Performance of trigonometric generating functions on some combinatorial problems

Yu. Nesterov *

October, 2004
September, 2005
(revised version)

Abstract

In this paper we analyze computational performance of dual trigonometric generating functions on some integer programming problems. We show that if the number of equality constraints is fixed, then this technique allows to solve the problems in time, which is polynomial in the dimension of the space of variables.

Keywords: Integer programming, generating functions, polynomial complexity, dynamic programming, knapsack problem.

CORE Discussion Paper #2005/69

*Center for Operations Research and Econometrics (CORE), Catholic University of Louvain (UCL), 34 voie du Roman Pays, 1348 Louvain-la-Neuve, Belgium; e-mail: nesterov@core.ucl.ac.be.

This paper presents research results of the Belgian Program on Interuniversity Attraction Poles, initiated by the Belgian Federal Science Policy Office.
The scientific responsibility rests with its author.

1 Introduction

Motivation. During last years we can see a permanent rise of interest to generating functions and polynomial technique as applied to combinatorial problems. This technique was developing in two main directions. The primal approach (see [1, 2] and the references therein), is focused on studying the properties of *primal* generating function

$$f(S,x) = \sum_{m \in S} x^m, \quad x = (x^{(1)}, \ldots, x^{(n)})^T \in C^n,$$

with $x^m = \prod_{i=1}^{n} (x^{(i)})^{m^{(i)}}$, which is defined for a set of integer vectors $S \subset Z^n$. Denote by $1_n \in Z^n$ the vector vector of all ones. Since the value $f(S, 1_n)$ is equal to the number of integer points inside the set S, this approach fits very well the counting problems. Moreover, the unit coefficients of the monomials in $f(S,x)$ ensure acceptable numerical stability of these objects. Probably, this explains the existence of several successful implementations of the primal technique (see, for example, [5], [6]).

The structure of *dual* generating functions is more complicated [3, 4]. In primal space we define a parametric family of sets

$$X(y) \subset Z^n, \quad y \in \Delta \subset Z^m.$$

Further, for each set $X \subset Z^n$, we define a *characteristic function*

$$\psi_X(c), \quad c \in R^n.$$

Then the dual generating function is defined as

$$g(v) = \sum_{y \in \Delta} \psi_{X(y)}(c)\, v^y, \quad v \in C^m.$$

The dual generating function is a powerful tool. As the primal generating function, it can be used for counting integer points in convex sets. However, it can be used also for approximating the optimal values of linear objective function over the corresponding feasible set. Moreover, for some simple cases, like knapsack problem, these technique coupled with fast Fourier transform, results in an algorithm [8], which is more efficient than the standard Dynamic Programming methods (see, for example, Section II.6.1 [7]).

To the best of our knowledge, the computational performance of the dual generating functions was not studied yet. In this paper we are addressing some complexity issues of this approach. At the same time, we show that the natural domain of the dual generating functions is not the whole C^m, but a direct product of m unit circles. Thus, on this domain the dual generating functions become the *trigonometric* polynomials. This transformation improves stability of all operations as applied to these functions. Moreover, it significantly facilitates the numerical computation of the characteristic functions.

Contents. The paper is organized as follows. In Section 2 we introduce a definition of characteristic functions and prove the representation theorem for (dual) trigonometric generating functions. We discuss the computational complexity of the value of characteristic function of a given set and of a particular point from this set. In the next Section

3 we apply the result to a family of integer polytopes formed as the intersection of n-dimensional boxes and a set of solutions to a system of linear equations parameterized by its right-hand side. We show that for m being fixed the complexity of our computations is polynomial in n. These results can be seen as an extension of the Dynamic Programming technique developed for knapsack problem onto a more general situation.

Notation. The main part of our notation is quite standard. For two real vectors x and y we denote by $\langle x, y\rangle$ their standard inner product:

$$\langle x, y\rangle = \sum_i x^{(i)} y^{(i)}.$$

Dimension of the arguments is always clear from the context. Notation $\mathbf{j}$ is used for $\sqrt{-1}$. Finally, for a discrete set X, notation $|X|$ is used for its cardinality, while for a complex point $v = x + \mathbf{j}\, y \in C$, we denote by $|v|$ its absolute value.

2 Trigonometric generating functions

Let us start by describing the main idea of proposed technique. Consider a parametric family of sets in Z^n:

$$\mathcal{X} = \{X(y),\ y \in \Delta\} \subset Z^n,$$

where Δ is a subset of Z^m; for simplicity, we assume Δ to be finite. For a given y, we are interested in a possibility of computing a value of the *characteristic function* of the set $X(y)$, that is

$$\psi_{X(y)}(c) = \begin{cases} \sum\limits_{x \in X(y)} e^{\langle c, x\rangle}, & \text{if } X(y) \neq \emptyset, \\ 0, & \text{otherwise,} \end{cases} \qquad (c \in R^n).$$

This function can be interesting in different situations. For example, the problem of counting the integer vectors in $X(y)$ is solved by relation

$$|X(y)| = \psi_{X(y)}(0).$$

Characteristic function can be used for approximating the optimal value of an optimization problem over $X(y)$:

$$\begin{aligned} \mu \ln \psi_{X(y)}\left(\tfrac{1}{\mu} c\right) &\geq \max_x \{\langle c, x\rangle :\ x \in X(y)\} \\ &\geq \mu \ln \psi_{X(y)}\left(\tfrac{1}{\mu} c\right) - \mu \ln |X(y)|, \quad \mu > 0. \end{aligned}$$

There is a convenient way for representing the whole system of characteristic functions using so-called (dual) *generating function* [3, 4]:

$$g_{\mathcal{X},c}(v) = \sum_{y \in \Delta} \psi_{X(y)}(c) \cdot v^y, \quad v \in C^m, \tag{2.1}$$

where $v^y = \prod\limits_{i=1}^{m} (v^{(i)})^{y^{(i)}}$. Note that quite often it is possible to find a compact expression for the value $g_{\mathcal{X},c}(v)$ (see [3]). However, it is very difficult to use it in practical computations since the polynomials of complex variables of high degree are numerically unstable.

In this paper we suggest to change the sense of variables in (2.1) and consider the generating functions with arguments restricted to a direct product of unit circles, the set

$$\mathcal{S}_m = \{v \in C^m, \quad |v^{(i)}| = 1,\ i = 1, \dots, m\}.$$

Then $g_{\mathcal{X},c}(v)$ becomes a *trigonometric polynomial*. The advantages of this modification are immediate. Firstly, now the powers of the variables are numerically stable:

$$z \in C,\ |z| = 1 \quad \Rightarrow \quad z = e^{\mathbf{j}\,\varphi} \quad \Rightarrow \quad z^k = \cos(k\varphi) + \mathbf{j}\,\sin(k\varphi).$$

Secondly, the system of monomials $\{v^y\}_{y \in Z^m}$, $v \in \mathcal{S}_m$, becomes *orthogonal*. This leads to many useful consequences. In this paper we analyze only one of them.

Lemma 1 *For $\varphi \in R^m$ denote $e^{\mathbf{j}\,\varphi} = (e^{\mathbf{j}\,\varphi^{(1)}}, \dots, e^{\mathbf{j}\,\varphi^{(m)}})^T$, and $d\varphi = d\varphi^{(1)} \dots d\varphi^{(m)}$. Then*

$$\psi_{X(y)}(c) = \frac{1}{(2\pi)^m} \int_0^{2\pi} \dots \int_0^{2\pi} e^{-\mathbf{j}\,\langle y, \varphi \rangle} g_{\mathcal{X},c}(e^{\mathbf{j}\,\varphi})\, d\varphi. \tag{2.2}$$

Proof:
This follows immediately from orthogonality of monomials v^y and $v^{\bar{y}}$ for $y \neq \bar{y}$:

$$\begin{aligned} \int_0^{2\pi} \dots \int_0^{2\pi} e^{-\mathbf{j}\,\langle y, \varphi \rangle} e^{\mathbf{j}\,\langle \bar{y}, \varphi \rangle}\, d\varphi &= \int_0^{2\pi} \dots \int_0^{2\pi} e^{\mathbf{j}\,\langle \bar{y} - y, \varphi \rangle}\, d\varphi \\ &= \prod_{i=1}^m \int_0^{2\pi} e^{\mathbf{j}\,(\bar{y}^{(i)} - y^{(i)})\varphi^{(i)}} d\varphi^{(i)} = \begin{cases} 0, & \text{if } \bar{y} \neq y, \\ (2\pi)^m, & \text{otherwise.} \end{cases} \end{aligned}$$

□

Note that in (2.2) we need to integrate a *polynomial*. Therefore we can compute the value of this integral by *exact* cubature formulas. Indeed, for $y \in Z^m$ denote

$$p_y(v) = v^y, \quad v \in \mathcal{S}_m.$$

Lemma 2 *For $L \in Z_+^m$ define the following grid*

$$\begin{aligned} \mathcal{G}_L &= \left\{ \varphi \in R^m :\ \varphi^{(i)} = \tfrac{2\pi}{L^{(i)}} k_i,\ k_i \in Z,\ 0 \leq k_i \leq L^{(i)} - 1,\ i = 1, \dots, m \right\}, \\ |\mathcal{G}_L| &= \prod_{i=1}^m L^{(i)}. \end{aligned}$$

Then, for any $y \in Z^m$, $|y^{(i)}| < L^{(i)}$, $i = 1, \dots, m$, we have

$$\frac{1}{(2\pi)^m} \int_0^{2\pi} \dots \int_0^{2\pi} p_y(e^{\mathbf{j}\,\varphi})\, d\varphi = \frac{1}{|\mathcal{G}_L|} \sum_{\varphi \in \mathcal{G}_L} p_y(e^{\mathbf{j}\,\varphi}). \tag{2.3}$$

Proof:
Indeed, we have seen that

$$\frac{1}{(2\pi)^m} \int_0^{2\pi} \dots \int_0^{2\pi} p_y(e^{\mathbf{j}\,\varphi})\, d\varphi = \begin{cases} 0, & \text{if } y \neq 0, \\ 1, & \text{otherwise.} \end{cases}$$

On the other hand,

$$\sum_{\varphi\in\mathcal{G}_L} p_y(e^{\mathbf{j}\varphi}) \;=\; \sum_{\varphi\in\mathcal{G}_L}\prod_{i=1}^{m} e^{\mathbf{j}\varphi^{(i)}y^{(i)}} \;=\; \sum_{k_1=0}^{L^{(1)}-1}\cdots\sum_{k_m=0}^{L^{(m)}-1}\prod_{i=1}^{m} e^{\mathbf{j}\frac{2\pi k_i}{L^{(i)}}y^{(i)}}$$

$$= \left(\sum_{k=0}^{L^{(1)}-1} e^{\mathbf{j}\frac{2\pi y^{(1)}}{L^{(1)}}k}\right)\cdots\left(\sum_{k=0}^{L^{(m)}-1} e^{\mathbf{j}\frac{2\pi y^{(m)}}{L^{(m)}}k}\right).$$

Since $|y^{(i)}| < L^{(i)}$, $i = 1,\ldots,m$, for $y^{(i)} \neq 0$ we have

$$s_i \stackrel{\rm def}{=} \sum_{k=0}^{L^{(i)}-1} e^{\mathbf{j}\frac{2\pi y^{(i)}}{L^{(i)}}k} = \left[e^{\mathbf{j}\,2\pi y^{(i)}} - 1\right]\cdot\left[e^{\mathbf{j}\frac{2\pi y^{(i)}}{L^{(i)}}} - 1\right]^{-1} = 0.$$

If $y^{(i)} = 0$, then $s_i = L^{(i)}$. □

We can easily prove now the following theorem.

Theorem 1 *Let $L^{(i)} > |y^{(i)}|$, for any $i = 1,\ldots,m$, and $y \in \Delta$. Then*

$$\psi_{X(y)}(c) \;=\; \tfrac{1}{|\mathcal{G}_L|}\sum_{\varphi\in\mathcal{G}_L} g_{\mathcal{X},c}(e^{\mathbf{j}\varphi})e^{-\mathbf{j}\,\langle y,\varphi\rangle}, \quad y\in\Delta. \tag{2.4}$$

Proof:
In view of Lemma 1 and Lemma 2 we have

$$\psi_{X(y)}(c) \;=\; \tfrac{1}{(2\pi)^m}\int_0^{2\pi}\cdots\int_0^{2\pi} e^{-\mathbf{j}\,\langle y,\varphi\rangle}\left(\sum_{u\in\Delta}\psi_{X(u)}(c)\cdot e^{\mathbf{j}\,\langle u,\varphi\rangle}\right)d\varphi$$

$$= \sum_{u\in\Delta}\psi_{X(u)}(c)\cdot\tfrac{1}{(2\pi)^m}\int_0^{2\pi}\cdots\int_0^{2\pi} e^{\mathbf{j}\,\langle u-y,\varphi\rangle}d\varphi$$

$$= \sum_{u\in\Delta}\psi_{X(u)}(c)\cdot\tfrac{1}{|\mathcal{G}_L|}\sum_{\varphi\in\mathcal{G}_L} e^{\mathbf{j}\,\langle u-y,\varphi\rangle} \;=\; \tfrac{1}{|\mathcal{G}_L|}\sum_{\varphi\in\mathcal{G}_L} g_{\mathcal{X},c}(e^{\mathbf{j}\varphi})e^{-\mathbf{j}\,\langle y,\varphi\rangle}.$$

□

Let us mention now some computational advantages of this representation. First of all, this is a straightforward computation provided that we are able to compute the values $g_{\mathcal{X},c}(e^{\mathbf{j}\varphi})$ for different $\varphi \in R^m$. Moreover, representation (2.4) can be used for computing a point from the convex hull of the set $X(y)$. Indeed, by definition of gradient, we have

$$\tfrac{1}{\psi_{X(y)}(c)}\nabla\psi_{X(y)}(c) \;=\; \tfrac{1}{\psi_{X(y)}(c)}\sum_{x\in X(y)} x\cdot e^{\langle c,x\rangle} \in \mathrm{Conv}\,(X(y)). \tag{2.5}$$

On the other hand,

$$\nabla\psi_{X(y)}(c) \;=\; \tfrac{1}{|\mathcal{G}_L|}\sum_{\varphi\in\mathcal{G}_L}\nabla_c\, g_{\mathcal{X},c}(e^{\mathbf{j}\varphi})\, e^{-\mathbf{j}\,\langle y,\varphi\rangle}. \tag{2.6}$$

The computational complexity of this vector depends on the complexity of computation of the value of function $g_{\mathcal{X},c}$. However, if this function is given by a formula, then usually the computational complexity of the gradient $\nabla\psi_{X(y)}(c)$ is proportional (up to an absolute multiplicative factor) to the complexity of computing the value $g_{\mathcal{X},c}(e^{\mathbf{j}\,\varphi})$. Of course, in general $\nabla\psi_{X(y)}(c)$ is not an integer vector. However, it is possible to approach any boundary integer point of $X(y)$ by applying an appropriate cost vector c. Note that for $c = 0$ we get an arithmetic mean of all vectors in $X(y)$.

Note also that the space R^n and its dimension n have no *direct* impact on the complexity of expression (2.4). If we are able to find an efficient way for computing the values $g_{\mathcal{X},c}(e^{\mathbf{j}\,\varphi})$ (see [3, 4] for examples of short generating functions), and if the set Δ is not too big, then the computation by (2.4) can be quite efficient. In some sense, using the technique of generating functions we can replace the combinatorial objects $X(y)$ in Z^n by another combinatorial object in R^m. Of course, in the latter space we have to apply a kind of complete enumeration. However, if the dimension m is small, the resulting complexity may be reasonably good. We consider a corresponding example in the next section.

3 Application example

Consider a family of sets of nonnegative integer solutions for a system of linear equations parameterized by its right-hand side. Namely, for $u \in Z^n_+$ and $y \in Z^m$ denote

$$\begin{aligned} B(u) &= \{x \in Z^n :\ 0 \le x \le u\} \\ X_u(y) &= \{x \in B(u) :\ Ax = y\}, \\ \mathcal{X} &= \{X_u(y),\ y \in \Delta \stackrel{\rm def}{=} AB(u)\}. \end{aligned}$$

where A is an $m \times n$-matrix with integer coefficients. Let us introduce the trigonometric generating function of our problem:

$$g_{\mathcal{X},c}(v) = \sum_{y\in\Delta} \psi_{X_u(y)}(c) \cdot v^y, \quad v \in \mathcal{S}_m. \tag{3.1}$$

The following result is quite useful.

Lemma 3

$$g_{\mathcal{X},c}(e^{\mathbf{j}\,\varphi}) = \prod_{j=1}^{n} \left[1 + \sum_{k=1}^{u^{(j)}} e^{k\cdot(c^{(j)} + \mathbf{j}\,\langle a_j,\varphi\rangle)}\right], \quad \varphi \in R^m, \tag{3.2}$$

where a_j is the jth column of matrix A.

Proof:
The proof of this statement is quite straightforward. Indeed, each variable $v^{(i)} \stackrel{\rm def}{=} e^{\mathbf{j}\,\varphi^{(i)}}$ of function $g_{\mathcal{X},c}(\cdot)$ is used for describing the status of the ith constraint. Consider the case $n = 1$. Then the set of all feasible values for y is comprised of $u^{(1)} + 1$ vectors

$$0,\ a_1,\ 2a_1,\ \ldots,\ u^{(1)}a_1.$$

In this case, each set $X_u(y)$ consists of a single integer vector. Thus, the expressions (3.1) and (3.2) are identical.

Assuming now that the representation (3.2) is valid for certain $n = p$, we can easily see that multiplication of this representation by the term

$$1 + \sum_{k=1}^{u^{(p+1)}} e^{k \cdot (c^{(p+1)} + \mathbf{j}\, \langle a_{p+1}, \varphi \rangle)}$$

properly modifies the feasible set for right-hand side y taking into account the appearance of the new variable $x^{(p+1)}$. Each term in this sum corresponds to a possible value of variable $x^{(p+1)}$, that is $0, 1, \ldots, u^{(p+1)}$. $\square$

Note that the representation (3.2) can be written in a short form:

$$g_{\mathcal{X},c}(e^{\mathbf{j}\varphi}) \;=\; \prod_{j=1}^{n} \left[1 + \sum_{k=1}^{u^{(j)}} e^{k \cdot (c^{(j)} + \mathbf{j}\, \langle a_j, \varphi \rangle)} \right] \;=\; \prod_{j=1}^{n} \frac{e^{(u^{(j)}+1) \cdot (c^{(j)} + \mathbf{j}\, \langle a_j, \varphi \rangle)} - 1}{e^{c^{(j)} + \mathbf{j}\, \langle a_j, \varphi \rangle} - 1}.$$

Thus, the value $g_{\mathcal{X},c}(e^{\mathbf{j}\varphi})$ can be computed in $O(mn)$ operations.

Let us estimate now the complexity of computing the value $\psi_{X(y)}(c)$ by (2.4). In order to do that, we need to estimate the size of the set $\Delta = AB(u)$. Let us assume that the elements of matrix A are bounded:

$$|A^{(i,j)}| \le \alpha, \quad i = 1, \ldots, m,\ j = 1, \ldots, n,$$

and that the box $B(u)$ is uniform:

$$u = \beta \cdot e,$$

where $e \in R^n$ is the vector of all ones. Then, for any $x \in B(u)$ we have

$$|\sum_{j=1}^{n} A^{(i,j)} x^{(j)}| \le \alpha\beta \cdot n. \quad i = 1, \ldots, m.$$

Hence, we can take

$$L^{(i)} = 1 + \alpha\beta \cdot n, \quad i = 1, \ldots, m.$$

In this case, computation of the value $\psi_{X(y)}(c)$ by expression (2.4) takes

$$O\left(mn \cdot (1 + \alpha\beta \cdot n)^m\right) \tag{3.3}$$

operations. If m is fixed, then this dependence is polynomial in n. On the contrary, the direct inspection of all integer vectors $x \in B(u)$, and verification of the system of linear equations $Ax = b$ takes

$$O\left(mn \cdot (1 + \beta)^n\right)$$

operations. The latter complexity bound is exponential in n.

Note that the above computation allows to solve also optimization problems by applying a bisection strategy with respect to the value of the objective function.

References

[1] A. Barvinok, and J.E. Pommersheim. An algorithmic theory of lattice points in polyhedra. In "New perspectives in algebraic combinatorics", *MSRI Publications*, 38 (1999), 91 – 147.

[2] A. Barvinok, and K. Woods. Short rational generating functions for lattice point problems. *Journal of the American Math. Society*, 16 (2003), 957 – 979.

[3] M. Brion, and M. Vergne. Residue formulae, vector partition functions and lattice points in rational polytopes. *Journal of the American Math. Society*, 10/4 (1997), 797 – 833.

[4] J.B. Lasserre. Generating functions and duality for integer programs. To be published in *Discrete Optimization*.

[5] J.A. De Loera, D. Haws, R. Hemmecke, P.Huggins, and R. Yoshida. Effective lattice point counting in rational convex polytopes. To appear in *Journal of Symbolic Computation*.

[6] J.A. De Loera, D. Haws, R. Hemmecke, P.Huggins, and R. Yoshida. Three kinds of integer programming algorithms based on Barvinok's rational functions. Accepted to IPCO 2004.

[7] G.L. Nemhauzer, and L.A. Wolsey. *Integer and Combinatorial Optimization*. Willey & Sons, New York 1988.

[8] Yu.Nesterov. Fast Fourier trasform and its applications to integer knapsack problems. CORE Discussion Paper #2004/64, September 2004.