



# Summing a polynomial function over integral points of a polygon. User's guide.

Velleda Baldoni, Nicole Berline, Michèle Vergne

## ► To cite this version:

Velleda Baldoni, Nicole Berline, Michèle Vergne. Summing a polynomial function over integral points of a polygon. User's guide.. 2009. hal-00383196

**HAL Id: hal-00383196**

**<https://hal.science/hal-00383196>**

Preprint submitted on 12 May 2009

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# SUMMING A POLYNOMIAL FUNCTION OVER INTEGRAL POINTS OF A POLYGON. USER'S GUIDE.

VELLEDA BALDONI, NICOLE BERLINE, AND MICHÈLE VERGNE

ABSTRACT. This document is a companion for the Maple program **Summing a polynomial function over integral points of a polygon**. It contains two parts. First, we see what this programs does. In the second part, we briefly recall the mathematical background.

## 1. INTRODUCTION

The present article is a user's guide for the Maple program **Summing a polynomial function over integral points of a polygon**, available at <http://www.math.polytechnique.fr/~berline/maple.html>. The Maple program contains two types of computation. The first computation does just what the title says. The input consists of a finite set of rational points in  $\mathbb{Q}^2$ , whose convex hull is a polygon  $\mathbf{p}$ , and a polynomial  $h(x, y)$  with rational coefficients. The output is the sum

$$\sum_{(x,y) \in \mathbf{p} \cap \mathbb{Z}^2} h(x, y).$$

The second computation returns the function of  $t \in \mathbb{N}$  which arises when the polytope  $\mathbf{p}$  is dilated by  $t$ .

$$E(t) := \sum_{(x,y) \in t\mathbf{p} \cap \mathbb{Z}^2} h(x, y).$$

This function is a *quasi-polynomial*, meaning that it has the form

$$E(t) = \sum_{i=0}^{\deg h + 2} E_i(t) t^i,$$

where the coefficients depend only on  $t \bmod q$ , where  $q$  is the smallest integer such that  $q\mathbf{p}$  has integral vertices. The function  $E(t)$  is called the *weighted Ehrhart quasi-polynomial* of  $\mathbf{p}$  with respect to the weight  $h(x, y)$ .

---

*Date:* May 2009.

We apply two methods, the first one for a fixed polygon, the second one for the computation of the weighted Ehrhart quasi-polynomial. The first method is based directly on Brion's formula (2), [3], while the second method is based on the local Euler-Maclaurin formula of [2]. Both methods use Barvinok's decomposition into unimodular cones [1]. Although they are very similar, the first method is faster when we deal with a fixed polygon, while the second is faster when we want the Ehrhart quasi-polynomial.

The software libraries *LattE* [4] (improved version in [5]) and *Barvinok* [6] include the computation of the number of points of a rational polytope in any dimension, together with many other applications. Moreover, the weighted Ehrhart polynomials in any dimension are computed in *Barvinok*. The present program, in dimension two, is based on the same principles: Brion's formula and Barvinok's decomposition of cones. We use however some new ideas on "renormalisation" of Laurent series from [2] to speed up the computation. In the future, we will generalize it to higher dimensions.

## 2. MAIN COMMANDS

**2.1. Summing a polynomial function over the set of integral points of a polygon.** Let  $P \subset \mathbb{Q}^2$  be a finite set of points. Let  $\mathbf{p} \subset \mathbb{R}^2$  be the polygon obtained as the convex hull of the set  $P$ . The program computes the sum

$$\sum_{(x,y) \in \mathbf{p} \cap \mathbb{Z}^2} h(x,y)$$

of the values of a polynomial  $h(x,y)$  over the set of integral points contained in  $\mathbf{p}$ . In particular, when  $h = 1$ , it computes the number of integral points in  $\mathbf{p}$ .

For a single monomial  $h(x,y) = x^{m_1}y^{m_2}$ , the command is

```
>sum_monomial_polygon(P,m);
```

Here  $P$  is a set of pairs of rational numbers, and  $m = [m_1, m_2]$  is a pair of non negative integers, the multidegree of the monomial  $x^{m_1}y^{m_2}$ .

If we want just the number of integral of integral points, we can use the command

```
>number_points_polygon(P);
```

This number can be also obtained by the command

```
>sum_monomial_polygon(polygon,[0,0]);
```

We compute the sum of a polynomial  $h(x,y)$  by the command

```
>sum_polynomial_polygon(P,h);
```

Here  $P$  is a set of pairs of rational numbers, and  $h = \sum_m h_m x^{m_1} y^{m_2}$  is entered as an expression in  $x, y$ .

**Example 1.**  $P$  is the square  $\{[0, 0], [1, 0], [1, 1], [0, 1]\}$ .

```
>square:= {[0,0],[1,0],[1,1],[0,1]};
```

```
>number_points_polygon(square);
```

4

The sum of values  $x^5 y^5$  over the 4 integral points in the square is

```
>sum_monomial_polygon(square,[5,5]);
```

1

**Example 2.** Here  $P$  is a randomly chosen set of 15 points.

```
> P := {[77/8,97/59],[93/44,70/29],[0,25/12],[25/32,29/48],
[92/41,57/91],[9/4,1/7],[64/43,31/75],[91/17,33/86],[12/37,77/8],
[8/5,41/27],[80/67,11/9],[16/73,11/89],[41/20,43/88],
[32/49,59/23],[77/94,65/46]};
```

The number of integral points in the convex hull is 45.

```
>number_points_polygon(P);
```

45

The vertices of the convex hull  $\mathbf{p}$  of  $P$  (listed in counter-clockwise order) are obtained with the command:

```
>vertices_in_counter_clock_order:=proc(polygon)
>vertices_in_counter_clock_order(P);
[[0,25/12],[16/73,11/89],[9/4,1/7],[91/17,33/86],[77/8,97/59],
[12/37,77/8]]
```

We compute the sum of  $x^{32} y^{32}$  over the set of integral points  $(x, y)$  of the convex hull  $\mathbf{p}$  of  $P$ .

```
>sum_monomial_polygon(P,[32,32]);
```

987532646688766560932727042325214847653263886

We compute the sum of  $x^{32} y^{32} + 7$  over all the integral points  $(x, y)$  of the polygon  $\mathbf{p}$ . (the preceding number  $+7$  times 45)

```
>h:= x^{32}y^{32}+7;
>sum_polynomial_polygon(P,h);
```

987532646688766560932727042325214847653264201

**2.2. Weighted Ehrhart polynomial of a polygon.** Our program computes also the weighted Ehrhart quasi-polynomial of a polygon. For brevity, we treat only the case where the weight is a monomial  $h(x, y) = x^{m_1} y^{m_2}$ . When the polygon is dilated by a non negative

integer  $t$ , and if  $q$  is a positive integer such that  $q\mathbf{p}$  has integral vertices, the function of  $t$  given by

$$t \mapsto \sum_{(x,y) \in t\mathbf{p} \cap \mathbb{Z}^2} x^{m_1} y^{m_2}$$

is a quasi-polynomial  $S(t) = \sum_{i=0}^{m_1+m_2+2} E_i(t)t^i$  of degree  $m_1 + m_2 + 2$ . The coefficients  $E_i(t)$  are functions of  $t$  modulo  $q$ . This program computes these coefficients  $E_i(t)$  in terms of the symbolic function  $fmod(p * t, q)$  which stands for  $(t \mapsto pt \bmod q)$ . We can either obtain each individual coefficient  $E_i(t)$  or the full weighted Ehrhart polynomial  $S(t)$ .

Here are the commands:

```
> coeff_t_Ehrhart_polygon(i,t,P,m);
```

The input consists of  $i$  an integer,  $t$  a letter,  $P$  a set of points and  $m = [m_1, m_2]$  a pair of integers which represents the weight; the output is the coefficient  $E_i(t)$ .

```
> Ehrhart_polynomial_polygon(t,P,m);
```

Input is as in the previous command, except  $i$  is not needed. The output is the full Ehrhart polynomial  $S(t)$ .

### Examples

```
> transsquare:={[-1/2,-1/2]{[1/2,-1/2],[1/2,1/2],[-1/2,1/2]};
```

```
> coeff_t_Ehrhart_polygon(0,t,square,[0,0]);
```

1

```
> coeff_t_Ehrhart_polygon(0,t,transsquare,[0,0]);
```

```
-2*fmod(t, 2)+3/2+1/2*fmod(t, 2)^2
```

```
> Ehrhart_polynomial_polygon(t,square,[0,0]);
```

1 + 2 t + t^2

```
> Ehrhart_polynomial_polygon(t,transsquare,[0,0]);
```

(fmod(t, 2)-1)^2+(-2\*fmod(t, 2)+2)\*t+t^2

**2.3. Experiments.** The following experiments were done with a laptop, processor 1,86 GHz, RAM 782 MHz, 0,99 Go.

```
> A:={[( -567337)/102495, -1414975/95662], [1/3, 1/5], [-88141/20499, 12732/47831]}
```

```
> largeA:={ [1000*( -567337)/102495, 1000*( -1414975/95662)],
[1000*1/3, 1000*1/5], [-1000*88141/20499, 1000*12732/47831]};
```

```
> number_points_polygon(A);
```

```
> number_points_polygon(largeA);
34922612
```

In the next experiments, we indicate the time of computation  $T$  in seconds. The number of integral points in the rational triangle with vertices  $A$  is 36. If we dilate  $A$  by the factor 1000, we obtain the triangle  $largeA$  where the number of points (34922612) is approximatively  $10^6$  times larger. Observe that we compute in 14 seconds the sum of the large degree monomial  $h(x, y) = x^{64}y^{64}$  over the set of integral points of  $A$ , and that we compute in 16 seconds the sum of the same monomial over the integral points of  $largeA$ . The computation time is almost the same, although any computation by enumeration would be  $10^6$  times longer.

```
> T:=time(): sum_monomial_polygon(A, [32,32]); Time:=time()-T;
11156693714080121436809683716369682546812787494001398139657
Time := 1.766
```

```
> T:=time(): sum_monomial_polygon(A, [64,64]); Time:=time()-T;
10691662746975383171690687952963005219723639375189814
217756070191566530558879\
3836513555847334896253718879462978590217
Time := 13.640
```

```
> T:=time(): sum_monomial_polygon(largeA, [64,64]): Time:=time()-T;
1783103591372206604358967784049666198798919356345005767183297976710270822606
90519522342895765988221612337480372436229072894493363579270305297678267123
40160119137597718403779959778986161713238013119891186401529359213636522185
95244921491613319792892241946298996049559369929767070065285383458443917290
85791611969462010599632957347801451344938373887397255088905193762020134177
82911075684183735887058845417207962477000059284528111310251783657942905087
20099703621578931359063825440122383120351301766010118556183
Time:= 15.516
```

Finally, we computed the weighted Ehrhart polynomial with weight  $x^{32}y^{32}$  over the triangle with vertices  $[[-567337/102495, -1414975/95662], [88141, 292844676/6833], [-88141/20499, 12732/47831]]$ . We compute the coefficient of  $t^2$  for example. The time of computation is 268 seconds. The result is too big to be printed here, as it involves many functions  $f \bmod(c * t, D)$  where  $D$  runs through the denominators of the coordinates of the vertices of  $A$  (large numbers).

```
> T:=time(): coeff_t_Ehrhart_polygon(2,t, [[-567337/102495,
```

```

-1414975/95662], [88141, 292844676/6833], [-88141/20499,
12732/47831]], [32, 32]): Time:=time()-T;
Time := 268.266

```

### 3. MATHEMATICAL BACKGROUND

The first method is for a fixed polygon, the second one for the computation of the weighted Ehrhart quasi-polynomial.

**3.1. First method: Brion's formula, Barvinok's decomposition into unimodular cones and iterated Laurent series.** Let  $\mathbf{p}$  be a convex polygon in  $\mathbb{R}^2$  with rational vertices  $s_i$ ,  $1 \leq i \leq n+1$ . We want to compute the sum

$$(1) \quad \sum_{x \in \mathbf{p} \cap \mathbb{Z}^2} x^{m_1} y^{m_2}.$$

We start by observing that (1) is equal to the coefficient of  $\frac{\xi_1^{m_1} \xi_2^{m_2}}{m_1! m_2!}$  in

$$\sum_{x \in \mathbf{p} \cap \mathbb{Z}^2} e^{\langle \xi, x \rangle}.$$

Our method is based on Brion's formula (2). Brion's formula is the generalization of the following formula for the sum of geometric progressions over the interval  $[A, B]$  (with  $A \leq B$  integers):

$$\sum_A^B e^{n\xi} = \frac{e^{A\xi}}{1 - e^\xi} + \frac{e^{B\xi}}{1 - e^{-\xi}}.$$

For any rational polygon  $\mathbf{q} \subset \mathbb{R}^2$  define

$$S(\mathbf{q})(\xi) = \sum_{x \in \mathbf{q} \cap \mathbb{Z}^2} e^{\langle \xi, x \rangle}.$$

This is a meromorphic function near  $\xi = 0$ . Moreover the map  $\mathbf{q} \mapsto S(\mathbf{q})(\xi)$  is a valuation on the set of rational polyhedra, and  $S(\mathbf{q}) = 0$  if  $\mathbf{q}$  contains a line. Brion's formula is the following. Let  $\mathbf{c}_i$  be the cone at vertex  $s_i$  of the polygon.

$$(2) \quad S(\mathbf{p}) = \sum_{i=1}^{n+1} S(\mathbf{c}_i).$$

Each term  $S(\mathbf{c}_i)(\xi) = \sum_{x \in \mathbf{c}_i \cap \mathbb{Z}^2} e^{\langle \xi, x \rangle}$  in (2) is a meromorphic function near  $\xi = 0$ . The poles cancel and the sum is a holomorphic function of  $\xi$ . Thus we compute (1) as the coefficient of  $\frac{\xi_1^{m_1} \xi_2^{m_2}}{m_1! m_2!}$  in the right-hand-side of (2). We actually compute the individual contribution of each cone  $\mathbf{c}_i$  (associated to the vertex  $s_i$ ) to the sum. The coefficient of

$\frac{\xi_1^{m_1}\xi_2^{m_2}}{m_1!m_2!}$  in the meromorphic function  $S(\mathbf{c}_i)$  of two variables  $\xi_1, \xi_2$  has no intrinsic meaning. Our method consists in applying **iterated Laurent series** expansions to  $S(\mathbf{c}_i)(\xi)$  with respect to the variables  $\xi_1$  then  $\xi_2$ . We obtain a Laurent series  $L(\mathbf{c}_i)$  in the ring  $\mathbb{Q}[\xi_1, \xi_1^{-1}, \xi_2, \xi_2^{-1}]$  and we compute the coefficient  $\frac{\xi_1^{m_1}\xi_2^{m_2}}{m_1!m_2!}$  in  $L(\mathbf{c}_i)$ .

Thus, in order to compute the contribution of a vertex  $s$  to the sum (2), we need to compute  $S(\mathbf{c})(\xi)$  for the supporting cone  $\mathbf{c}$ . The crucial tool here is Barvinok's decomposition into unimodular cones. Actually, we use the following variant of Barvinok's decomposition, (procedure `signed_decomp`).

Let  $\mathbf{c}$  be a simplicial cone in  $\mathbb{R}^d$ . Let  $V_i$ , for  $i = 1, \dots, d$ , be the generators of  $\mathbf{c}$ . Let  $V$  be a vector in  $\mathbb{R}^d$ . We write  $V = \sum_i u_i V_i$ . We split  $[V_1, \dots, V_d]$  into three parts, as follows.

$$L_+ := [X_1, \dots, X_k]$$

formed by the  $V_i$  such that  $u_i > 0$ ,

$$L_- := [Y_1, \dots, Y_m]$$

formed by the  $V_i$  such that  $u_i < 0$ ,

$$L_0 := \{Z_1, \dots, Z_b\}$$

formed by the  $V_i$  such that  $u_i = 0$ .

Then we have the equality of characteristic functions modulo characteristic functions of cones containing lines.

$$\begin{aligned} (-1)^{(k+1)}[\mathbf{c}] &= \sum_{i=1}^k (-1)^{i+1} [\mathbf{c}(X_1, \dots, X_{i-1}, -X_{i+1}, \dots, -X_k, V, L_-, L_0)] + \\ &\sum_{j=1}^m (-1)^{j+k} [\mathbf{c}(L_+, -V, -Y_1, \dots, -Y_{j-1}, Y_{j+1}, \dots, Y_m, L_0)]. \end{aligned}$$

**Remark.** This decomposition is not the stellar decomposition. It involves only cones of maximal dimension  $d$ . It avoids the dualizing trick of Brion.

**Example.**  $\mathbf{c} = \mathbb{R}^+ e_1 \oplus \mathbb{R}^+ e_2$ ,  $V = e_1 + e_2$ , so that  $L_-$  and  $L_0$  are empty and  $k = 2$ . Then

$$-[\mathbf{c}] = \mathbf{c}(V, -e_2) - \mathbf{c}(e_1, V) - \mathbf{c}[e_2, -e_2, e_1].$$

Indeed  $[\mathbf{c}(e_2, -e_2, e_1)] - [\mathbf{c}]$  is equal to the characteristic function of the quadrant  $(e_1, -e_2)$  minus that of the half-line  $\mathbb{R}^+ e_1$ . This is also the case for  $[\mathbf{c}(V, -e_2)] - [\mathbf{c}(e_1, V)]$ .



If we use a lattice vector  $V$  with sufficiently small coordinates in the basis  $(V_i)$ , the cones appearing in this decomposition have indices smaller than  $\mathbf{c}$ . One obtains such a *short* vector  $V$  by the Lenstra-Lenstra-Lovasz algorithm. By a repeated application of this decomposition, one obtains a decomposition of  $\mathbf{c}$  in a signed sum of unimodular cones  $\mathbf{c}_z$  (modulo cones containing lines). As  $S(\mathbf{a}) = 0$  for a cone  $\mathbf{a}$  which contains a line, we can use this decomposition to compute  $S(\mathbf{c})$ .

For a unimodular cone  $\mathbf{c}$ , the sum  $S(\mathbf{c})$  has a simple closed expression. Let  $(V_1, V_2)$  be primitive generators of the edges of  $\mathbf{c}$  and let  $s$  be its vertex. Let  $\tilde{s}$  be the unique integral point contained in the *semi-closed box*

$$\{s + t_1 V_1 + t_2 V_2, 0 \leq t_i < 1\}$$

If  $s = s_1 V_1 + s_2 V_2$ , then  $\tilde{s} = \tilde{s}_1 V_1 + \tilde{s}_2 V_2$  with  $\tilde{s}_i = \text{ceil}(s_i)$ . Then

$$(3) \quad S(\mathbf{c})(\xi) = \frac{e^{\langle \xi, \tilde{s} \rangle}}{(1 - e^{\langle \xi, V_1 \rangle})(1 - e^{\langle \xi, V_2 \rangle})}.$$

In order to simplify the computation of iterated Laurent series, we introduce the analytic function

$$B(X, u) = \frac{e^{uX}}{1 - e^X} + \frac{1}{X} = - \sum_{n=0}^{\infty} \frac{b(n+1, u)}{(n+1)!} X^n$$

where  $b(n, u)$  are the Bernoulli polynomials. Writing

$$(4) \quad \frac{e^{uX}}{1 - e^X} = B(X, u) - \frac{1}{X},$$

we obtain

$$S(\mathbf{c})(\xi) = A + G + R,$$

where

$$A = B(\langle \xi, V_1 \rangle, \tilde{s}_1) B(\langle \xi, V_2 \rangle, \tilde{s}_2)$$

is an analytic function of  $\xi$ ,

$$G := -\frac{1}{\langle \xi, V_1 \rangle} B(\langle \xi, V_2 \rangle, \tilde{s}_2) - \frac{1}{\langle \xi, V_2 \rangle} B(\langle \xi, V_1 \rangle, \tilde{s}_1),$$

$$R := \frac{1}{\langle \xi, V_1 \rangle \langle \xi, V_2 \rangle}.$$

We replace  $\frac{1}{\langle \xi, V_1 \rangle}$  and  $\frac{1}{\langle \xi, V_2 \rangle}$  by their iterated Laurent series expansion in the ring  $R[\xi_1, \xi_1^{-1}, \xi_2, \xi_2^{-1}]$ . For example, if  $V_1 = [2, 1]$ , we write

$$\frac{1}{2\xi_1 + \xi_2} = \frac{1}{\xi_2} \frac{1}{(1 + 2\xi_1/\xi_2)} = \frac{1}{\xi_2} \sum_{k=0}^{\infty} (-1)^k 2^k (\xi_1/\xi_2)^k.$$

We then replace  $S(\mathbf{c})$  by the corresponding element in  $\mathbb{Q}[[\xi_1, \xi_2, \xi^{-1}, \xi^{-2}]]$  and we take the coefficient of  $\xi_1^{m_1} \xi_2^{m_2}$ .

**Remark** The weighted Ehrhart polynomial can also be computed by this method. We did not write the corresponding algorithm in the Maple file, because we observed that a faster algorithm is given by the second method which we describe in the next section. However, let us explain what one should do. When the polytope  $\mathbf{p}$  is dilated in  $t\mathbf{p}$ , its vertices are dilated by  $t$ , while the edges of the cones at vertices do not change. Thus we have to compute

$$(5) \quad S(\mathbf{c}_t)(\xi) = \frac{e^{\langle \xi, \tilde{s}_t \rangle}}{(1 - e^{\langle \xi, V_1 \rangle})(1 - e^{\langle \xi, V_2 \rangle})}$$

where now  $s_t$  is the unique point with integral coordinates in the box

$$\{ts + u_1 V_1 + u_2 V_2, 0 \leq u_i < 1\}.$$

If  $s = s_1 V_1 + s_2 V_2$ , with  $s_i = p_i/q_i$ , we see that

$$s_t = [ts_1 + \text{mod}(-tp_1, q_1)/q_1, ts_2 - \text{mod}(-tp_2, q_2)/q_2].$$

The iterated Laurent series in  $\xi_1, \xi_2$  has coefficients which are polynomials in  $t$  and the periodic functions  $\text{mod}(tp_i, q_i)$ . We extract the coefficient of  $t^j \xi_1^{m_1} \xi_2^{m_2}$ .

**3.2. Second method. Weighted Ehrhart quasi-polynomial using local Euler-Maclaurin formula.** We now recall the results of [2] and explain how they can be applied to the computation of the weighted Ehrhart quasi-polynomials. Let  $\mathbf{p}$  be a convex polytope in  $\mathbb{R}^d$ , with rational vertices. Let  $h(x)$  be a polynomial function of degree  $r$  on  $\mathbb{R}^d$ . We want to compute the sum  $\sum_{x \in \mathbf{p} \cap \Lambda} h(x)$  of values  $h(x)$  over the set of integral points of the polytope  $\mathbf{p}$ .

The local Euler-Maclaurin formula has the following form.

$$(6) \quad \sum_{x \in \mathbf{p} \cap \Lambda} h(x) = \sum_{\mathbf{f} \in \mathcal{F}(\mathbf{p})} \int_{\mathbf{f}} D(\mathbf{p}, \mathbf{f}) \cdot h$$

where  $\mathcal{F}(\mathbf{p})$  is the set of all faces of  $\mathbf{p}$ . For each face  $\mathbf{f}$ ,  $D(\mathbf{p}, \mathbf{f})$  is a differential operator of infinite degree with constant coefficients associated to  $\mathbf{f}$ . The operator  $D(\mathbf{p}, \mathbf{f})$  is *local*, in the sense that it depends only on the intersection of  $\mathbf{p}$  with a neighborhood of any generic point of  $\mathbf{f}$ . The integral on the face  $\mathbf{f}$  is taken with respect to the Lebesgue measure on  $\langle \mathbf{f} \rangle$  defined by the lattice  $\mathbb{Z}^d \cap \text{lin}(\mathbf{f})$ . Here  $\langle \mathbf{f} \rangle$  is the affine span of the face  $\mathbf{f}$  and  $\text{lin}(\mathbf{f})$  is the linear subspace parallel to  $\langle \mathbf{f} \rangle$ .

Let us recall the construction of the operators  $D(\mathbf{p}, \mathbf{f})$ . We denote by  $\mathbf{t}(\mathbf{p}, \mathbf{f})$  the transverse cone to  $\mathbf{p}$  along  $\mathbf{f}$ . Using the standard scalar

product,  $\mathbf{t}(\mathbf{p}, \mathbf{f})$  is described as the following affine cone in  $\mathbb{R}^d$ . Let  $\text{lin}(\mathbf{f})^\perp$  be the vector subspace orthogonal to  $\text{lin}(\mathbf{f})$ . Then  $\mathbf{t}(\mathbf{p}, \mathbf{f})$  is the orthogonal projection on  $\text{lin}(\mathbf{f})^\perp$  of the supporting cone of  $\mathbf{p}$  along  $\mathbf{f}$ . The operator  $D(\mathbf{p}, \mathbf{f})$  is defined in terms of the transverse cone  $\mathbf{t}(\mathbf{p}, \mathbf{f})$ , as follows.

For every rational affine cone  $\mathbf{a} \subset V$ , we construct in [2] an analytic function  $\xi \mapsto \mu(\mathbf{a})(\xi)$  on  $\mathbb{R}^d$ . This construction depends on the choice of a scalar product. Here we use the standard scalar product. These functions  $\mu(\mathbf{a})$  have nice properties which play a crucial role in our method. First, the assignment  $\mathbf{a} \mapsto \mu(\mathbf{a})$  is a *valuation* on the set of affine cones with a given vertex. Second, it is *invariant under lattice translations*. Furthermore,  $\mu(\mathbf{a}) = 0$  if  $\mathbf{a}$  contains a line.

We define

$$D(\mathbf{p}, \mathbf{f}) = D(\mu(\mathbf{t}(\mathbf{p}, \mathbf{f})))$$

as the differential operator of infinite degree with constant coefficients, with symbol  $\mu(\mathbf{t}(\mathbf{p}, \mathbf{f}))(\xi)$ . In other words, if  $\xi = (\xi_1, \dots, \xi_d)$ , we obtain  $D(\mathbf{p}, \mathbf{f})$  by replacing  $\xi_i$  by  $\frac{\partial}{\partial x_i}$  in the Taylor series of  $\mu(\mathbf{t}(\mathbf{p}, \mathbf{f}))(\xi)$ .

For any positive integer  $t$ , we consider the dilated polytope  $t\mathbf{p}$  and the corresponding sum

$$S(t\mathbf{p}, h) = \sum_{x \in t\mathbf{p} \cap \Lambda} h(x).$$

From (6), it follows easily that the function  $t \mapsto S(t\mathbf{p}, h)$  is given by a quasi-polynomial: there exist periodic functions  $t \mapsto E_i(\mathbf{p}, h, t)$  on  $\mathbb{N}$  such that

$$(7) \quad S(t\mathbf{p}, h) = \sum_{i=0}^{d+r} E_i(\mathbf{p}, h, t) t^i$$

whenever  $t$  is a positive integer. Moreover the coefficients  $E_i(\mathbf{p}, h, t)$  are computed using the functions  $\mu(\mathbf{t}(t\mathbf{p}, t\mathbf{f}))$ . Indeed, let  $s$  be the vertex of  $\mathbf{t}(\mathbf{p}, \mathbf{f})$  so that  $\mathbf{t}(\mathbf{p}, \mathbf{f}) = s + \mathbf{t}_0$ . Then the dilated transverse cone is  $\mathbf{t}(t\mathbf{p}, t\mathbf{f}) = ts + \mathbf{t}_0$ . As  $\mathbf{a} \mapsto \mu(\mathbf{a})$  is invariant under lattice translations, we have

$$\mu(\mathbf{t}((t+q)\mathbf{p}, t\mathbf{f})) = \mu(\mathbf{t}(t\mathbf{p}, t\mathbf{f})),$$

if  $q$  is an integer such that  $qs$  is a lattice point for the projected lattice, or equivalently, such that  $q < \mathbf{f} >$  contains a lattice point. Thus, the coefficients  $E_i(\mathbf{p}, h, t)$  depend only on  $t \bmod q$ , where  $q$  is the smallest integer such that  $q\mathbf{p}$  has integral vertices.

When  $\mathbf{a}$  is a *unimodular* affine cone of dimension 1 or 2, the functions  $\mu(\mathbf{a})$  have an explicit form, in terms of the functions  $B(X, u)$  introduced in (4).

Let  $\mathfrak{d}$  be a one dimensional affine cone of the form  $(s + \mathbb{R}_+)V$  where  $V$  is a primitive vector and  $s \in \mathbb{Q}$ . We have

$$(8) \quad \mu(\mathfrak{d})(\xi) = B(\langle \xi, V \rangle, \text{ceil}(s) - s).$$

Let  $\mathfrak{a}$  be a two dimensional *unimodular* affine cone. Let  $V_1, V_2$  be primitive generators of its edges, such that  $\det(V_1, V_2) = 1$ . For  $\xi = (\xi_1, \xi_2) \in \mathbb{R}^2$ , let  $y_i = \langle \xi, V_i \rangle$ , for  $i = 1, 2$ , be the coordinates of  $\xi$  relative to the dual basis  $(V_1^*, V_2^*)$ . We write the vertex of  $\mathfrak{a}$  as  $s_1 V_1 + s_2 V_2$  with  $s_i \in \mathbb{Q}$ . Let  $\epsilon_i = \text{ceil}(s_i) - s_i$ , and let  $C_i = \frac{\langle V_1, V_2 \rangle}{\langle V_i, V_i \rangle}$ , for  $i = 1, 2$ . With these notations, we have

$$(9) \quad \mu(\mathfrak{a})(\xi) = \frac{e^{\epsilon_1 y_1 + \epsilon_2 y_2}}{(1 - e^{y_1})(1 - e^{y_2})} + \frac{1}{y_1} B(y_2 - C_1 y_1, \epsilon_2) + \frac{1}{y_2} B(y_1 - C_2 y_2, \epsilon_1) - \frac{1}{y_1 y_2}.$$

The function  $\mu(\mathfrak{a})(\xi)$  is actually analytic, although this is not obvious on (9). In order to compute the contribution of a vertex  $s$  of  $\mathfrak{p}$  to the sum (6), we need to compute  $\mu(\mathfrak{c})(\xi)$  when  $\mathfrak{c}$  is the two-dimensional supporting cone at  $s$ . The crucial tool here is Barvinok's decomposition into unimodular cones. The valuation property of  $\mathfrak{a} \mapsto \mu(\mathfrak{a})$  makes it possible to reduce the computation to the unimodular case, and use (9). Notice that (9) returns a function of the relative coordinates  $(y_1, y_2)$ , which we must convert back to a function of the standard coordinates  $(\xi_1, \xi_2)$ , in order to add the contributions of the various unimodular cones in Barvinok's decomposition. Actually, since  $\mu(\mathfrak{a}) = 0$  if the cone  $\mathfrak{a}$  contains a line, we use the variant of Barvinok's decomposition described in the first method.

## REFERENCES

- [1] **Barvinok A. I.**, *A polynomial time algorithm for counting integral points in polyhedra when the dimension is fixed*, Math. Oper. Res. **19** (1994), 769-779.
- [2] **Berline N. and Vergne M.**, *Local Euler-Maclaurin formula for polytopes*, Moscow Math. Journal, **7** (2007), 355-386. arXiv:math.CO/0507256.
- [3] **Brion M.**, *Points entiers dans les polyèdres convexes*, Ann. Sci. Ecole Norm. Sup. **21** (1988), 653-663.
- [4] **De Loera J.A., Haws D., Hemmecke R., Huggins H., Tauzer J. and Yoshida R.**, *A Users Guide for LattE v1.1, 2003*, software package LattE, available at <http://www.math.ucdavis.edu/latte>.
- [5] **Koepe M.**, *A primal Barvinok algorithm based on irrational decompositions*, SIAM Journal on Discrete Mathematics, **21** (2007), pp. 220-236. Software *LattE macchiato* available at <http://www.math.ucdavis.edu/mkoepe/latte/>.

- [6] **Verdoolaege S. and Bruynooghe M.**, *Algorithms for weighted counting over parametric polytopes: A survey and a practical comparison*, Eighth ACES Symposium, Edegem, Belgium, September 2008. <https://lirias.kuleuven.be/handle/123456789/197757>. Software available at <http://freshmeat.net/projects/barvinok/>.

UNIVERSITA DI ROMA TOR VERGATA, DIPARTIMENTO DI MATEMATICA, VIA  
DELLA RICERCA SCIENTIFICA, 00133 ROMA, ITALY  
*E-mail address:* `baldoni@mat.uniroma2.it`

ECOLE POLYTECHNIQUE, CENTRE DE MATHÉMATIQUES LAURENT SCHWARTZ,  
91128, PALAISEAU, FRANCE  
*E-mail address:* `berline@math.polytechnique.fr`

INSTITUT DE MATHÉMATIQUES DE JUSSIEU, THÉORIE DES GROUPES, CASE  
7012, 2 PLACE JUSSIEU, 75251 PARIS CEDEX 05, FRANCE

ECOLE POLYTECHNIQUE, CENTRE DE MATHÉMATIQUES LAURENT SCHWARTZ,  
91128, PALAISEAU, FRANCE  
*E-mail address:* `vergne@math.polytechnique.fr`