These studies can include performing optimization or uncertainty quantification of the underlying model, which requires multiple function evaluations at different input values. In such cases, we may choose to construct a cheap surrogate (also referred to as a response surface, emulator, or metamodel) of the original function [29, 32]. Methods for building such surrogate models include radial basis functions [15], splines [41], and polynomial approximations (referred to as polynomial chaos expansions in the uncertainty quantification literature) [43]. Once we have constructed the surrogate model, we can perform the study on it much more cheaply than on the original function.

Unfortunately, building surrogate approximations suffers from the curse of dimensionality [14, 38, 42]. Roughly speaking, the number of function evaluations required to construct a surrogate of fixed accuracy grows exponentially with the dimension (i.e., the number of inputs to the function). To combat this curse, we can look for

*Submitted to the editors June 15, 2018.

Funding: This work was funded by the DARPA DSO Enabling Quantification of Uncertainty in Physical Systems program.

†Department of Computer Science, University of Colorado Boulder, (andrew.glaws@colorado.edu).

‡Department of Computer Science, University of Colorado Boulder, (paul.constantine@colorado.edu).
anisotropy in the function, which can enable input space dimension reduction. Sensitivity analysis determines which inputs are the most/least important in terms of changing the function output \([34]\). Inputs that have relatively little impact on the output can then be fixed at a nominal value. This reduces the dimension of the function by exploiting coordinate-aligned anisotropy. Ridge functions generalize this concept by finding directions (i.e., linear combinations of inputs) that are most/least important in terms of changing the function output \([31]\). Methods for finding such directions include active subspaces \([3]\), sufficient dimension reduction \([10, 18]\), among others \([5, 23, 2, 40, 37]\).

In this work, we focus on the case when a single direction captures all (or most) of the variation in the output. In this case, the function depends (mostly) on a single input defined by a specific linear combination of the original inputs. Such an assumption may appear restrictive; however, exploitable near-one-dimensional structure has been identified in models for lithium ion batteries \([4]\), magnetohydrodynamical generators \([19]\), car aerodynamics \([30]\), integrated hydrologic models \([24]\), hypersonic scramjet designs \([7]\), among others \([9, 17, 12]\). Given a candidate vector, we can check for this structure using shadow plots (also called sufficient summary plots in the context of statistical regression) \([10]\). These are scatter plots of function evaluations against the linearly-transformed input. Figure 1 contains one-dimensional shadow plots for input/output maps derived from a variety of computational models. Each shadow plot shows function evaluations at a sample of multidimensional input values plotted against a single linear combination of the original inputs. The strong trends in these figures suggest that the functions exhibit strong one-dimensional ridge structure.

We assume that the one important direction (i.e., the coefficients of the linear combination) has been computed by some method using a set of exploratory pilot runs, and we want to fit a polynomial surrogate as a function of the derived variable. By exploiting known one-dimensional structure, we can build an approximation using exponentially fewer function evaluations—which are assumed to be very expensive—than would be necessary for a comparable polynomial approximation on the full-dimensional input space. Figure 2 illustrates this point. This figure shows \(L^2\) approximation errors for a five-dimensional function that is a one-dimensional ridge function. In terms of the total polynomial degree, there is no difference in performance between building the polynomial surrogate on the full five-dimensional input space versus the one-dimensional reduced input space. However, in terms of the number of function evaluations required to build each approximation, exploiting the one-dimension structure in the function results in an exponential savings in the computational costs.

Current approaches for exploiting this structure use Gaussian processes on the one-dimensional subspace (referred to as a single-index model) \([22]\) or transform the reduced inputs through an empirical CDF to enable the use of Legendre polynomials \([39]\). Our approach first approximates the induced density function on the reduced input space using convolutions. We then use the Lanczos iterative method as a discrete approximation of the Stieltjes procedure for constructing orthonormal polynomials and Gauss-Christoffel quadrature rules with respect to arbitrary densities. This allows us to accurately fit polynomial surrogates on the one-dimensional ridge subspace.

Throughout this paper, we use the following notation unless stated otherwise. Matrices and vectors will be represented by bold uppercase and lowercase letters, respectively. Unbolded versions of these letters with subscripts denote specific entries of the given matrix or vector. Also, we frequently relate matrix and vector elements
Fig. 1: One-dimensional shadow plots of data derived from computational models across applications. The plots verify the near-one-dimensional structure in the input/output map.

Fig. 2: A comparison of the cost and accuracy of the five-dimensional (in red) and the one-dimensional (in blue) polynomial approximations for a given function. They perform equally well in terms of polynomial degree, but the one-dimensional approximation significantly outperforms the five-dimensional approximation.

to polynomials of increasing degree. For example, the first column of a matrix or entry in a vector will often be connected to a zero-degree polynomial. For this reason, we use zero-based indexing throughout this paper.

The remainder of this paper is structured as follows. Section 2 contains important
background information, including discussions on ridge functions, Fourier expansions in terms of orthogonal polynomials, and the Lanczos and Stieltjes methods. In Section 3, we introduce the new Lanczos-Stieltjes method for one-dimensional ridge approximation and integration. Section 4 provides a heuristic extension of the method for approximate one-dimensional ridge functions, such as those shown in Figure 1.

2. Background. Consider a scalar-valued functions of $m$ inputs,

\begin{equation}
(2.1) \quad y = f(x), \quad y \in \mathbb{R}, \quad x \in \mathbb{R}^m,
\end{equation}

and assume the input space is weighted by a given probability density function $p(x)$, which describes uncertainty in the model inputs. For simplicity, we assume this density is uniform over the $m$-dimension hypercube $[-1,1]^m$ such that

\begin{equation}
(2.2) \quad p(x) = \begin{cases} 
\frac{1}{2^m} & \text{if } ||x||_{\infty} \leq 1, \\
0 & \text{otherwise}.
\end{cases}
\end{equation}

2.1. Ridge functions. We assume that (2.1) is a one-dimensional ridge function. That is,

\begin{equation}
(2.3) \quad y = f(x) = g(a^\top x)
\end{equation}

for $a \in \mathbb{R}^m \setminus \{0\}$ and $g : \mathbb{R} \to \mathbb{R}$. Equation (2.3) is a special case of the generalized ridge function

\begin{equation}
(2.4) \quad y = f(x) = g(A^\top x)
\end{equation}

for $A \in \mathbb{R}^{m \times n} \setminus \{0\}$ and $g : \mathbb{R}^n \to \mathbb{R}$ [31].

In terms of the original $m$ inputs, (2.3) implies that the given function is constant along directions orthogonal $a$. Consider $w \in \mathbb{R}^m$ with $a^\top w = 0$. Then,

\begin{equation}
(2.5) \quad f(x + w) = g(a^\top (x + w)) = g(a^\top x) = f(x).
\end{equation}

By ignoring directions along which the function is constant, we reduce the dimension of $f$.

The vector $a$ in (2.3) is known as the ridge direction and the function $g$ is the ridge profile [28]. We assume without loss of generality that $a$ is normalized to have unit 2-norm (i.e., $||a||_2 = 1$). In this work, we assume that the ridge direction is known, but that the ridge profile is unknown. We seek to exploit the structure in (2.3) to build an approximation of (2.1). In the next section, we discuss such approximations via orthonormal polynomials.

2.2. Polynomial approximation. Assume that $f$ is square-integrable with respect to the input density $p(x)$. Then, we may express $f$ as a Fourier expansion in terms of orthogonal polynomials with respect to $p$,

\begin{equation}
(2.6) \quad y = f(x) = \sum_{|\alpha|=0}^{\infty} f_\alpha \psi_\alpha(x),
\end{equation}

where equality is denoted in the $L^2$ sense [16]. The multivariate orthogonal polynomials $\psi_\alpha(x)$ are indexed by the multi-index $\alpha \in \mathbb{N}_0^m$ which denotes the degree of
the polynomial with respect to each of the components of $x$. Given the assumption $x \sim U([-1, 1]^m)$, we can write these multivariate polynomials as

$$
\psi_\alpha(x) = \prod_{i=1}^{m} \psi_{\alpha_i}(x_i),
$$

where each $\psi_{\alpha_i}$ is the univariate Legendre polynomial of degree $\alpha_i$. Without loss of generality, we also assume that the $\psi_\alpha$ are normalized so that the coefficients in (2.6) are the inner product of $f$ with the appropriate polynomial,

$$
f_\alpha = \int f(x) \psi_\alpha(x) p(x) \, dx.
$$

This method of approximation by orthogonal polynomials also appears in the uncertainty quantification literature under the name polynomial chaos \cite{43}.

In practice, we may choose to compute the pseudospectral expansion of $f(x)$ \cite{6}. We do this by truncating (2.6) to include only polynomials of total degree $d$ or less and approximating the integral in (2.8) numerically,

$$
y = f(x) \approx \sum_{|\alpha| \leq d} \hat{f}_\alpha \psi_\alpha(x), \quad \text{where} \quad f_\alpha \approx \hat{f}_\alpha = \sum_{j=0}^{M-1} \omega_j f(\xi_j) \psi_\alpha(\xi_j),
$$

where $(\xi_j, \omega_j), j = 0, \ldots, M-1$ denote the nodes and weights of an $M$-point numerical integration rule (e.g., tensor product Gauss quadrature) with respect to $p$.

Pseudospectral polynomial approximations can serve as quick-to-evaluate surrogates for the original function. However, as the input dimension $m$ grows, the cost of constructing (2.9) can quickly increase—a total degree $d$ polynomial in $m$ dimensions has $\binom{m+d}{d}$ coefficients. Recall from (2.3) that we assume $f(x)$ is a one-dimensional ridge function. We can exploit this structure to enable pseudospectral polynomial approximation.

To construct an orthogonal polynomial expansion of $g(a^T x)$ similar to (2.6), we must first understand the transformed input space. For convenience, let $u = a^T x$ denote the scalar-valued input of the ridge profile. Recall that the full-dimensional input space is weighted with the input probability density function $p(x)$. The linear transform $a^T x$ induces a new density, which we denote by $q(u)$. Figure 3 shows different rotations and projections defined by different vectors $a$ of the three-dimensional cube $[-1, 1]^3$ and the resulting one-dimensional probability densities $q(u)$.

We address the computation $q(u)$ in Section 3. For now, assume $q(u)$ is known. We write the polynomial expansion of $g(u)$ as

$$
y = g(u) = \sum_{i=0}^{\infty} g_i \phi_i(u), \quad \text{where} \quad g_i = \int g(u) \phi_i(u) q(u) \, du,
$$

where the polynomials $\phi_i$ are orthonormal with respect to $q$. By truncating the expansion and numerically approximating the coefficients, we obtain the pseudospectral approximation of the ridge profile,

$$
y = g(u) \approx \sum_{i=0}^{d} \hat{g}_i \phi_i(u), \quad \text{where} \quad \hat{g}_i = \sum_{j=0}^{M-1} \nu_j g(\lambda_j) \phi_i(\lambda_j),
$$

where $(\lambda_j, \nu_j)$ define a numerical integration rule with respect to $q$. 

Constructing the pseudospectral polynomial expansion of the one-dimensional ridge profile \( q(u) \) significantly reduces the number of function evaluations required compared to working in the \( m \)-dimensional space. However, (2.11) requires knowledge of the orthonormal polynomials \( \phi_i \) and an integration rule \((\lambda_j, \nu_j)\) with respect to \( q \), as well as the ability to evaluate the ridge profile, which is assumed to be unknown, at the \( \lambda_j \)'s. We address this latter issue in Section 3. In the next section, we discuss how we can obtain the orthonormal polynomials and integration nodes/weights using the Stieltjes and Lanczos iterative methods.

2.3. The Stieltjes and Lanczos iterative methods. In this section, we review the Stieltjes and Lanczos iterative methods. We discuss the key components of each and show that, under certain circumstances, the Lanczos method can be viewed as a discrete approximation to the Stieltjes procedure. These methods and the relationships between them have been rigorously studied \([27, 16, 20]\). The discussion in this section is based on these references.

The Stieltjes procedure—given in Algorithm 2.1—is a method for iteratively constructing a sequence of orthonormal polynomials \( \{\phi_0(u), \phi_1(u), \phi_2(u), \ldots\} \) with respect to a given density \([36]\). Step (iv) in Algorithm 2.1 contains the three-term recurrence relationship that must be satisfied by any sequence of orthonormal polynomials.

By truncating the Stieltjes procedure after \( d + 1 \) terms, we can rewrite the three-term recurrence relationship in vector form as

\[
(2.12) \quad u \phi(u) = J \phi(u) + \beta_{d+1} \phi_{d+1}(u)e_{d+1},
\]

where \( \phi(u) = [\phi_0(u), \phi_1(u), \ldots, \phi_d(u)]^T \), \( e_{d+1} \in \mathbb{R}^{d+1} \) is a vector of zeros with a one in the last entry, and the matrix \( J \in \mathbb{R}^{(d+1) \times (d+1)} \)—referred to as the Jacobi
Algorithm 2.1 Stieltjes procedure [16, Section 2.2.3.1]

Given: probability density function \( q(u) \)

Assumptions: \( \phi_{-1}(u) = 0 \) and \( \phi_0(u) = 1 \)

For \( i = 0, 1, 2, \ldots \)

(i) \( \beta_i = \int \phi_i(u)^2 q(u) \, du \)

(ii) \( \phi_i(u) = \tilde{\phi}_i(u) / \beta_i \)

(iii) \( \alpha_i = \int u \phi_i(u)^2 q(u) \, du \)

(iv) \( \tilde{\phi}_{i+1}(u) = (u - \alpha_i) \phi_i(u) - \beta_i \phi_{i-1}(u) \)

Output: the orthonormal polynomials \( \{ \phi_0(u), \phi_1(u), \phi_2(u), \ldots \} \) and recurrence coefficients \( \alpha_i, \beta_i \) for \( i = 0, 1, 2, \ldots \)

matrix—is a symmetric, tridiagonal matrix of recurrence coefficients,

\[
\begin{array}{cccccc}
\alpha_0 & \beta_1 & & & & \\
\beta_1 & \alpha_1 & \beta_2 & & & \\
& \ddots & \ddots & \ddots & & \\
& & \beta_{d-1} & \alpha_{d-1} & \beta_d & \\
& & & \beta_d & \alpha_d & \\
\end{array}
\]

(2.13)

Let \( J = Q \Lambda Q^T \) denote the eigendecomposition of \( J \). From (2.12), the \( d + 1 \) eigenvalues of \( J \) correspond to the zeros of the \( (d+1) \)-degree orthonormal polynomial \( \phi_{d+1} \). The normalized eigenvector associated with the eigenvalue \( \lambda_j, j = 0, 1, \ldots, d \) has the form

\[
(Q)_j = \frac{\phi(\lambda_j)}{\sqrt{\phi(\lambda_j) \phi(\lambda_j)}},
\]

(2.14)

where \((\cdot)_j\) denotes the \( j \)th column of the given matrix. Additionally, the \( (d+1) \)-point Gauss-Christoffel quadrature rule with respect to \( q \) can be obtained from the eigendecomposition of \( J \) [21]. The quadrature nodes are the eigenvalues of \( J \) and the associated quadrature weights are the square of the first entry of the associated normalized eigenvector,

\[
\nu_j = (Q)^2_{0,j} = \frac{1}{\phi(\lambda_j) \phi(\lambda_j)}.
\]

(2.15)

The Lanczos algorithm was introduced as an iterative approach to estimating eigenvalues of large symmetric matrices [26]. Given symmetric \( A \in \mathbb{R}^{N \times N} \), the Lanczos algorithm constructs the system

\[
A V = V T + \beta_{d+1} v_{d+1}^T e_{d+1},
\]

(2.16)

where \( T \in \mathbb{R}^{(d+1) \times (d+1)} \) is a symmetric, tridiagonal matrix of recurrence coefficients, \( V \in \mathbb{R}^{N \times (d+1)} \) contains the Lanczos vectors, and \( e_{d+1} \in \mathbb{R}^{d+1} \) is a vector of zeros with a one in the last entry. The eigenvalues of \( T \) approximate those of \( A \), and \( V \) transforms the eigenvectors of \( T \) into approximate eigenvectors of \( A \). For the purposes of this paper, we consider circumstances under which the Lanczos algorithm serves as a discrete approximation to the Stieltjes procedure described above. Algorithm 2.2
contains the Lanczos algorithm. For notational convenience, we put the outputs of Algorithm 2.2 into matrices. Define the matrix \( V \) of Lanczos vectors as

\[
V = \begin{bmatrix}
  v_0 & v_1 & \cdots & v_{d-1}
\end{bmatrix},
\]

and symmetric, tridiagonal Jacobi matrix \( T \) as

\[
T = \begin{bmatrix}
  \alpha_0 & \beta_1 & & \\
  \beta_1 & \alpha_1 & \beta_2 & \\
  & \ddots & \ddots & \ddots \\
  & & \beta_{d-2} & \alpha_{d-2} & \beta_{d-1} \\
  & & & \beta_{d-1} & \alpha_{d-1}
\end{bmatrix}.
\]

Algorithm 2.2: Lanczos algorithm [16, Section 3.1.7.1]

**Given:** an \( N \times N \) symmetric matrix \( A \)

**Assumptions:** \( v_{-1} = 0 \in \mathbb{R}^N \) and \( \tilde{v}_0 \in \mathbb{R}^N \setminus \{0\} \)

For \( i = 0, 1, \ldots, d - 1 \),
1. \( \beta_i = \sqrt{v_i^T \tilde{v}_i} \)
2. \( v_i = \tilde{v}_i / \beta_i \)
3. \( \alpha_i = v_i^T A v_i \)
4. \( \tilde{v}_{i+1} = (A - \alpha_i I) v_i - \beta_{i-1} v_{i-1} \)

Let \((u_j, v_j), j = 0, \ldots, N - 1\) be the nodes and weights of an \( N \)-point numerical integration rule with respect to a given density function \( q(u) \). These nodes and weights define a discrete approximation of \( q(u) \), which we denote by \( q^{(N)}(u) \). Performing Algorithm 2.2 on

\[
A = \begin{bmatrix}
  u_0 & & \\
  & \ddots & \\
  & & u_{N-1}
\end{bmatrix}, \quad \tilde{v}_0 = \begin{bmatrix}
  \sqrt{v_0} \\
  \vdots \\
  \sqrt{v_{N-1}}
\end{bmatrix},
\]

is equivalent to performing Algorithm 2.1 on the discrete density function \( q^{(N)}(u) \) [8]. The recurrence coefficients in \( T \) from (2.18) converge to those in \( J \) from (2.13) as \( N \) goes to infinity [16]. Therefore, we can use these recurrence coefficients to produce approximations to the orthonormal polynomials \( \{\phi_0(u), \phi_1(u), \phi_2(u), \ldots\} \). Additionally, the eigendecomposition of \( T \) provides us with an approximate Gauss-Christoffel quadrature rule with respect to \( q(u) \).

3. Methodology for one-dimensional ridge functions. Recall from (2.3) that we assume \( f(x) \) is a one-dimensional ridge function,

\[
y = f(x) = g(u),
\]

where \( u = a^T x \). We want to build a pseudospectral expansion of \( g \)

\[
y = g(u) \approx \sum_{i=0}^{d} \hat{g}_i \phi_i(u), \quad \hat{g}_i = \sum_{j=0}^{M-1} \nu_j g(\lambda_j) \phi_i(\lambda_j),
\]
where \( \phi_i \) denote the orthonormal polynomials with respect to the induced density function \( q(u) \).

We compute \( q(u) \) using convolution of probability densities [1, Ch. 4]. Consider two independent random variables \( x_0 \) and \( x_1 \) with density functions \( p_0 \) and \( p_1 \), respectively. The density function of \( u = x_0 + x_1 \) is

\[
q(u) = (p_0 * p_1)(u) = \int p_0(t) p_1(u - t) \, dt.
\]

Equation (3.3) is referred to as the convolution of \( p_0 \) and \( p_1 \).

Recall from (2.2) that we assume the input space is weighted by a uniform density over the hypercube, \( x \sim U([−1, 1]^m) \). By independence, we have that \( p(x) = p_0(x_0)p_1(x_1) \cdots p_{m−1}(x_{m−1}) \) with each

\[
p_i(x_i) = \begin{cases} 
\frac{1}{2} & \text{if } |x_i| \leq 1, \\
0 & \text{otherwise}.
\end{cases}
\]

We can write the linear transform \( u = a^\top x = a_0 x_0 + \cdots + a_{m−1} x_{m−1} \) and recognize that \( a_i x_i \sim U([−a_i, a_i]) \). Thus, we can obtain \( q(u) \) by iteratively applying convolutions to each \( a_i x_i \). In practice, we approximate the integral in (3.3) using a trapezoidal rule. However, this integral is one-dimensional and does not require us to evaluate \( f \), so we can use a high density of points to approximate the convolution.

Algorithm 3.1 details the process of approximating \( q(u) \) using iterative convolutions. Note that this algorithm uses the \text{sign}(\cdot) \) function, which returns a vector of \pm 1 based on the sign of each entry of the given vector. If \( a_i = 0 \) for some \( i = 0, \ldots, m−1 \), then the corresponding \( x_i \) has no influence on \( f \). For our purposes, we may assume that \( \text{sign}(0) = 1 \). Also note that we define \( N \) equally-spaced points along the one-dimensional interval, where \( N \) is odd. This requirement is an artifact of the numerical approximation of (3.3).

We next address the issue of evaluating the ridge profile at the Gauss-Christoffel quadrature nodes, \( g(\lambda_j) \). Consider the linear transformation \( \xi_j = \lambda_j a \), and recognize that

\[
f(\xi_j) = f(\lambda_j a) = g(a^\top (\lambda_j a)) = g(\lambda_j a^\top a) = g(\lambda_j),
\]

since \( a \) is assumed to be normalized. Thus, we can evaluate \( f(\xi_j) \) in place of \( g(\lambda_j) \), provided that \( \xi_j \in [−1, 1]^m \).

To ensure we find a valid point at which to evaluate \( f \), we draw a line between the two corners of the \( m \)-dimensional hypercube that define the endpoints of the one-dimensional interval. Then, we project \( \lambda_j \) onto that line. The endpoints of the one-dimensional interval are

\[
u_ℓ = a^\top \text{sign}(-a) \quad \text{and} \quad u_r = a^\top \text{sign}(a),
\]

and the corresponding corners of the hypercube are

\[
x_ℓ = \text{sign}(-a) \quad \text{and} \quad x_r = \text{sign}(a).
\]

Then, define

\[
\xi_j = (1 - \gamma_j) x_ℓ + \gamma_j x_r.
\]
Algorithm 3.1: Discrete convolution of densities

**Given:** the vector $a \in \mathbb{R}^m \setminus \{0\}$

**Assumptions:** $x \sim U([-1,1]^m)$

1. Find the inputs of the one-dimensional interval

$$u_\ell = a^\top \text{sign}(-a), \quad u_r = a^\top \text{sign}(a),$$

and define $N$ (where $N$ is odd) equally-spaced points along the interval

$$u_j = u_\ell + j \Delta u, \quad j = 0, \ldots, N - 1,$$

where $\Delta u = (u_r - u_\ell)/(N - 1)$.

2. Initialize the vector $q = [q(u_0) \ldots q(u_{N-1})]^\top$ where

$$q(u_j) = \begin{cases} 1/(2a_0) & \text{if } |u_j| \leq a_0, \\ 0 & \text{otherwise}. \end{cases}$$

3. for $i = 1, \ldots, m - 1$

   (i) Define $p = [p(u_0) \ldots p(u_{N-1})]^\top$ where

   $$p(u_j) = \begin{cases} 1/(2a_i) & \text{if } |u_j| \leq a_i, \\ 0 & \text{otherwise}. \end{cases}$$

   (ii) For $j = 0, \ldots, N - 1$, define

   $$k_0 = \max \left\{ 0, j - \frac{N - 1}{2} \right\},$$
   $$k_1 = \min \left\{ \frac{N - 1}{2} + j, N - 1 \right\},$$

   and compute

   $$q_j = \sum_{k=k_0}^{k_1} q_k p_{\frac{k_1 - j + k}{2}}$$

**NOTE:** skip any $i$ for which $a_i = 0$

**Output:** $q = [q(u_0) \ldots q(u_{N-1})]^\top$

where $\gamma_j = (\lambda_j - u_\ell)/(u_r - u_\ell)$. Figure 4 illustrates this process for a three-dimensional cube.

Algorithm 3.2 outlines the steps for building a pseudospectral expansion of a one-dimensional ridge function. To approximate the integral of (3.1), we recognize that

$$\hat{g}_0 = \sum_{j=0}^{d} \nu_j g(\lambda_j) \approx \int g(u) q(u) \, du = \int f(x) p(x) \, dx.$$

Algorithm 3.2 contains two levels of approximation: (i) the discrete approximation of $q(u)$ using an $N$-point trapezoidal rule and (ii) the number $d+1$ of Lanczos iterations.
performed. The number of Lanczos iterations corresponds to the number of terms in the polynomial expansion of \( g \) and the number of Gauss-Christoffel quadrature nodes. The latter is important as this is the number of model evaluations required to compute the pseudospectral coefficients. In general, we should choose \( N \gg d \) since the approximation of \( q \) at the \( N \) trapezoidal points does not require any function evaluations, which are typically the most expensive step. In the next section, we numerically study this two-level approximation.

3.1. Numerical study. In this section, we numerically study the behavior of Algorithm 3.2 for

\[
y = \sin\left(2\pi (a^\top x)\right) + \cos\left(\frac{\pi}{2} (a^\top x)\right), \quad x \in \mathbb{R}^{25}.
\]

We assume \( x \sim U([-1,1]^{25}) \). Notice that (3.21) is an exact one-dimensional ridge function.

Figure 5 contains the results from Algorithm 3.2 for \( N = 10,000 \) and \( d = 50 \). The first plot shows the one-dimensional ridge profile of (3.21) in black with the pseudospectral approximation laid on top by a dashed red line. The absolute error is at each point is shown in the second plot, and the density \( q(u) \) is in the third plot. We see the pseudospectral approximation performing well, particularly in regions of high probability. For \( u \in [-3,3] \), the absolute error is near machine accuracy. Near the endpoints, the error begins to increase. However, \( q(u) \) in these regions is many orders of magnitude smaller than in the middle of the domain.

Next, we examine the two levels of approximation in Algorithm 3.2 Recall that these two levels are (i) the number \( N \) of trapezoid points used to construct the discrete approximation of \( q(u) \) and (ii) the number \( d \) of Lanczos iterations performed. The latter corresponds to the degree of the polynomial expansion as well as the number of Gauss-Christoffel quadrature nodes.

Figure 6 contains the results of this study. The first plot shows the approximated \( L^2 \) norm of the error between \( f \) and the pseudospectral expansions for varying values of \( N \) and \( d \). The error appears to depend strongly on \( d \). This is because a high-degree polynomial is required to fit the highly-oscillatory nature of \( f(x) \). The second plot contains approximations of the integral of \( f(x) \) using the first coefficient in the pseudospectral expansion. Here, we see a strong dependence on \( N \). This is because integration errors in the Gauss-Christoffel decay quickly with \( d \). To improve the
Algorithm 3.2  Lanczos-Stieljes one-dimensional ridge function approximation

**Given:** function \( f : \mathbb{R}^m \rightarrow \mathbb{R} \) and unit vector \( a \in \mathbb{R}^m \setminus \{0\} \) such that
\[
y = f(x) = g(a^\top x)
\]
for some unknown \( g : \mathbb{R} \rightarrow \mathbb{R} \).

**Assumptions:** \( x \sim U([-1, 1]^m) \) and \( f \) is square-integrable with respect to the input density \( p(x) \).

1. Perform Algorithm 3.1 to obtain \( q = [q(u_0) \ldots q(u_{N-1})]^\top \).
2. Perform Algorithm 2.2 on (3.16)
\[
A = \begin{bmatrix}
u_0 & \cdots & \nu_{N-1}
\end{bmatrix}, \quad \tilde{v}_0 = \begin{bmatrix}\sqrt{q(u_0)} \\ \vdots \\ \sqrt{q(u_{N-1})}\end{bmatrix},
\]
to obtain the Jacobi matrix \( T \).
3. Take the eigendecomposition of \( T \),
\[
T = Q \Lambda Q^\top,
\]
where the eigenvalues are \( \{\lambda_0, \lambda_1, \ldots, \lambda_d\} \) and the eigenvectors are normalized. Define \( \nu_j = (QQ^\top)^{1/2} \) for \( j = 0, \ldots, d \).
4. For \( j = 0, \ldots, d \), compute
\[
\xi_j = (1 - \gamma_j) x_\ell + \gamma_j x_r, \quad \gamma_j = (\lambda_j - u_\ell) / (u_r - u_\ell),
\]
where \( x_\ell = \text{sign}(-a), x_r = \text{sign}(a), u_\ell = a^\top \text{sign}(-a), \) and \( u_r = a^\top \text{sign}(a) \).
5. Compute the pseudospectral coefficients
\[
\hat{g}_i = \sum_{j=0}^d \nu_j f(\xi_j) \phi_i(\lambda_j).
\]
6. Build the pseudospectral expansion
\[
y = g(u) \approx \sum_{i=0}^d \hat{g}_i \phi_i(u).
\]

approximation, we must improve our discrete approximation of the density \( q(u) \).

4. Extension of methodology to near-1D ridge functions. We next consider how Algorithm 3.2 can be extended for the case when \( f(x) \) is well-approximated by a one-dimensional ridge function,
\[
y = f(x) \approx g(u), \quad \text{where} \quad u = a^\top x.
\]
The best \( L^2 \) approximation of \( f \) by \( g \) is the expected value of the output conditioned on \( u = a^\top x \) \cite[Ch. 8]{31}. That is,
\[
g(u) = \mathbb{E} [y = f(x) | u = a^\top x].
\]
We can measure the degree to which $f$ is a near one-dimensional ridge function using the conditional variance $\text{Var}[y = f(x)|u = \mathbf{a}^\top \mathbf{x}]$. This conditional variance results in a baseline error for any one-dimensional approximation of $f(x)$. In shadow plots (see Figure 1), this variance is visually represented by the spread of the data for a fixed value of $u = \mathbf{a}^\top \mathbf{x}$.

The difficulty in applying the methodology introduced in Section 3 to (4.1) is how to compute $g(\lambda_j)$. Recall from (3.11) that, in the case of the exact ridge function, we can transform $\lambda_j$ into a corresponding input $\xi_j$ in the full-dimensional space and evaluate $f$ at this point, provided that $\xi_j$ is in the domain of $f$. In this scenario, we want to approximate the conditional expectation of $f(x)$ for all $\mathbf{x} \in [-1, 1]^m$ such that $\lambda_j = \mathbf{a}^\top \mathbf{x}$. We write the sample approximation of this conditional expectation as

\[
(4.3) \quad g(\lambda_j) \approx \hat{g}(\lambda_j) = \frac{1}{M_j} \sum_{i=1}^{M_j} f(\xi_{j,i}),
\]

where the $M_j$ input values $\xi_{j,i} \in [-1, 1]^m$ are sampled uniformly conditioned on $\mathbf{a}^\top \xi_{j,i} = \lambda_j$. We recommend using a hit-and-run type of algorithm to obtain the
\(\xi_{j,i}\). First, initialize the sampling procedure using the method described for (3.14) to obtain \(\xi_{i,0}\). Then, given sample \(\xi_{j,i}\), we can obtain \(\xi_{j,i+1}\) using the following approach. Choose a random normalized vector \(\mathbf{w} \in \mathbb{R}^m \setminus \{0\}\) with \(\mathbf{w}^\top \mathbf{a} = 0\) and a random value \(t \in [u_{\ell}, u_r]\), where \(u_{\ell}\) and \(u_r\) are the endpoints of the one-dimensional interval from (3.12). Define

\[
\xi_{j,i+1} = \xi_{j,i} + t \mathbf{w},
\]

provided that \(\xi_{j,i+1} \in [-1,1]^m\). If this is not true, then repeatedly sample \(t \in [u_{\ell}, u_r]\) until a valid point is obtained.

The sample approximation of the conditional expectation in (4.3) results in noisy estimates of \(g(\lambda_j)\). For this reason, constructing an interpolating polynomial, as described in Section 3, may not be the best approach given a restricted computational budget. We recommend truncating the pseudospectral polynomial expansion further to avoid overfitting the approximation. For each Gauss–Christoffel quadrature node \(\lambda_j\), we have \(\xi_{j,i}\) for \(i = 0, \ldots, M_j - 1\). Estimate the standard error in each approximation of the conditional expectation, \(s_j = \bar{\sigma}_j / \sqrt{M_j}\), where \(\bar{\sigma}_j\) is the standard deviation of the \(f(\xi_{j,i})\), \(i = 0, \ldots, M_j - 1\). Assuming that \(g\) can be expressed as a polynomial expansion, we expect a decay in the coefficients \(g_i\) from (2.10) for sufficiently large \(i\). Recall that the pseudospectral coefficients, \(\hat{g}_i\) from (2.11), approximate the true coefficients. We suggest truncating the expansion at a degree \(d = \hat{d}\) polynomial, where \(|g_i| < \sum_{j=0}^d s_j / (d + 1)\) for all \(i > \hat{d}\). This heuristic removes terms whose contribution to the expansion is smaller than the noise in the sample approximations \(\hat{g}(\lambda_j)\).

**4.1. Numerical studies: Example 1.** In this section, we numerically study the behavior of the extension of Algorithm 3.2 to nearly one-dimensional ridge functions. We consider the function

\[
y = \sin\left(\frac{\pi}{5}(\mathbf{a}^\top \mathbf{x})\right) + \frac{1}{5} \cos\left(\frac{4\pi}{9}(\mathbf{a}^\top \mathbf{x})\right) + \frac{1}{40}\mathbf{x}^\top \mathbf{B} \mathbf{1}, \quad \mathbf{x} \in \mathbb{R}^{25},
\]

here \(\mathbf{a} \in \mathbb{R}^5 \setminus \{0\}\) defines the ridge-like structure, \(\mathbf{B} \in \mathbb{R}^{5 \times 24}\) contains an orthonormal basis for the subspace orthogonal to \(\mathbf{a}\), and \(\mathbf{1} \in \mathbb{R}^{24}\) is a vector of ones. We assume \(\mathbf{x} \sim \mathcal{U}([-1,1]^{25})\).

Figure 7 shows the results of using the extension of Algorithm 3.2 discussed in the previous section to (4.5). The plot on the left is a shadow plot of evaluations of (4.5) against \(u = \mathbf{a}^\top \mathbf{x}\). The spread in the plot is due to variations in the 24 directions orthogonal to \(\mathbf{a}\). The red line is the polynomial approximation of the ridge profile—recall that this is \(g(u) = \mathbb{E}[y = f(\mathbf{x})|u = \mathbf{a}^\top \mathbf{x}]\). This is computed using \(d = 11\) and \(M = 50\) total functions evaluations distributed among the 12 Gauss–Christoffel quadrature nodes. The polynomial expansion is truncated at \(d = 6\) to avoid overfitting the noise in the approximation of \(g(\lambda_j)\). Note that fitting a polynomial of total degree 6 in 25 dimensions would require at least \(M = \binom{25+6}{6} = 736, 281\) to have a well-posed problem.

**4.2. Numerical studies: Example 2.** Next, we consider a physically-motivated problem in magnetohydrodynamics: the Hartmann problem [13]. The Hartmann problem models the flow of an ionized fluid in the presence of a perpendicular magnetic field along an infinitely-long channel (see Figure 8). The fluid flow induces a horizontal component in the magnetic field. The total magnitude of this induced magnetic field, denoted \(B_{\text{ind}}\), is the output of interest for this problem and can be
Fig. 7: The results of the extension to Algorithm 3.2 applied to the approximate ridge function (4.5). The left plot contains a show plot of \( f(x) \) along the \( u = a^\top \mathbf{x} \) axis with the \( d = 11 \) polynomial approximation on top of it. The right plot shows the density \( q(u) \) written as

\[
B_{\text{ind}}(x) = \frac{\partial p_0}{\partial x} \frac{\ell \mu_0}{2B_0} \left( 1 - 2\sqrt{\eta \mu} \tanh \left( \frac{B_0 \ell}{2\sqrt{\eta \mu}} \right) \right),
\]

where the five input variables are described in Table 1. The constant values \( \ell \) and \( \mu_0 \) are the width of the channel and the magnetic permeability of free space (i.e., a universal constant), respectively. It has been shown that (4.6) exhibits approximate one-dimensional ridge structure with respect to the log-transformed inputs [19]. Thus, we consider (4.6) with respect to the inputs

\[
\mathbf{x} = [\log(\mu) \ \log(\rho) \ \log(\partial p_0/\partial x) \ \log(\eta) \ \log(B_0)]^\top.
\]

We assume a uniform density function over the range of values given in Table 1.

Fig. 8: An illustration of the Hartmann problem, which models the flow of an ionized fluid in the presence of a perpendicular magnetic field along an infinitely-long channel.

Figure 9 contains the results of applying the extension of Algorithm 3.2 to (4.6). The left plot is a shadow plot of \( B_{\text{ind}} \) against \( u = a^\top \mathbf{x} \) overlaid with a polynomial approximation of the ridge profile. This approximation was constructed using \( d = 5 \) with a total computational budget of 60 function evaluations (10 for each of the 6 quadrature nodes). The polynomial expansion was truncated at \( \tilde{d} = 3 \) according the heuristic described in Section 4. The right figure contains approximate \( L^2 \) errors in the one-dimensional polynomial approximation of \( B_{\text{ind}} \) for increasing polynomial degree. The various lines represent different numbers of samples used to approximate
Table 1: Descriptions and ranges of the five variable inputs to (4.6). Note that the range of values is given in terms of the log of each input.

| Symbol | Description       | Range of log(input) |
|--------|-------------------|---------------------|
| µ      | fluid viscosity   | [log(0.05), log(0.2)] |
| ρ      | fluid density     | [log(1), log(5)]    |
| ∂p₀/∂x| applied pressure gradient | [log(0.5), log(3)] |
| η      | magnetic resistivity | [log(0.5), log(3)] |
| B₀     | applied magnetic field | [log(0.25), log(0.75)] |

the conditional expectation at each quadrature node. The total computational budget for computing degree \( d \) polynomial approximation is \( M_j (d + 1) \). Note that the errors level off as we increase the polynomial degree. This baseline error is due to the fact that \( B_{ind} \) is not an exact one-dimensional ridge function. We incur this error by choosing to build a one-dimensional polynomial approximation. However, constructing this approximation is exponentially cheaper than trying to build an approximation on the full input space.

Fig. 9: The results of applying the extension of Algorithm 3.2 to (4.6). The left plot is a shadow plot of \( B_{ind} \) against \( u = a^\top x \). The right plot shows approximation errors for various polynomial degrees and computational budgets.

5. Summary and conclusion. We introduce a novel algorithm for constructing polynomial approximations of one-dimensional ridge functions using the Lanczos iterative method. In general, building a polynomial surrogate of an \( m \)-dimensional function suffers from the curse of dimensionality—an exponential increase in computational costs associated with increases in \( m \). We also introduce an approach to extending this algorithm to functions that are well-approximated by a one-dimensional ridge function.

We numerically study the new algorithm on several test problems, including exact and approximate ridge functions. We show that exploiting low-dimensional structure can result in exponential savings while maintaining accuracy. Additionally, we study the two-level approximation behavior of the algorithm. The first level is a discrete approximation of the induced density function \( q(u) \). The second level is the number of Lanczos iterations performed. This corresponds to the degree of the polynomial approximation of the ridge function as well as the number of Gauss-Christoffel quadrature points. In studying the extension of the algorithm to nearly one-dimensional
ridge functions, we show that we can quickly achieve the baseline error using very few function evaluations.

REFERENCES

[1] P. Billingsley, *Probability and Measure*, Wiley, New York, 1986, https://www.wiley.com/en-us/Probability-and-Measure%2C-An-Anniversary-Edition-p-9781118341919.

[2] A. Cohen, I. Daubechies, R. DeVore, G. Kerkyacharian, and D. Picard, Capturing ridge functions in high dimensions from point queries, Constructive Approximation, 35 (2012), pp. 225–243, https://doi.org/10.1007/s00365-011-9147-6.

[3] P. G. Constantine, *Active Subspaces: Emerging Ideas for Dimension Reduction in Parameter Studies*, SIAM, Philadelphia, 2015, http://doi.org/10.1137/1.9781611973860.

[4] P. G. Constantine and A. Doostan, Time-dependent global sensitivity analysis with active subspaces for a lithium ion battery model, Statistical Analysis and Data Mining, 10 (2017), pp. 243–262, https://doi.org/10.1002/sam.11347.

[5] P. G. Constantine, A. Eftekhari, J. Hokanson, and R. Ward, A near-stationary subspace for ridge approximation, Computer Methods in Applied Mechanics and Engineering, 326 (2017), pp. 402–421, https://doi.org/10.1016/j.cma.2017.07.038.

[6] P. G. Constantine, M. S. Eldred, and E. T. Phipps, Sparse pseudospectral approximation method, Computer Methods in Applied Mechanics and Engineering, 229 (2012), pp. 1–12, https://doi.org/10.1016/j.cma.2012.03.019.

[7] P. G. Constantine, M. Emory, J. Larsson, and G. Iaccarino, Exploiting active subspaces to quantify uncertainty in the numerical simulation of the HyShot II scramjet, Journal of Computational Physics, 302 (2015), pp. 1–20, https://doi.org/10.1016/j.jcp.2015.09.001.

[8] P. G. Constantine and E. T. Phipps, A lanczos method for approximating composite functions, Applied Mathematics and Computation, 218 (2012), pp. 11751–11762, https://doi.org/10.1016/j.amc.2012.05.009.

[9] P. G. Constantine, B. Zaharatos, and M. Campanelli, Discovering an active subspace in a single-diode solar cell model, Statistical Analysis and Data Mining, 8 (2015), pp. 264–273, https://doi.org/10.1002/sam.11281.

[10] R. D. Cook, *Regression Graphics: Ideas for Studying Regression through Graphics*, John Wiley & Sons, Inc, New York, 1998, http://doi.org/10.1002/9780470516931.

[11] A. Cortesi, P. Constantine, T. Magin, and P. M. Congedo, Forward and backward uncertainty quantification with active subspaces: application to hypersonic flows around a cylinder, hal-01592591 (2017), https://hal.inria.fr/hal-01592591/

[12] A. Cortesi, P. G. Constantine, T. Magin, and P. M. Congedo, Forward and backward uncertainty quantification with active subspaces: application to hypersonic flows around a cylinder, PhD thesis, INRIA Bordeaux, équipe CARDAMOM, 2017.

[13] T. G. Cowling and R. B. Lindsay, *Magnetohydrodynamics*, Physics Today, 10 (1957), p. 40, http://doi.org/10.1063/1.3060498.

[14] D. L. Donoho, *High-dimensional data analysis: The curses and blessings of dimensionality*, in AMS Conference on Math Challenges of the 21st Century, 2000, http://www.stat.stanford.edu/~donoho/Lectures/CMBS/Curses.pdf.

[15] G. Fasshauer and M. McCourt, *Kernel-based Approximation Methods using MATLAB*, World Scientific Publishing Company, Singapore, 2016, https://doi.org/10.1142/9335.

[16] W. Gautschi, *Orthogonal Polynomials*, Oxford Press, Oxford, 2004, https://global.oup.com/academic/product/orthogonal-polynomials-9780198506720.

[17] J. M. Gilbert, J. L. Jefferson, P. G. Constantine, and R. M. Maxwell, Global spatial sensitivity of runoff to subsurface permeability using the active subspace method, Advances in water resources, 92 (2016), pp. 30–42.

[18] A. Glaws, P. G. Constantine, and R. D. Cook, Inverse regression for ridge recovery, arXiv:1702.02227v1 (2017), https://arxiv.org/abs/1702.02227.

[19] A. Glaws, P. G. Constantine, J. Shadid, and T. M. Wildey, Dimension reduction in MHD power generation models: dimensional analysis and active subspaces, Statistical Analysis and Data Mining, 10 (2017), pp. 312–325, https://doi.org/10.1002/sam.11355.

[20] G. H. Golub and G. Meurant, *Matrices, Moments, and Quadrature with Applications*, Princeton University, Princeton, 2010, http://press.princeton.edu/titles/9104.html.

[21] G. H. Golub and J. H. Welsch, Calculation of Gauss quadrature rules, Mathematics of Computation, 23 (1969), pp. 221–230, https://doi.org/10.1090/S0025-5718-69-99647-1.

[22] R. B. Gramacy and H. Li, Gaussian process single-index models as emulators for computer experiments, Technometrics, 54 (2012), pp. 39–41.
[23] J. M. Hokanson and P. G. Constantine, Data-driven polynomial ridge approximation using variable projection, SIAM Journal on Scientific Computing, 40 (2017), pp. A1566–A1589, https://doi.org/10.1137/17M117690.

[24] J. L. Jefferson, J. M. Gilbert, P. G. Constantine, and R. M. Maxwell, Active subspaces for sensitivity analysis and dimension reduction of an integrated hydrologic model, Computers & Geosciences, 83 (2016), pp. 127–138, http://doi.org/10.1016/j.cageo.2015.02.001.

[25] J. R. Kohler and A. B. Owen, Computer experiments, Handbook of Statistics, 13 (1996), pp. 261–308, http://doi.org/10.1016/S0169-7161(96)13011-X.

[26] J. L. Jefferson, J. M. Gilbert, P. G. Constantine, and R. M. Maxwell, Active subspaces for sensitivity analysis and dimension reduction of an integrated hydrologic model, Computers & Geosciences, 83 (2016), pp. 127–138, http://doi.org/10.1016/j.cageo.2015.02.001.

[27] J. R. Koehler and A. B. Owen, Computer experiments, Handbook of Statistics, 13 (1996), pp. 261–308, http://doi.org/10.1016/S0169-7161(96)13011-X.

[28] C. Lanczos, An iteration method for the solution of the eigenvalue problem of linear differential and integral operators, Journal of Research of the National Bureau of Standards, 45 (1950), pp. 255–282.

[29] C. Othmer, T. Lukaczyk, P. G. Constantine, and J. J. Alonso, On active subspaces in car aerodynamics, American Institute of Aeronautics and Astronautics, (2016), https://doi.org/10.2514/6.2016-4294.

[30] A. Pinkus, Ridge Functions, Cambridge University Press, 2015, https://doi.org/10.1017/CBO9781139087942.

[31] A. Pinkus, Ridge Functions, Cambridge University Press, 2015, https://doi.org/10.1017/CBO9781139087942.

[32] S. Razavi, B. A. Tolson, and D. H. Burn, Review of surrogate modeling in water resources, Water Resources Research, 48 (2012), https://doi.org/10.1029/2011WR011527.

[33] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn, Design and analysis of computer experiments, Statistical Science, 4 (1989), pp. 409–423, http://www.jstor.org/stable/2245838.

[34] A. Saltelli, M. Ratto, T. Andres, J. C. Francesca Campolongo, D. Galtelli, M. Saisana, and S. Tarantola, Global Sensitivity Analysis. The Primer, John Wiley & Sons Ltd, England, 2008, https://doi.org/10.1002/9780470725184.

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

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

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

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

[39] T. J. Stieltjes, Quelques recherches sur la theorie des quadratures dites mecaniques, Annales scientifiques de l’ecole Normale Superieure, 1 (1884), pp. 409–426, http://eudml.org/doc/80911.

[40] G. Wahba, Spline Models for Observational Data, SIAM, Philadelphia, 1990, https://doi.org/10.1137/1.9781611970128.

[41] H. Wozniakowski, Tractability and strong tractability of linear multivariate problems, Journal of Complexity, 10 (1994), pp. 96–128, http://doi.org/10.1006/jcom.1994.1004.

[42] D. Xiu and G. E. Karniadakis, The Wiener–Askey polynomial chaos for stochastic differential equations, SIAM Journal on Scientific Computing, 24 (2002), pp. 619–644, https://doi.org/10.1137/S1064827501387826.