

Chapter 2

Polynomial Optimization Over the Euclidean Ball

In this chapter, we shall present approximation methods for polynomial optimization. The focus will be placed on optimizing several classes of polynomial functions over the Euclidean ball. The models include maximizing a multilinear form over Cartesian product of the Euclidean balls, a homogeneous form over the Euclidean ball, a mixed form over Cartesian product of the Euclidean balls, and a general inhomogeneous polynomial over the Euclidean ball:

$$\begin{array}{ll} (T_{\bar{S}}) & \max F(\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^d) \\ & \text{s.t. } \mathbf{x}^k \in \bar{\mathbb{S}}^{n_k}, k = 1, 2, \dots, d \end{array}$$

$$\begin{array}{ll} (H_{\bar{S}}) & \max f(\mathbf{x}) \\ & \text{s.t. } \mathbf{x} \in \bar{\mathbb{S}}^n \end{array}$$

$$\begin{array}{ll} (M_{\bar{S}}) & \max f(\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^s) \\ & \text{s.t. } \mathbf{x}^k \in \bar{\mathbb{S}}^{n_k}, k = 1, 2, \dots, s \end{array}$$

$$\begin{array}{ll} (P_{\bar{S}}) & \max p(\mathbf{x}) \\ & \text{s.t. } \mathbf{x} \in \bar{\mathbb{S}}^n \end{array}$$

Among the above four polynomial optimization models, the degree of generality increases in the sequential order. Our focus is the design of polynomial-time approximation algorithms with guaranteed worst case performance ratios. There are two reasons for us to choose the Euclidean ball as a typical constraint set. The first is its simplicity, notwithstanding the wide applications. The second and more important reason is that, through this relatively simple case-study, we hope to clearly demonstrate how the new techniques work; much of the analysis can be adapted to other forms of constraints.

2.1 Multilinear Form

The first subclass of polynomial optimization models studied in this brief is the following multilinear form optimization over the Euclidean ball, i.e.,

$$\begin{array}{ll} (T_{\bar{S}}) & \max F(\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^d) \\ & \text{s.t. } \mathbf{x}^k \in \bar{\mathbb{S}}^{n_k}, k = 1, 2, \dots, d \end{array}$$

where $n_1 \leq n_2 \leq \dots \leq n_d$.

It is easy to see that the optimal value of $(T_{\bar{S}})$, denoted by $v(T_{\bar{S}})$, is positive by the assumption that \mathbf{F} is not a zero tensor. Moreover, $(T_{\bar{S}})$ is equivalent to

$$\begin{array}{ll} (T_S) & \max F(\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^d) \\ & \text{s.t. } \mathbf{x}^k \in \mathbb{S}^{n_k}, k = 1, 2, \dots, d. \end{array}$$

This is because we can always scale the decision variables such that $\|\mathbf{x}^k\| = 1$ for all $1 \leq k \leq d$ without decreasing the objective. Therefore in this section, for the ease of presentation, we use \mathbb{S}^{n_k} and $\bar{\mathbb{S}}^{n_k}$ interchangeably in the analysis.

Homogeneous polynomial functions play an important role in approximation theory. In a certain well-defined sense, homogeneous polynomials are fairly dense among all the continuous functions (see, e.g., [66, 113]). Multilinear form is a special class of homogeneous polynomials. In fact, one of the main reasons for us to study multilinear form optimization is its strong connection to homogenous polynomial optimization in deriving approximation bounds, whose details will be discussed in Sect. 2.2. This connection enables a new approach to solve polynomial optimization problems, and the fundamental issue is how to optimize a multilinear form over a set. Chen et al. [24] establish the tightness result of multilinear form relaxation for maximizing a homogeneous form over the Euclidean ball. The study of multilinear form optimization has become centrally important.

The low degree cases of $(T_{\bar{S}})$ are immediately recognizable. For $d = 1$, its optimal solution is $\mathbf{F}/\|\mathbf{F}\|$ due to the Cauchy–Schwartz inequality; for $d = 2$, $(T_{\bar{S}})$ is to compute the spectrum norm of the matrix \mathbf{F} with efficient algorithms readily available. As we shall prove later that $(T_{\bar{S}})$ is already NP-hard when $d = 3$, the focus of this section is to design polynomial-time approximation algorithms with worst-case performance ratios for any fixed degree d . Our basic approach to deal with a high degree multilinear form is to bring its order down step by step, finally leading to a multilinear form optimization in a very low order, hence solvable. Like any matrix can be treated as a long vector, any tensor can also be regarded as a reformed lower order tensor, e.g., by rewriting its corresponding multilinear form by one degree lower. (See the tensor operation in Sect. 1.4.1). After we solve the problem at a lower order, we need to decompose the solution to make it feasible for the original order. Then, specific decomposition methods are required, which will be the topic of this section.

2.1.1 Computational Complexity

To start, a special case of $(T_{\bar{S}})$ is worth noting, which plays an important role in our algorithms.

Proposition 2.1.1 *If $d = 2$, then $(T_{\bar{S}})$ can be solved in polynomial time, with $v(T_{\bar{S}}) \geq \|\mathbf{F}\|/\sqrt{n_1}$.*

Proof. The problem is essentially $\max_{\mathbf{x} \in \mathbb{S}^{n_1}, \mathbf{y} \in \mathbb{S}^{n_2}} \mathbf{x}^T \mathbf{F} \mathbf{y}$. For any fixed \mathbf{y} , the corresponding optimal \mathbf{x} must be $\mathbf{F} \mathbf{y} / \|\mathbf{F} \mathbf{y}\|$ due to the Cauchy–Schwartz inequality, and accordingly,

$$\mathbf{x}^T \mathbf{F} \mathbf{y} = \left(\frac{\mathbf{F} \mathbf{y}}{\|\mathbf{F} \mathbf{y}\|} \right)^T \mathbf{F} \mathbf{y} = \|\mathbf{F} \mathbf{y}\| = \sqrt{\mathbf{y}^T \mathbf{F}^T \mathbf{F} \mathbf{y}}.$$

Thus the problem is equivalent to $\max_{\mathbf{y} \in \mathbb{S}^{n_2}} \mathbf{y}^T \mathbf{F}^T \mathbf{F} \mathbf{y}$, whose solution is the largest eigenvalue and a corresponding eigenvector of the positive semidefinite matrix $\mathbf{F}^T \mathbf{F}$. We then have

$$\lambda_{\max}(\mathbf{F}^T \mathbf{F}) \geq \text{tr}(\mathbf{F}^T \mathbf{F}) / \text{rank}(\mathbf{F}^T \mathbf{F}) \geq \|\mathbf{F}\|^2 / n_1,$$

which implies $v(T_{\bar{S}}) = \sqrt{\lambda_{\max}(\mathbf{F}^T \mathbf{F})} \geq \|\mathbf{F}\| / \sqrt{n_1}$. \square

However, for any degree $d \geq 3$, $(T_{\bar{S}})$ becomes NP-hard. Before engaging in a formal proof, let us first quote a complexity result for a polynomial optimization over the Euclidean sphere due to Nesterov [89].

Lemma 2.1.2 *Suppose $\mathbf{A}_k \in \mathbb{R}^{n \times n}$ is symmetric for $k = 1, 2, \dots, m$, and $f(\mathbf{x})$ is a homogeneous cubic form, then*

$$\begin{aligned} & \max \sum_{k=1}^m (\mathbf{x}^T \mathbf{A}_k \mathbf{x})^2 \\ \text{s.t. } & \mathbf{x} \in \mathbb{S}^n \end{aligned}$$

and

$$\begin{aligned} & \max f(\mathbf{x}) \\ \text{s.t. } & \mathbf{x} \in \mathbb{S}^n \end{aligned}$$

are both NP-hard.

The proof is based on the reduction to the Motzkin–Straus formulation [82] of the stability number of the graph; for details, the reader is referred to Theorem 4 of [89].

Proposition 2.1.3 *If $d = 3$, then $(T_{\bar{S}})$ is NP-hard.*

Proof. In a special case $d = 3$, $n_1 = n_2 = n_3 = n$ and $\mathbf{F} \in \mathbb{R}^{n^3}$ satisfies $F_{ijk} = F_{jik}$ for all $1 \leq i, j, k \leq n$, the objective function of $(T_{\bar{S}})$ can be written as

$$F(\mathbf{x}, \mathbf{y}, \mathbf{z}) = \sum_{i,j,k=1}^n F_{ijk} x_i y_j z_k = \sum_{k=1}^n z_k \left(\sum_{i,j=1}^n F_{ijk} x_i y_j \right) = \sum_{k=1}^n z_k (\mathbf{x}^T \mathbf{A}_k \mathbf{y}),$$

where symmetric matrix $\mathbf{A}_k \in \mathbb{R}^{n \times n}$ with its (i, j) th entry being F_{ijk} for all $1 \leq i, j, k \leq n$. By the Cauchy–Schwartz inequality, $(T_{\mathcal{S}})$ is equivalent to

$$\begin{aligned} & \max \sum_{k=1}^n (\mathbf{x}^T \mathbf{A}_k \mathbf{y})^2 \\ & \text{s.t. } \mathbf{x}, \mathbf{y} \in \mathbb{S}^n. \end{aligned} \quad (2.1)$$

We shall first show that the optimal value of the above problem is always attainable at $\mathbf{x} = \mathbf{y}$. To see why, denote $(\hat{\mathbf{x}}, \hat{\mathbf{y}})$ to be any optimal solution pair, with optimal value v^* . If $\hat{\mathbf{x}} = \pm \hat{\mathbf{y}}$, then the claim is true; otherwise, we may suppose that $\hat{\mathbf{x}} + \hat{\mathbf{y}} \neq \mathbf{0}$. Let us denote $\hat{\mathbf{w}} := (\hat{\mathbf{x}} + \hat{\mathbf{y}}) / \|\hat{\mathbf{x}} + \hat{\mathbf{y}}\|$. Since $(\hat{\mathbf{x}}, \hat{\mathbf{y}})$ must be a KKT point, there exist (λ, μ) such that

$$\begin{cases} \sum_{k=1}^n \hat{\mathbf{x}}^T \mathbf{A}_k \hat{\mathbf{y}} \mathbf{A}_k \hat{\mathbf{y}} = \lambda \hat{\mathbf{x}} \\ \sum_{k=1}^n \hat{\mathbf{x}}^T \mathbf{A}_k \hat{\mathbf{y}} \mathbf{A}_k \hat{\mathbf{x}} = \mu \hat{\mathbf{y}}. \end{cases}$$

Pre-multiplying $\hat{\mathbf{x}}^T$ to the first equation and $\hat{\mathbf{y}}^T$ to the second equation yield $\lambda = \mu = v^*$. Summing up the two equations, pre-multiplying $\hat{\mathbf{w}}^T$, and then scaling, lead us to

$$\sum_{k=1}^n \hat{\mathbf{x}}^T \mathbf{A}_k \hat{\mathbf{y}} \hat{\mathbf{w}}^T \mathbf{A}_k \hat{\mathbf{w}} = v^*.$$

By applying the Cauchy–Schwartz inequality to the above equality, we have

$$v^* \leq \left(\sum_{k=1}^n (\hat{\mathbf{x}}^T \mathbf{A}_k \hat{\mathbf{y}})^2 \right)^{\frac{1}{2}} \left(\sum_{k=1}^n (\hat{\mathbf{w}}^T \mathbf{A}_k \hat{\mathbf{w}})^2 \right)^{\frac{1}{2}} = \sqrt{v^*} \left(\sum_{k=1}^n (\hat{\mathbf{w}}^T \mathbf{A}_k \hat{\mathbf{w}})^2 \right)^{\frac{1}{2}},$$

which implies that $(\hat{\mathbf{w}}, \hat{\mathbf{w}})$ is also an optimal solution. Problem (2.1) is then reduced to Nesterov’s quartic model in Lemma 2.1.2, and its NP-hardness thus follows. \square

2.1.2 Cubic Case

In the remainder of this section, we focus on approximation algorithms for $(T_{\mathcal{S}})$ with general degree d . To illustrate the main idea of the algorithms, we first work with the case $d = 3$ in this subsection

$$\begin{aligned} & (\hat{T}_{\mathcal{S}}) \max F(\mathbf{x}, \mathbf{y}, \mathbf{z}) = \sum_{1 \leq i \leq n_1, 1 \leq j \leq n_2, 1 \leq k \leq n_3} F_{ijk} x_i y_j z_k \\ & \text{s.t. } \mathbf{x} \in \mathbb{S}^{n_1}, \mathbf{y} \in \mathbb{S}^{n_2}, \mathbf{z} \in \mathbb{S}^{n_3}. \end{aligned}$$

Denote $\mathbf{W} = \mathbf{x}\mathbf{y}^T$, and we have

$$\|\mathbf{W}\|^2 = \text{tr}(\mathbf{W}\mathbf{W}^T) = \text{tr}(\mathbf{x}\mathbf{y}^T \mathbf{y}\mathbf{x}^T) = \text{tr}(\mathbf{x}^T \mathbf{x} \mathbf{y}^T \mathbf{y}) = \|\mathbf{x}\|^2 \|\mathbf{y}\|^2 \leq 1.$$

Model $(\hat{T}_{\bar{S}})$ can now be relaxed to

$$\begin{aligned} \max \quad & F(\mathbf{W}, \mathbf{z}) = \sum_{1 \leq i \leq n_1, 1 \leq j \leq n_2, 1 \leq k \leq n_3} F_{ijk} W_{ij} z_k \\ \text{s.t.} \quad & \mathbf{W} \in \bar{\mathbb{S}}^{n_1 \times n_2}, \mathbf{z} \in \mathbb{S}^{n_3}. \end{aligned}$$

Notice that the above problem is exactly $(T_{\bar{S}})$ with $d = 2$, which can be solved in polynomial-time by Proposition 2.1.1. Denote its optimal solution to be $(\hat{\mathbf{W}}, \hat{\mathbf{z}})$. Clearly $F(\hat{\mathbf{W}}, \hat{\mathbf{z}}) \geq v(\hat{T}_{\bar{S}})$. The key step is to recover solution $(\hat{\mathbf{x}}, \hat{\mathbf{y}})$ from the matrix $\hat{\mathbf{W}}$. Below we are going to introduce two basic decomposition routines: one is based on randomization and the other on eigen-decomposition. They play a fundamental role in our proposed algorithms; all solution methods to be developed later rely on these two routines as a basis.

Decomposition Routine 2.1.1

• *INPUT:* matrices $\mathbf{M} \in \mathbb{R}^{n_1 \times n_2}$, $\mathbf{W} \in \bar{\mathbb{S}}^{n_1 \times n_2}$.

1 *Construct*

$$\tilde{\mathbf{W}} = \begin{bmatrix} \mathbf{I}_{n_1 \times n_1} & \mathbf{W} \\ \mathbf{W}^T & \mathbf{W}^T \mathbf{W} \end{bmatrix} \succeq 0.$$

2 *Randomly generate*

$$\begin{pmatrix} \boldsymbol{\xi} \\ \boldsymbol{\eta} \end{pmatrix} \sim \mathcal{N}(\mathbf{0}_{n_1+n_2}, \tilde{\mathbf{W}})$$

and repeat if necessary, until $\boldsymbol{\xi}^T \mathbf{M} \boldsymbol{\eta} \geq \mathbf{M} \bullet \mathbf{W}$ and $\|\boldsymbol{\xi}\| \|\boldsymbol{\eta}\| \leq O(\sqrt{n_1})$.

3 *Compute* $\mathbf{x} = \boldsymbol{\xi} / \|\boldsymbol{\xi}\|$ and $\mathbf{y} = \boldsymbol{\eta} / \|\boldsymbol{\eta}\|$.

• *OUTPUT:* vectors $\mathbf{x} \in \mathbb{S}^{n_1}$, $\mathbf{y} \in \mathbb{S}^{n_2}$.

Now, let $\mathbf{M} = F(\cdot, \cdot, \hat{\mathbf{z}})$ and $\mathbf{W} = \hat{\mathbf{W}}$ in applying the above decomposition routine. For the randomly generated $(\boldsymbol{\xi}, \boldsymbol{\eta})$, we have

$$\mathbb{E}[F(\boldsymbol{\xi}, \boldsymbol{\eta}, \hat{\mathbf{z}})] = \mathbb{E}[\boldsymbol{\xi}^T \mathbf{M} \boldsymbol{\eta}] = \mathbf{M} \bullet \mathbf{W} = F(\hat{\mathbf{W}}, \hat{\mathbf{z}}).$$

He et al. [48] establish that if $f(\mathbf{x})$ is a homogeneous quadratic form and \mathbf{x} is drawn from a zero-mean multivariate normal distribution, then there is a universal constant $\theta \geq 0.03$ such that

$$\Pr\{f(\mathbf{x}) \geq \mathbb{E}[f(\mathbf{x})]\} \geq \theta.$$

Since $\boldsymbol{\xi}^T \mathbf{M} \boldsymbol{\eta}$ is a homogeneous quadratic form of the normal random vector $\begin{pmatrix} \boldsymbol{\xi} \\ \boldsymbol{\eta} \end{pmatrix}$, we know

$$\Pr\{\boldsymbol{\xi}^T \mathbf{M} \boldsymbol{\eta} \geq \mathbf{M} \bullet \mathbf{W}\} = \Pr\{F(\boldsymbol{\xi}, \boldsymbol{\eta}, \hat{\mathbf{z}}) \geq \mathbb{E}[F(\boldsymbol{\xi}, \boldsymbol{\eta}, \hat{\mathbf{z}})]\} \geq \theta.$$

Moreover, by using a property of normal random vectors (see Lemma 3.1 of [76]), we have

$$\begin{aligned} \mathbb{E} [\|\boldsymbol{\xi}\|^2 \|\boldsymbol{\eta}\|^2] &= \mathbb{E} \left[\sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \xi_i^2 \eta_j^2 \right] = \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \left(\mathbb{E}[\xi_i^2] \mathbb{E}[\eta_j^2] + 2(\mathbb{E}[\xi_i \eta_j])^2 \right) \\ &= \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \left[(\hat{\mathbf{W}}^T \hat{\mathbf{W}})_{jj} + 2\hat{W}_{ij}^2 \right] = (n_1 + 2) \text{tr}(\hat{\mathbf{W}}^T \hat{\mathbf{W}}) \leq n_1 + 2. \end{aligned}$$

By applying the Markov inequality, for any $t > 0$

$$\Pr \{ \|\boldsymbol{\xi}\|^2 \|\boldsymbol{\eta}\|^2 \geq t \} \leq \mathbb{E} [\|\boldsymbol{\xi}\|^2 \|\boldsymbol{\eta}\|^2] / t \leq (n_1 + 2)/t.$$

Therefore, by the so-called union inequality for the probability of joint events, we have

$$\begin{aligned} \Pr \{ F(\boldsymbol{\xi}, \boldsymbol{\eta}, \hat{\mathbf{z}}) \geq F(\hat{\mathbf{W}}, \hat{\mathbf{z}}), \|\boldsymbol{\xi}\|^2 \|\boldsymbol{\eta}\|^2 \leq t \} \\ \geq 1 - \Pr \{ F(\boldsymbol{\xi}, \boldsymbol{\eta}, \hat{\mathbf{z}}) < F(\hat{\mathbf{W}}, \hat{\mathbf{z}}) \} - \Pr \{ \|\boldsymbol{\xi}\|^2 \|\boldsymbol{\eta}\|^2 > t \} \\ \geq 1 - (1 - \theta) - (n_1 + 2)/t = \theta/2, \end{aligned}$$

where we let $t = 2(n_1 + 2)/\theta$. Thus we have

$$F(\mathbf{x}, \mathbf{y}, \hat{\mathbf{z}}) \geq \frac{F(\hat{\mathbf{W}}, \hat{\mathbf{z}})}{\sqrt{t}} \geq v(\hat{T}_{\hat{S}}) \sqrt{\frac{\theta}{2(n_1 + 2)}},$$

obtaining an $\Omega(1/\sqrt{n_1})$ -approximation ratio.

Below we present an alternative (and deterministic) decomposition routine.

Decomposition Routine 2.1.2

-
- *INPUT:* a matrix $\mathbf{M} \in \mathbb{R}^{n_1 \times n_2}$.
 - 1 *Find an eigenvector $\hat{\mathbf{y}}$ corresponding to the largest eigenvalue of $\mathbf{M}^T \mathbf{M}$.*
 - 2 *Compute $\mathbf{x} = \mathbf{M}\hat{\mathbf{y}}/\|\mathbf{M}\hat{\mathbf{y}}\|$ and $\mathbf{y} = \hat{\mathbf{y}}/\|\hat{\mathbf{y}}\|$.*
 - *OUTPUT:* vectors $\mathbf{x} \in \mathbb{S}^{n_1}$, $\mathbf{y} \in \mathbb{S}^{n_2}$.
-

This decomposition routine literally follows the proof of Proposition 2.1.1, which tells us that $\mathbf{x}^T \mathbf{M} \mathbf{y} \geq \|\mathbf{M}\|/\sqrt{n_1}$. Thus we have

$$F(\mathbf{x}, \mathbf{y}, \hat{\mathbf{z}}) = \mathbf{x}^T \mathbf{M} \mathbf{y} \geq \frac{\|\mathbf{M}\|}{\sqrt{n_1}} = \max_{\mathbf{z} \in \mathbb{S}^{n_1 \times n_2}} \frac{\mathbf{M} \bullet \mathbf{z}}{\sqrt{n_1}} \geq \frac{\mathbf{M} \bullet \hat{\mathbf{W}}}{\sqrt{n_1}} = \frac{F(\hat{\mathbf{W}}, \hat{\mathbf{z}})}{\sqrt{n_1}} \geq \frac{v(\hat{T}_{\hat{S}})}{\sqrt{n_1}}.$$

The complexity for DR 2.1.1 is $O(n_1 n_2 \ln(1/\varepsilon))$ with probability $1 - \varepsilon$, and for DR 2.1.2 it is $O(\max\{n_1^3, n_1 n_2\})$. However, DR 2.1.2 is indeed very easy to implement, and is deterministic. Both DR 2.1.1 and DR 2.1.2 lead to the following approximation result in terms of the order of the approximation ratio.

Theorem 2.1.4 *If $d = 3$, then $(T_{\tilde{S}})$ admits a polynomial-time approximation algorithm with approximation ratio $1/\sqrt{n_1}$.*

2.1.3 General Fixed Degree

Now we are ready to proceed to the general case of fixed degree d . Let $\mathbf{X} = \mathbf{x}^1(\mathbf{x}^d)^T$, and $(T_{\tilde{S}})$ can be relaxed to

$$\begin{aligned} (\tilde{T}_{\tilde{S}}) \quad & \max F(\mathbf{X}, \mathbf{x}^2, \mathbf{x}^3, \dots, \mathbf{x}^{d-1}) \\ \text{s.t.} \quad & \mathbf{X} \in \mathbb{S}^{n_1 \times n_d}, \mathbf{x}^k \in \mathbb{S}^{n_k}, k = 2, 3, \dots, d-1. \end{aligned}$$

Clearly it is a type of the model $(T_{\tilde{S}})$ with degree $d-1$. Suppose $(\tilde{T}_{\tilde{S}})$ can be solved approximately in polynomial time with approximation ratio τ , i.e., we find $(\hat{\mathbf{X}}, \hat{\mathbf{x}}^2, \hat{\mathbf{x}}^3, \dots, \hat{\mathbf{x}}^{d-1})$ with

$$F(\hat{\mathbf{X}}, \hat{\mathbf{x}}^2, \hat{\mathbf{x}}^3, \dots, \hat{\mathbf{x}}^{d-1}) \geq \tau v(\tilde{T}_{\tilde{S}}) \geq \tau v(T_{\tilde{S}}).$$

Observing that $F(\cdot, \hat{\mathbf{x}}^2, \hat{\mathbf{x}}^3, \dots, \hat{\mathbf{x}}^{d-1}, \cdot)$ is an $n_1 \times n_d$ matrix, using DR 2.1.2 we shall find $(\hat{\mathbf{x}}^1, \hat{\mathbf{x}}^d)$ such that

$$F(\hat{\mathbf{x}}^1, \hat{\mathbf{x}}^2, \dots, \hat{\mathbf{x}}^d) \geq F(\hat{\mathbf{X}}, \hat{\mathbf{x}}^2, \hat{\mathbf{x}}^3, \dots, \hat{\mathbf{x}}^{d-1})/\sqrt{n_1} \geq n_1^{-\frac{1}{2}} \tau v(T_{\tilde{S}}).$$

By induction this leads to the following.

Theorem 2.1.5 *$(T_{\tilde{S}})$ admits a polynomial-time approximation algorithm with approximation ratio $\tau(T_{\tilde{S}})$, where*

$$\tau(T_{\tilde{S}}) := \left(\prod_{k=1}^{d-2} n_k \right)^{-\frac{1}{2}}.$$

Below we summarize the above recursive procedure to solve $(T_{\tilde{S}})$ as in Theorem 2.1.5.

Algorithm 2.1.3

-
- *INPUT*: a d -th order tensor $\mathbf{F} \in \mathbb{R}^{n_1 \times n_2 \times \dots \times n_d}$ with $n_1 \leq n_2 \leq \dots \leq n_d$.
 - 1 Rewrite \mathbf{F} as a $(d-1)$ -th order tensor $\mathbf{F}' \in \mathbb{R}^{n_2 \times n_3 \times \dots \times n_{d-1} \times n_d n_1}$ by combining its first and last modes into one, and placing it in the last mode of \mathbf{F}' , i.e.,

$$F_{i_1, i_2, \dots, i_d} = F'_{i_2, i_3, \dots, i_{d-1}, (i_1-1)n_d + i_d} \quad \forall 1 \leq i_1 \leq n_1, 1 \leq i_2 \leq n_2, \dots, 1 \leq i_d \leq n_d.$$

- 2 For $(T_{\bar{S}})$ with the $(d-1)$ -th order tensor \mathbf{F}' : if $d-1=2$, then apply DR 2.1.2, with input $\mathbf{F}' = \mathbf{M}$ and output $(\hat{\mathbf{x}}^2, \hat{\mathbf{x}}^{1,d}) = (\mathbf{x}, \mathbf{y})$; otherwise obtain a solution $(\hat{\mathbf{x}}^2, \hat{\mathbf{x}}^3, \dots, \hat{\mathbf{x}}^{d-1}, \hat{\mathbf{x}}^{1,d})$ by recursion.
 - 3 Compute a matrix $\mathbf{M}' = F(\cdot, \hat{\mathbf{x}}^2, \hat{\mathbf{x}}^3, \dots, \hat{\mathbf{x}}^{d-1}, \cdot)$ and rewrite the vector $\hat{\mathbf{x}}^{1,d}$ as a matrix $\mathbf{X} \in \mathbb{S}^{n_1 \times n_d}$.
 - 4 Apply either DR 2.1.1 or DR 2.1.2, with input $(\mathbf{M}', \mathbf{X}) = (\mathbf{M}, \mathbf{W})$ and output $(\hat{\mathbf{x}}^1, \hat{\mathbf{x}}^d) = (\mathbf{x}, \mathbf{y})$.
 - *OUTPUT*: a feasible solution $(\hat{\mathbf{x}}^1, \hat{\mathbf{x}}^2, \dots, \hat{\mathbf{x}}^d)$.
-

2.2 Homogeneous Form

This section focuses on optimization of homogeneous polynomials (or forms) over the Euclidean ball:

$$\begin{aligned} (H_{\bar{S}}) \max & f(\mathbf{x}) \\ \text{s.t. } & \mathbf{x} \in \bar{\mathbb{S}}^n \end{aligned}$$

When the degree of the polynomial objective, d , is odd, $(H_{\bar{S}})$ is equivalent to

$$\begin{aligned} (H_S) \max & f(\mathbf{x}) \\ \text{s.t. } & \mathbf{x} \in \mathbb{S}^n. \end{aligned}$$

This is because we can always use $-\mathbf{x}$ to replace \mathbf{x} if its objective value is negative, and can also scale the vector \mathbf{x} along its direction to make it in \mathbb{S}^n . However, if d is even, then this equivalence may not hold. For example, the optimal value of $(H_{\bar{S}})$ may be negative, if the tensor \mathbf{F} is negative definite, i.e., $f(\mathbf{x}) < 0$ for all $\mathbf{x} \neq \mathbf{0}$, while the optimal value of (H_S) is always nonnegative, since $\mathbf{0}$ is always a feasible solution.

The model $(H_{\bar{S}})$ is in general NP-hard. In fact, when $d=1$, $(H_{\bar{S}})$ has a close-form solution, due to the Cauchy–Schwartz inequality; when $d=2$, $(H_{\bar{S}})$ is related

to the largest eigenvalue of the symmetric matrix \mathbf{F} ; when $n \geq 3$, $(H_{\bar{S}})$ becomes NP-hard, which was proven by Nesterov [89] (see Lemma 2.1.2). Interestingly, when $d \geq 3$, the model $(H_{\bar{S}})$ is also regarded as computing the largest eigenvalue of the supersymmetric tensor \mathbf{F} , like the case $d = 2$ (see, e.g., Qi [99]). Luo and Zhang [76] proposed the first polynomial-time randomized approximation algorithm with relative approximation ratio $\Omega(1/n^2)$ when $d = 4$, based on its quadratic SDP relaxation and randomization techniques.

Here in this section, we are going to present polynomial-time approximation algorithms with guaranteed worse-case performance ratios for the models concerned. Our algorithms are designed to solve polynomial optimization with any given degree d , and the approximation ratios improve the previous works specialized to their particular degrees. The major novelty in our approach here is the multilinear tensor relaxation, instead of quadratic SDP relaxation methods in [72, 76]. The relaxed multilinear form optimization problems admit polynomial-time approximation algorithms discussed in Sect. 2.1. After we solve the relaxed problem approximately, the solutions for the tensor model will then be used to produce a feasible solution for the original polynomial optimization model. The remaining task of the section is to illustrate how this can be done.

2.2.1 Link Between Multilinear Form and Homogeneous Form

Let \mathbf{F} be the supersymmetric tensor satisfying $F(\underbrace{\mathbf{x}, \mathbf{x}, \dots, \mathbf{x}}_d) = f(\mathbf{x})$. Then $(H_{\bar{S}})$ can be *relaxed* to multilinear form optimization model $(T_{\bar{S}}^d)$ discussed in Sect. 2.1, as follows:

$$\begin{aligned} (\hat{H}_{\bar{S}}) \quad & \max F(\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^d) \\ \text{s.t.} \quad & \mathbf{x}^k \in \bar{\mathbb{S}}^n, k = 1, 2, \dots, d. \end{aligned}$$

Theorem 2.1.5 asserts that $(\hat{H}_{\bar{S}})$ can be solved approximately in polynomial time, with approximation ratio $n^{-\frac{d-2}{2}}$. The key step is to draw a feasible solution of $(H_{\bar{S}})$ from the approximate solution of $(\hat{H}_{\bar{S}})$. For this purpose, we establish the following link between $(H_{\bar{S}})$ and $(\hat{H}_{\bar{S}})$.

Lemma 2.2.1 *Suppose $\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^d \in \mathbb{R}^n$, and $\xi_1, \xi_2, \dots, \xi_d$ are i.i.d. random variables, each taking values 1 and -1 with equal probability $1/2$. For any supersymmetric d -th order tensor \mathbf{F} and function $f(\mathbf{x}) = F(\mathbf{x}, \mathbf{x}, \dots, \mathbf{x})$, it holds that*

$$\mathbb{E} \left[\prod_{i=1}^d \xi_i f \left(\sum_{k=1}^d \xi_k \mathbf{x}^k \right) \right] = d! F(\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^d).$$

Proof. First we observe that

$$\begin{aligned} \mathbb{E} \left[\prod_{i=1}^d \xi_i f \left(\sum_{k=1}^d \xi_k \mathbf{x}^k \right) \right] &= \mathbb{E} \left[\prod_{i=1}^d \xi_i \sum_{1 \leq k_1, k_2, \dots, k_d \leq d} F \left(\xi_{k_1} \mathbf{x}^{k_1}, \xi_{k_2} \mathbf{x}^{k_2}, \dots, \xi_{k_d} \mathbf{x}^{k_d} \right) \right] \\ &= \sum_{1 \leq k_1, k_2, \dots, k_d \leq d} \mathbb{E} \left[\prod_{i=1}^d \xi_i \prod_{j=1}^d \xi_{k_j} F \left(\mathbf{x}^{k_1}, \mathbf{x}^{k_2}, \dots, \mathbf{x}^{k_d} \right) \right]. \end{aligned}$$

If $\{k_1, k_2, \dots, k_d\}$ is a permutation of $\{1, 2, \dots, d\}$, then

$$\mathbb{E} \left[\prod_{i=1}^d \xi_i \prod_{j=1}^d \xi_{k_j} \right] = \mathbb{E} \left[\prod_{i=1}^d \xi_i^2 \right] = 1.$$

Otherwise, there must be an index k_0 with $1 \leq k_0 \leq d$ and $k_0 \neq k_j$ for all $1 \leq j \leq d$. In the latter case,

$$\mathbb{E} \left[\prod_{i=1}^d \xi_i \prod_{j=1}^d \xi_{k_j} \right] = \mathbb{E} [\xi_{k_0}] \mathbb{E} \left[\prod_{1 \leq i \leq d, i \neq k_0} \xi_i \prod_{j=1}^d \xi_{k_j} \right] = 0.$$

Since the number of different permutations of $\{1, 2, \dots, d\}$ is $d!$, by taking into account of the supersymmetric property of the tensor \mathbf{F} , the claimed relation follows. \square

Note that the coefficients of the link identity in Lemma 2.2.1, $\prod_{i=1}^d \xi_i$, are not always positive. Therefore, whether the degree of the polynomial objective d is even or odd makes a difference.

2.2.2 The Odd Degree Case

When d is odd, the identity in Lemma 2.2.1 can be rewritten as

$$d! F(\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^d) = \mathbb{E} \left[\prod_{i=1}^d \xi_i f \left(\sum_{k=1}^d \xi_k \mathbf{x}^k \right) \right] = \mathbb{E} \left[f \left(\sum_{k=1}^d \left(\prod_{i \neq k} \xi_i \right) \mathbf{x}^k \right) \right].$$

Since $\xi_1, \xi_2, \dots, \xi_d$ are i.i.d. random variables taking values 1 or -1 , by randomization we may find a particular binary vector $\boldsymbol{\beta} \in \mathbb{B}^d$, such that

$$f \left(\sum_{k=1}^d \left(\prod_{i \neq k} \beta_i \right) \mathbf{x}^k \right) \geq d! F(\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^d). \quad (2.2)$$

We remark that d is considered a constant parameter in this brief. Therefore, searching over all the combinations can be done, in principle, in constant time.

Let $\tilde{\mathbf{x}} = \sum_{k=1}^d (\prod_{i \neq k} \beta_i) \mathbf{x}^k$, and $\hat{\mathbf{x}} = \tilde{\mathbf{x}} / \|\tilde{\mathbf{x}}\|$. By the triangle inequality, we have $\|\tilde{\mathbf{x}}\| \leq d$, and thus

$$f(\hat{\mathbf{x}}) \geq d! d^{-d} F(\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^d).$$

Combining with Theorem 2.1.5, we have

Theorem 2.2.2 *When $d \geq 3$ is odd, $(H_{\bar{S}})$ admits a polynomial-time approximation algorithm with approximation ratio $\tau(H_{\bar{S}})$, where*

$$\tau(H_{\bar{S}}) := d! d^{-d} n^{-\frac{d-2}{2}} = \Omega\left(n^{-\frac{d-2}{2}}\right).$$

The algorithm for approximately solving $(H_{\bar{S}})$ with odd d is highlighted below.

Algorithm 2.2.1

- *INPUT: a d -th order supersymmetric tensor $\mathbf{F} \in \mathbb{R}^{n^d}$*
- 1 *Apply Algorithm 2.1.3 to solve the problem*

$$\begin{aligned} & \max F(\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^d) \\ & \text{s.t. } \mathbf{x}^k \in \bar{\mathbb{S}}^n, k = 1, 2, \dots, d \end{aligned}$$

approximately, with input \mathbf{F} and output $(\hat{\mathbf{x}}^1, \hat{\mathbf{x}}^2, \dots, \hat{\mathbf{x}}^d)$.

- 2 *Compute $\boldsymbol{\beta} = \arg \max_{\boldsymbol{\xi} \in \mathbb{B}^d} \{f(\sum_{k=1}^d \xi_k \hat{\mathbf{x}}^k)\}$, or randomly generate $\boldsymbol{\beta}$ uniformly on \mathbb{B}^d and repeat if necessary, until $f(\sum_{k=1}^d \beta_k \hat{\mathbf{x}}^k) \geq d! F(\hat{\mathbf{x}}^1, \hat{\mathbf{x}}^2, \dots, \hat{\mathbf{x}}^d)$.*
 - 3 *Compute $\hat{\mathbf{x}} = \sum_{k=1}^d \beta_k \hat{\mathbf{x}}^k / \|\sum_{k=1}^d \beta_k \hat{\mathbf{x}}^k\|$.*
 - *OUTPUT: a feasible solution $\hat{\mathbf{x}} \in \mathbb{S}^n$.*
-

We remark that it is unnecessary to enumerate all possible 2^d combinations in Step 2 of Algorithm 2.2.1, as (2.2) suggests that a simple randomization process will serve the same purpose, especially when d is large. In the latter case, we will end up with a *polynomial-time randomized approximation algorithm*; otherwise, the computational complexity of Algorithm 2.2.1 is deterministic and runs in polynomial time for fixed d .

2.2.3 The Even Degree Case

When d is even, the only easy case of $(H_{\bar{S}})$ appears to be $d = 2$, and even worse, we have the following.

Proposition 2.2.3 *If $d = 4$, then there is no polynomial-time approximation algorithm with a positive approximation ratio for $(H_{\bar{S}})$ unless $P = NP$.*

Proof. Let $f(\mathbf{x}) = F(\mathbf{x}, \mathbf{x}, \mathbf{x}, \mathbf{x})$ with \mathbf{F} being supersymmetric. We say quartic form $F(\mathbf{x}, \mathbf{x}, \mathbf{x}, \mathbf{x})$ is positive semidefinite if $F(\mathbf{x}, \mathbf{x}, \mathbf{x}, \mathbf{x}) \geq 0$ for all $\mathbf{x} \in \mathbb{R}^n$. It is well known that checking the positive semidefiniteness of $F(\mathbf{x}, \mathbf{x}, \mathbf{x}, \mathbf{x})$ is co-NP-complete. If we were able to find a polynomial-time approximation algorithm to get a positive approximation ratio $\tau \in (0, 1]$ for $v^* = \max_{\mathbf{x} \in \bar{S}^n} -F(\mathbf{x}, \mathbf{x}, \mathbf{x}, \mathbf{x})$, then this algorithm can be used to check the positive semidefiniteness of $F(\mathbf{x}, \mathbf{x}, \mathbf{x}, \mathbf{x})$. To see why, suppose this algorithm returns a feasible solution $\hat{\mathbf{x}}$ with $-F(\hat{\mathbf{x}}, \hat{\mathbf{x}}, \hat{\mathbf{x}}, \hat{\mathbf{x}}) > 0$, then $F(\mathbf{x}, \mathbf{x}, \mathbf{x}, \mathbf{x})$ is not positive semidefinite. Otherwise the algorithm must return a feasible solution $\hat{\mathbf{x}}$ with $0 \geq -F(\hat{\mathbf{x}}, \hat{\mathbf{x}}, \hat{\mathbf{x}}, \hat{\mathbf{x}}) \geq \tau v^*$, which implies $v^* \leq 0$; hence, $F(\mathbf{x}, \mathbf{x}, \mathbf{x}, \mathbf{x})$ is positive semidefinite in this case. Therefore, such algorithm cannot exist unless $P = NP$. \square

This negative result rules out any polynomial-time approximation algorithm with a positive *absolute* approximation ratio for $(H_{\bar{S}})$ when $d \geq 4$ is even. Thus we can only speak of *relative* approximation ratio. The following algorithm slightly modifies Algorithm 2.2.1, and works for $(H_{\bar{S}})$ when d is even.

Algorithm 2.2.2

- *INPUT:* a d -th order supersymmetric tensor $\mathbf{F} \in \mathbb{R}^{n^d}$
- 1 Apply Algorithm 2.1.3 to solve the problem

$$\begin{aligned} & \max F(\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^d) \\ \text{s.t. } & \mathbf{x}^k \in \bar{S}^n, k = 1, 2, \dots, d \end{aligned}$$

approximately, with input \mathbf{F} and output $(\hat{\mathbf{x}}^1, \hat{\mathbf{x}}^2, \dots, \hat{\mathbf{x}}^d)$.

- 2 Compute $\boldsymbol{\beta} = \arg \max_{\boldsymbol{\xi} \in \mathbb{B}^d, \prod_{i=1}^d \xi_i = 1} \{f(\sum_{k=1}^d \xi_k \hat{\mathbf{x}}^k)\}$.
 - 3 Compute $\hat{\mathbf{x}} = \sum_{k=1}^d \beta_k \hat{\mathbf{x}}^k / d$.
 - *OUTPUT:* a feasible solution $\hat{\mathbf{x}} \in \bar{S}^n$.
-

Theorem 2.2.4 *When $d \geq 4$ is even, $(H_{\bar{S}})$ admits a polynomial-time approximation algorithm with relative approximation ratio $\tau(H_{\bar{S}})$.*

Proof. Like in the proof of Theorem 2.2.2, by relaxing $(H_{\bar{S}})$ to $(\hat{H}_{\bar{S}})$, we are able to find a set of vectors $(\hat{\mathbf{x}}^1, \hat{\mathbf{x}}^2, \dots, \hat{\mathbf{x}}^d)$ in the Euclidean ball, such that

$$F(\hat{\mathbf{x}}^1, \hat{\mathbf{x}}^2, \dots, \hat{\mathbf{x}}^d) \geq n^{-\frac{d-2}{2}} v(\hat{H}_{\bar{S}}).$$

Besides, we observe that $v(H_{\bar{S}}) \leq v(\hat{H}_{\bar{S}})$ and $\underline{v}(H_{\bar{S}}) \geq \underline{v}(\hat{H}_{\bar{S}}) = -v(\hat{H}_{\bar{S}})$. Therefore

$$2v(\hat{H}_{\bar{S}}) \geq v(H_{\bar{S}}) - \underline{v}(H_{\bar{S}}). \quad (2.3)$$

Let $\xi_1, \xi_2, \dots, \xi_d$ be i.i.d. random variables, each taking values 1 and -1 with equal probability $1/2$. Obviously, $\Pr \left\{ \prod_{i=1}^d \xi_i = 1 \right\} = \Pr \left\{ \prod_{i=1}^d \xi_i = -1 \right\} = 1/2$. By the triangle inequality, it follows that $\frac{1}{d} \sum_{k=1}^d \xi_k \hat{\mathbf{x}}^k \in \mathbb{S}^n$, and so $f(\frac{1}{d} \sum_{k=1}^d \xi_k \hat{\mathbf{x}}^k) \geq \underline{v}(H_{\bar{S}})$. Applying Lemma 2.2.1 and we have

$$\begin{aligned} & \frac{1}{2} \mathbb{E} \left[f \left(\frac{1}{d} \sum_{k=1}^d \xi_k \hat{\mathbf{x}}^k \right) - \underline{v}(H_{\bar{S}}) \middle| \prod_{i=1}^d \xi_i = 1 \right] \\ & \geq \mathbb{E} \left[f \left(\frac{1}{d} \sum_{k=1}^d \xi_k \hat{\mathbf{x}}^k \right) - \underline{v}(H_{\bar{S}}) \middle| \prod_{i=1}^d \xi_i = 1 \right] \Pr \left\{ \prod_{i=1}^d \xi_i = 1 \right\} \\ & \quad - \mathbb{E} \left[f \left(\frac{1}{d} \sum_{k=1}^d \xi_k \hat{\mathbf{x}}^k \right) - \underline{v}(H_{\bar{S}}) \middle| \prod_{i=1}^d \xi_i = -1 \right] \Pr \left\{ \prod_{i=1}^d \xi_i = -1 \right\} \\ & = \mathbb{E} \left[\prod_{i=1}^d \xi_i \left(f \left(\frac{1}{d} \sum_{k=1}^d \xi_k \hat{\mathbf{x}}^k \right) - \underline{v}(H_{\bar{S}}) \right) \right] \\ & = d^{-d} \mathbb{E} \left[\prod_{i=1}^d \xi_i f \left(\sum_{k=1}^d \xi_k \hat{\mathbf{x}}^k \right) \right] - \underline{v}(H_{\bar{S}}) \mathbb{E} \left[\prod_{i=1}^d \xi_i \right] \\ & = d^{-d} d! F(\hat{\mathbf{x}}^1, \hat{\mathbf{x}}^2, \dots, \hat{\mathbf{x}}^d) \geq \tau(H_S) v(\hat{H}_{\bar{S}}) \geq (\tau(H_S)/2) (v(H_{\bar{S}}) - \underline{v}(H_{\bar{S}})). \end{aligned}$$

Thus we may find a binary vector $\boldsymbol{\beta} \in \mathbb{B}^d$ with $\prod_{i=1}^d \beta_i = 1$, such that

$$f \left(\frac{1}{d} \sum_{k=1}^d \beta_k \hat{\mathbf{x}}^k \right) - \underline{v}(H_{\bar{S}}) \geq \tau(H_S) (v(H_{\bar{S}}) - \underline{v}(H_{\bar{S}})). \quad \square$$

2.3 Mixed Form

In this section, we extend the study on the multilinear form and the homogeneous form to a general mixed form, i.e.,

$$\text{Function M} \quad f(\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^s) = F(\underbrace{\mathbf{x}^1, \mathbf{x}^1, \dots, \mathbf{x}^1}_{d_1}, \underbrace{\mathbf{x}^2, \mathbf{x}^2, \dots, \mathbf{x}^2}_{d_2}, \dots, \underbrace{\mathbf{x}^s, \mathbf{x}^s, \dots, \mathbf{x}^s}_{d_s}),$$

where $d = d_1 + d_2 + \dots + d_s$ is deemed a fixed constant, and d -th order tensor $\mathbf{F} \in \mathbb{R}^{n_1^{d_1} \times n_2^{d_2} \times \dots \times n_s^{d_s}}$ has partial symmetric property. Here we assume that $n_1 \leq n_2 \leq \dots \leq n_s$. The mixed-form optimization model considered here is

$$\begin{array}{ll} (M_{\bar{S}}) & \max f(\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^s) \\ & \text{s.t. } \mathbf{x}^k \in \bar{\mathbb{S}}^{n_k}, k = 1, 2, \dots, s \end{array}$$

The model $(M_{\bar{S}})$ is a generalization of $(T_{\bar{S}})$ in Sect. 2.1 and $(H_{\bar{S}})$ in Sect. 2.2. For the computational complexity, it is similar to its special cases $(T_{\bar{S}})$ and $(H_{\bar{S}})$. It is solvable in polynomial time when $d \leq 2$, and is NP-hard when $d \geq 3$, which will be shown shortly later. Moreover, when $d \geq 4$ and all d_i ($1 \leq k \leq s$) are even, there is no polynomial-time approximation algorithm with a positive approximation ratio unless $P = NP$. This can be verified in its simplest case of $d = 4$ and $d_1 = d_2 = 2$ by using a similar argument as in Ling et al. [72]. In fact, the biquadratic optimization model considered in Ling et al. [72] is slightly different from $(M_{\bar{S}})$, and is exactly the model (M_S) when $d = 4$ and $d_1 = d_2 = 2$, i.e., the Euclidean sphere is considered instead of the Euclidean ball. In particular, they established the equivalence between (M_S) and its quadratic SDP relaxation, based on which they proposed a polynomial-time randomized approximation algorithm with relative approximation ratio $\Omega(1/n_2^2)$.

Like we did earlier, below we are going to present polynomial-time approximation algorithms with guaranteed worse-case performance ratios. Our algorithms work for any fixed degree d , and the approximation ratios improve that of Ling et al. [72] specialized to the quartic case. Instead of using the quadratic SDP relaxation methods in [72], we resort to the multilinear form relaxation, similar as for $(H_{\bar{S}})$. However, one has to adjust Lemma 2.2.1 carefully, and a more general link from the multilinear form to the mixed form need be established, which is the objective of this section.

2.3.1 Complexity and a Step-by-Step Adjustment

First, let us settle the following hardness issue.

Proposition 2.3.1 *If $d = 3$, then $(M_{\bar{S}})$ is NP-hard.*

Proof. We need to verify the NP-hardness for three cases under $d = 3$: $(d_1, d_2, d_3) = (3, 0, 0)$, $(d_1, d_2, d_3) = (2, 1, 0)$ and $(d_1, d_2, d_3) = (1, 1, 1)$. The first case is exactly $(H_{\bar{S}})$ with $d = 3$, whose NP-hardness was claimed in Lemma 2.1.2, and the last case is exactly $(T_{\bar{S}})$ with $d = 3$, whose NP-hardness was shown in Proposition 2.1.3.

It remains to consider the second case $(d_1, d_2, d_3) = (2, 1, 0)$. As a special case, we focus on $n_1 = n_2 = n$ and $\mathbf{F} \in \mathbb{R}^{n^3}$ satisfying $F_{ijk} = F_{jik}$ for all $1 \leq i, j, k \leq n$. We notice that the following form of $(T_{\bar{S}})$ is NP-hard (cf. the proof of Proposition 2.1.3):

$$\begin{array}{ll} (\check{T}_{\bar{S}}) & \max F(\mathbf{x}, \mathbf{y}, \mathbf{z}) \\ & \text{s.t. } \mathbf{x}, \mathbf{y}, \mathbf{z} \in \bar{\mathbb{S}}^n. \end{array}$$

We shall show that the optimal value of $(\check{T}_{\bar{S}})$ is equal to the optimal value of this special case

$$\begin{aligned} &(\check{M}_{\bar{S}}) \max F(\mathbf{x}, \mathbf{x}, \mathbf{z}) \\ &\text{s.t. } \mathbf{x}, \mathbf{z} \in \bar{\mathbb{S}}^n. \end{aligned}$$

It is obvious that $v(\check{T}_{\bar{S}}) \geq v(\check{M}_{\bar{S}})$. Now choose any optimal solution $(\mathbf{x}^*, \mathbf{y}^*, \mathbf{z}^*)$ of $(\check{T}_{\bar{S}})$ and compute the matrix $\mathbf{M} = F(\cdot, \cdot, \mathbf{z}^*)$. Since \mathbf{M} is symmetric, we can compute an eigenvector $\hat{\mathbf{x}}$ corresponding to the largest absolute eigenvalue λ (which is also the largest singular value) in polynomial time. Observe that

$$|F(\hat{\mathbf{x}}, \hat{\mathbf{x}}, \mathbf{z}^*)| = |\hat{\mathbf{x}}^T \mathbf{M} \hat{\mathbf{x}}| = \lambda = \max_{\mathbf{x}, \mathbf{y} \in \bar{\mathbb{S}}^n} \mathbf{x}^T \mathbf{M} \mathbf{y} = \max_{\mathbf{x}, \mathbf{y} \in \bar{\mathbb{S}}^n} F(\mathbf{x}, \mathbf{y}, \mathbf{z}^*) = F(\mathbf{x}^*, \mathbf{y}^*, \mathbf{z}^*) = v(\check{T}_{\bar{S}}),$$

which implies either $(\hat{\mathbf{x}}, \hat{\mathbf{x}}, \mathbf{z}^*)$ or $(\hat{\mathbf{x}}, \hat{\mathbf{x}}, -\mathbf{z}^*)$ is an optimal solution of $(\check{T}_{\bar{S}})$. Therefore $v(\check{T}_{\bar{S}}) \leq v(\check{M}_{\bar{S}})$, and this proves $v(\check{T}_{\bar{S}}) = v(\check{M}_{\bar{S}})$. If $(\check{M}_{\bar{S}})$ can be solved in polynomial time, then its optimal solution is also an optimal solution for $(\check{T}_{\bar{S}})$, which would solve $(\check{T}_{\bar{S}})$ in polynomial time, a contradiction to its NP-hardness. \square

Thus we shall focus on polynomial-time approximation algorithms. Similar to the relaxation in Sect. 2.2, we relax $(M_{\bar{S}})$ to the multilinear form optimization $(T_{\bar{S}})$ as follows:

$$\begin{aligned} &\max F(\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^d) \\ &\text{s.t. } \mathbf{x}^k \in \bar{\mathbb{S}}^{n_1}, 1 \leq k \leq d_1, \\ &\quad \mathbf{x}^k \in \bar{\mathbb{S}}^{n_2}, d_1 + 1 \leq k \leq d_1 + d_2, \\ &\quad \vdots \\ &\quad \mathbf{x}^k \in \bar{\mathbb{S}}^{n_s}, d_1 + d_2 + \dots + d_{s-1} + 1 \leq k \leq d, \end{aligned}$$

then by Theorem 2.1.5 we are able to find $(\hat{\mathbf{x}}^1, \hat{\mathbf{x}}^2, \dots, \hat{\mathbf{x}}^d)$ with $\|\mathbf{x}^k\| \leq 1$ for all $1 \leq k \leq d$ in polynomial time, such that

$$F(\hat{\mathbf{x}}^1, \hat{\mathbf{x}}^2, \dots, \hat{\mathbf{x}}^d) \geq \tilde{\tau}(M_S) v(M_{\bar{S}}), \quad (2.4)$$

where

$$\tilde{\tau}(M_S) := \begin{cases} \left(\frac{\prod_{k=1}^{s-1} n_k^{d_k}}{n_{s-1}} \right)^{-\frac{1}{2}} & d_s = 1, \\ \left(\frac{\prod_{k=1}^s n_k^{d_k}}{n_s^2} \right)^{-\frac{1}{2}} & d_s \geq 2. \end{cases}$$

In order to draw a feasible solution for $(M_{\bar{S}})$ from $(\hat{\mathbf{x}}^1, \hat{\mathbf{x}}^2, \dots, \hat{\mathbf{x}}^d)$, we need to apply the identity stipulated in Lemma 2.2.1 in a careful manner. Approximation results for $(H_{\bar{S}})$ can be similarly derived for the odd case.

Theorem 2.3.2 *If $d \geq 3$ and one of d_k ($k = 1, 2, \dots, s$) is odd, then $(M_{\bar{S}})$ admits a polynomial-time approximation algorithm with approximation ratio $\hat{\tau}(M_{\bar{S}})$, where*

$$\begin{aligned} \hat{\tau}(M_{\bar{S}}) &:= \tilde{\tau}(M_{\bar{S}}) \prod_{1 \leq k \leq s, 3 \leq d_k} \frac{d_k!}{d_k^{d_k}} = \Omega(\tilde{\tau}(M_{\bar{S}})) \\ &= \begin{cases} \left(\prod_{1 \leq k \leq s, 3 \leq d_k} \frac{d_k!}{d_k^{d_k}} \right) \left(\frac{\prod_{k=1}^{s-1} n_k^{d_k}}{n_{s-1}} \right)^{-\frac{1}{2}} & d_s = 1, \\ \left(\prod_{1 \leq k \leq s, 3 \leq d_k} \frac{d_k!}{d_k^{d_k}} \right) \left(\frac{\prod_{k=1}^s n_k^{d_k}}{n_s^2} \right)^{-\frac{1}{2}} & d_s \geq 2. \end{cases} \end{aligned}$$

To prevent the notations from getting out of hand, here we shall only consider a special case $(\hat{M}_{\bar{S}})$, which is easily extended to general $(M_{\bar{S}})$:

$$\begin{aligned} &(\hat{M}_{\bar{S}}) \max F(\mathbf{x}, \mathbf{x}, \mathbf{x}, \mathbf{x}, \mathbf{y}, \mathbf{y}, \mathbf{z}, \mathbf{z}, \mathbf{z}) \\ \text{s.t. } &\mathbf{x} \in \bar{\mathbb{S}}^{n_1}, \mathbf{y} \in \bar{\mathbb{S}}^{n_2}, \mathbf{z} \in \bar{\mathbb{S}}^{n_3}. \end{aligned}$$

By (2.4), we are able to find $\mathbf{x}^1, \mathbf{x}^2, \mathbf{x}^3, \mathbf{x}^4 \in \bar{\mathbb{S}}^{n_1}$, $\mathbf{y}^1, \mathbf{y}^2 \in \bar{\mathbb{S}}^{n_2}$, and $\mathbf{z}^1, \mathbf{z}^2, \mathbf{z}^3 \in \bar{\mathbb{S}}^{n_3}$ in polynomial time, such that

$$F(\mathbf{x}^1, \mathbf{x}^2, \mathbf{x}^3, \mathbf{x}^4, \mathbf{y}^1, \mathbf{y}^2, \mathbf{z}^1, \mathbf{z}^2, \mathbf{z}^3) \geq \tilde{\tau}(M_{\bar{S}}) v(\hat{M}_{\bar{S}}).$$

Let us first fix $(\mathbf{y}^1, \mathbf{y}^2, \mathbf{z}^1, \mathbf{z}^2, \mathbf{z}^3)$ and try to get a solution for the problem

$$\begin{aligned} &\max F(\mathbf{x}, \mathbf{x}, \mathbf{x}, \mathbf{x}, \mathbf{y}^1, \mathbf{y}^2, \mathbf{z}^1, \mathbf{z}^2, \mathbf{z}^3) \\ \text{s.t. } &\mathbf{x} \in \bar{\mathbb{S}}^{n_1}. \end{aligned}$$

Using the same argument as in the proof of Theorem 2.2.2, we are able to find $\hat{\mathbf{x}} \in \bar{\mathbb{S}}^{n_1}$, such that either $F(\hat{\mathbf{x}}, \hat{\mathbf{x}}, \hat{\mathbf{x}}, \hat{\mathbf{x}}, \mathbf{y}^1, \mathbf{y}^2, \mathbf{z}^1, \mathbf{z}^2, \mathbf{z}^3)$ or $F(\hat{\mathbf{x}}, \hat{\mathbf{x}}, \hat{\mathbf{x}}, \hat{\mathbf{x}}, \mathbf{y}^1, \mathbf{y}^2, -\mathbf{z}^1, \mathbf{z}^2, \mathbf{z}^3)$ will be no less than $4!4^{-4}F(\mathbf{x}^1, \mathbf{x}^2, \mathbf{x}^3, \mathbf{x}^4, \mathbf{y}^1, \mathbf{y}^2, \mathbf{z}^1, \mathbf{z}^2, \mathbf{z}^3)$, whereas in the latter case we use $-\mathbf{z}^1$ to update \mathbf{z}^1 . In this context the even degree ($d_1 = 4$) of \mathbf{x} does not raise any issue, as we can always move the negative sign to \mathbf{z}^1 . This process may be considered variable adjustment, and the approximation bound is $4!4^{-4}\tilde{\tau}(M_{\bar{S}})$.

Next we work on adjustment of variable \mathbf{y} and consider the problem

$$\begin{aligned} &\max |F(\hat{\mathbf{x}}, \hat{\mathbf{x}}, \hat{\mathbf{x}}, \hat{\mathbf{x}}, \mathbf{y}, \mathbf{y}, \mathbf{z}^1, \mathbf{z}^2, \mathbf{z}^3)| \\ \text{s.t. } &\mathbf{y} \in \bar{\mathbb{S}}^{n_2}. \end{aligned}$$

The problem is equivalent to finding the largest absolute eigenvalue of a matrix, which can be solved in polynomial time. Denote its optimal solution to be $\hat{\mathbf{y}}$, and update \mathbf{z}^1 with $-\mathbf{z}^1$ if necessary. This process leads to an approximation bound $4!4^{-4}\tilde{\tau}(M_{\bar{S}})$ for the solution $(\hat{\mathbf{x}}, \hat{\mathbf{x}}, \hat{\mathbf{x}}, \hat{\mathbf{x}}, \hat{\mathbf{y}}, \hat{\mathbf{y}}, \mathbf{z}^1, \mathbf{z}^2, \mathbf{z}^3)$.

The last adjustment of the variable \mathbf{z} is straightforward. Similar to the adjustment on \mathbf{x} , now we work with

$$\begin{aligned} & \max F(\hat{\mathbf{x}}, \hat{\mathbf{x}}, \hat{\mathbf{x}}, \hat{\mathbf{x}}, \hat{\mathbf{y}}, \hat{\mathbf{y}}, \mathbf{z}, \mathbf{z}, \mathbf{z}) \\ & \text{s.t. } \mathbf{z} \in \mathbb{S}^{n_3}, \end{aligned}$$

and we can find $\hat{\mathbf{z}} \in \mathbb{S}^{n_3}$ in polynomial time, such that the solution $(\hat{\mathbf{x}}, \hat{\mathbf{x}}, \hat{\mathbf{x}}, \hat{\mathbf{x}}, \hat{\mathbf{y}}, \hat{\mathbf{y}}, \hat{\mathbf{z}}, \hat{\mathbf{z}}, \hat{\mathbf{z}})$ admits an approximation bound $3!3^{-3}4!4^{-4}\tilde{\tau}(M_S)$.

We remark here that the variable \mathbf{z} is the last variable for adjustment, since we cannot move the negative sign to other adjusted variables if the degree of \mathbf{z} is even. That is why we need one of d_k 's to be odd, which allows us to ensure that the last variable for adjustment has an odd degree.

2.3.2 Extended Link Between Multilinear Form and Mixed Form

If all d_k 's ($k = 1, 2, \dots, s$) are even, then we can only hope for a *relative* approximation ratio. For the simplest case where $d = 4$ and $d_1 = d_2 = 2$, the biquadratic optimization model $\max_{\mathbf{x} \in \mathbb{S}^{n_1}, \mathbf{y} \in \mathbb{S}^{n_2}} F(\mathbf{x}, \mathbf{x}, \mathbf{y}, \mathbf{y})$ does not admit any polynomial-time approximation algorithm with a positive approximation ratio. Before working out this case, let us first introduce the following link between the multilinear form and the mixed form, extended from Lemma 2.2.1.

Lemma 2.3.3 *Suppose that $\mathbf{x}^k \in \mathbb{R}^{n_1}$ ($1 \leq k \leq d_1$), $\mathbf{x}^k \in \mathbb{R}^{n_2}$ ($d_1 + 1 \leq k \leq d_1 + d_2$), \dots , $\mathbf{x}^k \in \mathbb{R}^{n_s}$ ($d_1 + d_2 + \dots + d_{s-1} + 1 \leq k \leq d_1 + d_2 + \dots + d_s = d$), and $\xi_1, \xi_2, \dots, \xi_d$ are i.i.d. random variables, each taking values 1 and -1 with equal probability $1/2$. Denote*

$$\mathbf{x}_\xi^1 = \sum_{k=1}^{d_1} \xi_k \mathbf{x}^k, \mathbf{x}_\xi^2 = \sum_{k=d_1+1}^{d_1+d_2} \xi_k \mathbf{x}^k, \dots, \mathbf{x}_\xi^s = \sum_{k=d_1+d_2+\dots+d_{s-1}+1}^d \xi_k \mathbf{x}^k. \quad (2.5)$$

For any partial symmetric d -th order tensor $\mathbf{F} \in \mathbb{R}^{n_1^{d_1} \times n_2^{d_2} \times \dots \times n_s^{d_s}}$ and function

$$f(\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^s) = F(\underbrace{\mathbf{x}^1, \mathbf{x}^1, \dots, \mathbf{x}^1}_{d_1}, \underbrace{\mathbf{x}^2, \mathbf{x}^2, \dots, \mathbf{x}^2}_{d_2}, \dots, \underbrace{\mathbf{x}^s, \mathbf{x}^s, \dots, \mathbf{x}^s}_{d_s}),$$

it holds that

$$\mathbb{E} \left[\prod_{i=1}^d \xi_i f(\mathbf{x}_\xi^1, \mathbf{x}_\xi^2, \dots, \mathbf{x}_\xi^s) \right] = \prod_{k=1}^s d_k! F(\mathbf{x}^1, \mathbf{x}^2, \dots, \mathbf{x}^d).$$

This lemma is easy to prove by invoking Lemma 2.2.1 repeatedly s times. Now, with this extended link in hand, we can then apply a similar argument as in the proof of Theorem 2.2.4.

Theorem 2.3.4 *If $d \geq 4$ and all d_k ($k = 1, 2, \dots, s$) are even, then $(M_{\bar{S}})$ admits a polynomial-time approximation algorithm with relative approximation ratio $\tau(M_S)$, where*

$$\begin{aligned} \tau(M_S) &:= \tilde{\tau}(M_S) \prod_{k=1}^s \frac{d_k!}{d_k^{d_k}} = \Omega(\tilde{\tau}(M_S)) \\ &= \begin{cases} \left(\prod_{k=1}^s \frac{d_k!}{d_k^{d_k}} \right) \left(\frac{\prod_{k=1}^{s-1} n_k^{d_k}}{n_{s-1}} \right)^{-\frac{1}{2}} & d_s = 1, \\ \left(\prod_{k=1}^s \frac{d_k!}{d_k^{d_k}} \right) \left(\frac{\prod_{k=1}^s n_k^{d_k}}{n_s^2} \right)^{-\frac{1}{2}} & d_s \geq 2. \end{cases} \end{aligned}$$

Remark that the case $d_s = 1$ is theoretically not relevant for Theorem 2.3.4 since it assumes all d_k to be even. However, we shall keep this definition of $\tau(M_S)$ for the interest of Sect. 3.2 where this definition will be used.

2.4 Inhomogeneous Polynomial

The last section of this chapter tackles an important and useful extension of the models studied in the previous sections: a generic inhomogeneous polynomial objective function. As is evident, many important applications of polynomial optimization involve an objective that is intrinsically inhomogeneous. Specifically, we consider the following model:

$$\begin{array}{ll} (P_{\bar{S}}) & \max p(\mathbf{x}) \\ & \text{s.t. } \mathbf{x} \in \bar{S}^n \end{array}$$

The above model can be solved in polynomial time when $d \leq 2$ and becomes NP-hard when $d \geq 3$. Even worse, for $d \geq 3$ there is no polynomial-time approximation algorithm with a positive approximation ratio unless $P = NP$, which we shall show later. Therefore, the whole section is focused on *relative* approximation algorithms. The inapproximability of $(P_{\bar{S}})$ differs greatly from that of the homogeneous model $(H_{\bar{S}})$ discussed in Sect. 2.2, since when d is odd, $(H_{\bar{S}})$ admits a polynomial-time approximation algorithm with a positive approximation ratio by Theorem 2.2.2. Consequently, the optimization of an inhomogeneous polynomial is much harder than a homogeneous one.

Extending the solution methods and the corresponding analysis from *homogeneous* polynomial optimization to the general *inhomogeneous* polynomials is not straightforward. As a matter of fact, so far all the successful approximation algorithms with provable approximation ratios in the literature, e.g., the quadratic models considered in [48, 74, 86, 87, 117] and the quartic models considered in [72, 76], are dependent on the homogeneity in a crucial way. Technically, a homogenous polynomial allows one to *scale* the overall function value along a given direction, which is an essential operation in proving the quality bound of the approximation algorithms. The current section breaks its path from the preceding practices, by directly dealing with a *homogenizing* variable. Although homogenization is a natural way to deal with inhomogeneous polynomial functions, it is quite a different matter when it comes to the worst-case performance ratio analysis. In fact, the usual homogenization does not lead to any assured performance ratio. In this section we shall point out a specific route to get around this difficulty, in which we actually provide a general scheme to approximately solve such problems via homogenization.

Let us now focus on the approximation methods for $(P_{\bar{S}})$. As this section is concerned with the relative approximation ratios, we may without loss of generality assume $p(\mathbf{x})$ to have no constant term, i.e., $p(\mathbf{0}) = 0$. Thus the optimal value of this problem is obviously nonnegative, i.e., $v(P_{\bar{S}}) \geq 0$. The complexity to solve $(P_{\bar{S}})$ is summarized in the following proposition.

Proposition 2.4.1 *If $d \leq 2$, then $(P_{\bar{S}})$ can be solved in polynomial time. If $d \geq 3$, then $(P_{\bar{S}})$ is NP-hard, and there is no polynomial-time approximation algorithm with a positive approximation ratio unless $P = NP$.*

Proof. For $d \leq 2$, $(P_{\bar{S}})$ is a standard trust region subproblem. As such it is well known to be solvable in polynomial time (see, e.g., [110, 111] and the references therein). For $d \geq 3$, in a special case where $p(\mathbf{x})$ is a homogeneous cubic form, $(P_{\bar{S}})$ is equivalent to $\max_{\mathbf{x} \in \mathbb{S}^n} p(\mathbf{x})$, which is shown to be NP-hard by Nesterov [89]; see also Lemma 2.1.2.

Let us now consider a special class of $(P_{\bar{S}})$ when $d = 3$:

$$\begin{aligned} v(\alpha) &= \max_{\mathbf{x} \in \mathbb{S}^n} f(\mathbf{x}) - \alpha \|\mathbf{x}\|^2 \\ \text{s.t. } &\mathbf{x} \in \mathbb{S}^n, \end{aligned}$$

where $\alpha \geq 0$, and $f(\mathbf{x})$ is a homogeneous cubic form associated with a nonzero supersymmetric tensor $\mathbf{F} \in \mathbb{R}^{n \times n \times n}$. If $v(\alpha) > 0$, then its optimal solution \mathbf{x}^* satisfies

$$f(\mathbf{x}^*) - \alpha \|\mathbf{x}^*\|^2 = \|\mathbf{x}^*\|^3 f\left(\frac{\mathbf{x}^*}{\|\mathbf{x}^*\|}\right) - \alpha \|\mathbf{x}^*\|^2 = \|\mathbf{x}^*\|^2 \left(\|\mathbf{x}^*\| f\left(\frac{\mathbf{x}^*}{\|\mathbf{x}^*\|}\right) - \alpha \right) > 0.$$

Thus by the optimality of \mathbf{x}^* , we have $\|\mathbf{x}^*\| = 1$. If we choose $\alpha = \|\mathbf{F}\| \geq \max_{\mathbf{x} \in \mathbb{S}^n} f(\mathbf{x})$, then $v(\alpha) = 0$. Since otherwise we must have $v(\alpha) > 0$ and $\|\mathbf{x}^*\| = 1$, with

$$v(\alpha) = f(\mathbf{x}^*) - \alpha \|\mathbf{x}^*\|^2 \leq \max_{\mathbf{x} \in \mathbb{S}^n} f(\mathbf{x}) - \alpha \leq 0,$$

which is a contradiction. Moreover, $v(0) > 0$ simply because \mathbf{F} is a nonzero tensor, and it is also easy to see that $v(\alpha)$ is nonincreasing as $\alpha \geq 0$ increases. Hence, there is a threshold $\alpha_0 \in [0, \|\mathbf{F}\|]$, such that $v(\alpha) > 0$ if $0 \leq \alpha < \alpha_0$, and $v(\alpha) = 0$ if $\alpha \geq \alpha_0$.

Suppose there exists a polynomial-time approximation algorithm with a positive approximation ratio τ for $(P_{\mathbb{S}})$ when $d \geq 3$. Then for every $\alpha \geq 0$, we can find $\mathbf{z} \in \mathbb{S}^n$ in polynomial time, such that $g(\alpha) := f(\mathbf{z}) - \alpha \|\mathbf{z}\|^2 \geq \tau v(\alpha)$. It is obvious that $g(\alpha) \geq 0$ since $v(\alpha) \geq 0$. Together with the fact that $g(\alpha) \leq v(\alpha)$ we have that $g(\alpha) > 0$ if and only if $v(\alpha) > 0$, and $g(\alpha) = 0$ if and only if $v(\alpha) = 0$. Therefore, the threshold value α_0 also satisfies $g(\alpha) > 0$ if $0 \leq \alpha < \alpha_0$, and $g(\alpha) = 0$ if $\alpha \geq \alpha_0$. By applying the bisection search over the interval $[0, \|\mathbf{F}\|]$ with this polynomial-time approximation algorithm, we can find α_0 and $\mathbf{z} \in \mathbb{S}^n$ in polynomial time, such that $f(\mathbf{z}) - \alpha_0 \|\mathbf{z}\|^2 = 0$. This implies that $\mathbf{z} \in \mathbb{S}^n$ is the optimal solution for the problem $\max_{\mathbf{x} \in \mathbb{S}^n} f(\mathbf{x})$ with the optimal value α_0 , which is an NP-hard problem mentioned in the beginning of the proof. Therefore, such approximation algorithm cannot exist unless $P = NP$. \square

The negative result in Proposition 2.4.1 rules out any polynomial-time approximation algorithm with a positive approximation ratio for $(P_{\mathbb{S}})$ when $d \geq 3$. However, a positive *relative* approximation ratio is still possible, which is the main subject of this section. Below we shall first present a polynomial-time algorithm for approximately solving $(P_{\mathbb{S}})$, which admits a (relative) worst-case performance ratio. In fact, here we present a general scheme aiming at solving the polynomial optimization $(P_{\mathbb{S}})$. This scheme breaks down to the following four major steps:

1. Introduce an equivalent model with the objective being a homogenous form.
2. Solve a relaxed model with the objective being a multilinear form.
3. Adjust to get a solution based on the solution of the relaxed model.
4. Assemble a solution for the original inhomogeneous model.

Some of these steps can be designed separately. The algorithm below is one realization of the general scheme for solving $(P_{\mathbb{S}})$, with each step being carried out by a specific procedure. We first present the specialized algorithm, and then in the remainder of the section, we elaborate on these four general steps, and prove that in combination they lead to a polynomial-time approximation algorithm with a quality-assured solution.

Algorithm 2.4.1

- *INPUT: an n -dimensional d -th degree polynomial function $p(\mathbf{x})$.*
 - 1** Rewrite $p(\mathbf{x}) - p(\mathbf{0}) = F(\underbrace{\bar{\mathbf{x}}, \bar{\mathbf{x}}, \dots, \bar{\mathbf{x}}}_d)$ when $x_h = 1$ as in (2.7), with \mathbf{F} being an $(n+1)$ -dimensional d -th order supersymmetric tensor.
 - 2** Apply Algorithm 2.1.3 to solve the problem

$$\begin{aligned} & \max F(\bar{\mathbf{x}}^1, \bar{\mathbf{x}}^2, \dots, \bar{\mathbf{x}}^d) \\ & \text{s.t. } \bar{\mathbf{x}}^k \in \mathbb{S}^{n+1}, k = 1, 2, \dots, d \end{aligned}$$
 approximately, with input \mathbf{F} and output $(\bar{\mathbf{y}}^1, \bar{\mathbf{y}}^2, \dots, \bar{\mathbf{y}}^d)$.
 - 3** Compute $(\bar{\mathbf{z}}^1, \bar{\mathbf{z}}^2, \dots, \bar{\mathbf{z}}^d) = \arg \max \left\{ F \left(\left(\xi_1 \mathbf{y}_1^{1/d} \right), \left(\xi_2 \mathbf{y}_1^{2/d} \right), \dots, \left(\xi_d \mathbf{y}_1^{d/d} \right) \right), \boldsymbol{\xi} \in \mathbb{B}^d \right\}$.
 - 4** Compute $\mathbf{z} = \arg \max \left\{ p(\mathbf{0}); p(\mathbf{z}(\boldsymbol{\beta})/z_h(\boldsymbol{\beta})), \boldsymbol{\beta} \in \mathbb{B}^d \text{ and } \beta_1 = \prod_{k=2}^d \beta_k = 1 \right\}$, with $\bar{\mathbf{z}}(\boldsymbol{\beta}) = \beta_1(d+1)\bar{\mathbf{z}}^1 + \sum_{k=2}^d \beta_k \bar{\mathbf{z}}^k$.
 - *OUTPUT: a feasible solution $\mathbf{z} \in \mathbb{S}^n$.*
-

In Step 2 of Algorithm 2.4.1, Algorithm 2.1.3 is called to approximately solve multilinear form optimization over the Euclidean ball, which is a deterministic polynomial-time algorithm. Notice the degree of the polynomial $p(\mathbf{x})$ is deemed a fixed parameter in this brief, and thus Algorithm 2.4.1 runs in polynomial time, and is deterministic too. Our main result in this section is the following.

Theorem 2.4.2 ($P_{\bar{\mathbb{S}}}$) admits a polynomial-time approximation algorithm with relative approximation ratio $\tau(P_{\bar{\mathbb{S}}})$, where

$$\tau(P_{\bar{\mathbb{S}}}) := 2^{-\frac{5d}{2}} (d+1)! d^{-2d} (n+1)^{-\frac{d-2}{2}} = \Omega \left(n^{-\frac{d-2}{2}} \right).$$

Below we study in detail how a particular implementation of these four steps of the scheme (which becomes Algorithm 2.4.1) leads to the promised worst-case relative performance ratio in Theorem 2.4.2.

2.4.1 Homogenization

The method of homogenization depends on the form of the polynomial $p(\mathbf{x})$. Without losing generality henceforth we assume $p(\mathbf{x})$ to have no constant term, although Algorithm 2.4.1 applies for any polynomial. If $p(\mathbf{x})$ is given as a summation of

homogeneous polynomial functions of different degrees, i.e., $f_k(\mathbf{x})$ ($1 \leq k \leq d$) is a homogeneous polynomial function of degree k , then we may first write

$$f_k(\mathbf{x}) = F_k(\underbrace{\mathbf{x}, \mathbf{x}, \dots, \mathbf{x}}_k) \quad (2.6)$$

with \mathbf{F}_k being a k th order supersymmetric tensor. Then by introducing a homogenizing variable x_h , which is always equal to 1, we may rewrite $p(\mathbf{x})$ as

$$\begin{aligned} p(\mathbf{x}) &= \sum_{k=1}^d f_k(\mathbf{x}) = \sum_{k=1}^d f_k(\mathbf{x}) x_h^{d-k} = \sum_{k=1}^d F_k(\underbrace{\mathbf{x}, \mathbf{x}, \dots, \mathbf{x}}_k) x_h^{d-k} \\ &= F\left(\underbrace{\begin{pmatrix} \mathbf{x} \\ x_h \end{pmatrix}, \begin{pmatrix} \mathbf{x} \\ x_h \end{pmatrix}, \dots, \begin{pmatrix} \mathbf{x} \\ x_h \end{pmatrix}}_d\right) = F(\underbrace{\bar{\mathbf{x}}, \bar{\mathbf{x}}, \dots, \bar{\mathbf{x}}}_d) = f(\bar{\mathbf{x}}), \end{aligned} \quad (2.7)$$

where \mathbf{F} is an $(n+1)$ -dimensional d -th order supersymmetric tensor, whose last component is 0 (since $p(\mathbf{x})$ has no constant term).

If the polynomial $p(\mathbf{x})$ is given in terms of summation of monomials, then we should first group them according to their degrees, and then rewrite the summation of monomials in each group as homogeneous polynomial function. After that, we proceed according to (2.6) and (2.7) to obtain the tensor form \mathbf{F} , as required.

Finally, we may equivalently reformulate $(P_{\bar{S}})$ as

$$\begin{aligned} (\bar{P}_{\bar{S}}) \quad & \max f(\bar{\mathbf{x}}) \\ \text{s.t.} \quad & \bar{\mathbf{x}} = \begin{pmatrix} \mathbf{x} \\ x_h \end{pmatrix}, \\ & \mathbf{x} \in \bar{\mathbb{S}}^n, x_h = 1. \end{aligned}$$

Obviously, we have $v(P_{\bar{S}}) = v(\bar{P}_{\bar{S}})$ and $\underline{v}(P_{\bar{S}}) = \underline{v}(\bar{P}_{\bar{S}})$.

2.4.2 Multilinear Form Relaxation

Multilinear form relaxation has proven to be effective, as discussed in Sects. 2.2 and 2.3. Specifically, Lemmas 2.2.1 and 2.3.3 are the key link formulae. Now we relax $(\bar{P}_{\bar{S}})$ to an inhomogeneous multilinear form optimization model

$$\begin{aligned} (TP_{\bar{S}}) \quad & \max F(\bar{\mathbf{x}}^1, \bar{\mathbf{x}}^2, \dots, \bar{\mathbf{x}}^d) \\ \text{s.t.} \quad & \bar{\mathbf{x}}^k = \begin{pmatrix} \mathbf{x}^k \\ x_h^k \end{pmatrix}, k = 1, 2, \dots, d, \\ & \mathbf{x}^k \in \bar{\mathbb{S}}^n, x_h^k = 1, k = 1, 2, \dots, d. \end{aligned}$$

Obviously, we have $v(TP_{\bar{S}}) \geq v(\bar{P}_{\bar{S}}) = v(P_{\bar{S}})$. Before proceeding, let us first settle the computational complexity issue for solving $(TP_{\bar{S}})$.

Proposition 2.4.3 $(TP_{\bar{S}})$ is NP-hard whenever $d \geq 3$.

Proof. Notice that in Proposition 2.1.3, we proved the following problem is NP-hard:

$$\begin{aligned} & \max F(\mathbf{x}, \mathbf{y}, \mathbf{z}) \\ & \text{s.t. } \mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathbb{S}^n. \end{aligned}$$

For $d = 3$ and a special case where \mathbf{F} satisfies $F_{n+1,j,k} = F_{i,n+1,k} = F_{i,j,n+1} = 0$ for all $1 \leq i, j, k \leq n+1$, $(TP_{\bar{S}})$ is equivalent to the above model, and so it is NP-hard in general. \square

$(TP_{\bar{S}})$ is still difficult to solve, and moreover it remains inhomogeneous, since x_h^k is required to be 1. To our best knowledge, no polynomial-time approximation algorithm is available in the literature to solve this problem. Furthermore, we shall relax the constraint $x_h^k = 1$, and introduce the following parameterized and homogenized problem:

$$\begin{aligned} (TP_{\bar{S}}(t)) \quad & \max F(\bar{\mathbf{x}}^1, \bar{\mathbf{x}}^2, \dots, \bar{\mathbf{x}}^d) \\ & \text{s.t. } \|\bar{\mathbf{x}}^k\| \leq t, \bar{\mathbf{x}}^k \in \mathbb{R}^{n+1}, k = 1, 2, \dots, d. \end{aligned}$$

Obviously, $(TP_{\bar{S}})$ can be relaxed to $(TP_{\bar{S}}(\sqrt{2}))$, since if $\bar{\mathbf{x}}$ is feasible for $(TP_{\bar{S}})$ then $\|\bar{\mathbf{x}}\|^2 = \|\mathbf{x}\|^2 + x_h^2 \leq 1 + 1 = 2$. Consequently, $v(TP_{\bar{S}}(\sqrt{2})) \geq v(TP_{\bar{S}})$.

Both the objective and the constraints are now homogeneous, and it is obvious that for all $t > 0$, $(TP_{\bar{S}}(t))$ is equivalent (in fact scalable) to each other. Moreover, $(TP_{\bar{S}}(1))$ is

$$\begin{aligned} & \max F(\bar{\mathbf{x}}^1, \bar{\mathbf{x}}^2, \dots, \bar{\mathbf{x}}^d) \\ & \text{s.t. } \bar{\mathbf{x}}^k \in \mathbb{S}^{n+1}, k = 1, 2, \dots, d, \end{aligned}$$

which is exactly $(T_{\bar{S}})$ as we discussed in Sect. 2.1. By using Algorithm 2.1.3 and applying Theorem 2.1.5, $(TP_{\bar{S}}(1))$ admits a polynomial-time approximation algorithm with approximation ratio $(n+1)^{-\frac{d-2}{2}}$. Therefore, for all $t > 0$, $(TP_{\bar{S}}(t))$ also admits a polynomial-time approximation algorithm with approximation ratio $(n+1)^{-\frac{d-2}{2}}$, and $v(TP_{\bar{S}}(t)) = t^d v(TP_{\bar{S}}(1))$. After this relaxation step (Step 2 in Algorithm 2.4.1), we are able to find a feasible solution $(\bar{\mathbf{y}}^1, \bar{\mathbf{y}}^2, \dots, \bar{\mathbf{y}}^d)$ of $(TP_{\bar{S}}(1))$ in polynomial time, such that

$$\begin{aligned} F(\bar{\mathbf{y}}^1, \bar{\mathbf{y}}^2, \dots, \bar{\mathbf{y}}^d) & \geq (n+1)^{-\frac{d-2}{2}} v(TP_{\bar{S}}(1)) \\ & = 2^{-\frac{d}{2}} (n+1)^{-\frac{d-2}{2}} v(TP_{\bar{S}}(\sqrt{2})) \\ & \geq 2^{-\frac{d}{2}} (n+1)^{-\frac{d-2}{2}} v(TP_{\bar{S}}). \end{aligned} \tag{2.8}$$

Algorithm 2.1.3 is the engine which enables the second step of our scheme. In fact, any polynomial-time approximation algorithm of $(TP_{\bar{S}}(1))$ can be used as an engine to yield a realization (algorithm) of our scheme. As will become evident later, any improvement of the approximation ratio of $(TP_{\bar{S}}(1))$ leads to the improvement of relative approximation ratio in Theorem 2.4.2. For example, recently So [106] improved the approximation bound of $(TP_{\bar{S}}(1))$ to $\Omega\left(\left(\frac{\ln n}{n}\right)^{-\frac{d-2}{2}}\right)$ (though the algorithm is mainly of theoretical interest), and consequently the relative approximation ratio under our scheme is improved to $\Omega\left(\left(\frac{\ln n}{n}\right)^{\frac{d-2}{2}}\right)$ too. Of course, one may apply any other favorite algorithm to solve the relaxation $(TP_{\bar{S}}(1))$. For instance, the alternating least square (ALS) algorithm (see, e.g., [65] and the references therein) and the maximum block improvement (MBI) method of Chen et al. [24] can be the other alternatives for the second step.

2.4.3 Adjusting the Homogenizing Components

The approximate solution $(\bar{\mathbf{y}}^1, \bar{\mathbf{y}}^2, \dots, \bar{\mathbf{y}}^d)$ of $(TP_{\bar{S}}(1))$ satisfies $\|\bar{\mathbf{y}}^k\| \leq 1$ for all $1 \leq k \leq d$, which implies $\|\mathbf{y}^k\| \leq 1$. Other from that, we do not have any control on the size of y_h^k , and thus $(\bar{\mathbf{y}}^1, \bar{\mathbf{y}}^2, \dots, \bar{\mathbf{y}}^d)$ may not be a feasible solution for $(TP_{\bar{S}})$. The following lemma plays a link role in our analysis to ensure that the construction of a feasible solution for the inhomogeneous model $(TP_{\bar{S}})$ is possible.

Lemma 2.4.4 *Suppose $\bar{\mathbf{x}}^k \in \mathbb{R}^{n+1}$ with $|x_h^k| \leq 1$ for all $1 \leq k \leq d$. Let $\eta_1, \eta_2, \dots, \eta_d$ be independent random variables, each taking values 1 and -1 with $E[\eta_k] = x_h^k$ for all $1 \leq k \leq d$, and let $\xi_1, \xi_2, \dots, \xi_d$ be i.i.d. random variables, each taking values 1 and -1 with equal probability $1/2$. If the last component of the tensor \mathbf{F} is 0, then*

$$E \left[\prod_{k=1}^d \eta_k F \left(\begin{pmatrix} \eta_1 \mathbf{x}^1 \\ 1 \end{pmatrix}, \begin{pmatrix} \eta_2 \mathbf{x}^2 \\ 1 \end{pmatrix}, \dots, \begin{pmatrix} \eta_d \mathbf{x}^d \\ 1 \end{pmatrix} \right) \right] = F(\bar{\mathbf{x}}^1, \bar{\mathbf{x}}^2, \dots, \bar{\mathbf{x}}^d), \quad (2.9)$$

and

$$E \left[F \left(\begin{pmatrix} \xi_1 \mathbf{x}^1 \\ 1 \end{pmatrix}, \begin{pmatrix} \xi_2 \mathbf{x}^2 \\ 1 \end{pmatrix}, \dots, \begin{pmatrix} \xi_d \mathbf{x}^d \\ 1 \end{pmatrix} \right) \right] = 0. \quad (2.10)$$

Proof. The claimed equations readily result from the following observations:

$$\begin{aligned} & E \left[\prod_{k=1}^d \eta_k F \left(\begin{pmatrix} \eta_1 \mathbf{x}^1 \\ 1 \end{pmatrix}, \begin{pmatrix} \eta_2 \mathbf{x}^2 \\ 1 \end{pmatrix}, \dots, \begin{pmatrix} \eta_d \mathbf{x}^d \\ 1 \end{pmatrix} \right) \right] \\ &= E \left[F \left(\begin{pmatrix} \eta_1^2 \mathbf{x}^1 \\ \eta_1 \end{pmatrix}, \begin{pmatrix} \eta_2^2 \mathbf{x}^2 \\ \eta_2 \end{pmatrix}, \dots, \begin{pmatrix} \eta_d^2 \mathbf{x}^d \\ \eta_d \end{pmatrix} \right) \right] \quad (\text{multilinearity of } F) \end{aligned}$$

$$\begin{aligned}
&= F\left(E\left[\begin{pmatrix} \mathbf{x}^1 \\ \eta_1 \end{pmatrix}\right], E\left[\begin{pmatrix} \mathbf{x}^2 \\ \eta_2 \end{pmatrix}\right], \dots, E\left[\begin{pmatrix} \mathbf{x}^d \\ \eta_d \end{pmatrix}\right]\right) \quad (\text{independence of } \eta_k \text{'s}) \\
&= F(\bar{\mathbf{x}}^1, \bar{\mathbf{x}}^2, \dots, \bar{\mathbf{x}}^d),
\end{aligned}$$

and

$$\begin{aligned}
&E\left[F\left(\begin{pmatrix} \xi_1 \mathbf{x}^1 \\ 1 \end{pmatrix}, \begin{pmatrix} \xi_2 \mathbf{x}^2 \\ 1 \end{pmatrix}, \dots, \begin{pmatrix} \xi_d \mathbf{x}^d \\ 1 \end{pmatrix}\right)\right] \\
&= F\left(E\left[\begin{pmatrix} \xi_1 \mathbf{x}^1 \\ 1 \end{pmatrix}\right], E\left[\begin{pmatrix} \xi_2 \mathbf{x}^2 \\ 1 \end{pmatrix}\right], \dots, E\left[\begin{pmatrix} \xi_d \mathbf{x}^d \\ 1 \end{pmatrix}\right]\right) \quad (\text{independence of } \xi_k \text{'s}) \\
&= F\left(\begin{pmatrix} \mathbf{0} \\ 1 \end{pmatrix}, \begin{pmatrix} \mathbf{0} \\ 1 \end{pmatrix}, \dots, \begin{pmatrix} \mathbf{0} \\ 1 \end{pmatrix}\right) \quad (\text{zero-mean of } \xi_k \text{'s}) \\
&= 0,
\end{aligned}$$

where the last equality is due to the fact that the last component of \mathbf{F} is 0. \square

Lemma 2.4.4 suggests that one may enumerate the 2^d possible combinations of $\left(\begin{pmatrix} \xi_1 \mathbf{y}^1 \\ 1 \end{pmatrix}, \begin{pmatrix} \xi_2 \mathbf{y}^2 \\ 1 \end{pmatrix}, \dots, \begin{pmatrix} \xi_d \mathbf{y}^d \\ 1 \end{pmatrix}\right)$ and pick the one with the largest value of function F (or via a simple randomization procedure) to generate a feasible solution for the inhomogeneous multilinear form optimization ($TP_{\bar{S}}$) from a feasible solution for the homogeneous multilinear form optimization ($TP_S(1)$), with a controlled possible quality deterioration. This fact plays a key role in proving the approximation ratio for ($TP_{\bar{S}}$).

Theorem 2.4.5 ($TP_{\bar{S}}$) *admits a polynomial-time approximation algorithm with approximation ratio $2^{-\frac{3d}{2}}(n+1)^{-\frac{d-2}{2}}$.*

Proof. Let $(\bar{\mathbf{y}}^1, \bar{\mathbf{y}}^2, \dots, \bar{\mathbf{y}}^d)$ be the feasible solution found in Step 2 of Algorithm 2.4.1 satisfying (2.8), and let $\boldsymbol{\eta} = (\eta_1, \eta_2, \dots, \eta_d)^T$ with all η_k 's being independent and taking values 1 and -1 such that $E[\eta_k] = y_h^k$. Applying Lemma 2.4.4, we have (2.9) which implies

$$\begin{aligned}
&F(\bar{\mathbf{y}}^1, \bar{\mathbf{y}}^2, \dots, \bar{\mathbf{y}}^d) \\
&= - \sum_{\boldsymbol{\beta} \in \mathbb{B}^d, \prod_{k=1}^d \beta_k = -1} \Pr\{\boldsymbol{\eta} = \boldsymbol{\beta}\} F\left(\begin{pmatrix} \beta_1 \mathbf{y}^1 \\ 1 \end{pmatrix}, \begin{pmatrix} \beta_2 \mathbf{y}^2 \\ 1 \end{pmatrix}, \dots, \begin{pmatrix} \beta_d \mathbf{y}^d \\ 1 \end{pmatrix}\right) \\
&\quad + \sum_{\boldsymbol{\beta} \in \mathbb{B}^d, \prod_{k=1}^d \beta_k = 1} \Pr\{\boldsymbol{\eta} = \boldsymbol{\beta}\} F\left(\begin{pmatrix} \beta_1 \mathbf{y}^1 \\ 1 \end{pmatrix}, \begin{pmatrix} \beta_2 \mathbf{y}^2 \\ 1 \end{pmatrix}, \dots, \begin{pmatrix} \beta_d \mathbf{y}^d \\ 1 \end{pmatrix}\right),
\end{aligned}$$

and (2.10) which implies

$$\sum_{\boldsymbol{\beta} \in \mathbb{B}^d} F\left(\begin{pmatrix} \beta_1 \mathbf{y}^1 \\ 1 \end{pmatrix}, \begin{pmatrix} \beta_2 \mathbf{y}^2 \\ 1 \end{pmatrix}, \dots, \begin{pmatrix} \beta_d \mathbf{y}^d \\ 1 \end{pmatrix}\right) = 0.$$

Combing the above two equalities, for any constant c , we have

$$\begin{aligned} & F(\bar{\mathbf{y}}^1, \bar{\mathbf{y}}^2, \dots, \bar{\mathbf{y}}^d) \\ &= \sum_{\boldsymbol{\beta} \in \mathbb{B}^d, \prod_{k=1}^d \beta_k = -1} (c - \Pr\{\boldsymbol{\eta} = \boldsymbol{\beta}\}) F\left(\begin{pmatrix} \beta_1 \mathbf{y}^1 \\ 1 \end{pmatrix}, \begin{pmatrix} \beta_2 \mathbf{y}^2 \\ 1 \end{pmatrix}, \dots, \begin{pmatrix} \beta_d \mathbf{y}^d \\ 1 \end{pmatrix}\right) \\ &+ \sum_{\boldsymbol{\beta} \in \mathbb{B}^d, \prod_{k=1}^d \beta_k = 1} (c + \Pr\{\boldsymbol{\eta} = \boldsymbol{\beta}\}) F\left(\begin{pmatrix} \beta_1 \mathbf{y}^1 \\ 1 \end{pmatrix}, \begin{pmatrix} \beta_2 \mathbf{y}^2 \\ 1 \end{pmatrix}, \dots, \begin{pmatrix} \beta_d \mathbf{y}^d \\ 1 \end{pmatrix}\right). \end{aligned} \quad (2.11)$$

If we let

$$c = \max_{\boldsymbol{\beta} \in \mathbb{B}^d, \prod_{k=1}^d \beta_k = -1} \Pr\{\boldsymbol{\eta} = \boldsymbol{\beta}\},$$

then the coefficients of each term in (2.11) will be nonnegative. Therefore we are able to find $\boldsymbol{\beta}' \in \mathbb{B}^d$, such that

$$F\left(\begin{pmatrix} \beta'_1 \mathbf{y}^1 \\ 1 \end{pmatrix}, \begin{pmatrix} \beta'_2 \mathbf{y}^2 \\ 1 \end{pmatrix}, \dots, \begin{pmatrix} \beta'_d \mathbf{y}^d \\ 1 \end{pmatrix}\right) \geq \tau_0 F(\bar{\mathbf{y}}^1, \bar{\mathbf{y}}^2, \dots, \bar{\mathbf{y}}^d), \quad (2.12)$$

where

$$\begin{aligned} \tau_0 &= \left(\sum_{\boldsymbol{\beta} \in \mathbb{B}^d, \prod_{k=1}^d \beta_k = 1} (c + \Pr\{\boldsymbol{\eta} = \boldsymbol{\beta}\}) + \sum_{\boldsymbol{\beta} \in \mathbb{B}^d, \prod_{k=1}^d \beta_k = -1} (c - \Pr\{\boldsymbol{\eta} = \boldsymbol{\beta}\}) \right)^{-1} \\ &\geq \left(2^{d-1} c + \sum_{\boldsymbol{\beta} \in \mathbb{B}^d, \prod_{k=1}^d \beta_k = 1} \Pr\{\boldsymbol{\eta} = \boldsymbol{\beta}\} + (2^{d-1} - 1)c \right)^{-1} \\ &\geq \left(2^{d-1} + 1 + 2^{d-1} - 1 \right)^{-1} = 2^{-d}. \end{aligned}$$

Let us denote $\bar{\mathbf{z}}^k := \begin{pmatrix} \beta'_k \mathbf{y}^k \\ 1 \end{pmatrix}$ for $k = 1, 2, \dots, d$. Since $\|\bar{\mathbf{z}}^k\| = \|\beta'_k \mathbf{y}^k\| \leq 1$, we know that $(\bar{\mathbf{z}}^1, \bar{\mathbf{z}}^2, \dots, \bar{\mathbf{z}}^d)$ is a feasible solution for $(TP_{\bar{S}})$. By combining with (2.8), we have

$$\begin{aligned} F(\bar{\mathbf{z}}^1, \bar{\mathbf{z}}^2, \dots, \bar{\mathbf{z}}^d) &\geq \tau_0 F(\bar{\mathbf{y}}^1, \bar{\mathbf{y}}^2, \dots, \bar{\mathbf{y}}^d) \\ &\geq 2^{-d} 2^{-\frac{d}{2}} (n+1)^{-\frac{d-2}{2}} v(TP_{\bar{S}}) \\ &= 2^{-\frac{3d}{2}} (n+1)^{-\frac{d-2}{2}} v(TP_{\bar{S}}). \end{aligned} \quad \square$$

One may notice that our proposed algorithm for solving $(TP_{\bar{S}})$ is very similar to Steps 2 and 3 of Algorithm 2.4.1, with only a minor modification at Step 3, namely we choose a solution in $\arg \max \left\{ F \left((\beta_1 y^1), (\beta_2 y^2), \dots, (\beta_d y^d) \right), \boldsymbol{\beta} \in \mathbb{B}^d \right\}$, instead of choosing a solution in $\arg \max \left\{ F \left((\beta_1 y^1/d), (\beta_2 y^2/d), \dots, (\beta_d y^d/d) \right), \boldsymbol{\beta} \in \mathbb{B}^d \right\}$. The reason to divide d at Step 3 in Algorithm 2.4.1 (to solve $(P_{\bar{S}})$) will become clear later. Finally, we remark again that it is unnecessary to enumerate all possible 2^d combinations in this step, as (2.11) suggests that a simple randomization process will serve the same purpose, especially when d is large. In the latter case, we will end up with a *polynomial-time randomized approximation algorithm*; otherwise, the computational complexity of the procedure is deterministic and is polynomial-time.

2.4.4 Feasible Solution Assembling

Finally we come to the last step of the scheme. In Step 4 of Algorithm 2.4.1, a polarization formula $\bar{\mathbf{z}}(\boldsymbol{\beta}) = \beta_1(d+1)\bar{\mathbf{z}}^1 + \sum_{k=2}^d \beta_k \bar{\mathbf{z}}^k$ with $\boldsymbol{\beta} \in \mathbb{B}^d$ and $\beta_1 = \prod_{k=2}^d \beta_k = 1$ is proposed. In fact, searching over all $\boldsymbol{\beta} \in \mathbb{B}^d$ will possibly improve the solution, although the worst-case performance ratio will remain the same. Moreover, one may choose $\bar{\mathbf{z}}^1$ or any other $\bar{\mathbf{z}}^k$ to play the same role here; alternatively one may enumerate $\beta_\ell(d+1)\bar{\mathbf{z}}^\ell + \sum_{1 \leq k \leq d, k \neq \ell} \beta_k \bar{\mathbf{z}}^k$ over all $\boldsymbol{\beta} \in \mathbb{B}^d$ and $1 \leq \ell \leq d$, and take the best possible solution; again, this will not change the theoretical performance ratio. The polarization formula at Step 4 of Algorithm 2.4.1 works for any fixed degree d , and we shall complete the final stage of the proof of Theorem 2.4.2. Specifically, we shall prove that by letting

$$\mathbf{z} = \arg \max \left\{ p(\mathbf{0}); p \left(\frac{\mathbf{z}(\boldsymbol{\beta})}{z_h(\boldsymbol{\beta})} \right), \boldsymbol{\beta} \in \mathbb{B}^d \text{ and } \beta_1 = \prod_{k=2}^d \beta_k = 1 \right\}$$

with $\bar{\mathbf{z}}(\boldsymbol{\beta}) = \beta_1(d+1)\bar{\mathbf{z}}^1 + \sum_{k=2}^d \beta_k \bar{\mathbf{z}}^k$, we have

$$p(\mathbf{z}) - v(P_{\bar{S}}) \geq \tau(P_{\bar{S}}) (v(P_{\bar{S}}) - v(P_{\bar{S}})). \quad (2.13)$$

First, the solution $(\bar{\mathbf{z}}^1, \bar{\mathbf{z}}^2, \dots, \bar{\mathbf{z}}^d)$ as established at Step 3 of Algorithm 2.4.1 satisfies $\|\bar{\mathbf{z}}^k\| \leq 1/d$ (notice we divided d in each term at Step 3) and $z_h^k = 1$ for $k = 1, 2, \dots, d$. A same proof of Theorem 2.4.5 can show that

$$F(\bar{\mathbf{z}}^1, \bar{\mathbf{z}}^2, \dots, \bar{\mathbf{z}}^d) \geq d^{-d} 2^{-\frac{3d}{2}} (n+1)^{-\frac{d-2}{2}} v(TP_{\bar{S}}) \geq 2^{-\frac{3d}{2}} d^{-d} (n+1)^{-\frac{d-2}{2}} v(P_{\bar{S}}). \quad (2.14)$$

It is easy to see that

$$2 \leq |z_h(\boldsymbol{\beta})| \leq 2d \text{ and } \|\mathbf{z}(\boldsymbol{\beta})\| \leq (d+1)/d + (d-1)/d = 2. \quad (2.15)$$

Thus $\bar{\mathbf{z}}(\beta)/z_h(\beta)$ is a feasible solution for $(\bar{P}_{\bar{S}})$, and so $f(\bar{\mathbf{z}}(\beta)/z_h(\beta)) \geq \underline{v}(\bar{P}_{\bar{S}}) = \underline{v}(P_{\bar{S}})$. Moreover, we shall argue below that

$$\beta_1 = 1 \implies f(\bar{\mathbf{z}}(\beta)) \geq (2d)^d \underline{v}(P_{\bar{S}}). \quad (2.16)$$

If this were not the case, then $f(\bar{\mathbf{z}}(\beta)/(2d)) < \underline{v}(P_{\bar{S}}) \leq 0$. Notice that $\beta_1 = 1$ implies $z_h(\beta) > 0$, and thus we have

$$f\left(\frac{\bar{\mathbf{z}}(\beta)}{z_h(\beta)}\right) = \left(\frac{2d}{z_h(\beta)}\right)^d f\left(\frac{\bar{\mathbf{z}}(\beta)}{2d}\right) \leq f\left(\frac{\bar{\mathbf{z}}(\beta)}{2d}\right) < \underline{v}(P_{\bar{S}}),$$

which contradicts the feasibility of $\bar{\mathbf{z}}(\beta)/z_h(\beta)$.

Suppose $\xi_1, \xi_2, \dots, \xi_d$ are i.i.d. random variables, each taking values 1 and -1 with equal probability $1/2$. By the link Lemma 2.2.1, noticing that $f(\bar{\mathbf{z}}(-\xi)) = f(-\bar{\mathbf{z}}(\xi)) = (-1)^d f(\bar{\mathbf{z}}(\xi))$, we have

$$\begin{aligned} d!F\left((d+1)\bar{\mathbf{z}}^1, \bar{\mathbf{z}}^2, \dots, \bar{\mathbf{z}}^d\right) &= \mathbb{E}\left[\prod_{k=1}^d \xi_k f(\bar{\mathbf{z}}(\xi))\right] \\ &= \frac{1}{4}\mathbb{E}\left[f(\bar{\mathbf{z}}(\xi)) \mid \xi_1 = 1, \prod_{k=2}^d \xi_k = 1\right] - \frac{1}{4}\mathbb{E}\left[f(\bar{\mathbf{z}}(\xi)) \mid \xi_1 = 1, \prod_{k=2}^d \xi_k = -1\right] \\ &\quad - \frac{1}{4}\mathbb{E}\left[f(\bar{\mathbf{z}}(\xi)) \mid \xi_1 = -1, \prod_{k=2}^d \xi_k = 1\right] + \frac{1}{4}\mathbb{E}\left[f(\bar{\mathbf{z}}(\xi)) \mid \xi_1 = -1, \prod_{k=2}^d \xi_k = -1\right] \\ &= \frac{1}{4}\mathbb{E}\left[f(\bar{\mathbf{z}}(\xi)) \mid \xi_1 = 1, \prod_{k=2}^d \xi_k = 1\right] - \frac{1}{4}\mathbb{E}\left[f(\bar{\mathbf{z}}(\xi)) \mid \xi_1 = 1, \prod_{k=2}^d \xi_k = -1\right] \\ &\quad - \frac{1}{4}\mathbb{E}\left[f(\bar{\mathbf{z}}(-\xi)) \mid \xi_1 = 1, \prod_{k=2}^d \xi_k = (-1)^{d-1}\right] \\ &\quad + \frac{1}{4}\mathbb{E}\left[f(\bar{\mathbf{z}}(-\xi)) \mid \xi_1 = 1, \prod_{k=2}^d \xi_k = (-1)^d\right]. \end{aligned}$$

By inserting and canceling a constant term, the above expression further leads to

$$\begin{aligned} d!F\left((d+1)\bar{\mathbf{z}}^1, \bar{\mathbf{z}}^2, \dots, \bar{\mathbf{z}}^d\right) &= \mathbb{E}\left[\prod_{k=1}^d \xi_k f(\bar{\mathbf{z}}(\xi))\right] \\ &= \frac{1}{4}\mathbb{E}\left[\left(f(\bar{\mathbf{z}}(\xi)) - (2d)^d \underline{v}(P_{\bar{S}})\right) \mid \xi_1 = 1, \prod_{k=2}^d \xi_k = 1\right] \end{aligned}$$

$$\begin{aligned}
& -\frac{1}{4} \mathbb{E} \left[\left(f(\bar{\mathbf{z}}(\xi)) - (2d)^d \underline{v}(P_{\bar{S}}) \right) \middle| \xi_1 = 1, \prod_{k=2}^d \xi_k = -1 \right] \\
& + \frac{(-1)^{d-1}}{4} \mathbb{E} \left[\left(f(\bar{\mathbf{z}}(\xi)) - (2d)^d \underline{v}(P_{\bar{S}}) \right) \middle| \xi_1 = 1, \prod_{k=2}^d \xi_k = (-1)^{d-1} \right] \\
& + \frac{(-1)^d}{4} \mathbb{E} \left[\left(f(\bar{\mathbf{z}}(\xi)) - (2d)^d \underline{v}(P_{\bar{S}}) \right) \middle| \xi_1 = 1, \prod_{k=2}^d \xi_k = (-1)^d \right] \\
& \leq \frac{1}{2} \mathbb{E} \left[\left(f(\bar{\mathbf{z}}(\xi)) - (2d)^d \underline{v}(P_{\bar{S}}) \right) \middle| \xi_1 = 1, \prod_{k=2}^d \xi_k = 1 \right], \tag{2.17}
\end{aligned}$$

where the last inequality is due to (2.16). Therefore, there is a binary vector $\beta' \in \mathbb{B}^d$ with $\beta'_1 = \prod_{k=2}^d \beta'_k = 1$, such that

$$\begin{aligned}
f(\bar{\mathbf{z}}(\beta')) - (2d)^d \underline{v}(P_{\bar{S}}) & \geq 2d! F((d+1)\bar{\mathbf{z}}^1, \bar{\mathbf{z}}^2, \dots, \bar{\mathbf{z}}^d) \\
& \geq 2^{-\frac{3d}{2}+1} (d+1)! d^{-d} (n+1)^{-\frac{d-2}{2}} v(P_{\bar{S}}),
\end{aligned}$$

where the last step is due to (2.14).

Below we argue $\mathbf{z} = \arg \max \left\{ p(\mathbf{0}); p\left(\frac{\mathbf{z}(\beta)}{z_h(\beta)}\right), \beta \in \mathbb{B}^d \text{ and } \beta_1 = \prod_{k=2}^d \beta_k = 1 \right\}$ satisfies (2.13). In fact, if $-\underline{v}(P_{\bar{S}}) \geq \tau(P_{\bar{S}})(v(P_{\bar{S}}) - \underline{v}(P_{\bar{S}}))$, then $\mathbf{0}$ trivially satisfies (2.13), and so does \mathbf{z} in this case. Otherwise, if $-\underline{v}(P_{\bar{S}}) < \tau(P_{\bar{S}})(v(P_{\bar{S}}) - \underline{v}(P_{\bar{S}}))$, then we have

$$v(P_{\bar{S}}) > (1 - \tau(P_{\bar{S}}))(v(P_{\bar{S}}) - \underline{v}(P_{\bar{S}})) \geq \frac{v(P_{\bar{S}}) - \underline{v}(P_{\bar{S}})}{2},$$

which implies

$$\begin{aligned}
f\left(\frac{\bar{\mathbf{z}}(\beta')}{2d}\right) - \underline{v}(P_{\bar{S}}) & \geq (2d)^{-d} 2^{-\frac{3d}{2}+1} (d+1)! d^{-d} (n+1)^{-\frac{d-2}{2}} v(P_{\bar{S}}) \\
& \geq \tau(P_{\bar{S}})(v(P_{\bar{S}}) - \underline{v}(P_{\bar{S}})).
\end{aligned}$$

The above inequality also implies that $f(\bar{\mathbf{z}}(\beta')/(2d)) > 0$. Recall that $\beta'_1 = 1$ implies $z_h(\beta') > 0$, and thus $2d/z_h(\beta') \geq 1$ by (2.15). Therefore, we have

$$p(\mathbf{z}) \geq p\left(\frac{\mathbf{z}(\beta')}{z_h(\beta')}\right) = f\left(\frac{\bar{\mathbf{z}}(\beta')}{z_h(\beta')}\right) = \left(\frac{2d}{z_h(\beta')}\right)^d f\left(\frac{\bar{\mathbf{z}}(\beta')}{2d}\right) \geq f\left(\frac{\bar{\mathbf{z}}(\beta')}{2d}\right).$$

This shows that \mathbf{z} satisfies (2.13) in both cases, which concludes the whole proof.

Approximation Methods for Polynomial Optimization
Models, Algorithms, and Applications

Li, Z.; He, S.; Zhang, S.

2012, VIII, 124 p., Softcover

ISBN: 978-1-4614-3983-7