arXiv:1810.08505v2 [math.NA] 30 Apr 2019

Generation of point sets by convex optimization for interpolation in reproducing kernel Hilbert spaces

Ken'ichiro Tanaka*

April 30, 2019

Abstract

We propose algorithms to take point sets for kernel-based interpolation of functions in reproducing kernel Hilbert spaces (RKHSs) by convex optimization. We consider the case of kernels with the Mercer expansion and propose an algorithm by deriving a second-order cone programming (SOCP) problem that yields n points at one sitting for a given integer n. In addition, by modifying the SOCP problem slightly, we propose another sequential algorithm that adds an arbitrary number of new points in each step. Numerical experiments show that in several cases the proposed algorithms compete with the P-greedy algorithm, which is known to provide nearly optimal points.

Keywords: reproducing kernel Hilbert space, kernel interpolation, point set, power function, optimal design, second order cone programming

Mathematics Subject Classification (2010): 65D05, 65D15, 41A05, 46E22

1 Introduction

We propose algorithms to take point sets for kernel-based interpolation of functions in reproducing kernel Hilbert spaces (RKHSs). The algorithms are based on a convex optimization problem derived by the Mercer series of the kernel.

Kernel-based methods are useful in various fields of scientific computing and engineering. Their details and applications can be found in the references [6, 8, 9, 21, 24]. Many of the applications include approximation of multivariate functions by scattered data as a fundamental component. Therefore we focus on kernel interpolation in this paper. Let d be a positive integer and let $\Omega \subset \mathbf{R}^d$ be a region. For a finite set $\{x_1, \ldots, x_n\} \subset \Omega$ of n pairwise distinct points and a set $\{y_1, \ldots, y_n\} \subset \mathbf{R}$, we aim to find a function $s : \Omega \to \mathbf{R}$ with the interpolation condition $s(x_i) = y_i$ $(i = 1, \ldots, n)$. To this end, we consider a positive definite kernel $K : \Omega \times \Omega \to \mathbf{R}$ and the function s given by

$$s(x) = \sum_{j=1}^{n} c_j \, K(x, x_j), \tag{1.1}$$

where c_j are determined by the interpolation condition.

*Department of Mathematical Informatics, Graduate School of Information Science and Technology, University of Tokyo. 7-3-1 Hongo, Bunkyo-ku, Tokyo, 113-8656, Japan. kenichiro@mist.i.u-tokyo.ac.jp

For a positive definite kernel K, there exists a unique Hilbert space $\mathcal{H}_K(\Omega)$ of functions from Ω to $\mathbf{R}$ where K is a reproducing kernel, that is,

$$\text{(a) } \forall x \in \Omega,\ K(\cdot, x) \in \mathcal{H}_K(\Omega), \tag{1.2}$$

$$\text{(b) } \forall x \in \Omega,\ \forall f \in \mathcal{H}_K(\Omega),\ \langle f, K(\cdot, x)\rangle_{\mathcal{H}_K(\Omega)} = f(x). \tag{1.3}$$

The reproducing kernel Hilbert space (RKHS) $\mathcal{H}_K(\Omega)$ is called the native space of K. As the kernel interpolation, we consider the case that $y_j = f(x_j)$ for $f \in \mathcal{H}_K(\Omega)$. Then, we encounter a problem of approximating functions in $\mathcal{H}_K(\Omega)$ by the kernel interpolation.

Such approximation is widely used, for example, for the numerical solution of partial differential equations via collocation [14]. In such situations, we can choose the point set $\{x_1, \ldots, x_n\}$ to achieve accurate computation. The choice is critical for the kernel interpolation. Actually, there are several fundamental results about the approximation power of the kernel interpolation for well-distributed points [24, Chapter 11]. Based on these, many methods for choosing well-distributed points have been proposed: geometric greedy algorithm [11, 12, 13], P-greedy algorithm [11, 13, 19], f-greedy algorithm [20], and f/P-greedy algorithm [15]. Furthermore, there are methods based on optimal designs and similar concepts to them. In the paper [5], the authors compute (approximate) Fekete points, the points maximizing the Vandermonde determinants with respect to a polynomial basis. For the computation, they use the routines provided in Matlab for general nonlinear optimization. For other methods, see [9, Appendix B] and the references therein.

In this paper, we focus on such methods that depend just on the space $\mathcal{H}_K(\Omega)$ and provide universal interpolation on it. The P-greedy algorithm and the algorithms for the Fekete points are in this category. The P-greedy algorithm is a sequential one adding a new point one by one in each step as described in Section 2. It is easy to implement and known to be nearly optimal [19]. On the other hand, the others are non-sequential algorithms providing a point set at one sitting. The Fekete points for a polynomial basis are known to be well-distributed, although they are given by non-convex optimization. Therefore we expect that algorithms of this type for a general kernel K would provide good point sets for kernel interpolation in $\mathcal{H}_K(\Omega)$. To investigate whether this expectation is the case or not, we propose an algorithm for generating point sets by approximately maximizing the determinant of the kernel matrix $\mathcal{K} = (K(x_i, x_j))_{ij}$. More precisely, we show that they can be computed by *convex optimization* in the case of kernels with the Mercer expansion. Actually, we can formulate a second-order cone programming (SOCP) problem yielding point sets. As a by-product of this formulation, we provide a variant sequential algorithm that adds an arbitrary number of new points in each step. By some numerical experiments, we observe that the point sets provided by the proposed algorithms are well-distributed and compete with the points provided by the P-greedy algorithm in several cases.

The rest of this paper is organized as follows. In Section 2, we summarize the fundamental facts of positive definite kernels and reproducing kernel Hilbert spaces. In Section 3, we derive a relaxed D-optimal design problem to generate a point set by using the Mercer expansion of a kernel. Then, according to the paper [17], we describe the equivalent formulation of the design problem as a SOCP problem in Section 4. We show the results of numerical experiments in Section 5 and conclude this paper by Section 6.

2 Interpolation in reproducing kernel Hilbert spaces

2.1 Positive definite kernels

Let $\Omega \subset \mathbf{R}^d$ be a region and let $K : \Omega \times \Omega \to \mathbf{R}$ be a continuous, symmetric, and positive definite kernel. That is, for any set $\mathcal{X}_n = \{x_1, \ldots, x_n\} \subset \Omega$ of distinct points, the kernel matrix

$$\mathcal{K} = \begin{bmatrix} K(x_1, x_1) & \cdots & K(x_n, x_1) \\ \vdots & \ddots & \vdots \\ K(x_1, x_n) & \cdots & K(x_n, x_n) \end{bmatrix} \tag{2.1}$$

is positive definite. Throughout this paper, we assume that K has the Mercer series expansion given by

$$K(x, y) = \sum_{\ell=1}^{\infty} \lambda_\ell \, \varphi_\ell(x) \, \varphi_\ell(y), \tag{2.2}$$

where $\lambda_1 \geq \lambda_2 \geq \cdots > 0$, and $\{\varphi_\ell\}$ is an orthonormal system in $L^2(\Omega, \rho)$, the Lebesgue space with a density function ρ. This expansion is known to be possible on appropriate conditions [9, Theorem 2.2].

Example 1 (Brownian motion kernel on $\Omega = [0, 1] \subset \mathbf{R}$ [9, Section A.1.2])**.** The kernel

$$K(x, y) = \min\{x, y\} \qquad (x, y \in [0, 1])$$

is the Brownian motion kernel on the domain $\Omega = [0, 1]$. Its Mercer series is given by letting

$$\lambda_\ell = \frac{4}{(2\ell - 1)^2 \pi^2}, \quad \varphi_\ell(x) = \sqrt{2} \sin\left((2\ell - 1)\frac{\pi x}{2}\right)$$

in (2.2).

Example 2 (Spherical inverse multiquadric kernel on $\Omega = S^2$ [9, Section A.9.1])**.** Let γ be a real number with $0 < \gamma < 1$. The kernel

$$K(x, y) = \frac{1}{\sqrt{1 + \gamma^2 - 2\gamma \, x^T y}} \qquad (x, y \in S^2)$$

is the spherical inverse multiquadric kernel on the sphere $S^2 = \{\|x\| = 1 \mid x \in \mathbf{R}^3\}$. Its Mercer series is given by

$$K(x, y) = \sum_{n=0}^{\infty} \frac{4\pi\gamma^n}{2n + 1} \sum_{\ell=1}^{2n+1} Y_{n,\ell}(x) Y_{n,\ell}(y), \tag{2.3}$$

where $Y_{n,\ell}$ are the spherical harmonics [3, 7].

Example 3 (Gaussian kernel on $\Omega \subset \mathbf{R}^d$ [9, Section 12.2.1])**.** Let ε be a positive real number. The kernel

$$K(x, y) = \exp(-\varepsilon^2 \|x - y\|^2)$$

is the multivariate Gaussian kernel. By letting $\beta = (1 + (2\varepsilon/\alpha)^2)^{1/4}$ and $\delta = \alpha^2(\beta^2 - 1)/2$ for a positive real number α, its Mercer series is given by

$$K(x, y) = \sum_{\boldsymbol{n} \in \mathbf{N}^d} \boldsymbol{\lambda_n}\, \boldsymbol{\varphi_n}(x)\, \boldsymbol{\varphi}_n(y), \qquad \boldsymbol{\lambda_n} = \prod_{\ell=1}^{d} \lambda_{n_\ell}, \qquad \boldsymbol{\varphi_n}(x) = \prod_{\ell=1}^{d} \varphi_{n_\ell}(x_\ell), \tag{2.4}$$

where

$$\lambda_n = \sqrt{\frac{\alpha^2}{\alpha^2 + \delta^2 + \varepsilon^2}} \left(\frac{\varepsilon^2}{\alpha^2 + \delta^2 + \varepsilon^2} \right)^{n-1}, \qquad \varphi_n(x) = \sqrt{\frac{\beta}{2^{n-1}\Gamma(n)}}\, \mathrm{e}^{-\delta^2 x^2} H_{n-1}(\alpha\beta x). \tag{2.5}$$

The functions H_n are Hermite polynomials of degree n.

2.2 Interpolation of functions in reproducing kernel Hilbert spaces

Let $\mathcal{H}_K(\Omega)$ be the native space of K on Ω. We denote the inner product of $\mathcal{H}_K(\Omega)$ by $\langle\ ,\ \rangle_{\mathcal{H}_K(\Omega)}$. Then, the kernel K satisfies the reproducing property:

$$f \in \mathcal{H}_K(\Omega), \quad f(x) = \langle f, K(\cdot, x) \rangle_{\mathcal{H}_K(\Omega)}. \tag{2.6}$$

For a function $f \in \mathcal{H}_K(\Omega)$, we consider its interpolant of the form

$$s_f(x) = \sum_{j=1}^{n} c_j K(x, x_j), \qquad x \in \Omega \subset \mathbf{R}^d, \tag{2.7}$$

where $\mathcal{X}_n = \{x_1, \ldots, x_n\} \subset \Omega$ is a set of points for interpolation. The coefficients c_j are determined by the interpolation equations

$$s_f(x_i) = f(x_i), \quad i = 1, \ldots, n. \tag{2.8}$$

The interpolant in (2.7) can be rewritten in the form

$$s_f(x) = \sum_{j=1}^{n} f(x_j)\, u_j(x), \tag{2.9}$$

where u_j are cardinal bases satisfying the Lagrange property

$$u_j(x_i) = \delta_{ij}, \qquad i, j = 1, \ldots, n. \tag{2.10}$$

The vector $u(x) = (u_1(x), \ldots, u_n(x))^T$ is determined by the linear equation

$$\mathcal{K}\, u(x) = k(x), \tag{2.11}$$

where $k(x) = (K(x, x_1), \ldots, K(x, x_n))^T$.

Using the reproducing property in (2.6), we can derive a well-known error bound of the interpolation as follows:

$$|f(x) - s_f(x)| = \left| f(x) - \sum_{j=1}^{n} f(x_j)\, u_j(x) \right| = \left| \left\langle f, K(\cdot, x) - \sum_{j=1}^{n} K(\cdot, x_j)\, u_j(x) \right\rangle_{\mathcal{H}_K(\Omega)} \right|$$

$$
\begin{aligned}
&= \left| \left\langle f, K(\cdot, x) - k(\cdot)^T \mathcal{K}^{-1}\, k(x) \right\rangle_{\mathcal{H}_K(\Omega)} \right| \\
&\le \|f\|_{\mathcal{H}_K(\Omega)} \left\| K(\cdot, x) - k(\cdot)^T \mathcal{K}^{-1}\, k(x) \right\|_{\mathcal{H}_K(\Omega)} \\
&= \|f\|_{\mathcal{H}_K(\Omega)}\, P_{K,\mathcal{X}_n}(x),
\end{aligned} \tag{2.12}
$$

where

$$
P_{K,\mathcal{X}_n}(x) = \sqrt{K(x,x) - k(x)^T \mathcal{K}^{-1}\, k(x)} \tag{2.13}
$$

is called the power function. The set of its zeros is $\mathcal{X}_n = \{x_1, \ldots, x_n\}$.

By using the power function, the P-greedy algorithm generating points for interpolation is described as follows[1]. Start with $\mathcal{X}_1 = \{x_1\}$ for a point $x_1 \in \Omega$ maximizing $\sqrt{K(x,x)}$, and

$$
\mathcal{X}_j = \mathcal{X}_{j-1} \cup \{x_j\} \quad \text{with} \quad P_{K,\mathcal{X}_{j-1}}(x_j) = \max_{x \in \Omega} P_{K,\mathcal{X}_{j-1}}(x) \quad (j = 2, \ldots, n).
$$

3 Optimization problems to generate good point configurations

3.1 Upper bound of the power function

If we can find a minimizer $\mathcal{X}_n^\dagger = \{x_1^\dagger, \ldots, x_n^\dagger\}$ of the worst case error

$$
\max_{x \in \Omega} P_{K,\mathcal{X}_n}(x), \tag{3.1}
$$

we can take $\mathcal{X}_n^\dagger$ as one of the best point sets for the interpolation. However, this optimization problem is difficult. Then, we provide an upper bound of the power function and consider its minimization to obtain an approximate minimizer of the value in (3.1). To this end, We start with providing an expression of the power function $P_{K,\mathcal{X}_n}(x)$ by using determinants of matrices.

Proposition 3.1 ([9, §14.1.1], [11])**.** The power function $P_{K,\mathcal{X}_n}(x)$ in (2.13) satisfies that

$$
P_{K,\mathcal{X}_n}(x) = \left(\frac{1}{\det \mathcal{K}} \det \begin{bmatrix} K(x,x) & k(x)^T \\ k(x) & \mathcal{K} \end{bmatrix} \right)^{1/2}. \tag{3.2}
$$

Proof. The assertion is shown as follows:

$$
\begin{aligned}
P_{K,\mathcal{X}_n}(x)^2 &= K(x,x) - k(x)^T \mathcal{K}^{-1}\, k(x) \\
&= \frac{1}{\det \mathcal{K}} \det \begin{bmatrix} K(x,x) - k(x)^T \mathcal{K}^{-1}\, k(x) & k(x)^T \\ \mathbf{0} & \mathcal{K} \end{bmatrix} \\
&= \frac{1}{\det \mathcal{K}} \det \left(\begin{bmatrix} K(x,x) & k(x)^T \\ k(x) & \mathcal{K} \end{bmatrix} \begin{bmatrix} 1 & \mathbf{0} \\ -\mathcal{K}^{-1} k(x) & I \end{bmatrix} \right) \\
&= \frac{1}{\det \mathcal{K}} \det \begin{bmatrix} K(x,x) & k(x)^T \\ k(x) & \mathcal{K} \end{bmatrix}.
\end{aligned}
$$

□

[1] If K is translation-invariant, we can choose a starting point x_1 arbitrarily.

Next, for a positive integer r we define κ_r by

$$\kappa_r := \max_{\mathcal{X}_r=\{x_1,\ldots,x_r\}\subset\Omega} \det(K(x_i,x_j))_{ij}. \tag{3.3}$$

We call a maximizer providing the value of κ_r a set of Fekete type points, because it resembles the set of the Fekete points that maximizes the determinant of a Vandermonde matrix. Then, for any $x \in \Omega$ we have

$$P_{K,\mathcal{X}_n}(x)^2 \le \frac{\kappa_{n+1}}{\det \mathcal{K}}, \tag{3.4}$$

in which the square root of the RHS gives an upper bound of the worst case error in (3.1). Therefore, we consider the maximization of $\det \mathcal{K}$ with respect to $\mathcal{X}_n = \{x_1, \ldots, x_n\}$. However, because this maximization problem is still difficult, we consider

1. approximation of the determinant $\det \mathcal{K}$ by using the Mercer expansion in (2.2), and
2. approximate reduction of the maximization problem to a convex optimization problem.

We show these procedures in Sections 3.2 and 3.3, respectively.

3.2 Approximation of the determinant $\det \mathcal{K}$

Based on Proposition 3.1 and the expansion of the kernel K in (2.2), we provide an approximation of $P_{K,\mathcal{X}_n}(x)$. By truncating the expansion, we have

$$\begin{aligned}
\mathcal{K} &\approx \left(\sum_{\ell=1}^{n} \lambda_\ell\, \varphi_\ell(x_j)\, \varphi_\ell(x_i)\right)_{ij} \\
&= \begin{bmatrix} \lambda_1\varphi_1(x_1) & \lambda_2\varphi_2(x_1) & \cdots & \lambda_n\varphi_n(x_1) \\ \lambda_1\varphi_1(x_2) & \lambda_2\varphi_2(x_2) & \cdots & \lambda_n\varphi_n(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_1\varphi_1(x_n) & \lambda_2\varphi_2(x_n) & \cdots & \lambda_n\varphi_n(x_n) \end{bmatrix} \begin{bmatrix} \varphi_1(x_1) & \varphi_1(x_2) & \cdots & \varphi_1(x_n) \\ \varphi_2(x_1) & \varphi_2(x_2) & \cdots & \varphi_2(x_n) \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_n(x_1) & \varphi_n(x_2) & \cdots & \varphi_n(x_n) \end{bmatrix} \\
&= \begin{bmatrix} \varphi_1(x_1) & \varphi_2(x_1) & \cdots & \varphi_n(x_1) \\ \varphi_1(x_2) & \varphi_2(x_2) & \cdots & \varphi_n(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_1(x_n) & \varphi_2(x_n) & \cdots & \varphi_n(x_n) \end{bmatrix} \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n \end{bmatrix} \begin{bmatrix} \varphi_1(x_1) & \varphi_1(x_2) & \cdots & \varphi_1(x_n) \\ \varphi_2(x_1) & \varphi_2(x_2) & \cdots & \varphi_2(x_n) \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_n(x_1) & \varphi_n(x_2) & \cdots & \varphi_n(x_n) \end{bmatrix}.
\end{aligned} \tag{3.5}$$

Then, letting

$$\Phi_n(x_1, x_2, \ldots, x_n) = \begin{bmatrix} \varphi_1(x_1) & \varphi_1(x_2) & \cdots & \varphi_1(x_n) \\ \varphi_2(x_1) & \varphi_2(x_2) & \cdots & \varphi_2(x_n) \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_n(x_1) & \varphi_n(x_2) & \cdots & \varphi_n(x_n) \end{bmatrix}, \tag{3.6}$$

we have

$$\det \mathcal{K} \approx (\lambda_1\lambda_2\cdots\lambda_n)\left(\det \Phi_n(x_1, x_2, \ldots, x_n)\right)^2. \tag{3.7}$$

3.3 Approximate reduction to a convex optimization problem

Based on the approximation in (3.7), we consider the optimization problem

$$\text{maximize} \quad (\det \Phi_n(x_1, \ldots, x_n))^2 \qquad \text{subject to} \quad (x_1, \ldots, x_n) \in \Omega^n. \tag{3.8}$$

First, we consider the equivalent form of this problem as shown below.

Theorem 3.2. Problem (3.8) is equivalent to the problem given by

$$\begin{aligned} &\text{maximize} && \det\left(\sum_{k=1}^{n} \begin{bmatrix} \varphi_1(x_k) \\ \vdots \\ \varphi_n(x_k) \end{bmatrix} \begin{bmatrix} \varphi_1(x_k) & \cdots & \varphi_n(x_k) \end{bmatrix}\right) \\ &\text{subject to} && (x_1, \ldots, x_n) \in \Omega^n. \end{aligned} \tag{3.9}$$

Proof. The conclusion is derived from the following relations:

$$\begin{aligned} &(\det \Phi_n(x_1, \ldots, x_n))^2 \\ &= \det\left(\Phi_n(x_1, \ldots, x_n)\, \Phi_n(x_1, \ldots, x_n)^T\right) \\ &= \det\left(\begin{bmatrix} \varphi_1(x_1) & \varphi_1(x_2) & \cdots & \varphi_1(x_n) \\ \varphi_2(x_1) & \varphi_2(x_2) & \cdots & \varphi_2(x_n) \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_n(x_1) & \varphi_n(x_2) & \cdots & \varphi_n(x_n) \end{bmatrix} \begin{bmatrix} \varphi_1(x_1) & \varphi_2(x_1) & \cdots & \varphi_n(x_1) \\ \varphi_1(x_2) & \varphi_2(x_2) & \cdots & \varphi_n(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_1(x_n) & \varphi_2(x_n) & \cdots & \varphi_n(x_n) \end{bmatrix}\right) \\ &= \det\left(\left(\sum_{k=1}^{n} \varphi_i(x_k)\, \varphi_j(x_k)\right)_{ij}\right). \end{aligned} \tag{3.10}$$

□

Next, we consider approximation of Problem (3.9) because it is not easily tractable. We approximate it by preparing a sufficient number of candidate points $y_1, \ldots, y_m \in \Omega$ ($n \ll m$) and choosing n points $y_{i_1}, \ldots y_{i_n}$ which maximize

$$\det\left(\sum_{\ell=1}^{n} \begin{bmatrix} \varphi_1(y_{i_\ell}) \\ \vdots \\ \varphi_n(y_{i_\ell}) \end{bmatrix} \begin{bmatrix} \varphi_1(y_{i_\ell}) & \cdots & \varphi_n(y_{i_\ell}) \end{bmatrix}\right).$$

Then, we can rewrite this problem as follows:

$$\begin{aligned} &\text{maximize} && \det\left(\sum_{j=1}^{m} w_j \begin{bmatrix} \varphi_1(y_j) \\ \vdots \\ \varphi_n(y_j) \end{bmatrix} \begin{bmatrix} \varphi_1(y_j) & \cdots & \varphi_n(y_j) \end{bmatrix}\right) \\ &\text{subject to} && w_j \in \{0, 1\}, \;\; w_1 + \cdots + w_m = n. \end{aligned} \tag{3.11}$$

In fact, Problem (3.11) is well-known as a *D-optimal experimental design problem.*

Finally, we relax the constraint $w_j \in \{0,1\}$ of Problem (3.11) because it is still difficult. More precisely, we consider the relaxed problem given by

$$\begin{array}{ll} \text{maximize} & \det\left(\displaystyle\sum_{j=1}^{m} w_j \begin{bmatrix} \varphi_1(y_j) \\ \vdots \\ \varphi_n(y_j) \end{bmatrix} \begin{bmatrix} \varphi_1(y_j) & \cdots & \varphi_n(y_j) \end{bmatrix}\right) \\ \text{subject to} & 0 \le w_j \le 1, \;\; w_1 + \cdots + w_m = n. \end{array} \tag{3.12}$$

In fact, the objective function of this problem is log-concave with respect to a vector $w = (w_1, \ldots, w_m)^T$ (see e.g. [4]). Therefore, in principle, we can obtain an optimal solution $w^* \in [0,1]^m$ of Problem (3.12) by using a standard solver for convex optimization. Furthermore, as shown in Section 5, we can reduce the solution w^* to a 0-1 vector, which becomes an approximate solution of Problem (3.11). Taking these facts into account, in Section 4, we begin with reformulating Problem (3.12) to solve it efficiently.

Remark 3.1. In general, the matrix of Equation (3.6) can be singular for certain choices of the set $\mathcal{X}_n = \{x_1, \ldots, x_n\}$. Hence there can exist a set $\{y_1, \ldots, y_m\}$ such that the optimal value of Problem (3.11) is zero. We need to avoid such sets because it is not desirable in view of our purpose. However, we leave this issue as a theme for future work. In the results of the numerical experiments in Section 5 below, such sets do not seem to appear.

4 Reformulation by a second order cone programming (SOCP) problem

To solve Problem (3.12) efficiently, we use a reformulation of Problem (3.12) as a second order cone programming (SOCP) problem [16, 17]. In this section, we decribe this reformulation, which is proposed by Sagnol and Harman [17]. To this end, we consider the general form of the relaxed D-optimal design given by

$$\begin{array}{lll} \text{maximize} & \det\left(\displaystyle\sum_{j=1}^{m} w_j a_j a_j^T\right) & \\ \text{subject to} & 0 \le w_j \le 1 & (j = 1, \ldots, m) \\ & w_1 + \cdots + w_m = n & \end{array} \tag{4.1}$$

where $a_j \in \mathbf{R}^\ell$ $(j = 1, \ldots, m)$ for $\ell \le m$. If we set $\ell = n$ and $a_j = (\varphi_1(y_j), \ldots, \varphi_n(y_j))^T$, Problem (4.1) is reduced to Problem (3.12).

We describe a reformulation of Problem (4.1) as a SOCP problem in the same manner as that in [17]. It consists of the following two steps.

(1) Rewriting the determinant in Problem (4.1) as the optimal value of an optimization problem for a fixed $w = (w_1, \ldots, w_m)^T$.

(2) Adding the constraint of w to the optimization problem and expressing it as a SOCP problem.

In the following, we show the details of (1) and (2) in Sections 4.1 and 4.2, respectively. In fact, Problem (4.1) is a simplified case of the problem treated in [17]. Therefore, we just describe the results for the reformulation in these sections and show their proofs in Appendix A for readers' convenience.

4.1 Optimization problems yielding a determinant

Let the matrix $H \in \mathbf{R}^{\ell \times m}$ be given by

$$H = (\sqrt{w_1} a_1, \ldots, \sqrt{w_m} a_m) \tag{4.2}$$

for $w = (w_1, \ldots, w_m)^T \geq 0$. Then, the determinant in Problem (4.1) is written as $\det(HH^T)$. This determinant is given by the optimal value of an optimization problem as shown by the following theorem.

Theorem 4.1 (Special case of Theorem 4.2 in [17])**.** Let ℓ and m be integers with $\ell \leq m$ and let $H \in \mathbf{R}^{\ell \times m}$ be given by (4.2). Furthermore, let $\mathrm{OPT}_1(H)$ be the optimal value of the following optimization problem:

$$\begin{array}{lll} \underset{Q \in \mathbf{R}^{m \times \ell},\, S \in \mathbf{R}^{\ell \times \ell}}{\text{maximize}} & (\det S)^2 & \\ \text{subject to} & S \text{ is lower triangular}, & \\ & HQ = S, & \\ & \|Q\,\mathbf{e}_i\| \leq 1 & (i = 1, \ldots, \ell). \end{array} \tag{4.3}$$

Then $\mathrm{OPT}_1(H) = \det(HH^T)$.

Theorem 4.2 (Special case of Theorem 4.3 in [17])**.** Let ℓ and m be integers with $\ell \leq m$, let $a_1, \ldots, a_m \in \mathbf{R}^\ell$, and let $w = (w_1, \ldots, w_m)^T \geq 0$. Furthermore, let $\mathrm{OPT}_2(\{a_i\}, w)$ be the optimal value of the following optimization problem:

$$\begin{array}{lll} \underset{\substack{Z=(z_{ij}),\, T=(t_{ij}) \in \mathbf{R}^{m \times \ell}, \\ G=(g_{ij}) \in \mathbf{R}^{\ell \times \ell}}}{\text{maximize}} & \displaystyle\prod_{j=1}^{\ell} g_{jj} & \\ \text{subject to} & G \text{ is lower triangular}, & \\ & (a_1, \ldots, a_m)\, Z = G, & \\ & z_{ij}^2 \leq t_{ij} w_i & (i = 1, \ldots, m,\ j = 1, \ldots, \ell), \\ & \displaystyle\sum_{i=1}^{m} t_{ij} \leq g_{jj} & (j = 1, \ldots, \ell). \end{array} \tag{4.4}$$

Then $\mathrm{OPT}_2(\{a_i\}, w) = \det\left(HH^T\right)$, where H is given by (4.2).

4.2 SOCP form of Problem (4.1)

We show that Problem (4.1) can be expressed as a SOCP problem. To this end, we start with showing that the product of the variables in the objective function of Problem (4.4) is represented as the optimal value of an optimization problem with first or second order constraints.

Theorem 4.3 ([2, 10])**.** Let $\ell \geq 2$ be an integer and let $h_1, \ldots, h_\ell \in \mathbf{R}$ be non-negative numbers. Furthermore, let p be the integer with $2^{p-1} < \ell \leq 2^p$. We consider the optimization problem given by

$$\begin{array}{lll} \text{maximize} & u_1 & \\ \text{subject to} & u_i^2 \leq u_{2i} u_{2i+1} & (i = 1, \ldots, 2^{p-1} - 1), \\ & u_i^2 \leq g_{2i-2^p+1} g_{2i-2^p+2} & (i = 2^{p-1}, \ldots, 2^{p-1} + \ell/2 - 1), \\ & u_i^2 \leq u_1^2 & (i = 2^{p-1} + \ell/2, \ldots, 2^p - 1), \\ & u_i \geq 0 & (i = 1, \ldots, 2^p - 1) \end{array} \tag{4.5}$$

in the case that ℓ is even, and

$$\begin{array}{lll} \text{maximize} & u_1 & \\ \text{subject to} & u_i^2 \le u_{2i}u_{2i+1} & (i=1,\dots,2^{p-1}-1), \\ & u_i^2 \le g_{2i-2^p+1}g_{2i-2^p+2} & (i=2^{p-1},\dots,2^{p-1}+(\ell-3)/2), \\ & u_i^2 \le g_{2i-2^p+1}u_1 & (i=2^{p-1}+(\ell-1)/2), \\ & u_i^2 \le u_1^2 & (i=2^{p-1}+(\ell+1)/2,\dots,2^p-1), \\ & u_i \ge 0 & (i=1,\dots,2^p-1) \end{array} \tag{4.6}$$

in the case that ℓ is odd. Then, its optimal value is equal to the geometric mean $\left(\prod_{j=1}^{\ell} g_j\right)^{1/\ell}$.

By Theorems 4.2 and 4.3, we can show that the value $(\det(HH^T))^{1/\ell}$ is equal to the optimal value of Problem (4.4) whose objective function is rewritten in the form of Problem (4.5) or (4.6). Consequently, regarding $w_1,\dots,w_m$ as variables, we obtain the SOCP form of Problem (4.1). For simplicity, we show only the case that ℓ is even as follows:

$$\begin{array}{lll} \underset{\substack{Z=(z_{ij})\in\mathbf{R}^{m\times\ell},\\ T=(t_{ij})\in\mathbf{R}^{m\times\ell},\\ \{\tilde{g}_1,\dots,\tilde{g}_\ell\}\subset\mathbf{R},\\ \{u_1,\dots,u_{2^p-1}\}\subset\mathbf{R},\\ \{w_1,\dots,w_m\}\subset\mathbf{R}}}{\text{maximize}} & u_1 & \\ \text{subject to} & (a_1,\dots,a_m)\,Z = \begin{bmatrix} \tilde{g}_1 & 0 & \cdots & 0 \\ * & \tilde{g}_2 & \ddots & \vdots \\ * & * & \ddots & 0 \\ * & * & * & \tilde{g}_\ell \end{bmatrix}, & \\ & z_{ij}^2 \le t_{ij}w_i & (i=1,\dots,m,\ j=1,\dots,\ell), \\ & \displaystyle\sum_{i=1}^{m} t_{ij} \le \tilde{g}_j & (j=1,\dots,\ell). \\ & u_i^2 \le u_{2i}u_{2i+1} & (i=1,\dots,2^{p-1}-1), \\ & u_i^2 \le \tilde{g}_{2i-2^p+1}\tilde{g}_{2i-2^p+2} & (i=2^{p-1},\dots,2^{p-1}+\ell/2-1), \\ & u_i^2 \le u_1^2 & (i=2^{p-1}+\ell/2,\dots,2^p-1), \\ & u_i \ge 0 & (i=1,\dots,2^p-1) \\ & 0 \le w_j \le 1 & (j=1,\dots,m), \\ & w_1+\cdots+w_m = n, & \end{array} \tag{4.7}$$

whose optimal value is the ℓ-th root of that of Problem (4.1). We can similarly deal with the case that ℓ is odd. Therefore, we can obtain the optimal solution $w^* = (w_1^*,\dots,w_m^*)^T$ of Problem (4.1) by solving Problem (4.7) with a SOCP solver.

5 Algorithms and numerical experiments

In this section, we propose algorithms for generating point sets and apply them to Examples 1, 2, and 3 in Section 2.

Algorithm 1 Generation of sampling points

Input: Region $\Omega \subset \mathbf{R}^d$, Orthonormal system $\{\varphi_1, \ldots, \varphi_n\}$, Points $y_1, \ldots, y_m \in \Omega$, Neighborhood sets $\{\mathcal{N}_j\}_{j=1}^m$, Integer n.

Output: Points $x_1, \ldots, x_n \in \Omega$.

```
for j = 1 to m do
    a_j ← (φ_1(y_j), ..., φ_n(y_j))^T
end for
Solve SOCP problem (4.7) with ℓ = n to obtain the optimal solution w* = (w*_1, ..., w*_m)^T
J = ∅
for j = 1 to m do
    if w*_j ≥ w*_k for any k ∈ N_j then
        J ← J ∪ {j}
    end if
end for
Sort {w*_j}_{j∈J} in descending order: w*_{j_1} ≥ w*_{j_2} ≥ ··· ≥ w*_{j_|J|}
if |J| ≥ n then
    x_k ← y_{j_k} (k = 1, ..., n)
else
    print "Error"
end if
```

5.1 Algorithms for generating point sets

We propose two algorithms: Algorithms 1 and 2. In Algorithm 1, we solve SOCP problem (4.7) with $a_j = (\varphi_1(y_j), \ldots, \varphi_n(y_j))^T$ and $\ell = n$ to obtain the optimal solution $w^* = (w_1^*, \ldots, w_m^*)^T$. Then, we choose the "local maxima" of w^*, which mean the components w_j^* satisfying $w_j^* \geq w_k^*$ for any "neighbors" $k \in \mathcal{N}_j := \{k \mid y_k \text{ is a neighbor of } y_j \text{ in } \Omega\}$. We determine the neighbors of y_j according to the geometric property of Ω and the arrangement of $y_1, \ldots, y_m \in \Omega$ as shown in Section 5.2.

We propose Algorithm 2 to show that we can invent a sequential algorithm based on the SOCP formulation. To generate n' points, the algorithm requires a sequence of positive integers $\{n_i\}_{i=1}^I$ with $n_1 + \cdots + n_I = n'$. Then, the algorithm consists of I steps and new sampling points are added in each step. The integer n_i indicates the number of the new sampling points added in the i-th step. To achieve this procedure, in the i-th step we solve SOCP problem (4.7) with $n = n_1 + \cdots + n_i$ and the weights w_j fixed to 1 for $j \in \mathcal{W}$, where $\mathcal{W}$ is the set of the indices corresponding to the points y_j chosen in the previous steps. We can set the SOCP problems easily because we have only to add the linear constraints $w_j = 1$ $(j \in \mathcal{W})$. Algorithm 2 can be regarded as a generalization of greedy algorithms that choose points one by one.

5.2 Methods of numerical experiments for Algorithms 1 and 2

For numerical experiments, we take Examples 1, 2, and 3 in Section 2. For Example 2, we set $\gamma = 0.1$. For Example 3, we set $\alpha = \varepsilon = 1$ and consider the four versions:

3-1. $d = 1$, $\Omega = [-1, 1]$,

3-2. $d = 2$, $\Omega = [-1, 1]^2$,

3-3. $d = 2$, $\Omega = \triangle := \{(x_1, x_2) \in [-1, 1]^2 \mid x_1 + x_2 \geq 0\}$,

Algorithm 2 Sequential generation of sampling points

Input: Region $\Omega \subset \mathbf{R}^d$, Orthonormal system $\{\varphi_1, \ldots, \varphi_{n'}\}$, Points $y_1, \ldots, y_m \in \Omega$, Neighborhood sets $\{\mathcal{N}_j\}_{j=1}^m$, Sequence of positive integers $\{n_i\}_{i=1}^I$ with $n_1 + \cdots + n_I = n'$.

Output: Points $x_1, \ldots, x_{n'} \in \Omega$.

```
W = ∅
for i = 1 to I do
  n = n_1 + ··· + n_i
  for j = 1 to m do
    a_j ← (φ_1(y_j), ..., φ_n(y_j))^T
  end for
  Solve SOCP problem (4.7) with ℓ = n and the additional constraints w_j = 1 (j ∈ W) to
  obtain the optimal solution w* = (w*_1, ..., w*_m)^T
  J = ∅
  for j = 1 to m do
    if w*_j ≥ w*_k for any k ∈ N_j then
      J ← J ∪ {j}
    end if
  end for
  Sort {w*_j}_{j∈J} in descending order: w*_{j_1} ≥ w*_{j_2} ≥ ··· ≥ w*_{j_|J|}
  if |J| ≥ n then
    x_k ← y_{j_k} (k = 1, ..., n)
    W ← W ∪ {j_1, ..., j_n}
  else
    print "Error"
  end if
end for
```

3-4. $d = 2$, $\Omega = D := \{(x_1, x_2) \in [-1, 1]^2 \mid x_1^2 + x_2^2 \leq 1\}$.

We chose the candidate points $y_1, \ldots, y_m \in \Omega$ and the neighborhood sets $\mathcal{N}_j$ for these examples. For Example 1, we took $m = 250$, $y_j = (j-1)/(m-1)$ $(j = 1, \ldots, m)$ on $[0, 1]$, and $\mathcal{N}_j = \{j-1, j+1\} \cap \{1, \ldots, m\}$. For Example 2, we took $k = 25$, $m = (k-1)(k-2)+2 = 554$, and and the angles $\theta_p = \pi(p-1)/(k-1)$ $(p = 1, \ldots, k)$ and $\phi_q = 2\pi(q-1)/(k-1)$ $(q = 1, \ldots, k-1)$ to generate the points

$$y_j = (\sin\theta_{p_j}\cos\phi_{q_j},\ \sin\theta_{p_j}\sin\phi_{q_j},\ \cos\theta_{p_j}) \in S^2, \tag{5.1}$$

where

$$(p_j, q_j) = \begin{cases} (1, 1) & (j = 1), \\ (k, 1) & (j = m), \\ (\lfloor (j-2)/(k-1) \rfloor + 2,\ \mathrm{mod}(j-2, k-1) + 1) & (j \neq 1, m). \end{cases}$$

Here $\mathrm{mod}(a, b)$ denotes the remainder after division of a by b. Furthermore, we defined $\mathcal{N}_j$ by

$$\mathcal{N}_j = \begin{cases} \{2, \ldots, k\} & (j = 1), \\ \{m-(k-1), \ldots, m-1\} & (j = m), \\ \{j + l_j, j + r_j, j-(k-1), j+(k+1)\} & (j \neq 1, m), \end{cases} \tag{5.2}$$

where

$$l_j = \begin{cases} -1 & (\mathrm{mod}(j-2, k-1) \neq 0), \\ +(k-2) & (\mathrm{mod}(j-2, k-1) = 0), \end{cases}$$

$$r_j = \begin{cases} +1 & (\mathrm{mod}(j-2, k-1) \neq k-2), \\ -(k-2) & (\mathrm{mod}(j-2, k-1) = k-2). \end{cases}$$

For Example 3-1, we took $m = 250$, $y_j = -1 + 2(j-1)/(m-1)$ $(j = 1, \ldots, m)$ on $[-1, 1]$, and $\mathcal{N}_j = \{j-1, j+1\} \cap \{1, \ldots, m\}$. For Examples 3-2, 3-3, 3-4, first we took the set $\tilde{Y}_k := \{(-1 + 2(p-1)/(k-1), -1 + 2(q-1)/(k-1) \mid p = 1, \ldots, k,\ q = 1, \ldots, k\} \subset [-1, 1]^2$ for some k and then let the intersection $\tilde{Y}_k \cap \Omega$ be the set $\{y_j\}_{j=1}^m$. We chose $k = 23, 32, 27$ to obtain $m = 529, 520, 529$ for Examples 3-2, 3-3, 3-4, respectively.

For the cases with $d \geq 2$ in Examples 2 and 3, we need to decide the order of the eigenfunctions of the kernels because there are several eigenfunctions corresponding to an eigenvalue. We chose the orders as follows:

$$\begin{aligned} &\text{Example 2 } (d = 3): \\ &Y_{0,1},\ Y_{1,1}, Y_{1,2}, Y_{1,3},\ \ldots,\ Y_{n,1}, \ldots, Y_{n,2n+1},\ \ldots \end{aligned} \tag{5.3}$$

$$\begin{aligned} &\text{Example 3-2, 3-3, 3-4 } (d = 2): \\ &\varphi_1\varphi_1,\ \varphi_2\varphi_1, \varphi_1\varphi_2,\ \ldots,\ \varphi_{n-1}\varphi_1, \ldots, \varphi_1\varphi_{n-1},\ \ldots, \end{aligned} \tag{5.4}$$

where $Y_{n,\ell}$ are the spherical harmonics that appear in (2.3), φ_i are given in (2.5), and $\varphi_i\varphi_j = \varphi_i(x_1)\varphi_j(x_2)$ for $(x_1, x_2) \in \mathbf{R}^2$. Under these settings, we applied Algorithms 1 and 2 to these examples to generate sampling points. In applying Algorithm 2, we chose $\{4, 8, 12, 16, 20, 24\}$ and $\{8, 12, 16, 20, 24, 28, 32, 35\}$ as the sequence $\{n_1, \ldots, n_I\}$ for the one-dimensional examples (Examples 1 and 3-1) and the others, respectively. Then, we computed the maximum values of the power functions and the condition numbers of the kernel matrix $\mathcal{K} = (K(x_i, x_j))_{ij}$.

To implement Algorithms 1 and 2, we used MATLAB R2018b and the MOSEK optimization toolbox for MATLAB 8.1.0.56 provided by MOSEK ApS in Denmark (`https://www.mosek.com/`, last accessed on 19 October 2018). The toolbox contains a solver for SOCP problems. All computation in this section was done with the double precision floating point numbers on a computer with Intel Xeon 2.1 GHz CPU and 31.9 GB RAM. The programs used for the computation are available on the web page [23].

5.3 Results

We show the results by Figures 1–19. Figures 1–3 display the numerical solutions for the weights w^* and power functions for some representative cases. Figures 4–5 show the generated points for the higher-dimensional kernels in the case $n = 35$. One common feature of the computed weights is that most components of them are nearly zero except for those corresponding to the "local maxima" satisfying $w_j^* \geq w_k^*$ for any $j \in \mathcal{N}_j$. Therefore it is expected that Problem (3.12) given by the convex relaxation can find good approximate solution of Problem (3.11), although this phenomenon has not been proved theoretically and is not observed in some cases. Actually, we observed that it did not occur and Algorithm 1 failed in the case $n \geq 18$ for Example 3-1.

Figures 6–11 show the maximum values of the power functions given by Algorithms 1, 2, and the P-greedy algorithm. In addition, Figures 12–17 show the condition numbers of the kernel matrix $\mathcal{K} = (K(x_i, x_j))_{ij}$. For the one-dimensional examples, Examples 1 (Figures 6

and 12) and 3-1 (Figures 8 and 14), we can observe that Algorithm 1 outperforms the P-greedy algorithm for most of n, although the decay rates given by them seem to be similar. The performance of Algorithm 2 is a bit worse. For the higher-dimensional examples, in most cases the P-greedy algorithm outperforms the others, although Algorithm 2 compete with it. Algorithm 1 does not show stable performance, but it compete with the others for some n. We guess that Problem (3.12) can be precisely solved in the one-dimensional case, whereas it becomes more difficult in the higher-dimensional cases. In such cases Algorithm 2 performs better than Algorithm 1.

Remark 5.1. As n gets large, it tends to be difficult to obtain appropriate points because the weights in w^* often fails to have an apparent pattern for which n "local maxima" can be found easily. For example, it has less than n "local maxima" in some cases[2]. Even in the cases that we can find n "local maxima", the chosen points tend to be ill-balanced. The saturation of the values of the power functions and condition numbers shown respectively by Figures 8 and 14 is owing to this phenomenon. Resolution of this difficulty is one of the themes for future work.

Figures 18 and 19 show the computation times of Algorithm 1 and the P-greedy algorithm applied to Examples 1 and 2. We present only these computation times because those of Example 3-1 are similar to those of Example 1, and those of Examples 3-2, 3-3, and 3-4 are similar to those of Example 2. Recall that we used the same number $m = 250$ (the number of the candidate points) for Examples 1 and 3-1, and used the similar numbers as m for Examples 2, 3-2, 3-3, and 3-4. In each figure, the left and right graphs show the times for making the SOCP instances of Algorithm 1 and those for executing the SOCP optimizer and the P-greedy algorithm, respectively. Making an instance means constructing the structure of the MOSEK toolbox expressing the constraints of Problem (4.7) by a matrix and vectors. Its details are described in Appendix C. For the P-greedy algorithm, we plot the total time $t_1 + \cdots + t_n$ for each n, where t_i is the time for generating i-th point by the P-greedy algorithm. In addition, we omit the times for Algorithm 2 because there were little differences between Algorithms 1 and 2 for common numbers n. From these figures, we can observe that it took longer to make the SOCP instances than to solve them and the times for solving the instances are a bit longer than those for executing the P-greedy algorithm. Therefore we think that the efficiency of solving the SOCP problems is sufficient and expect that more efficient implementation of generating the instances would make our algorithms competitive with the P-greedy algorithm.

Finally, we show the results of investigation whether the points generated by the proposed algorithms are influenced by

1. the parameter(s) and
2. the order

of the eigenfunctions in the Mercer expansion. For example, the eigenfunctions for the Gaussian kernel in Example 3 contain the parameter α that can take an arbitrary positive value. It depends on the density function ρ of the space $L^2(\Omega, \rho)$ to which the eigenfunctions belong. In addition, we used the order of the eigenfunctions given by (5.3) and (5.4) for Example 2 and Examples 3-2, 3-3, 3-4, respectively. We investigate whether these factors influence the generated points by numerical experiments. To this end, we take Example 3-2 with

[2] For example, we observed that Algorithm 1 failed for $n = 100$ in the case of Example 2 because of this phenomenon.

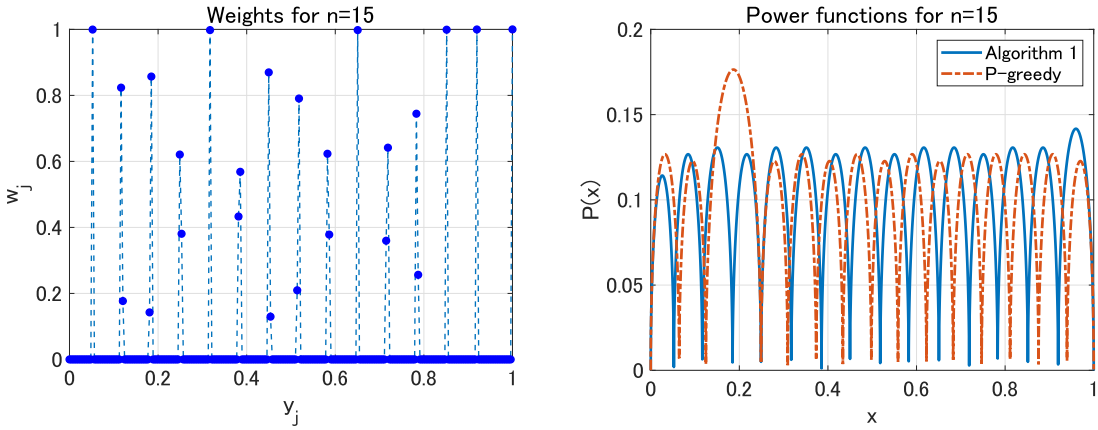


Figure 1: Weights and power functions for the Brownian kernel on $[0, 1]$ (Example 1). Left: the weights computed by Algorithm 1 for $n = 15$, Right: the power functions given by Algorithm 1 and the P-greedy algorithm for $n = 15$. Their zeros are the points given by these algorithms.

1. α replaced by 2 or
2. the order of the eigenfunctions changed as

$$\varphi_1\varphi_1,\ \varphi_1\varphi_2, \varphi_2\varphi_1,\ \ldots,\ \varphi_1\varphi_{n-1}, \ldots, \varphi_{n-1}\varphi_1,\ \ldots. \tag{5.5}$$

Then, we show the results of generating points by Algorithm 1 for $n = 35$ by Figure 20. We can observe that both factors influence the generated points.

6 Concluding remarks

We proposed Algorithms 1 and 2 generating point sets $\{x_1, \ldots x_n\}$ by using the second order cone programming (SOCP) problem in (4.7) for interpolation in reproducing kernel Hilbert spaces (RKHSs). Algorithm 1 is not a sequential algorithm in that it generates a point set by solving the SOCP problem at one sitting for a given number n. On the other hand, Algorithm 2 is a sequential one that generates n_i points in the i-th step for any sequence of positive integers $n_1, \ldots, n_L$. The SOCP problem is equivalent to the relaxed D-optimal experimental design problem in (3.12) derived from the maximization problem of the kernel matrix $\mathcal{K} = (K(x_i, x_j))_{ij}$, where K is the kernel of the RKHS. Therefore, we can regard Algorithm 1 as an algorithm yielding approximate Fekete points. In the results of the numerical experiments, we observed that the proposed algorithms compete with the P-greedy algorithm in several cases, although they are a bit time-consuming and Algorithm 1 often yields worse results. From the observation, we expect that the approximate Fekete points will also provide nearly optimal interpolation in RKHSs. To show this, further improvement of the algorithms and their theoretical analysis will be necessary.

Funding information

K. Tanaka is supported by the grant-in-aid of Japan Society of the Promotion of Science with KAKENHI Grant Number 17K14241.

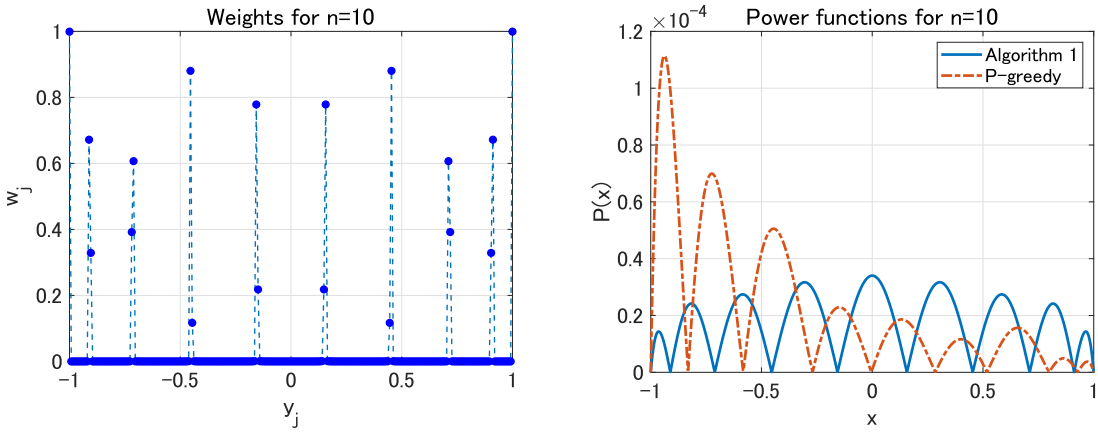


Figure 2: Weights and power functions for the Gaussian kernel on $[-1, 1]$ (Example 3-1). Left: the weights computed by Algorithm 1 for $n = 10$, Right: the power functions given by Algorithm 1 and the P-greedy algorithm for $n = 10$. Their zeros are the points given by these algorithms.

Acknowledgements

The author thanks Takanori Maehara for his valuable comment about the Matlab programs used in the numerical experiments. Thanks to the comment, much faster execution of the proposed algorithms has been realized than that in the initially submitted version of this article. Furthermore, the author gives thanks to the anonymous reviewers for their valuable suggestions about this article.

References

[1] K. M. Abadir and J. R. Magnus: Matrix Algebra. Cambridge University Press, New York, 2005.

[2] F. Alizadeh and D. Goldfarb: Second-order cone programming, Math. Program., Ser. B **95** (2003), pp. 3–51.

[3] K. Atkinson and W. Han: Spherical Harmonics and Approximations on the Unit Sphere: An Introduction. Springer-Verlag Berlin Heidelberg, 2012.

[4] J. M. Borwein and A. S. Lewis: Convex Analysis and Nonlinear Optimization: Theory and Examples, 2nd ed. Springer, New York, 2006.

[5] M. Briani, A. Sommariva, and M. Vianello: Computing Fekete and Lebesgue points: Simplex, square, disk, J. Comput. Appl. Math. **236** (2012), pp. 2477–2486.

[6] M. Buhmann: Radial basis functions, Acta Numerica **10** (2000), pp. 1–38.

[7] F. Dai and Y. Xu: Approximation Theory and Harmonic Analysis on Spheres and Balls. Springer-Verlag, New York, 2013.

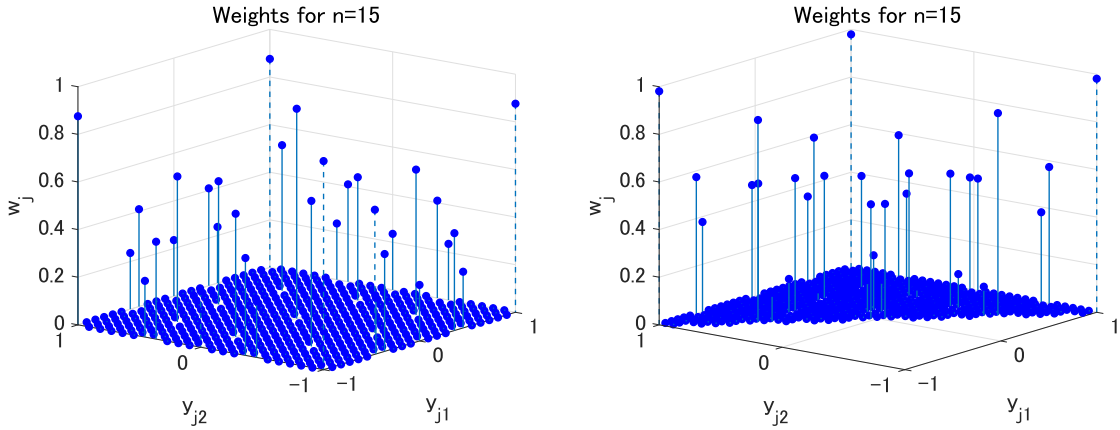


Figure 3: Weights for the Gaussian kernel on $[-1,1]^2$ and $\triangle$ (Example 3-2, 3-3). Left: the weights on $[-1,1]^2$ given by Algorithm 1 for $n = 15$, Right: the weights on $\triangle$ given by Algorithm 1 for $n = 15$.

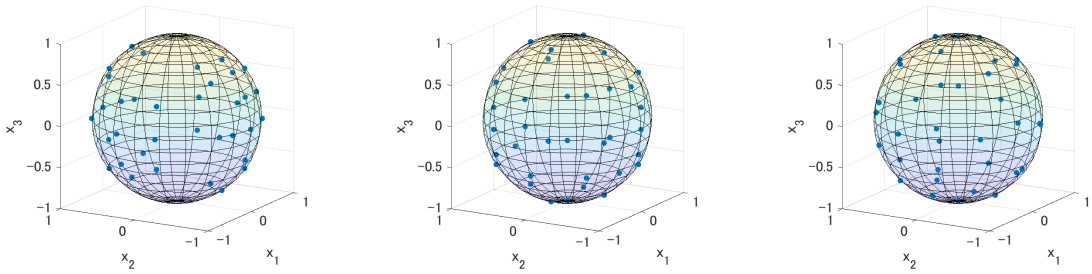


Figure 4: Generated points on the sphere for the spherical inverse multiquadric kernels (Example 2) in the case $n = 35$. Left: Algorithm 1, Middle: Algorithm 2, Right: P-greedy algorithm.

[8] G. Fasshauer: Meshfree Approximation Methods with MATLAB, World Scientific, Singapore, 2007.

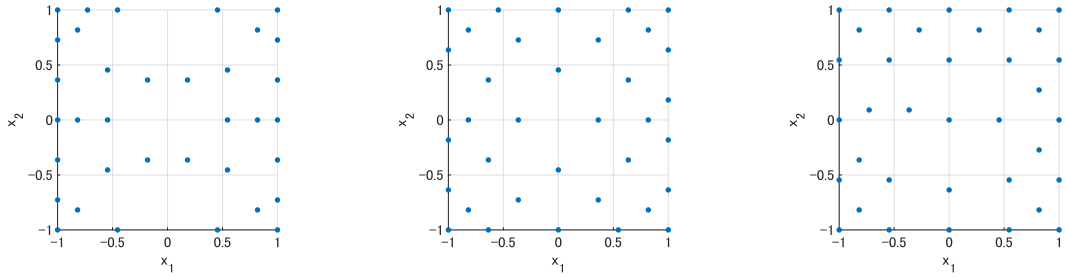
[9] G. Fasshauer and M. McCourt: Kernel-based Approximation Methods using MATLAB, World Scientific, Singapore, 2015.

[10] M. S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret: Applications of second-order cone programming. Linear Algebra Appl. **284** (1998), pp. 193–228.

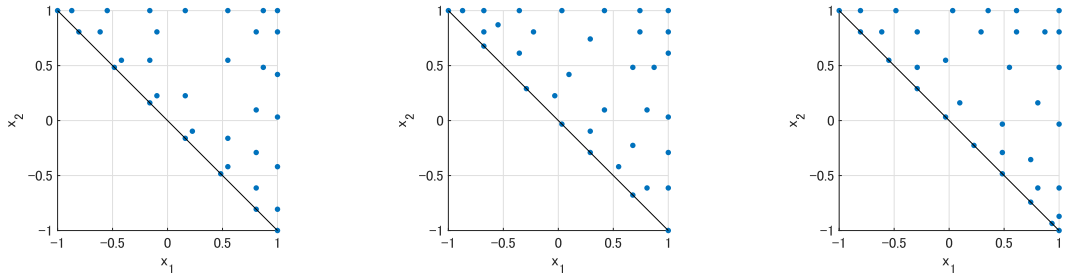
[11] S. De Marchi: On optimal center locations for radial basis function interpolation: computational aspects, Rend. Sem. Mat. Univ. Pol. Torino **61** (2003), pp. 343–358.

[12] S. De Marchi: Geometric greedy and greedy points for RBF interpolation, Proceedings of the International Conference on Computational and Mathematical Methods in Science and Engineering, CMMSE 2009, 2009.

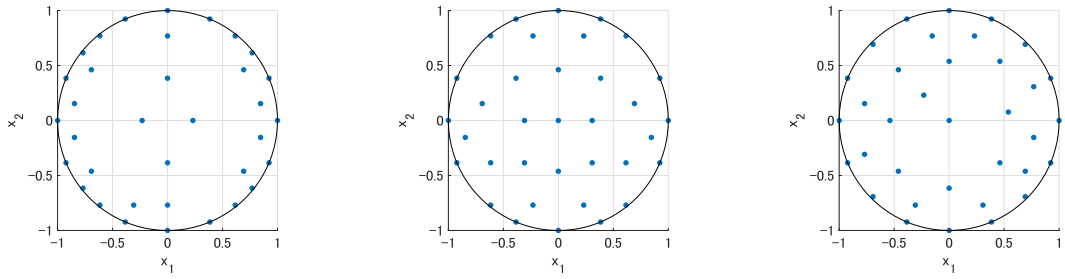
[13] S. De Marchi, R. Schaback and H. Wendland: Near-optimal data-independent point locations for radial basis function interpolation. Adv. Comput. Math. **23** (2005), pp. 317–330.



(a) Generated points on $[-1,1]^2$ (Example 3-2).



(b) Generated points on $\triangle \subset \mathbf{R}^2$ (Example 3-3).



(c) Generated points on $D \subset \mathbf{R}^2$ (Example 3-4).

Figure 5: Generated points for the Gaussian kernels (Example 3) in the case $n = 35$. Left: Algorithm 1, Middle: Algorithm 2, Right: P-greedy algorithm.

[14] V. P. Nguyen, T. Rabczuk, S. Bordas, and M. Duflot: Meshless methods: A review and computer implementation aspects. Math. Comput. Simulation **79** (2008), pp. 763–813.

[15] S. Müller: Complexity and Stability of Kernel-based Reconstructions. Dissertation, Georg-August-Universität Göttingen, Institut für Numerische und Angewandte Mathematik, Lotzestrasse 16-18, D-37083 Göttingen, Jan 2009. Göttinger Online Klassifikation: EDDF 050 (in German).

[16] G. Sangol: Computing optimal designs of multiresponse experiments reduces to second order cone programming. J. Statist. Plann. Inference **141** (2011), pp. 1684–1708.

[17] G. Sangol and R. Harman: Computing exact D-optimal designs by mixed integer second-order cone programming. The Annals of Statistics **43** (2015), pp. 2198–2224.

[18] G. Santin and R. Schaback: Approximation of eigenfunctions in kernel-based spaces. Adv. Comput. Math. **42** (2016) pp. 973–993.

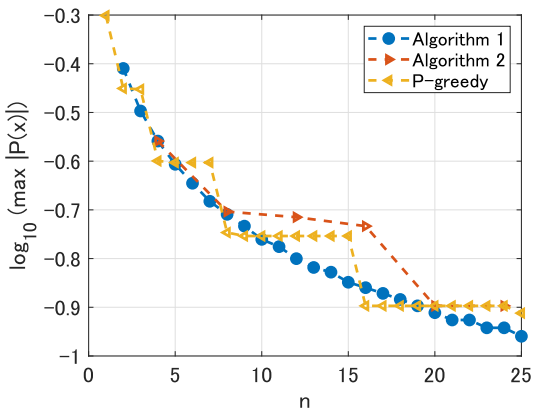


Figure 6: Maximum values of the power functions for Brownian kernel on $[0, 1]$ (Example 1).

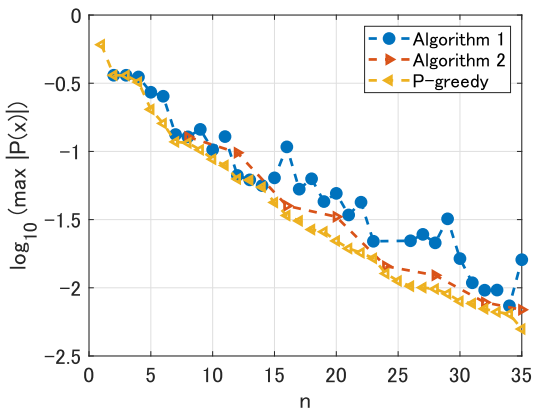


Figure 7: Maximum values of the power functions for the spherical inverse multiquadric kernel on $S^2 \subset \mathbf{R}^3$ (Example 2).

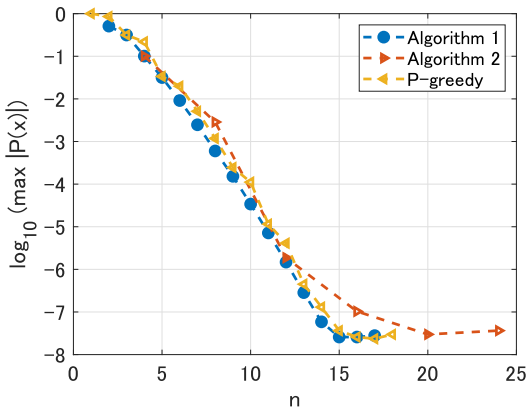


Figure 8: Maximum values of the power functions for the Gaussian kernel on $[-1, 1]$ (Example 3-1).

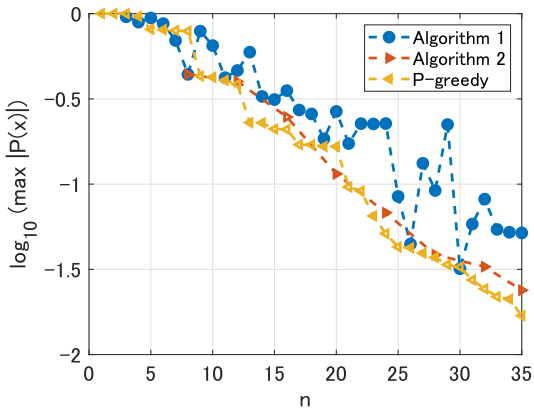


Figure 9: Maximum values of the power functions for the Gaussian kernel on $[-1, 1]^2 \subset \mathbf{R}^2$ (Example 3-2).

[19] G. Santin and B. Haasdonk: Convergence rate of the data-independent P-greedy algorithm in kernel-based approximation. Dolomites Research Notes on Approximation **10** (2017), pp. 68–78.

[20] R. Schaback and H. Wendland: Adaptive greedy techniques for approximate solution of large RBF systems, Numerical Algorithms **24** (2000) pp. 239–254.

[21] R. Schaback and H. Wendland: Kernel techniques: from machine learning to meshless methods. Acta Numerica **15** (2006), pp. 543–639.

[22] M. Singh and W. Xie: Approximate positive correlated distributions and approximation algorithms for D-optimal design. in: Proceedings of the 29th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2018), 2018.

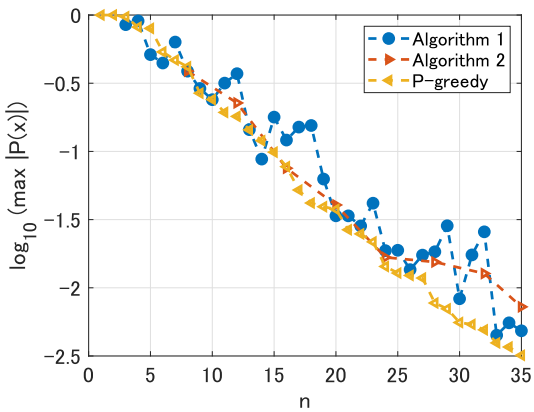


Figure 10: Maximum values of the power functions for the Gaussian kernel on $\triangle \subset \mathbf{R}^2$ (Example 3-3).

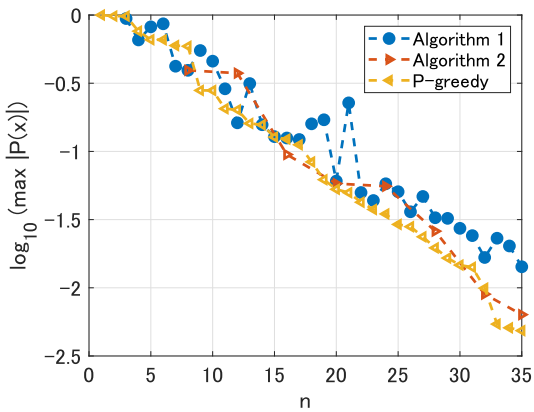


Figure 11: Maximum values of the power functions for the Gaussian kernel on $D \subset \mathbf{R}^2$ (Example 3-4).

[23] K. Tanaka: Matlab programs for generating point sets for interpolation in reproducing kernel Hilbert spaces. `https://github.com/KeTanakaN/mat_points_interp_rkhs` (last accessed on April 28, 2019)

[24] H. Wendland: Scattered Data Approximation, Cambridge University Press, Cambridge, 2005.

A Proofs of the theorems for the SOC-representablity of the D-optimal design

A.1 Proof of Theorem 4.1

Proof. Let $\tilde{Q} \in \mathbf{R}^{m\times m}$ and $\tilde{R} \in \mathbf{R}^{m\times \ell}$ be the matrices that give the QR-decomposition $H^T = \tilde{Q}\tilde{R}$. They have the following partitions:

$$\tilde{Q} = \begin{bmatrix} Q_* & \hat{Q} \end{bmatrix}, \quad \tilde{R} = \begin{bmatrix} S_*^T \\ O^T \end{bmatrix}, \tag{A.1}$$

where $Q_* \in \mathbf{R}^{m\times\ell}$, $\hat{Q} \in \mathbf{R}^{m\times(m-\ell)}$, and $S_* \in \mathbf{R}^{\ell\times\ell}$ is lower triangular. Then, $(Q, S) = (Q_*, S_*)$ is a feasible solution of Problem (4.3). In fact,

$$HQ_* = \tilde{R}^T\tilde{Q}^TQ_* = \begin{bmatrix} S_* & O \end{bmatrix} \begin{bmatrix} Q_*^T \\ \hat{Q}^T \end{bmatrix} Q_* = \begin{bmatrix} S_* & O \end{bmatrix} \begin{bmatrix} I_\ell \\ O^T \end{bmatrix} = S_*, \tag{A.2}$$

and $\|Q_*\mathbf{e}_i\| = 1$ because $\tilde{Q}$ is an orthogonal matrix. Furthermore, because

$$HH^T = \tilde{R}^T\tilde{Q}^T\tilde{Q}\tilde{R} = \tilde{R}^T\tilde{R} = S_*^TS_*, \tag{A.3}$$

we have $\det(HH^T) = \det(S_*^TS_*) = (\det S_*)^2$.

Therefore, what remains is to show that $(\det S)^2 \le \det(HH^T)$ for any feasible solution (Q, S) of Problem (4.3). This inequality holds true because we have

$$(\det S)^2 = (\det(HQ))^2 \le \det(HH^T)\det(Q^TQ) \le \det(HH^T). \tag{A.4}$$

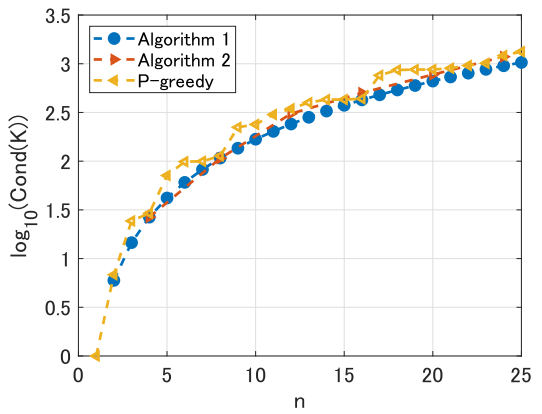


Figure 12: Condition numbers of the kernel matrices for Brownian kernel on $[0, 1]$ (Example 1).

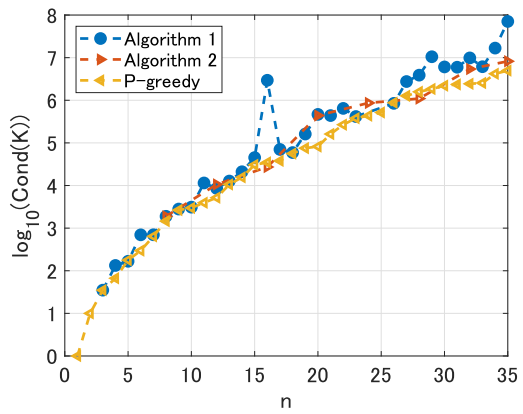


Figure 13: Condition numbers of the kernel matrices for the spherical inverse multiquadric kernel on $S^2 \subset \mathbf{R}^3$ (Example 2).

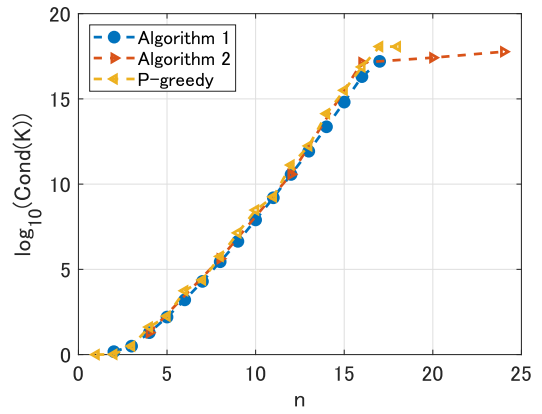


Figure 14: Condition numbers of the kernel matrices for the Gaussian kernel on $[-1, 1]$ (Example 3-1).

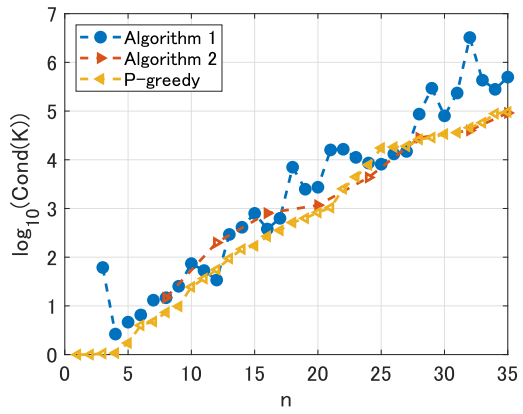


Figure 15: Condition numbers of the kernel matrices for the Gaussian kernel on $[-1, 1]^2 \subset \mathbf{R}^2$ (Example 3-2).

The first inequality in (A.4) is the Cauchy-Schwarz inequality for determinants [1, Exercise 12.15] (see Proposition B.2 in Appendix B). Moreover, the second inequality in (A.4) is derived by the Hadamard inequality [1, Exercise 12.26] and the last constraint of Problem (4.3) as follows:

$$\det(Q^T Q) \leq \prod_{i=1}^{\ell} \big(\text{The } (i, i) \text{ component of } Q^T Q\big) = \prod_{i=1}^{\ell} \|Q\mathbf{e}_i\|^2 \leq 1. \tag{A.5}$$

□

A.2 Proof of Theorem 4.2

Proof. We show that every feasible solution $(Z, T, G) = ((z_{ij}), (t_{ij}), (g_{ij}))$ of Problem (4.4) yields a feasible solution $(Q, S) = ((q_{ij}), (s_{ij}))$ of Problem (4.3) in which $g_{jj} = s_{jj}^2$ for all

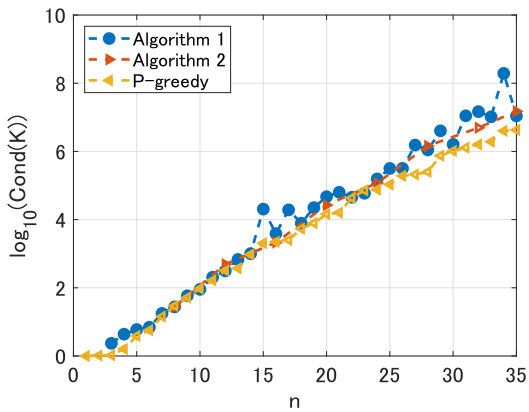


Figure 16: Condition numbers of the kernel matrices for the Gaussian kernel on $\triangle \subset \mathbf{R}^2$ (Example 3-3).

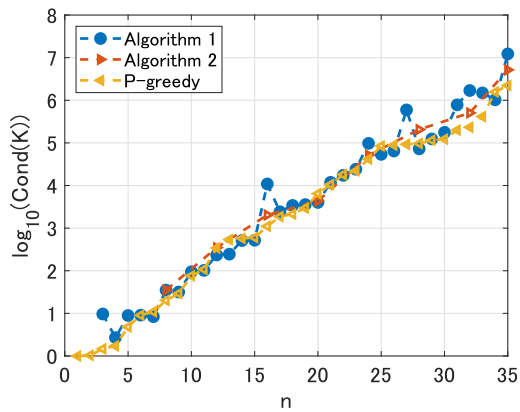


Figure 17: Condition numbers of the kernel matrices for the Gaussian kernel on $D \subset \mathbf{R}^2$ (Example 3-4).

$j = 1, \ldots, \ell$, and vice versa. If we do these, we have $\mathrm{OPT}_2(\{a_i\}, w) = \det J = (\det S)^2 = \mathrm{OPT}_1(H)$, from which the conclusion follows.

First, suppose that $(Z, T, G) = ((z_{ij}), (t_{ij}), (g_{ij}))$ is a feasible solution of Problem (4.4). Then, we define $(Q, S) = ((q_{ij}), (s_{ij}))$ by

$$q_{ij} = \begin{cases} \dfrac{z_{ij}}{\sqrt{w_i}\sqrt{g_{jj}}} & (w_i > 0,\ g_{jj} > 0), \\ 0 & (\text{otherwise}), \end{cases} \qquad s_{ij} = \begin{cases} \dfrac{g_{ij}}{\sqrt{g_{jj}}} & (g_{jj} > 0), \\ 0 & (\text{otherwise}). \end{cases}$$

Then, clearly S is lower triangular. For j with $g_{jj} > 0$, we have

$$\begin{aligned} (HQ)\,\mathbf{e}_j &= \sum_{i=1}^{m} \sqrt{w_i}\, q_{ij}\, a_i = \sum_{i:\, w_i > 0} \sqrt{w_i}\, q_{ij}\, a_i = \frac{1}{\sqrt{g_{jj}}} \sum_{i:\, w_i > 0} z_{ij}\, a_i \\ &= \frac{1}{\sqrt{g_{jj}}} \sum_{i=1}^{m} z_{ij}\, a_i = \frac{1}{\sqrt{g_{jj}}} G\,\mathbf{e}_j = S\,\mathbf{e}_j. \end{aligned}$$

For the fourth equality, we used the fact that $w_i = 0 \Rightarrow z_{ij} = 0$, which follows from the constraint $z_{ij}^2 \le t_{ij} w_i$ in Problem (4.4). For j with $g_{jj} = 0$, we have $(HQ)\,\mathbf{e}_j = \mathbf{0} = S\,\mathbf{e}_j$. Hence $HQ = S$ holds true. Furthermore, for j with $g_{jj} > 0$, we have

$$\|Q\,\mathbf{e}_j\|^2 = \sum_{i=1}^{m} q_{ij}^2 = \sum_{i:\, w_i > 0} \frac{1}{g_{jj}} \frac{z_{ij}^2}{w_i} \le \frac{1}{g_{jj}} \sum_{i:\, w_i > 0} t_{ij} \le 1,$$

and for j with $g_{jj} > 0$, we have $\|Q\,\mathbf{e}_j\|^2 = 0 \le 1$. Consequently, (Q, S) is a feasible solution of Problem (4.3) with $g_{jj} = s_{jj}^2$.

Next, suppose that $(Q, S) = ((q_{ij}), (s_{ij}))$ is a feasible solution of Problem (4.3). Then, we define $(Z, T, G) = ((z_{ij}), (t_{ij}), (g_{ij}))$ by

$$z_{ij} = \sqrt{w_i}\, s_{jj}\, q_{ij}, \qquad t_{ij} = s_{jj}^2\, q_{ij}^2, \qquad g_{ij} = s_{jj}\, s_{ij}.$$

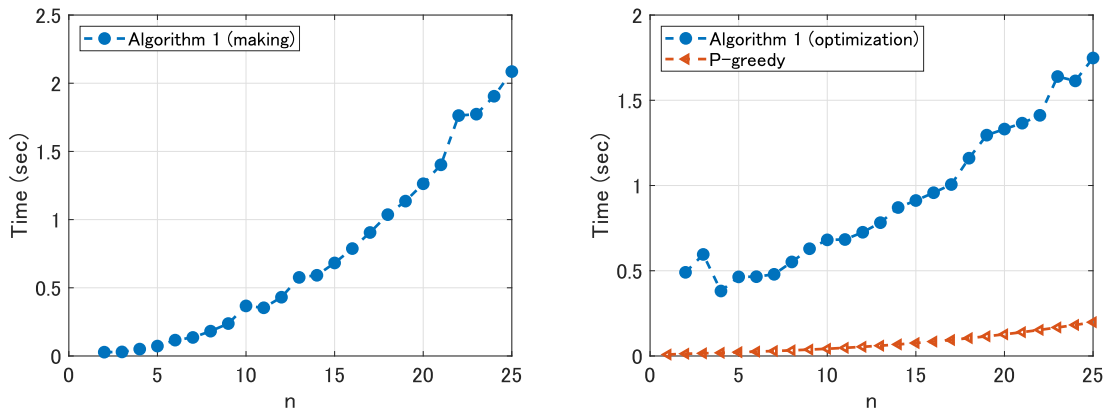


Figure 18: Computation times for Brownian kernel on $[0, 1]$ (Example 1). Left: the times for making the SOCP instances of Algorithm 1, Right: the times for executing the SOCP optimizer and the P-greedy algorithm.

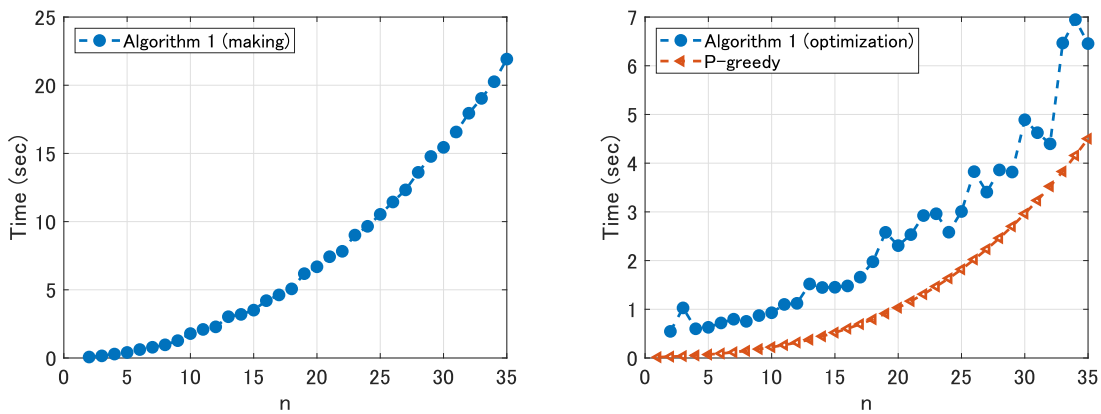


Figure 19: Computation times for the spherical inverse multiquadric kernel on S^2 (Example 2). Left: the times for making the SOCP instances of Algorithm 1, Right: the times for executing the SOCP optimizer and the P-greedy algorithm.

Then, clearly G is lower triangular. Furthermore, we have

$$(a_1, \ldots, a_m)\, Z\, \mathbf{e}_j = \sum_{i=1}^{m} a_i\, z_{ij} = s_{jj} \sum_{i=1}^{m} \sqrt{w_i}\, a_i\, q_{ij} = s_{jj} HQ\, \mathbf{e}_j = s_{jj} S\, \mathbf{e}_j = G\, \mathbf{e}_j,$$

$z_{ij}^2 = w_i\, s_{jj}^2\, q_{ij}^2 = w_i\, t_{ij}$, and

$$\sum_{i=1}^{m} t_{ij} = s_{jj}^2 \sum_{i=1}^{m} q_{ij}^2 \leq s_{jj}^2 = g_{jj}.$$

Therefore, (Z, T, G) is a feasible solution of Problem (4.4) with $s_{jj}^2 = g_{jj}$. $\square$

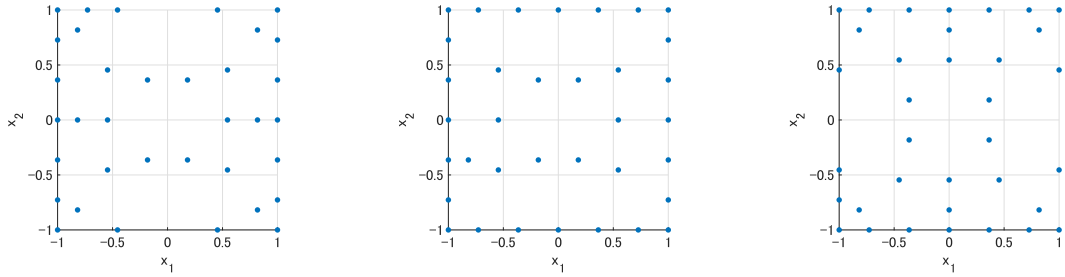


Figure 20: Generated points on $[-1,1]^2$ for the Gaussian kernels (Example 3-2) in the case $n = 35$. Left: the same picture as the left one of Figure 5 (a) (for comparison), Middle: the points for $\varepsilon = 1$, $\alpha = 2$, and the order of the eigenfunctions given by (5.4), Right: the points for $\varepsilon = 1$, $\alpha = 1$, and the order of the eigenfunctions given by (5.5).

A.3 Proof of Theorem 4.3

Proof. We show the assertion only in the case that ℓ is even, because the other case can be shown similarly. If $g_j = 0$ for some j, we can deduce that $u_1 = 0$. Therefore, it suffices to consider the case that $g_j > 0$ for $j = 1, \dots, \ell$.

First, we show that $u_1 \le \left(\prod_{j=1}^{\ell} g_j\right)^{1/\ell}$ for any feasible solution $\{u_i\}$. If $u_1 = 0$, the conclusion is trivial. Then, we assume that $u_1 > 0$. From the constraints with $1 \le i \le 2^{p-1} - 1$ in (4.5), we have

$$\begin{array}{ccccc}
 & u_1^{2^p} \le u_2^{2^{p-1}} u_3^{2^{p-1}}, & & & \\
u_2^{2^{p-1}} \le u_4^{2^{p-2}} u_5^{2^{p-2}}, & & u_3^{2^{p-1}} \le u_6^{2^{p-2}} u_7^{2^{p-2}}, & & \\
\iddots & & \vdots & & \ddots \\
u_{2^{p-2}}^{2^2} \le u_{2^{p-1}}^2 u_{2^{p-1}+1}^2, & u_{2^{p-2}+1}^{2^2} \le u_{2^{p-1}+2}^2 u_{2^{p-1}+3}^2, & & \cdots & u_{2^{p-1}-1}^{2^2} \le u_{2^p-2}^2 u_{2^p-1}^2.
\end{array}$$

Then, taking the product of all these inequalities, we have

$$u_1^{2^p} \le \prod_{i=2^{p-1}}^{2^p-1} u_i^2. \tag{A.6}$$

Similarly, taking the product of all the constraints with $2^{p-1} \le i \le 2^p - 1$, we have

$$\prod_{i=2^{p-1}}^{2^p-1} u_i^2 \le \left(\prod_{j=1}^{\ell} g_j\right) u_1^{2^p-\ell}. \tag{A.7}$$

Therefore, it follows from (A.6) and (A.7) that

$$u_1^{2^p} \le \left(\prod_{j=1}^{\ell} g_j\right) u_1^{2^p-\ell} \iff u_1 \le \left(\prod_{j=1}^{\ell} g_j\right)^{1/\ell}.$$

Next, we show that there exists a feasible solution $\{u_i\}$ with $u_1 = \left(\prod_{j=1}^{\ell} g_j\right)^{1/\ell}$. Set u_1 by this equality and set the other variables by

$$\begin{array}{ll} u_i = u_1 & (i = 2^{p-1} + \ell/2, \ldots, 2^p - 1), \\ u_i = \sqrt{g_{2i-2^p+1} g_{2i-2^p+2}} & (i = 2^{p-1}, \ldots, 2^{p-1} + \ell/2 - 1), \\ u_i = \sqrt{u_{2i} u_{2i+1}} & (i = 2, \ldots, 2^{p-1} - 1). \end{array}$$

Then, in a similar manner to that for deriving (A.6) and (A.7), we have

$$u_2^{2^{p-1}} u_3^{2^{p-1}} = \left(\prod_{j=1}^{\ell} g_j\right) u_1^{2^p - \ell} = u_1^{\ell} u_1^{2^p - \ell} = u_1^{2^p}, \tag{A.8}$$

which guarantees the solution $\{u_i\}$ is feasible. □

B Cauchy-Schwarz inequality for determinants

Lemma B.1 ([1, Exercise 12.7])**.** Let ℓ be a non-negative integer and let $C, D \in \mathbf{R}^{\ell \times \ell}$ be $\ell \times \ell$ positive semi-definite matrices. Then, the following inequality holds true:

$$\det(C + D) \geq \det C + \det D.$$

Proof. If C and D are singular, the inequality is trivial. Therefore, we assume that C is non-singular without loss of generality. Let $S := C^{-1/2} D C^{-1/2}$. Then, because

$$\begin{aligned} \det(C + D) - \det C - \det D &= \det(C^{1/2}(I + S)C^{1/2}) - \det C - \det(C^{1/2} S C^{1/2}) \\ &= \det C \, (\det(I + S) - 1 - \det S), \end{aligned}$$

it suffices to show that $\det(I + S) - 1 - \det S \geq 0$ for any $S \in \mathbf{R}^{\ell \times \ell}$ with $S \succeq O$. By letting $\lambda_1, \ldots, \lambda_\ell \geq 0$ be the eigenvalues of S, this inequality is reduced to

$$\prod_{i=1}^{\ell} (1 + \lambda_i) - 1 - \prod_{i=1}^{\ell} \lambda_i \geq 0,$$

which can be proved by expanding the product $\prod_{i=1}^{\ell}(1 + \lambda_i)$. □

Proposition B.2 (Cauchy-Schwarz inequality for determinants [1, Exercise 12.15])**.** Let ℓ and m be non-negative integers with $\ell \leq m$ and let $A, B \in \mathbf{R}^{m \times \ell}$ be $m \times \ell$ matrices. Then, the following inequality holds true:

$$(\det(A^T B))^2 \leq \det(A^T A) \det(B^T B). \tag{B.1}$$

Proof. If $\det(A^T B) = 0$, the inequality is trivial. Therefore, we assume that $\det(A^T B) \neq 0$. Then, A and B have full column rank, which implies that $A^T A$ and $B^T B$ are non-singular. Let $C, D \in \mathbf{R}^{\ell \times \ell}$ be given by

$$\begin{aligned} C &:= B^T A \, (A^T A)^{-1} A^T B, \\ D &:= B^T B - C = B^T (I - A \, (A^T A)^{-1} A^T) B. \end{aligned}$$

Then, clearly $C \succ O$ holds true. Furthermore, because $P := I - A \, (A^T A)^{-1} A^T$ satisfies $P^2 = P$, we have $P \succeq O$ and hence $D \succeq O$. Then, using Lemma B.1, we have

$$\det(B^T B) = \det(C + D) \geq \det C + \det D \geq \det C = (\det(A^T B))^2 \, (\det A^T A)^{-1},$$

which is equivalent to (B.1). □

C Remarks on implementation of SOCP (4.7) by MOSEK

C.1 MOSEK

As a software to solve SOCP problems, we use the MOSEK Optimization Toolbox for MATLAB 8.1.0.56, which is provided by MOSEK ApS in Denmark (`https://www.mosek.com/`, last accessed on 19 October 2018).

C.1.1 Expression of rotated second order cones

Recall that $p = \lceil \log_2 \ell \rceil$. In SOCP (4.7) in which ℓ is even, we need to consider the rotated second order cone constraints

$$z_{ij}^2 \le t_{ij} w_i \qquad (i = 1, \dots, m, j = 1, \dots, \ell), \tag{C.1}$$

$$u_i^2 \le u_{2i} u_{2i+1} \qquad (i = 1, \dots, 2^{p-1} - 1), \tag{C.2}$$

$$u_i^2 \le \tilde{g}_{2i-2^p+1} \tilde{g}_{2i-2^p+2} \qquad (i = 2^{p-1}, \dots, 2^{p-1} + \ell/2 - 1). \tag{C.3}$$

Furthermore, in the counterpart in which ℓ is odd, we need to consider the constraints

$$z_{ij}^2 \le t_{ij} w_i \qquad (i = 1, \dots, m, j = 1, \dots, \ell), \tag{C.4}$$

$$u_i^2 \le u_{2i} u_{2i+1} \qquad (i = 1, \dots, 2^{p-1} - 1), \tag{C.5}$$

$$u_i^2 \le \tilde{g}_{2i-2^p+1} \tilde{g}_{2i-2^p+2} \qquad (i = 2^{p-1}, \dots, 2^{p-1} + (\ell - 3)/2), \tag{C.6}$$

$$u_i^2 \le \tilde{g}_{2i-2^p+1} u_1 \qquad (i = 2^{p-1} + (\ell - 1)/2). \tag{C.7}$$

However, MOSEK permits only the expression $\eta_1^2 + \cdots + \eta_N^2 \le 2\xi\zeta$ for a rotated second order cone, where we just need the case that $N = 1$. Therefore, we employ the following variable transformations:

$$w_i = 2\hat{w}_i \quad (i = 1, \dots, m), \tag{C.8}$$

$$\tilde{g}_i = \sqrt{2}\hat{g}_i \quad (i = 1, \dots, \ell - 1), \tag{C.9}$$

$$\tilde{g}_\ell = \begin{cases} \sqrt{2}\hat{g}_\ell & (\ell \text{ is even}), \\ 2\hat{g}_\ell & (\ell \text{ is odd}). \end{cases} \tag{C.10}$$

We do not change u_i because of the following reason. If we change the constraints $u_i^2 \le u_{2i} u_{2i+1}$ to $u_i^2 \le 2u_{2i} u_{2i+1}$, they just change the optimal value by $2^{(p-1)2^{p-1}}$ times[3]. Consequently, we

[3]Therefore the optimal value of SOCP (4.7) is $2^{(p-1)2^{p-1}}$(The optimal value of Problem (4.1))$^{1/\ell}$.

deal with the SOCP problem

$$
\begin{array}{lll}
\underset{\substack{\{u_1,\ldots,u_{2^p-1}\}\subset\mathbf{R},\\ \{\hat{g}_1,\ldots,\hat{g}_\ell\}\subset\mathbf{R},\\ Z=(z_{ij})\in\mathbf{R}^{m\times\ell},\\ T=(t_{ij})\in\mathbf{R}^{m\times\ell},\\ \{\hat{w}_1,\ldots,\hat{w}_m\}\subset\mathbf{R}}}{\text{maximize}} & u_1 & \\
\text{subject to} & (a_1,\ldots,a_m)\,Z = \begin{bmatrix} \sqrt{2}\hat{g}_1 & 0 & \cdots & 0 \\ * & \sqrt{2}\hat{g}_2 & \ddots & \vdots \\ * & * & \ddots & 0 \\ * & * & * & \sqrt{2}\hat{g}_\ell \end{bmatrix}, & \\
& \hat{w}_1+\cdots+\hat{w}_m = n/2, & \\
& \displaystyle\sum_{i=1}^{m} t_{ij} \le \sqrt{2}\hat{g}_j & (j=1,\ldots,\ell), \\
& u_i^2 \le u_1^2 & (i=2^{p-1}+\ell/2,\ldots,2^p-1), \\
\hline
& u_i \ge 0 & (i=1,\ldots,2^p-1), \\
& 0 \le \hat{w}_j \le 1/2 & (j=1,\ldots,m), \\
\hline
& u_i^2 \le 2u_{2i}u_{2i+1} & (i=1,\ldots,2^{p-1}-1), \\
& u_i^2 \le 2\hat{g}_{2i-2^p+1}\hat{g}_{2i-2^p+2} & (i=2^{p-1},\ldots,2^{p-1}+\ell/2-1), \\
& z_{ij}^2 \le 2t_{ij}\hat{w}_i & (i=1,\ldots,m,\ j=1,\ldots,\ell),
\end{array}
\tag{C.11}
$$

in the case that ℓ is even, and

$$
\begin{array}{lll}
\underset{\substack{\{u_1,\ldots,u_{2^p-1}\}\subset\mathbf{R},\\ \{\hat{g}_1,\ldots,\hat{g}_\ell\}\subset\mathbf{R},\\ Z=(z_{ij})\in\mathbf{R}^{m\times\ell},\\ T=(t_{ij})\in\mathbf{R}^{m\times\ell},\\ \{\hat{w}_1,\ldots,\hat{w}_m\}\subset\mathbf{R}}}{\text{maximize}} & u_1 & \\
\text{subject to} & (a_1,\ldots,a_m)\,Z = \begin{bmatrix} \sqrt{2}\hat{g}_1 & 0 & \cdots & 0 \\ * & \sqrt{2}\hat{g}_2 & \ddots & \vdots \\ * & * & \ddots & 0 \\ * & * & * & 2\hat{g}_\ell \end{bmatrix}, & \\
& \hat{w}_1+\cdots+\hat{w}_m = n/2, & \\
& \displaystyle\sum_{i=1}^{m} t_{ij} \le \sqrt{2}\hat{g}_j & (j=1,\ldots,\ell-1), \\
& \displaystyle\sum_{i=1}^{m} t_{i\ell} \le 2\hat{g}_\ell, & \\
& u_i^2 \le u_1^2 & (i=2^{p-1}+(\ell+1)/2,\ldots,2^p-1), \\
\hline
& u_i \ge 0 & (i=1,\ldots,2^p-1), \\
& 0 \le \hat{w}_j \le 1/2 & (j=1,\ldots,m), \\
\hline
& u_i^2 \le 2u_{2i}u_{2i+1} & (i=1,\ldots,2^{p-1}-1), \\
& u_i^2 \le 2\hat{g}_{2i-2^p+1}\hat{g}_{2i-2^p+2} & (i=2^{p-1},\ldots,2^{p-1}+(\ell-3)/2), \\
& u_i^2 \le 2\hat{g}_{2i-2^p+1}u_1 & (i=2^{p-1}+(\ell-1)/2), \\
& z_{ij}^2 \le 2t_{ij}\hat{w}_i & (i=1,\ldots,m,\ j=1,\ldots,\ell),
\end{array}
\tag{C.12}
$$

in the case that ℓ is odd.

C.1.2 Restriction about cone constraints

MOSEK does not permit that a variable belongs to plural cones. However, we need to consider the cone constraints in (C.11) and (C.12) in which each u_i appears 2 times and each $\hat{w}_i$ appears ℓ times. Therefore we prepare the copies of these variables:

$$u_i \;\leftrightarrow\; u_{i1}, u_{i2}, \qquad \hat{w}_i \;\leftrightarrow\; \hat{w}_{i1}, \hat{w}_{i2}, \ldots, \hat{w}_{i\ell},$$

and set the linear constraints between these variables:

$$u_{i1} = u_{i2}, \tag{C.13}$$
$$\hat{w}_{i1} = \hat{w}_{i2} = \cdots = \hat{w}_{i\ell}. \tag{C.14}$$

Then, we introduce a vector

$$\begin{aligned} v = (&u_{11}, u_{12},\ u_{21}, u_{22},\ \ldots,\ u_{(2^p-1),1}, u_{(2^p-1),2}; \\ &\hat{g}_1, \hat{g}_2, \ldots, \hat{g}_\ell; \\ &z_{11}, \ldots, z_{m1},\ z_{12}, \ldots, z_{m2},\ \ldots,\ z_{1\ell}, \ldots, z_{m\ell}; \\ &t_{11}, \ldots, t_{m1},\ t_{12}, \ldots, t_{m2},\ \ldots,\ t_{1\ell}, \ldots, t_{m\ell}; \\ &\hat{w}_{11}, \ldots, \hat{w}_{m1},\ \hat{w}_{12}, \ldots, \hat{w}_{m2},\ \ldots,\ \hat{w}_{1\ell}, \ldots, \hat{w}_{m\ell})^T. \end{aligned} \tag{C.15}$$

C.2 Linear constraints

To describe the linear constraints in (C.11) and (C.12) in terms of the vector v, we provide a matrix A and vectors b_{l} and b_{u} that express the constraints as $b_{\mathrm{l}} \le Av \le b_{\mathrm{u}}$.

We begin with the case that ℓ is even. First, we express the constraints in (C.13) and (C.14) by

$$\underbrace{\begin{bmatrix} 1 & -1 & 0 & 0 & \cdots & 0 & 0 & 0^T_{\ell+3m\ell} \\ 0 & 0 & 1 & -1 & \cdots & 0 & 0 & 0^T_{\ell+3m\ell} \\ \vdots & & & & \ddots & & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 & -1 & 0^T_{\ell+3m\ell} \end{bmatrix}}_{A_1} v = 0_{2^p-1}, \tag{C.16}$$

and

$$\underbrace{\begin{bmatrix} O_{m,2(2^p-1)+\ell+2m\ell} & I_m & -I_m & O_{m,m} & \cdots & O_{m,m} \\ O_{m,2(2^p-1)+\ell+2m\ell} & O_{m,m} & I_m & -I_m & & O_{m,m} \\ \vdots & & & \ddots & & \\ O_{m,2(2^p-1)+\ell+2m\ell} & O_{m,m} & \cdots & O_{m,m} & I_m & -I_m \end{bmatrix}}_{A_2} v = 0_{m(\ell-1)}, \tag{C.17}$$

respectively. Next, we consider the constraint

$$(a_1, \ldots, a_m)\, Z = \begin{bmatrix} \sqrt{2}\hat{g}_1 & 0 & \cdots & 0 \\ * & \sqrt{2}\hat{g}_2 & \ddots & \vdots \\ * & * & \ddots & 0 \\ * & * & * & \sqrt{2}\hat{g}_\ell \end{bmatrix}.$$

By using v and the expression

$$(a_1, \ldots, a_m) = \begin{bmatrix} \boldsymbol{\alpha}_1 & \cdots & \boldsymbol{\alpha}_\ell \end{bmatrix}^T \qquad (\boldsymbol{\alpha}_j \in \mathbf{R}^m),$$

we express the linear constraint as follows:

$$\underbrace{\left[\begin{array}{c|cccccc|cccc|c} 0^T_{2(2^p-1)} & -\sqrt{2} & 0 & 0 & \cdots & 0 & 0 & \boldsymbol{\alpha}_1^T & 0^T_m & \cdots & 0^T_m & 0^T_{2m\ell} \\ 0^T_{2(2^p-1)} & 0 & 0 & 0 & \cdots & 0 & 0 & 0^T_m & \boldsymbol{\alpha}_1^T & & 0^T_m & 0^T_{2m\ell} \\ \vdots & & & \vdots & & \vdots & \vdots & & & \ddots & \vdots & \vdots \\ 0^T_{2(2^p-1)} & 0 & 0 & 0 & \cdots & 0 & 0 & 0^T_m & \cdots & 0^T_m & \boldsymbol{\alpha}_1^T & 0^T_{2m\ell} \\ \hline 0^T_{2(2^p-1)} & 0 & -\sqrt{2} & 0 & \cdots & 0 & 0 & 0^T_m & \boldsymbol{\alpha}_2^T & & 0^T_m & 0^T_{2m\ell} \\ \vdots & & & \vdots & & \vdots & \vdots & & & \ddots & \vdots & \vdots \\ 0^T_{2(2^p-1)} & 0 & 0 & 0 & \cdots & 0 & 0 & 0^T_m & \cdots & 0^T_m & \boldsymbol{\alpha}_2^T & 0^T_{2m\ell} \\ \hline \vdots & & & \vdots & & \vdots & \vdots & & & & \vdots & \vdots \\ \hline 0^T_{2(2^p-1)} & 0 & 0 & 0 & \cdots & 0 & -\sqrt{2} & 0^T_m & \cdots & 0^T_m & \boldsymbol{\alpha}_\ell^T & 0^T_{2m\ell} \end{array}\right]}_{A_3} v = 0_{\ell(\ell+1)/2}. \tag{C.18}$$

Next, we express the constraint $\hat{w}_{1\ell} + \cdots + \hat{w}_{m\ell} = n/2$ by

$$\underbrace{\begin{bmatrix} 0^T_{2(2^p-1)+\ell+3m(\ell-1)} & 1^T_m \end{bmatrix}}_{A_4} v = \frac{n}{2}. \tag{C.19}$$

Next, we express the inequality constraints

$$\sum_{i=1}^{m} t_{ij} \le \sqrt{2}\hat{g}_j \quad (j = 1, \ldots, \ell)$$

by

$$\underbrace{\left[\begin{array}{c|cccc|c|cccc|c} 0^T_{2(2^p-1)} & -\sqrt{2} & 0 & \cdots & 0 & 0^T_{m\ell} & 1^T_m & 0^T_m & \cdots & 0^T_m & 0^T_{m\ell} \\ 0^T_{2(2^p-1)} & 0 & -\sqrt{2} & \cdots & 0 & 0^T_{m\ell} & 0^T_m & 1^T_m & \cdots & 0^T_m & 0^T_{m\ell} \\ \vdots & \vdots & & \ddots & \vdots & \vdots & \vdots & & \ddots & \vdots & \vdots \\ 0^T_{2(2^p-1)} & 0 & \cdots & 0 & -\sqrt{2} & 0^T_{m\ell} & 0^T_m & \cdots & 0^T_m & 1^T_m & 0^T_{m\ell} \end{array}\right]}_{A_5} v \le 0_\ell. \tag{C.20}$$

Finally, we consider the constraints

$$u_i^2 \le u_1^2 \qquad (i = 2^{p-1} + \ell/2, \ldots, 2^p - 1), \tag{C.21}$$

which are equivalent to the linear constraints $u_i \le u_1$ owing to the non-negative constraints $u_i \ge 0$. These constraints are expressed by

$$\underbrace{\left[\begin{array}{cc|c|cccccc|c} -1 & 0 & 0^T_{2^p+\ell-4} & 1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0^T_{\ell+3m\ell} \\ -1 & 0 & 0^T_{2^p+\ell-4} & 0 & 0 & 1 & 0 & \cdots & 0 & 0 & 0^T_{\ell+3m\ell} \\ \vdots & \vdots & \vdots & \vdots & \vdots & & & \ddots & \vdots & \vdots & \vdots \\ -1 & 0 & 0^T_{2^p+\ell-4} & 0 & 0 & 0 & 0 & \cdots & 1 & 0 & 0^T_{\ell+3m\ell} \end{array}\right]}_{A_6} v \le 0_{2^{p-1}-\ell/2}. \tag{C.22}$$

Consequently, by setting

$$A = \begin{bmatrix} A_1^T & A_2^T & A_3^T & A_4^T & A_5^T & A_6^T \end{bmatrix}^T, \tag{C.23}$$

$$b_{\mathrm{l}} = \left(0^T_{2^p-1},\ 0^T_{m(\ell-1)},\ 0^T_{\ell(\ell+1)/2},\ n/2,\ -\infty \cdot 1^T_{\ell},\ -\infty \cdot 1_{2^{p-1}-\ell/2}\right)^T, \tag{C.24}$$

$$b_{\mathrm{u}} = \left(0^T_{2^p-1},\ 0^T_{m(\ell-1)},\ 0^T_{\ell(\ell+1)/2},\ n/2,\ 0^T_{\ell},\ 0^T_{2^{p-1}-\ell/2}\right)^T, \tag{C.25}$$

we can express the linear constraints as $b_{\mathrm{l}} \le Av \le b_{\mathrm{u}}$.

In the case that ℓ is odd, we need to modify A_3, A_5, A_6, b_{l}, and b_{u}. We replace the component "$-\sqrt{2}$" in the last rows of A_3 and A_5 by "-2". Furthermore, We modify A_6, b_{l}, and b_{u} as

$$A_6 = \left.\left[\begin{array}{cc|c|cccccc|c} -1 & 0 & 0^T_{2^p+\ell-3} & 1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0^T_{\ell+3m\ell} \\ -1 & 0 & 0^T_{2^p+\ell-3} & 0 & 0 & 1 & 0 & \cdots & 0 & 0 & 0^T_{\ell+3m\ell} \\ \vdots & \vdots & \vdots & \vdots & \vdots & & & \ddots & \vdots & \vdots & \vdots \\ -1 & 0 & 0^T_{2^p+\ell-3} & 0 & 0 & 0 & 0 & \cdots & 1 & 0 & 0^T_{\ell+3m\ell} \end{array}\right]\right\} 2^{p-1} - (\ell+1)/2, \tag{C.26}$$

$$b_{\mathrm{l}} = \left(0^T_{2^p-1},\ 0^T_{m(\ell-1)},\ 0^T_{\ell(\ell+1)/2},\ n/2,\ -\infty \cdot 1^T_{\ell},\ -\infty \cdot 1_{2^{p-1}-(\ell+1)/2}\right)^T, \tag{C.27}$$

$$b_{\mathrm{u}} = \left(0^T_{2^p-1},\ 0^T_{m(\ell-1)},\ 0^T_{\ell(\ell+1)/2},\ n/2,\ 0^T_{\ell},\ 0^T_{2^{p-1}-(\ell+1)/2}\right)^T, \tag{C.28}$$

respectively.

C.3 Constraints for the ranges of the variables

Besides the constraints $u_i \ge 0$ and $0 \le \hat{w}_j \le 1/2$ in (C.11) and (C.12), we can add $\hat{g}_i \ge 0$ and $t_{ij} \ge 0$ because any solution that does not satisfy these constraints is useless. Therefore by using the vectors given by

$$\tilde{b}_{\mathrm{l}} = \left(0^T_{2(2^p-1)+\ell},\ -\infty \cdot 1^T_{m\ell},\ 0^T_{2m\ell}\right)^T, \tag{C.29}$$

$$\tilde{b}_{\mathrm{u}} = \left(\infty \cdot 1^T_{2(2^p-1)+\ell+2m\ell},\ (1/2) \cdot 1^T_{m\ell}\right)^T, \tag{C.30}$$

we can express the constraints for the ranges of the variables as $\tilde{b}_{\mathrm{l}} \le v \le \tilde{b}_{\mathrm{u}}$.

C.4 Cone constraints

To express the cone constraints in (C.11) and (C.12) by MOSEK, we need to specify the components of the vector v that correspond to the constraints. For the constraints

$$u_i^2 \le 2u_{2i}u_{2i+1} \qquad (i = 1, \ldots, 2^{p-1} - 1),$$

we take the components v_{2i}, $v_{2(2i)-1}$, and $v_{2(2i+1)-1}$ for u_i, u_{2i}, and u_{2i+1}, respectively. For the constraints

$$u_i^2 \leq 2\hat{g}_{2i-2^p+1}\hat{g}_{2i-2^p+2} \qquad \begin{cases} (i = 2^{p-1}, \ldots, 2^{p-1} + \ell/2 + 1) & (\ell \text{ is even}), \\ (i = 2^{p-1}, \ldots, 2^{p-1} + (\ell - 3)/2) & (\ell \text{ is odd}), \end{cases}$$

we take the components v_{2i}, v_{2^p+2i-1}, and v_{2^p+2i} for u_i, $\hat{g}_{2i-2^p+1}$, and $\hat{g}_{2i-2^p+2}$, respectively. In the case that ℓ is odd, we take the components $v_{2^p+\ell-1}$, $v_{2(2^p-1)+\ell}$, and v_1 for the constraint $u^2_{2^{p-1}+(\ell-1)/2} \leq 2\hat{g}_\ell u_1$. Finally, for the constraints

$$z_{ij}^2 \leq 2t_{ij}\hat{w}_i \qquad (i = 1, \ldots, m,\ j = 1, \ldots, \ell),$$

we take the components $v_{2(2^p-1)+\ell+ij}$, $v_{2(2^p-1)+\ell+ij+m\ell}$, and $v_{2(2^p-1)+\ell+ij+2m\ell}$.