A Sweep-Plane Algorithm for Generating Random Tuples in Simple Polytopes

Josef Leydold
University of Economics and Business Administration
Department for Applied Statistics and Data Processing
Augasse 2-6, A-1090 Vienna, Austria
email: `Josef.Leydold@wu-wien.ac.at`

Wolfgang Hörmann
University of Economics and Business Administration
Department for Applied Statistics and Data Processing
Augasse 2-6, A-1090 Vienna, Austria

August 8, 1997

Abstract

A sweep-plane algorithm by Lawrence for convex polytope computation is adapted to generate random tuples on simple polytopes. In our method an affine hyperplane is swept through the given polytope until a random fraction (sampled from a proper univariate distribution) of the volume of the polytope is covered. Then the intersection of the plane with the polytope is a simple polytope with smaller dimension.

In the second part we apply this method to construct a black-box algorithm for log-concave and T-concave multivariate distributions by means of transformed density rejection.

Mathematics Subject Classification: 65C10 (Random Number Generation)

General Terms: Algorithms

Additional Key Words and Phrases: Uniform distributions, polytope, rejection method, multivariate log-concave distributions, universal method

Contents

1 Introduction

Several methods were suggested for the generation of uniformly distributed random points on a n-polytope P.

(1) If P is a simplex, then by [8, chapter XI.2.5, theorem 2.1] we get such a point by

$$\mathbf{x} = \sum_{j=0}^{n} U_j \, \mathbf{v}_j \quad \text{where} \quad \sum_{j=0}^{n} U_j = 1 \tag{1}$$

where $\mathbf{v}_0, \ldots, \mathbf{v}_n$ are the vertices of the simplex P and the U_j are generated by a uniform sample on $[0, 1]$.

(2) Consequently a method for arbitrary polytopes is to triangulate P [8, chapter XI.2.5], i.e. we split the polytopes into n-simplices. Triangulation works well for polygons ($\dim(P) = 2$) but is rather difficult for dimension greater than 2. The complexity of the decomposition step depends on the number of faces which is $O(m^{\lfloor n/2 \rfloor})$, where m is the number of vertices ([23], cf. [8, chapter XI.2.5]).

(3) Another approach are grid methods (see [8, chapter VIII.3.2]). The polytope is enclosed in a hyper-rectangle, which is decomposed in a set of grid rectangles. In a setup step the grid rectangles are classified into inside, outside or on the border of the polytope. Those inside and on the border are stored and randomly chosen by the sampling algorithm. Then a point inside the small rectangle is generated. It is accepted if it is inside the polytope. It is clear that the number of necessary grid rectangles explodes for higher dimensions. Thus for higher dimensions the method has advantages and disadvantages comparable with the below method (4).

(4) A possible variant of (2) and (3) is to enclose P into a simplex (L. Devroye, private correspondence). Here the rejection constant for an arbitrary polytope is of order $O(\dim(P)!) = O(n!)$ but strongly depends on the shape of the polytope. Furthermore it is not a simple task to find the enclosing simplex (e.g. if P has parallel constraints.).

A new approach to the problem is the use of a sweep-plane technique. This technique goes back to Hadwiger [16, 17], who used it in the context of Euler characteristic on the convex ring. It was applied to volume computation by Bieri&Nef [5] and (with a different name) by Lawrence [22] (see also [14]). In [4] a recursive algorithm is used to count the cells of a finite division in $\mathbb{R}^n$.

The general idea of sweep-plane algorithms is to "sweep" a hyperplane through a polytope P, keeping track of the changes that occur when the hyperplane sweeps through a vertex.

For our purpose the plane is swept through the given simple polytope until a random fraction of the volume of the polytope is covered. This fraction is given by a uniform sample on $[0, 1]$. The intersection of the plane with the polytope is a simple polytope with smaller dimension. By recursion we arrive at a polytope of dimension 0, that is, a single point. The complexity for the first recursion step is $O(m^2 + mn^3)$.

Although only derived for the convex case, the sweep plane algorithm also works for non-convex polytopes but in contrast to the methods (2), (3) and (4) only for simple polytopes. Compared with methods (2) and (3) the setup of the new algorithm is much faster but the generation time is slower. Compared to (3) and (4) the new algorithm has the advantage that there is no rejection necessary. The main advantage of our new generation procedure is that the complexity does not grow as fast with the dimension as the methods suggested in literature. There are problems with rounding errors in higher dimensions but they can be overcome using exact rational arithmetic.

In the second part we apply this method to construct a black-box algorithm for log-concave and T-concave multivariate distributions by following the idea of transformed density rejection, introduced in [12] and [18] for the univariate case. The sweep-plane technique is used to generate random variates with respect to the hat-function.

2 Sweep-plane algorithm for simple polytopes

2.1 Simple convex polytopes.

Let us first summarize the concept of polytopes (see [30]; an introduction into convex polytopes can further be found in [15, 3] or from a different point of view in [10, 25]).

A $\mathcal{H}$-polyhedron is an intersection of finitely many closed half-spaces in $\mathbb{R}^n$. In linear programming the inequalities $\langle \mathbf{c}, \mathbf{x} \rangle \leq c_0$ that define the half-spaces are called *constraints*. A $\mathcal{V}$-polyhedron is the convex hull of some points in $\mathbb{R}^n$. A *polyhedron* is a point set $P \in \mathbb{R}^n$ which can be presented either as a $\mathcal{V}$-polyhedron or as an $\mathcal{H}$-polyhedron. Both representations are equivalent and can be converted to each other (for example the reverse search algorithm [2] or the double description method [24, 11]).

A (convex) *polytope* is a bounded polyhedron. A polyhedron (polytope) of dimension n is called a n-polyhedron (n-polytope). A *face* of a polyhedron is a polyhedron $F \subseteq P$ which is the intersection of P with some supporting hyperplane $\{\mathbf{x} \in \mathbb{R}^n : \langle \mathbf{c}, \mathbf{x} \rangle = c_0\}$ where $\langle \mathbf{c}, \mathbf{x} \rangle \leq c_0$ is satisfied for all points $\mathbf{x} \in P$. A face of dimension i is called an i-face. The 0-faces, 1-faces and $(n-1)$-faces of an n-polytope are respectively its *vertices*, *edges* and *facets*. A n-polyhedron is called *simple* if every vertex is contained in the minimal number of only n facets, that is only n constraints are binding in each vertex. (In linear programming this is called the *non-degenerate* case.)

Notice that if we consider any set of constraints that are generic (i.e. they define hyperplanes in general position) then this defines a simple polyhedron. In the generic case a $\mathcal{H}$-polytope is always simple. On the other hand if we choose points in $\mathbb{R}^n$ in general position (i.e. no n of these are affine dependent) then its convex hull is simplicial (all proper faces are simplices) but may not be simple.

2.2 Sweeping planes

Sweep-plane. Let P a simple convex n-polytope and $f(x)$ the density function of the uniform distribution on P. For simplicity we set $f(x) \equiv 1$. Choose a nonzero vector $\mathbf{g}$. In what follows we assume

$$\|\mathbf{g}\| = 1 \qquad \text{and} \qquad \langle \mathbf{g}, \mathbf{x} \rangle \text{ is non-constant on every edge in } P \tag{2}$$

$\langle \cdot, \cdot \rangle$ denotes the scalar product. For a given $\mathbf{x}$ let $x = \langle \mathbf{g}, \mathbf{x} \rangle$. We denote the hyperplane perpendicular to $\mathbf{g}$ through $\mathbf{x}$ by

$$F(\mathbf{x}) = F(x) = \{\mathbf{y} \in \mathbb{R}^n : \langle \mathbf{g}, \mathbf{y} \rangle = x\} \tag{3}$$

and its intersection with the polytope P with $Q(\mathbf{x}) = Q(x) = P \cap F(x)$. ($F(\mathbf{x})$ and $Q(\mathbf{x})$ depend on x only; thus we write $F(x)$ and $Q(x)$, respectively, if there is no risk of confusion.) $Q(x)$ again is a convex polytope (see [15]). Now we can move this sweep-plane $F(x)$ through the domain P by varying x. Figure 1 illustrates the situation. The marginal density function $h_{\mathbf{g}}(x)$ along $\mathbf{g}$ of a uniform distribution with support P is simply given by the volume $A(x)$ of $Q(x)$. We can sample a variate x from the marginal distribution and get the polytope $Q(x)$ (see §2.3 and §2.4).

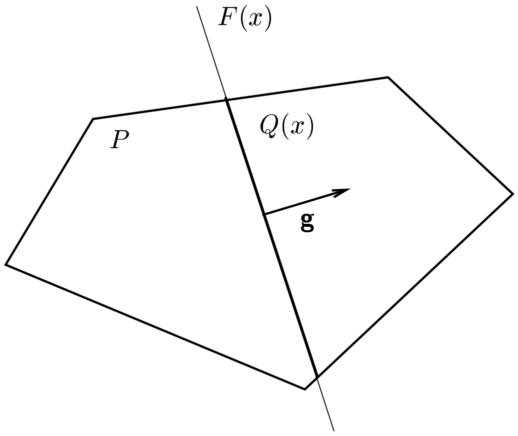

Figure 1: sweep-plane $F(x)$

Recursive sweep-plane algorithm. If $\dim(Q(x)) = 1$, $Q(x)$ is a line segment and we get the random point by

$$\mathbf{x} = U\,\mathbf{v}_0 + (1-U)\,\mathbf{v}_1 \tag{4}$$

where $\mathbf{v}_0$, $\mathbf{v}_1$ are the vertices of $Q(x)$ and U is a uniform sample on $[0,1]$.

If $\dim(Q(x)) \geq 2$ let $Q_{n-1} = Q(x)$. Then Q_{n-1} is a simple $(n-1)$-polytope (immediately from [30, proposition 2.16]). We embed Q_{n-1} into the $\mathbb{R}^{n-1}$ by eliminating the component in $\mathbf{x}$, where $\mathbf{g}$ takes its maximum.

$$\begin{gathered}\mathbf{x} = (x_1,\dots,x_M,\dots,x_n) \mapsto \mathbf{x}' = (x_1,\dots,x_{M-1},x_{M+1},\dots,x_n) \\ \text{where} \qquad g_M = \max_{j=1,\dots,n} g_j\end{gathered} \tag{5}$$

If the maximum of g_j is not unique, we set M to the first index that maximizes g_j.

We get a polytope $Q'_{n-1} \subset \mathbb{R}^{n-1}$ (see figure 2). We choose a proper $\mathbf{g}_{n-1} \in \mathbb{R}^{n-1}$

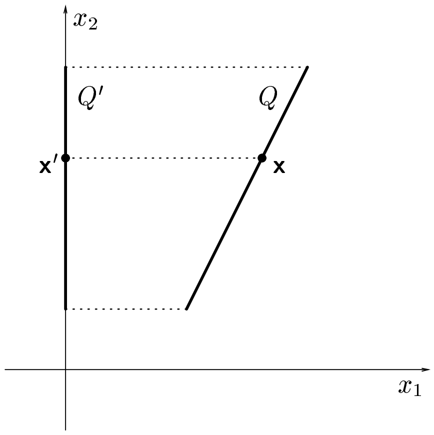

Figure 2: Projection $Q \subset \mathbb{R}^d \to Q' \subset \mathbb{R}^{d-1}$

which satisfies (2), that is, $\|\mathbf{g}_{n-1}\| = 1$ and $\mathbf{x}' \mapsto \langle \mathbf{g}_{n-1}, \mathbf{x}' \rangle$ is non-constant on every edge of Q'_{n-1}. Again we use a sweep plane $F_{n-1}(x)$ and a univariate random number generator to get a $(n-2)$-polytope Q_{n_2}.

Now we apply the same method to Q_{n-2} and get a polytope Q_{n-3} and so on, until we finish with a polytope Q_1 of dimension 1 for which we use (4).

We get the random point $\mathbf{x} \in Q(x) = Q_{n-1}$ from $\mathbf{x}' \in Q_{n-2}$ by the fact that $\langle \mathbf{g}, \mathbf{x} \rangle = x$ (see figure 2). Thus we find

$$\begin{aligned} &\mathbf{x} = (x_1, \ldots, x_n) = (y_1, \ldots, y_{M-1}, x_M, y_{M+1}, \ldots, y_n) \\ &\text{with} \quad x_M = \tfrac{x - \langle \mathbf{g}, \mathbf{y} \rangle}{g_M} \\ &\text{where} \quad \mathbf{y} = (y_1, \ldots, y_n) = (x'_1, \ldots, x'_{M-1}, 0, x'_M, \ldots, x'_{n-1}) \end{aligned} \tag{6}$$

2.3 Sample from marginal distribution

Computing volume $A(x)$. Let $F^-(x) = \{\mathbf{y} \in \mathbb{R}^n : \langle \mathbf{g}, \mathbf{y} \rangle \le x\}$. By assumptions $P \cap F^-(x)$ is bounded. Thus we find

$$A(x) = V_{n-1}(P \cap F(x)) = \frac{dV_n(P \cap F^-(x))}{dx} \tag{7}$$

where V_k denotes the k-dimensional volume. To compute $A(x)$ we modify the method in [5, 22, 14]. Denote the vertices of P by $\mathbf{v}_j \in \mathbb{R}^n$, $j = 1, \ldots, m$ and $v_j = \langle \mathbf{g}, \mathbf{v}_j \rangle$, such that

$$-\infty < v_1 \le v_2 \le \ldots \le v_m < \infty \tag{8}$$

Additionally we set $v_0 = -\infty$ and $v_{m+1} = \infty$.

The polytope P can be built up by simple *cones* at the vertices $\mathbf{v}_j$. Let $\mathbf{v}$ be a vertex of P and $\mathbf{t}_1^{\mathbf{v}}, \ldots, \mathbf{t}_n^{\mathbf{v}}$ be nonzero vectors in the directions of the edges of P (originated from $\mathbf{v}$), i.e. for each j and every $\mathbf{x} \in P$, $\langle \mathbf{t}_j^{\mathbf{v}}, \mathbf{x} \rangle \ge 0$. We define $\gamma_j = \operatorname{sgn}(\langle \mathbf{t}_j^{\mathbf{v}}, \mathbf{g} \rangle)$ and $\delta(\mathbf{v}) = \prod_{j=1}^{n} \gamma_j$. Notice that by assumption (2), $\langle \mathbf{t}_j^{\mathbf{v}}, \mathbf{g} \rangle \ne 0$. Then the vectors $\gamma_j \, \mathbf{t}_j^{\mathbf{v}}$ span the *forward cone* $C(\mathbf{v})$ at $\mathbf{v}$, i.e.

$$C(\mathbf{v}) = \{\mathbf{v} + \sum_{j=1}^{n} c_j \gamma_j \mathbf{t}_j^{\mathbf{v}} : c_j \ge 0\} \tag{9}$$

As can easily be seen, $\langle \mathbf{x}, \mathbf{g} \rangle \ge \langle \mathbf{v}, \mathbf{g} \rangle$ for all $\mathbf{x} \in C(\mathbf{v})$ (see figure 3). Since P is

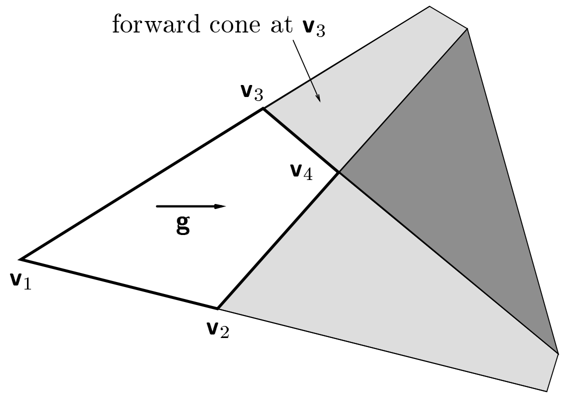

Figure 3: Forward cone and Gram's relation

simple $C(\mathbf{v}) \cap F^-(x)$ is an n-simplex for $x > v = \langle \mathbf{v}, \mathbf{g} \rangle$ and hence

$$V_n(C(\mathbf{v}) \cap F^-(x)) \quad = \quad \frac{1}{n!} \left| \det \left((x - v) \frac{\gamma_1 \mathbf{t}_1^{\mathbf{v}}}{\langle \gamma_1 \mathbf{t}_1^{\mathbf{v}}, \mathbf{g} \rangle}, \ldots, (x - v) \frac{\gamma_n \mathbf{t}_n^{\mathbf{v}}}{\langle \gamma_N \mathbf{t}_n^{\mathbf{v}}, \mathbf{g} \rangle} \right) \right|$$

$$= \quad (x-v)^n \, \frac{1}{n!} \, |\det(\mathbf{t}_1^{\mathbf{v}},\ldots,\mathbf{t}_n^{\mathbf{v}})| \, \delta(\mathbf{v}) \, \prod_{i=1}^{n} \langle \mathbf{t}_i^{\mathbf{v}}, \mathbf{g}\rangle^{-1} \tag{10}$$

for $x \geq v$ and 0 otherwise.

Let χ_M denote the characteristic function of the set $M \subseteq \mathbb{R}^n$, i.e. $\chi_M(\mathbf{x}) = 1$ if $\mathbf{x} \in M$ and 0 otherwise. By a version of Gram's relation (see [27]) it follows ([22])

$$\chi_P = \sum_{j=1}^{m} \delta(\mathbf{v}_j) \, \chi_{C(\mathbf{v_j})} \tag{11}$$

and thus

$$V_n(P \cap F^-(x)) = \sum_{j=1}^{m} \delta(\mathbf{v}_j) \, V_n(C(\mathbf{v_j}) \cap F^-(x)) \tag{12}$$

Define

$$a_j = \frac{1}{(n-1)!} \, |\det(\mathbf{t}_1^{\mathbf{v}_j},\ldots,\mathbf{t}_n^{\mathbf{v}_j})| \, \prod_{i=1}^{n} \langle \mathbf{t}_i^{\mathbf{v}_j}, \mathbf{g}\rangle^{-1} \tag{13}$$

Combining (7), (12), (10), (13) and the fact that $\delta(\mathbf{v}_j)^2 = 1$ we arrive at

$$A(x) = \sum_{\substack{1 \leq j \leq m \\ v_j \leq x}} a_j \, (x - v_j)^{n-1} \tag{14}$$

Using binomial theorem we get

$$A(x) = \sum_{k=0}^{n-1} b_k^{(x)} \, x^k \tag{15}$$

where the coefficients

$$b_k^{(x)} = \binom{n-1}{k} \sum_{\substack{1 \leq j \leq m \\ v_j \leq x}} a_j \, (-v_j)^{n-1-k} \tag{16}$$

depend on the intervals $[v_{j-1}, v_j)$ only. (Notice that $A(x) \equiv 0$ for $x \geq v_m$.)

Marginal density function. The coefficients $b_k^{(x)}$ in (15) are constants on the intervals $[v_{j-1}, v_j)$, i.e. $b_k^{(x)} = b_k^{(v_{j-1})}$ for all $x \in [v_{j-1}, v_j)$. On each of these intervals the marginal density function $h_{\mathbf{g}}(x) = A(x)$ is a polynomial of degree $n-1$ with both positive and negative coefficients.

We do not know particular generators for such marginal distributions (except for the special case $x \in [v_1, v_2)$, where $A(x)$ is a power function), but we can utilize the fact ([26, theorem 8]) that every marginal density of a log-concave distribution again is log-concave (The density of the uniform distribution is constant and thus log-concave.) Since the mode of the distribution is not known and the $b_k^{(x)}$ change in every recursion step of the algorithm it seems most convenient to use the algorithm of [12] on the interval $[v_{j-1}, v_j)$. Other possible choices are the inversion-rejection method (see [8, chapter VII.4]) or (especially in low dimensions) the inversion method.

Marginal distribution function. To find the interval $[v_{j-1}, v_j)$ we need the marginal distribution function $H(x)$, that is

$$H(x) = \int_{-\infty}^{x} h_{\mathbf{g}}(t) \, dt = \int_{-\infty}^{x} \sum_{k=0}^{n-1} b_k^{(t)} \, t^k \, dt \tag{17}$$

Generate a uniform sample U on $[0,1]$ and let $H_P = H(\infty)$ be the volume of P. Then $\mathbf{v}_j$ is the least vertex such that

$$H(v_{j-1}) \leq U \cdot H_P < H(v_j). \tag{18}$$

$H(x)$ can be calculated by recursion. Let $v = \max_{\substack{1\leq j\leq k \\ v_j\leq x}} v_j$. Then we have

$$H(x) = H(v) + \sum_{k=0}^{n-1} b_k^{(v)} \int_v^x t^k \, dt \tag{19}$$

Let

$$H_1 = 0 \qquad \text{and} \qquad H_j = \sum_{k=0}^{n-1} b_k^{(v_{j-1})} \int_{v_{j-1}}^{v_j} t^k \, dt \tag{20}$$

the volume of $P \cap \{\mathbf{x} \in \mathbb{R}: v_{j-1} \leq x \leq v_j\}$. Then we find

$$H(v_0) = 0 \qquad \text{and} \qquad H(v_j) = H(v_{j-1}) + H_j \qquad \text{for } j = 1, \ldots, m \tag{21}$$

2.4 Compute $Q(x)$

Cut polytopes. We get the vertices of the cut polytope $Q(x)$ by the intersection of the sweep-plane $F(x)$ with all edges $(\mathbf{v}_0, \mathbf{v}_1)$ of P. Obviously only those edges are of interest where $v_0 \leq x < v_1$. (Again $v_i = \langle \mathbf{g}, \mathbf{v}_i \rangle$.) Then we find for the vertex $\mathbf{v}'$ of $Q(x)$

$$\mathbf{v}' = \frac{v_1 - x}{v_1 - v_0} \mathbf{v}_0 + \frac{x - v_0}{v_1 - v_0} \mathbf{v}_1 \tag{22}$$

Hence we get the $\mathcal{V}$-representation of $Q(x)$.

The face lattice. By assumption, a d-face of P becomes a $(d-1)$-face of $Q(x)$ when we intersect P with the sweep-plane $F(x)$ (except when $d = 0$, i.e. the face is a vertex). To get the cut vertices by (22) we need the incident edges to all vertices of the cut polytopes in all steps of the recursive algorithm. Thus we need the *face lattice* (also called *incident graph* in [10]), i.e. the set of faces partially ordered by inclusion (see [30, chapter 2.2]). For determination notice that every point $\mathbf{x} \in P$ is an element of one or more facets. Thus we introduce an index $\rho(\mathbf{x})$ (see [2, 5]). Let $f_1, \ldots, f_N$ be the facets of P (in arbitrary but fixed order). Then we set

$$\rho(\mathbf{x}) = (\rho_1(\mathbf{x}), \ldots, \rho_N(\mathbf{x})) \qquad \text{with } \rho_i(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{x} \in f_i \\ 0 & \text{otherwise} \end{cases} \tag{23}$$

Let

$$\rho(\mathbf{x}) \leq \rho(\mathbf{y}) \quad \Leftrightarrow \quad \rho_i(\mathbf{x}) \leq \rho_i(\mathbf{y}) \quad \forall i = 1, \ldots, N \tag{24}$$

and

$$|\rho(\mathbf{x})| = \sum_{i=1}^{N} \rho_i(\mathbf{x}) \tag{25}$$

Notice that $\rho(\mathbf{x}) = \rho(\mathbf{y})$ if and only if $\mathbf{x}$ and $\mathbf{y}$ are in the relative interior of the same face f. Thus we get an index for each face by

$$\rho(f) = \rho(\mathbf{x}) \quad \text{for a } \mathbf{x} \in \text{relative interior of } f \tag{26}$$

By means of this index we can find the face lattice. Since P is simple by assumption we have for faces f, f_1 and f_2

$$f_1 \supseteq f_2 \quad \Leftrightarrow \quad \rho(f_1) \leq \rho(f_2) \tag{27}$$

$$\dim(f) = n - |\rho(f)| \tag{28}$$

Two d-faces f_1 and f_2 are joined by a $d+1$ face if and only if

$$\rho(f_1) > \rho(f) \text{ and } \rho(f_2) > \rho(f) \tag{29}$$

Using (28), this is equivalent to

$$|\rho(f_1) \wedge \rho(f_2)| = |\rho(f_1)| - 1 \tag{30}$$

where $\wedge$ denotes the bitwise AND-operator.

Recursion step. The d-faces of P become the $d-1$-faces of $Q(x)$. We use the same indices for the faces of the cut polytopes Q. Problems arise if $\langle \mathbf{g}, \mathbf{v}_0 \rangle = x$ in (22), i.e. the sweep-plane $F(x)$ contains a vertex $\mathbf{v}_0$ of P. Then there might be two or more edges with the same vertex $\mathbf{v}_0$. We get $\mathbf{v}_0$ as vertex of $Q(x)$ at least twice, but with different indices since the indices of the edges differ. In this case we append $\mathbf{v}_0$ to the vertex list of $Q(x)$ (without changing its index) and ignore the incident edges.

Let Q_k be the cut polytope after $n-k$ steps. Thus it has dimension k. Because of assumption (2), (28) holds analogously, except when f is a vertex.

$$\dim(f) = k - |\rho(f)| \qquad \text{if } f \text{ is not a vertex of } Q_k \tag{31}$$

Moreover (29) holds. But (30) is valid only if the faces f_i are not vertices of the cut polytope.

2.5 The algorithm

We are given the $\mathcal{H}$- or $\mathcal{V}$-representation of a *simple convex n-polytope*. (Notice that the convex hull of $n+2$ or more points in general position in $\mathbb{R}^n$ is *not* a simple polytope. A $\mathcal{H}$-polyhedron might not be bounded and thus the uniform distribution over the polyhedron does not exist. In both cases the algorithm does not work.) The algorithm SWEEP() requires a list of the vertices $\mathbf{v}_j$ of P (the $\mathcal{V}$-representation) and all the indices $\rho(\mathbf{v}_j)$ of these vertices. E.g. [1], [6] or [11] do the job. Some of these programs offer options to check the required conditions. Then algorithm SWEEP() runs as follows:

algorithm SWEEP()
Generates uniformly distributed random tuples over a simple polytope P.
Input: $\mathcal{V}$-representation of simple polytope P, indices $\rho(\mathbf{v}_j)$, $\dim(P)$

if $\dim(P) = 1$ **then**
  Use (4) to get random point $\mathbf{x}$ and **return** $\mathbf{x}$.
Find a proper $\mathbf{g}$ that satisfies (2).
Compute coefficients $b_k^{(v_j)}$ (16) and marginal distribution $H(v_j)$ (21) for all vertices $\mathbf{v}_j$. Compute H_P.
Generate a uniform $[0, H_P]$ random number U and get the interval $[v_{j-1}, v_j)$ such that $v_{j-1} \le U < v_j$.
Generate random variate X from marginal distribution (Use [12]).
Find all edges that intersect sweep-plane $F(x)$ (Use (29) or (30)).
Compute $Q(x)$ (22) and projection Q' (5).
$\mathbf{x}' \leftarrow$ **call** SWEEP() **with** Q', indices $\rho(\mathbf{v}_j)$, $\dim(P) - 1$.
Compute $\mathbf{x}$ (6).
return $\mathbf{x}$.

2.6 Remarks

The choice of $\mathbf{g}$. The algorithm is sensible about the choice of $\mathbf{g}$. The method requires summing a lot of numbers for computing the coefficients $b_k^{(v_j)}$ in (16). Some of these numbers are positive, some negative. If $\langle \mathbf{g}, \mathbf{x} \rangle$ is "nearly" constant on an edge of P, then these numbers can be quite large in magnitude, so that there can be considerable loss of significance due to round-off errors. As a consequence $A(v_m) \neq 0$ at the last vertex of the polytope. This is likely to happen for an arbitrary $\mathbf{g}$ if dimension is high and P has many faces (and vertices).

In the literature this problem is not really solved. [22] gives a solution only for the case when rational arithmetic is used for computation. [5] suggests the use of randomly chosen vectors.

A possible solution to this problem is: (1) Choose a $\mathbf{g}$. (2) Calculate all coefficients a_j. (3) If one of these is too big (e.g. larger than 10^5) reject $\mathbf{g}$ and try another one. A good choice for $\mathbf{g}$ are the unit vectors $(0, \ldots, 1, \ldots, 0)$ (which are easy to use). If these does not work possible choices are random vectors (as suggested in [5]) or the vectors $\mathbf{v}_j - \bar{\mathbf{v}}$, where $\bar{\mathbf{v}} = \frac{1}{m} \sum_{i=1}^{m} \mathbf{v}_i$ denotes the barycenter of P. Another possibility is to apply small random rotations of the axis and try the unit vectors again.

Non-convex polytopes. This sweep-plane algorithm can be modified for non-convex simple polytopes. The polytope P can be presented in Boolean Form, the face lattice (again) by the indices $\rho(f)$ (see [5] for details).

Complexity. We assume, that the $\mathcal{V}$-representation P is given. For computing the marginal density $h_{\mathbf{g}}(x) = A(x)$ we need determinants at all vertices of P a work taking $O(mn^3)$ steps. To get all vertices of the cut polytope $Q(x)$ in §2.4 we have to find all edges of P that intersect the sweep plane, which is of order $O(m^2)$. Thus for the first recursion step the amount of work is $O(m^2 + mn^3)$.

Comparison. It is difficult to compare the sweep-plane algorithm with the rejection methods (3) and (4) of the introduction. There the rejection constants for arbitrary polytope grow with $O(n!)$. On the other hand there are of course the special cases of the hyperrectangle and the simplex, respectively, where these methods are optimal. In contrast to method (3), no algorithm to construct an enclosing simplex is available in the literature for method (4).

Compared with the triangluation method (2) the main advantage of the new algorithm is the fact that practically no setup is necessary. On the other hand sampling is slower. For method (2) the complexity of the decomposition step depends on the number of faces which is $O(m^{\lfloor n/2 \rfloor})$, where m is the number of vertices ([23]). Thus for polytopes with a large number of vertices in high dimensions triangulation is very slow. Hence the presented new method is preferable if

- the dimension is high (≥ 3), and
- the polytope contains a large number of vertices, or
- we only need a few random points for the given polytope.

Volume computation. The generation of random tuples in a polytope is closely related to the determination of volumes. Volume computation is reported to be #P-hard (cf. [9, 14]).

3 A rejection technique for multivariate log-concave densities

For the generation of variates from bivariate and multivariate distributions papers are rare. Well known and discussed are only the generation of the multinormal and of the Wishart distribution (see e.g. [8] and [7]). One approach — especially considered by researchers interested in simulation – aims to develop new, easy to generate, classes of multivariate distributions; it is only necessary (and possible) to specify the marginal distribution and the degree of dependence measured by some correlation coefficient (see the monograph [21]). This idea seems to be attractive for most simulation practitioners interested in multivariate distributions but it is no help if variates from a distribution with given density should be generated.

The conditional distribution method requires the knowledge of and the ability to sample from the marginal and the conditional distributions (see [8, chapter XI.1.2]).

The multivariate extension of the ratio of uniforms method as described in [28] and [29] can be reformulated as rejection from a small family of table-mountain shaped multivariate distributions. This point of view explains the poor acceptance probability for high correlation. The practical problem how to obtain the enclosing multivariate rectangle for the ratio of uniforms method is not discussed in these papers and seems to be difficult for most distributions.

To our knowledge, no universal algorithms are known for multivariate distributions with given density function. In [8, chapter XI.1.3] it is even stressed that no general inequalities for multivariate densities are available, a fact which makes it impossible to design black-box algorithms similar to those in [8] for the univariate case. In [19] a universal algorithm for log-concave distributions is developed for the bivariate case. It uses the idea of transformed density rejection which is presented in a first form in [8, chapter VII.2.4] and with a different set-up in [12].

In this section we generalize this idea to the multivariate case. The sweep-plane technique is used to sample from the hat function.

3.1 Transformed density rejection

Density. We are given a multivariate distribution with differentiable density function

$$f\colon D \to [0,\infty), \qquad D \subseteq \mathbb{R}^n, \qquad \text{with mode } \mathbf{m}. \tag{32}$$

For simplicity we assume $D = \mathbb{R}^n$.

Transformation. To design a black-box algorithm utilizing the rejection method it is necessary to find an automatic way to construct a hat function for a given density. Transformed density rejection introduced under a different name in [12] and generalized in [18] is based on the idea that the density f is transformed by a monotone T (e.g. $T(x) = \log(x)$) in such a way that (see [18]):

(T1) $\tilde{f}(\mathbf{x}) = T(f(\mathbf{x}))$ is concave. We then say "f is *T-concave*".

(T2) $\lim_{x\to 0} T(x) = -\infty$;

(T3) $T(x)$ is differentiable and $T'(x) > 0$, which implies T^{-1} exists; and

(T4) the volume under the hat is finite.

Hat. It is then easy to construct a hat $\tilde{h}(\mathbf{x})$ for $\tilde{f}(\mathbf{x})$ as the minimum of N tangents. Since $\tilde{f}(\mathbf{x})$ is concave we clearly have $\tilde{f}(\mathbf{x}) \leq \tilde{h}(\mathbf{x})$ for all $\mathbf{x} \in \mathbb{R}^n$. Transforming $\tilde{h}(\mathbf{x})$ back into the original scale we get $h(\mathbf{x}) = T^{-1}(\tilde{h}(\mathbf{x}))$ as majorizing function or hat for f, i.e. with $f(\mathbf{x}) \leq h(\mathbf{x})$. Figure 4 illustrates the situation for the univariate case by means of the normal distribution and the transformation $T(x) = \log(x)$. The left hand side shows the transformed density with three tangents. The right hand side shows the density function with the resulting hat.

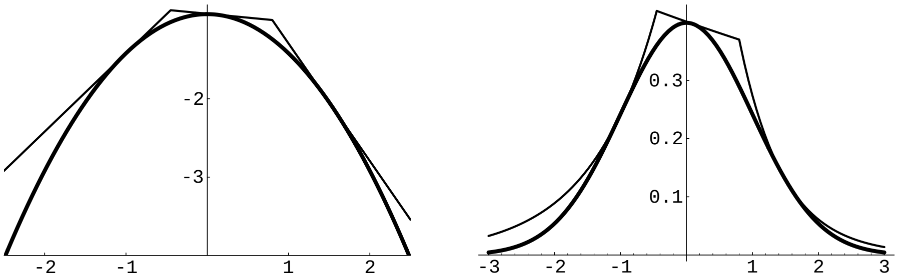

Figure 4: hat function for univariate normal density

Rejection. The basic form of the multivariate rejection method is given by

algorithm REJECTION()

1: **Set-up:** Construct a hat-function $h(\mathbf{x})$.
2: Generate a random tuple $\mathbf{X} = (X_1, \ldots, X_n)$ with density proportional to $h(\mathbf{X})$ and a uniform random number U.
3: If $Uh(\mathbf{X}) \leq f(\mathbf{X})$ return $\mathbf{X}$ else go to 2.

The main idea of this section is to extend transformed density rejection to the multivariate case.

3.2 Construct a hat-function

Tangents. To construct the hat function we choose N points $\mathbf{p}_i \in D \subseteq \mathbb{R}^n$ and take tangents $\ell_i(\mathbf{x})$ of the transformed density $\tilde{f}$ at these points:

$$\ell_i(\mathbf{x}) = \tilde{f}(\mathbf{p}_i) + \langle \nabla \tilde{f}(\mathbf{p}_i), (\mathbf{x} - \mathbf{p}_i) \rangle \tag{33}$$

The hat is then the pointwise minimum of these tangents.

$$\tilde{h}(\mathbf{x}) = \min_{i=1,\ldots,m} \ell_i(\mathbf{x}) \qquad \text{and} \qquad h(\mathbf{x}) = \exp(\tilde{h}(\mathbf{x})) \tag{34}$$

Although the main idea of multivariate transformed density rejection is simple, there is still hard work to collect all necessary details. One problem is the suitable choice of the points $\mathbf{p}_i$. In the univariate case we are able to optimize the choice of the points (see [18] and [20]). It is even possible to show that the execution time of the algorithm is uniformly bounded for a family of T-concave distributions. In the multivariate case this task seems to be impracticable. So we use the important idea of adaptive rejection sampling introduced in [12].

Adaptive rejection sampling. Adapted to our situation it works in the following way: For the start take at least $n+1$ points of contact, which only must have the property that the volume below the hat $h(\mathbf{x})$ is bounded. Then start the generation of random variates with this hat until a point $\mathbf{x}$ is rejected. Now use $\mathbf{x}$ to construct an additional tangent and thus a new hat and restart generation of random points. Every rejected point is taken for an additional tangent until the maximum number N of tangents is reached. The points of contact are thus chosen by a stochastic algorithm and it is clear that the multivariate density of the distribution of the next point for a new tangent is proportional to $h(\mathbf{x}) - f(\mathbf{x})$. Hence with N tending towards infinity the acceptance probability for a hat constructed in such a way converges to 1 with probability 1.

It is not difficult to show that the expected rejection constant is of order $1 + O(N^{-2/n})$: Since h is constructed by tangents over each polytope, we find $h(\mathbf{x}) - f(\mathbf{x}) = O(\|\mathbf{x} - \mathbf{p}_i\|^2)$ for polytope P_i and thus the volume between h and f in a ball is of order $O(\|\mathbf{x} - \mathbf{p}_i\|^{2+n})$. Without loss D is bounded. The volume of each polytope is $O(d^n)$, where d is the diameter of P_i. Assuming that all polytopes have same volume we find $d = O(N^{-1/n})$. Thus the total volume between h and f is given by $N \cdot O((N^{-1/n})^{2+n}) = O(N^{-2/n})$. Since the polytopes do not have same size but are constructed in order to minimize this volume, the statement follows.

3.3 Generate random tuples

We use a modified version of the sweep-plane algorithm in §2 to generate random tuples with respect to the hat function.

Polyhedra. The domain in which a particular tangent ℓ_i determines the hat function h is a convex n-polyhedron which may be bounded or not. We find for touching point $\mathbf{p}_i$ and its tangent $\ell_i(\mathbf{x})$

$$P_i = \{\mathbf{x} \in \mathbb{R}^n : h(\mathbf{x}) = T^{-1}(\ell_i(\mathbf{x}))\} = \bigcap_{j=1,\ldots,N} \{\mathbf{x} \in \mathbb{R}^n : \ell_i(\mathbf{x}) \leq \ell_j(\mathbf{x})\} \tag{35}$$

To avoid lots of indices we write $\mathbf{p}$, $\ell(\mathbf{x})$ and P without the index i if there is no risk of confusion.

We make the following assumptions about the polyhedron P:

(P1) P is a *simple* polyhedron.

(P2) There exists a maximum of ℓ in P.

(P3) ℓ is non-constant on every edge of P.

We always can find touching points such that these restrictions hold.

Sweep-plane algorithm. In each of these polyhedra, $\ell(\mathbf{x})$ is constant on every intersection of P with an affine hyperspace (called a *flat*) perpendicular to the gradient of $\ell(\mathbf{x})$. Since by condition (P3) $\nabla\ell = \nabla\tilde{f}(\mathbf{p}) \neq 0$, let

$$\mathbf{g} = -\frac{\nabla\tilde{f}(\mathbf{p})}{\|\nabla\tilde{f}(\mathbf{p})\|}. \tag{36}$$

Let again $x = \langle \mathbf{g}, \mathbf{x}\rangle$. $F(x)$ denotes the sweep-plane and $Q(x)$ its intersection with P (see §2.2). Notice that under conditions (P1)–(P3) all the sweep-plane technique derived in §2 still works for an unbounded polyhedron P. By conditions (P2) and (P3) the cut polytope $Q(x)$ is bounded. By setting

$$\alpha = \tilde{f}(\mathbf{p}) - \langle\nabla\tilde{f}(\mathbf{p}), \mathbf{p}\rangle \quad \text{and} \quad \beta = \|\nabla\tilde{f}(\mathbf{p})\| \tag{37}$$

and by inserting into (33) we find for the hat function in domain P

$$h|_P(\mathbf{x}) = T^{-1}(\ell(\mathbf{x})) = T^{-1}(\alpha - \beta\, x) \tag{38}$$

The marginal density function $h_{\mathbf{g}}$ of the hat $h|_P$ along $\mathbf{g}$ is then

$$h_{\mathbf{g}}(x) = \int_{Q(x)} h(\mathbf{y})dF(x) = A(x) \cdot T^{-1}(\alpha - \beta\, x) \tag{39}$$

where the volume $A(x)$ of $Q(x)$ is given by (15).

Log-concave densities. The transformation $T(x) = \log(x)$ satisfies (T1)–(T4). If $T(f(\mathbf{x})) = \log(f(\mathbf{x}))$ is concave, we say f is *log-concave.*

We have $T^{-1}(x) = \exp(x)$ and thus by (39) and (15)

$$h_{\mathbf{g}}(x) = \exp(\alpha - \beta\, x) \sum_{k=0}^{n-1} b_k^{(x)}\, x^k \tag{40}$$

$h_{\mathbf{g}}$ is again log-concave by [26, theorem 8], since it is the marginal density of the log-concave function $\exp(\ell(\mathbf{x}))\big|_P$. Again it is best to use the algorithm of [12] on the intervals $[v_{j-1}, v_j)$ (see §2.3).

The marginal distribution function $H(x) = \int_{-\infty}^{x} h_{\mathbf{g}}(t)\, dt$ is calculated analogously to (19) and (21) by recursion. For volume H_j below the hat in the interval $[v_{j-1}, v_j)$ we now have

$$H_j = e^{\alpha} \sum_{k=0}^{n-1} b_k^{(v_{j-1})} \int_{v_{j-1}}^{v_j} (\beta t)^k\, e^{\beta t}\, dt$$

By substituting $z = \beta\, t$ and using formula (2.323) in [13] we arrive at

$$\begin{aligned} H_j &= e^{\alpha} \sum_{k=0}^{n-1} b_k^{(v_{j-1})}\, \beta^{-k-1}\, k! \\ &\quad \cdot \left(e^{-\beta\, v_{j-1}} \sum_{l=0}^{k} \frac{(\beta\, v_{j-1})^l}{l!} - e^{-\beta\, v_j} \sum_{l=0}^{k} \frac{(\beta\, v_j)^l}{l!} \right) \end{aligned} \tag{41}$$

and for the unbounded interval $[v_m, \infty)$ (if P is not bounded)

$$H_{m+1} = e^{\alpha} \sum_{k=0}^{n-1} b_k^{(v_m)}\, k!\, \beta^{-k-1}\, e^{-\beta\, v_m} \sum_{l=0}^{k} \frac{(\beta\, v_m)^l}{l!} \tag{42}$$

Sample from marginal distribution. For the algorithm of [12] we first have to take one of the N polyhedra P_i and one of the intervals $[v_{j-1}, v_j)$ on this polyhedron. The latter is done by (18).

Let $H_{tot} = \sum_{P_i} H_{P_i}$ denote the total volume below the hat and U a random point from a uniform sample on $[0, 1]$. Then we select the polytope P_j such that

$$\sum_{i=1}^{j-1} H_{P_i} \leq U \cdot H_{tot} < \sum_{i=1}^{j} H_{P_i}. \tag{43}$$

3.4 Construct polyhedra

There are two differences to §2. (1) Some of the polyhedra are not bounded. (2) We cannot use e.g. [11] to compute all the vertices of the polyhedra whenever we add a new construction point.

Vertices and extreme rays. For an unbounded polyhedron we have vertices and extreme rays. The latter may be interpreted as "vertices at infinity". The set of all vertices of the polyhedron (in $\mathbb{R}^n$ or at infinity) can be written as $\mathbb{R}^n \cup S^{n-1}$, where S^{n-1} denotes the unit sphere in $\mathbb{R}^n$. For an algebraic description of this set, we define a space $\mathbb{R}^n_\infty$ by

$$\begin{aligned} \mathbb{R}^n_\infty &= \{(x_0;\mathbf{x})\colon x_0 \in \mathbb{R},\, \mathbf{x} \in \mathbb{R}^n,\, (x_0;\mathbf{x}) \neq (0;\mathbf{0}),\, x_0 \geq 0\} \\ \mathbf{x} = \mathbf{y} \quad &\Leftrightarrow \quad x_i = \alpha\, y_i \text{ for an } \alpha > 0 \text{ for all } i = 0,\ldots,n \end{aligned} \tag{44}$$

As can easily be seen we have $\{(x_0;\mathbf{x}) \in \mathbb{R}^n_\infty \colon x_0 \neq 0\} \cong \mathbb{R}^n$ and $\{(x_0;\mathbf{x}) \in \mathbb{R}^n_\infty \colon x_0 = 0\} \cong S^{n-1}$. We may write $\mathbb{R}^n_\infty \cong \mathbb{R}^n \cup S^{n-1}$. Notice that we get the projective space $\mathbb{PR}^n$ by identifying antipodal points in the subset S^{n-1}. In other words, we use the projective space $\mathbb{PR}^n$ but distinguish between two directions of points at infinity. For that reason we call $P_0 = \{(x_0;\mathbf{x}) \in \mathbb{R}^n_\infty \colon x_0 = 0\}$ the *hyperplane at infinity* (although it is not "really" a hyperplane). The facets of P_i in $\mathbb{R}^n_\infty$ are given by the equalities $\ell_i(\frac{\mathbf{x}}{x_0}) = \ell_j(\frac{\mathbf{x}}{x_0})$. We use the standard Euclidean metric in the subsets $\{(x_0;\mathbf{x}) \in \mathbb{R}^n_\infty \colon x_0 \neq 0\}$ and $\{(x_0;\mathbf{x}) \in \mathbb{R}^n_\infty \colon x_0 = 0\}$ (this cannot be extended to the whole space).

We embed the polyhedron P into the compact $\mathbb{R}^n_\infty$ set by

$$\begin{aligned} &\mathbf{v} \in \mathbb{R}^n \hookrightarrow (1;\mathbf{v}) = (1;v_1,\ldots,v_n) \in \{(x_0;\mathbf{x}) \in \mathbb{R}^n_\infty \colon x_0 \neq 0\} \\ &\qquad \text{for a vertex } \mathbf{v} \text{ of } P \\ &\mathbf{t} \in \mathbb{R}^n \hookrightarrow (0;\mathbf{t}) = (0;t_1,\ldots,t_n) \in \{(x_0;\mathbf{x}) \in \mathbb{R}^n_\infty \colon x_0 = 0\} \\ &\qquad \text{for a nonzero vector } \mathbf{t} \text{ in the directions of edges of } P \end{aligned} \tag{45}$$

An unbounded convex polyhedron is then a polyhedron with vertices in the hyperplane at infinity. The coordinates of a vertex at infinity are given by the vector in the direction of its corresponding unbounded edge, i.e. the extreme ray.

We call all faces with vertices in $\mathbb{R}^n$ and in the hyperplane at infinity *unbounded faces* and the faces with all vertices at hyperplane at infinity *faces at infinity.*

For a point $\mathbf{x}$ at infinity we set $\ell(\mathbf{x}) = \langle \nabla\ell, \mathbf{x}\rangle$, i.e. $\|\mathbf{x}\|$ times the directional derivative of ℓ along $\mathbf{x}$.

Edges. To calculate the coefficients a_j in (13) we need the vectors $\mathbf{t}_i^{\mathbf{v}}$ in the directions of the edges of P. Using the embedding into $\mathbb{R}^n_\infty$ we find for an edge $(\mathbf{v}_0, \mathbf{v}_1)$, $\mathbf{v}_0 \in \mathbb{R}^n$,

$$(0;\mathbf{t}) = \begin{cases} (1;\mathbf{v}_1) - (1;\mathbf{v}_0) & \text{if edge is bounded} \\ (0;\mathbf{v}_1) & \text{if edge is unbounded} \end{cases} \tag{46}$$

The points on such an edge are then given by

$$\mathbf{v} = \begin{cases} (1-t)\,\mathbf{v}_0 + t\,\mathbf{v}_1 & 0 \leq t \leq 1 & \text{if edge } (\mathbf{v}_0,\mathbf{v}_1) \text{ is bounded} \\ \mathbf{v}_0 + t\,\mathbf{v}_1 & t \geq 0 & \text{if } \mathbf{v}_0 \in \mathbb{R}^n \text{ and } \mathbf{v}_1 \text{ at infinity} \\ t_0\,\mathbf{v}_0 + t_1\,\mathbf{v}_1 & t_0, t_1 > 0 & \text{if edge } (\mathbf{v}_0,\mathbf{v}_1) \text{ is at infinity} \end{cases} \tag{47}$$

In the last case $t_0\,\mathbf{v}_0 + t_1\,\mathbf{v}_1$ again is a point at infinity and thus any multiple of this vector gives the same point.

Index $\rho(\mathbf{x})$ and face lattice. The indices $\rho(\mathbf{x})$ and $\rho(f)$ are defined similar to §2.4. Now we use the polyhedra P_i rather than facets.

$$\begin{aligned} &\rho(\mathbf{x}) = (\rho_0(\mathbf{x}); \rho_1(\mathbf{x}), \ldots, \rho_N(\mathbf{x})) \\ &\text{with} \qquad \rho_i(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{x} \in P_i \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{48}$$

P_0 denotes the hyperplane at infinity. Every vertex $\mathbf{v}$ is element of at least $n+1$ polyhedra. From now on we assume that

(P1') every (finite) vertex $\mathbf{v}$ is element of exactly $n+1$ polyhedra.

Notice that each vertex is element of at least $n+1$ polyhedra. Since we choose the construction points at random, this assumption holds with probability 1 if the Hessian of the transformed density is non-degenerated almost everywhere.

Using (P1') the definitions and equations of §2.4 are still valid; except of (28), which now is given by

$$\dim(f) = n + 1 - |\rho(f)| \tag{49}$$

Initial polyhedra. We start the algorithm with $n+1$ points for constructing the hat function. We can choose the vertices of a regular n-simplex, centered at the mode $\mathbf{m}$. For example, such points are given by

$$\begin{aligned} \mathbf{p}_j &= \sigma\sqrt{\tfrac{n+1}{n}}((a,a,a,\ldots,a) + \mathbf{e}_j + \mathbf{m}), \quad j = 1,\ldots,n \\ \mathbf{p}_{n+1} &= \sigma\sqrt{\tfrac{n+1}{n}}((b,b,b,\ldots,b) + \mathbf{m}). \end{aligned} \tag{50}$$

Here $\mathbf{e}_j$ denotes the j-th unit-vector and $b = -\frac{1}{\sqrt{n+1}}$, $a = -\frac{b+1}{n}$. The parameter σ can be chosen appropriate for the distribution.

We have one vertex in $\mathbb{R}^n$ and $n+1$ vertices at infinity for the initial polyhedra. For all these vertices $\mathbf{v}_j$ we find $\rho_j(\mathbf{v}) = 0$ for exactly one j. We get the coordinates of the vertex $\mathbf{v}_j$ by solving the following system of equalities in $\mathbb{R}^n_\infty$:

$$\begin{aligned} &\rho_0(\mathbf{v}_j)\cdot x_0 = 0 \\ &\rho_j(\mathbf{v}_j)\cdot(\ell_{n+1}(\mathbf{x}) - \ell_j(\mathbf{x})) = 0 \quad \text{for } j = 1,\ldots,n \end{aligned} \tag{51}$$

Notice that one of these equalities vanishes, since $\rho_j(\mathbf{v}_j) = 0$. To get the coordinates of the vertices at infinity we have to determine the direction of each solution $(0;\mathbf{t}_i)$. We must have

$$\ell_i(\mathbf{v}+\mathbf{t}_i) > \ell_j(\mathbf{v}+\mathbf{t}_i) \qquad \text{for at least one } j \neq i \tag{52}$$

By means of index $\rho(\mathbf{v})$ we can easily verify, which vertices belong to which initial polytopes P_i.

Violated conditions. If condition (P1), (P1') or (P3) fails we have to move the vertices of the regular simplex in (50) a little bit by adding a random point.

Condition (P2) might fail, if the gradient of the transformed density $\nabla\tilde{f}(\mathbf{x})$ is "far away" from pointing to the mode. For example, this happens for the normal density $f(x,y) = \exp(-x^2 - 1000y^2))$. A possible solution to this problem is to start with the points $\pm\mathbf{e}_j$ as construction points. (But then we have more than $n+1$ vertices for the $2n$ initial polyhedra.) Because of the convexity of the transformed density, condition (P2) always holds if each polyhedra can be moved into the cone $\{\mathbf{x} \in \mathbb{R}^n : x_i \geq 0\}$.

Adding a construction point. For the adaptive method it is necessary to add a new construction point whenever a point is rejected. Notice that $\ell(\mathbf{x})$ gives the directional derivative for points at infinity with $\|\mathbf{x}\| = 1$. Then the following procedure gives the new polyhedra:

(A1) get the new tangent $\ell(\mathbf{x})$ at this point (see (33));

(A2) find all edges $(\mathbf{v}_0, \mathbf{v}_1)$ with $\ell(\mathbf{v}_0) < \tilde{h}(\mathbf{v}_0)$ and $\ell(\mathbf{v}_1) > \tilde{h}(\mathbf{v}_1)$;

(A3) calculate new vertex $\mathbf{v} = t_0\,\mathbf{v}_0 + t_1\,\mathbf{v}_1$, such that $\ell(\mathbf{v}) = \tilde{h}(\mathbf{v})$. Using (47) and the linearity of ℓ and $\tilde{h}$ on every edge we find

$$\begin{array}{ccc|cc}
\text{case} & \mathbf{v}_0 & \mathbf{v}_1 & t_0 & t_1 \\
& \in \mathbb{R}^n & \in \mathbb{R}^n & \frac{\delta_1}{\delta_1-\delta_0} & \frac{\delta_0}{\delta_0-\delta_1} \\
& \in \mathbb{R}^n & inf & 1 & -\frac{\delta_0}{\delta_1} \\
& inf & \in \mathbb{R}^n & -\frac{\delta_1}{\delta_0} & 1 \\
& inf & inf & \delta_1 & \delta_0
\end{array} \tag{53}$$

where

$$\delta_i = \ell(\mathbf{v}_i) - \tilde{h}(\mathbf{v}_i)$$

(A4) remove all vertices $\mathbf{v}$ (and adjacent faces) with $\ell(\mathbf{v}) < \tilde{h}(\mathbf{v})$;

(A5) check assumptions (P1)–(P3):

(P1) and (P1') are violated if we find a new vertex $\mathbf{v}$ with $\ell(\mathbf{v}) = \tilde{h}(\mathbf{v})$.

(P3) fails if we find two new vertices $\mathbf{v}_0$ and $\mathbf{v}_1$, adjacent by an edge, with $\ell(\mathbf{v}_0) = \ell(\mathbf{v}_1)$.

(P2) always holds, if it holds for the initial polytope.

If one of the assumptions (P1)–(P3) is violated, we cannot use the point as new construction point.

3.5 Remarks

A C-implementation. We translated our algorithm into a working C-program. We tested the algorithm for the multivariate normal distribution. The program worked well up to dimensions 6–7. Then round-off errors occur too frequently, i.e. some of the rejected points could not be used for constructing the hat and many trials have to be made to find a vector $\mathbf{g}$ for the sweep-planes in the cut polytopes Q_j. Furthermore the average number of generated random points per second decreases rapidly, which is due to the great number of vertices of the polytopes P_i.

3.6 Possible variants

Subset of $\mathbb{R}^n$ as domain. Obviously we can restrict the domain D of density f to simple polyhedron , which may be bounded or not. Changes to the basic version are: (1) The initial polyhedra have more vertices (and thus the usage of a vertex finding algorithm, e.g. [2], is recommended). (2) The index $\rho(\mathbf{x})$ must be extended by the facets of the boundary of D, i.e we append $\rho_{-i}(\mathbf{x}) = 1$ if $\mathbf{x}$ is in the boundary facet f_i, and $\rho_{-i}(\mathbf{x}) = 0$ otherwise.

The mode as touching point. When we use the mode $\mathbf{m}$ of the density f as touching point for our hat, $\ell(\mathbf{x})$ is constant, on the corresponding polytope $P_{\mathbf{m}}$, since $\nabla\tilde{f}(\mathbf{m}) = 0$. Therefore we can use the algorithm from §2. (But we have to check if this polyhedron is bounded.)

Problems arise for all neighboring polyhedra P_i. Each of these must have an edge common with $P_{\mathbf{m}}$. $\ell_i(\mathbf{x})$ is constant on this edge and hence assumption (P3) fails for all P_i. This assumption is necessary for calculating $A(x)$ in (14). Thus we “ignore” the mode $\mathbf{m}$ when we construct the other polyhedra (see figure 5) and modify the marginal density in (39) to

$$h_{\mathbf{g}}(x) = \begin{cases} A(x) \cdot T^{-1}(\alpha - \beta\, x) & \text{for } x \geq v_l \\ 0 & \text{for } x < v_l \end{cases}$$
$$\text{where} \qquad v_l = \min_{\substack{\mathbf{x}\in P \\ \ell(\mathbf{x})=\tilde{h}(\mathbf{x})}} \langle \mathbf{g}, \mathbf{x} \rangle \tag{54}$$

If P does not intersect with the polytope $P_{\mathbf{m}}$, then $v_l = v_1$.

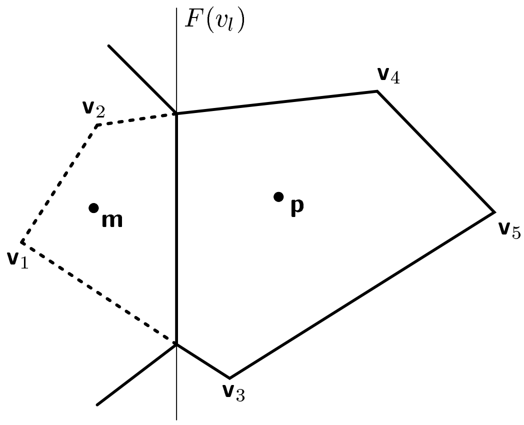

Figure 5: “hidden” part of a polytope

T_c-concave densities. A family T_c of transformations that fulfill conditions $(T1)$–$(T4)$ is introduced in [18]. Let $c \leq 0$. Then we set

c		$T_c(x)$	$T_c^{-1}(x)$	$T_c'(x)$
$c = 0$	$\mathbb{R}^+ \to \mathbb{R}$	$\log(x)$	$\exp(x)$	x^{-1}
$-\frac{1}{n} < c \leq 0$	$\mathbb{R}^+ \to \mathbb{R}^-$	$-x^c$	$(-x)^{1/c}$	$-c\, x^{c-1}$

It can easily be verified that condition (T4) holds if and only if $-\frac{1}{n} < c \leq 0$. Moreover for $c < 0$ we must have $h|_P < 0$. To ensure the negativity of the hat we always have to choose the mode $\mathbf{m}$ as construction point for a tangent plane ℓ.

In [18] it was shown that if a density f is T_c-concave then it is T_{c_1}-concave for all $c_1 \leq c$.

The case $c = 0$ is already described in §3.3. For the case $c < 0$ the marginal density function is given by

$$h_{\mathbf{g}}(x) = (\beta\, x - \alpha)^{\frac{1}{c}} \sum_{k=0}^{n-1} b_k^{(x)}\, x^k \tag{55}$$

Notice that $\alpha - \beta\, t < 0$ for all $t \geq v_1$ and that $\frac{1}{c} < -n$. By substituting $z = \beta\, t - \alpha$ and binomial theorem we find for the marginal distribution function

$$\begin{aligned} H_j &= \sum_{k=0}^{n-1} b_k^{(v_{j-1})}\, \beta^{-k-1} \sum_{l=0}^{k} \binom{k}{l} \frac{1}{l+\frac{1}{c}+1}\, \alpha^{k-l} \\ &\qquad \cdot \Big((\beta\, v_j - \alpha)^{l+\frac{1}{c}+1} - (\beta\, v_{j-1} - \alpha)^{l+\frac{1}{c}+1} \Big) \end{aligned} \tag{56}$$

and for the unbounded interval $[v_m, \infty)$ (if P is not bounded)

$$H_{m+1} = -\sum_{k=0}^{n-1} b_k^{(v_m)}\, \beta^{-k-1} \sum_{l=0}^{k} \binom{k}{l} \frac{1}{l+\frac{1}{c}+1}\, \alpha^{k-l}\, (\beta\, v_m - \alpha)^{l+\frac{1}{c}+1} \tag{57}$$

Due to the following lemma we can use the algorithm of [18] to generate random points in the intervals $[v_{j-1}, v_j)$ with respect to this marginal distribution.

Lemma 1 *The marginal density* $h_{\mathbf{g}}(x) = (\beta\, x - \alpha)^{\frac{1}{c}}\, A(x)$, $(-\frac{1}{n} < c \leq 0)$, *is* T_c-*concave.*

It remains to prove that $T_c(h_{\mathbf{g}}(x)) = (\alpha - \beta\, x)\, A(x)^c$ is concave for $c < 0$. Let $g(x) = \alpha - \beta\, x$ and $f(x) = A(x)^c$. Let $x_2 > x_1 \geq v_1$ and $\bar{x} = h\, x_1 + (1-h)\, x_2$. By assumption $g(x_2) \leq g(\bar{x}) \leq g(x_1) < 0$ and $g(\bar{x}) = h\, g(x_1) + (1-h)\, g(x_2)$. $f(x)$ is convex, since $A(x)$ is log-concave (see §2.3) and hence $T_c(A(x)) = -A(x)^c$ concave. We have to show that

$$\begin{aligned} f(\bar{x})\, g(\bar{x}) - (h\, f(x_1)\, g(x_1) + (1-h)\, f(x_2)\, g(x_2)) &= \\ h\, g(x_1)\, (f(\bar{x}) - f(x_1)) + (1-h)(f(\bar{x}) - f(x_2)) &\geq 0 \end{aligned} \tag{58}$$

Notice that $f(\bar{x}) \leq h\, f(x_1) + (1-h)\, f(x_2)$ by assumption. Thus the left hand side of (58) is $\geq h\,(1-h)\,(g(x_1) - g(x_2))\,(f(x_2) - f(x_1))$ which is ≥ 0 if $f(x_1) \leq f(x_2)$.

For the case $f(x_1) \geq f(x_2)$ notice that by the convexity of $f(x)$, $f(x_1) - f(\bar{x}) \geq (1-h)\,(f(x_1) - f(x_2))$ and $f(\bar{x}) - f(x_2) \leq h\,(f(x_1) - f(x_2))$. Thus the left hand side of (58) is $\geq h\,(1-h)\,(g(x_1) - g(x_2))\,(f(x_1) - f(x_2)) \geq 0$, as proposed.

Acknowledgements

We wish to note our appreciation for help rendered by Jörg Lenneis. He has given us lots of hints for the implementation of the algorithm. We also thank Gerhard Derflinger for his interest in our work.

References

[1] Avis, D. A C implementation of the reverse search vertex enumeration algorithm. Tech. rep., School of Computer Science, McGill University, Montreal, Quebec, 1993. report and code available at `ftp://mutt.cs.mcgill.ca/pub/C`.

[2] Avis, D., and Fukuda, K. A pivoting algorithm for convex hull and vertex enumeration of arrangements and polyhedra. *Discrete Comput. Geom. 8* (1992), 295–313.

[3] Berger, M. *Geometrie*, 2 ed., vol. 3. Cedic/Fernand Nathan, Paris, 1978. Convexes et polytopes, polyedres reguliers, aires et volumes.

[4] Bieri, H., and Nef, W. A recursive sweep-plane algorithm, determining all cells of a finite division of R^m. *Computing 28* (1982), 189–198.

[5] Bieri, H., and Nef, W. A sweep-plane algorithm for the volume of polyhedra represented in boolean form. *Linear Algebra Appl. 52/53* (1983), 69–97.

[6] Christof, T. *PORTA – A Polyhedron Representation Transformation.* Universität Heidelberg. code available at `ftp://elib.zib-berlin.de/pub/mathprog`.

[7] Dagpunar, J. *Principles of Random Variate Generation.* Clarendon Press, Oxford, 1988.

[8] Devroye, L. *Non-Uniform Random Variate Generation.* Springer-Verlag, New-York, 1986.

[9] Dyer, M. E., and Frieze, A. M. On the complexity of computing the volume of a polyhedron. *SIAM J. Comput. 17*, 5 (October 1988), 967–974.

[10] Edelsbrunner, H. *Algorithms in Combinatorial Geometry*, vol. 10 of *EATCS Monographs on Theoretical Computer Science.* Springer-Verlag, 1987.

[11] Fukuda, K. cdd reference manual. Tech. rep., ETH Zentrum, Zürich, Switzerland, 1995. report and code availeable at `ftp://ifor13.ethz.ch/pub/fukuda/cdd`.

[12] Gilks, W. R., and Wild, P. Adaptive rejection sampling for gibbs sampling. *Appl. Statistics 41* (1992), 337–348.

[13] Gradshteyn, I. S., and Ryzhnik, I. M. *Table of Integrals*, 5th ed. Academic Press, 1965.

[14] Gritzmann, P., and Klee, V. Polytopes: abstract, convex and computational. In *On the complexity of some basic problems in computational convexity. II: Volume and mixed volumes.* (Scarborough, Ontario, Canada, 1994), T. Bisztriczky et al., Eds., no. 440 in NATO ASI Series, Ser. C, Math. Phys. Sci., NATO Advanced Study Institute, Kluwer Academic Publishers, pp. 373–466.

[15] Grünbaum, B. *Convex Polytopes.* Interscience, 1967.

[16] Hadwiger, H. Eulers Charakteristik und kombinatorische Geometrie. *J. Reine Angew. Math. 194* (1955), 101–110.

[17] Hadwiger, H. Eine Schnittrekursion für die Eulersche Charakteristik euklidischer Polyeder. *Elem. Math. 23*, 6 (1968), 121–132.

[18] Hörmann, W. A rejection technique for sampling from T-concave distributions. *ACM Trans. Math. Software 21*, 2 (1995), 182–193.

[19] Hörmann, W. A universal generator for bivariate log-concave distributions. Preprint, 1995.

[20] Hörmann, W., and Derflinger, G. Universal generators for correlation induction. In *Compstat, Proceedings in Computational Statistics* (Heidelberg, 1994), R. Dutter and W. Grossmann, Eds., Physica-Verlag, pp. 52–57.

[21] Johnson, M. E. *Multivariate Statistical Simulation.* John Wiley & Sons, New York, 1987.

[22] Lawrence, J. Polytope volume computation. *Math. Comput. 57*, 195 (1991), 259–271.

[23] McMullen, P. The maximum number of faces of a convex polytope. *Mathematika 17* (1970), 179–184.

[24] Motzkin, T. S., Raiffa, H., Thompson, G. L., and Thrall, R. M. The double description method. In *Contribution to the Theory of Games, Vol. II* (1953), H. W. Kuhn and A. W. Tucker, Eds., vol. 28 of *Annals of Math. Studies*, Princton University Press, pp. 81–103.

[25] Nef, W. *Beiträge zur Theorie der Polyeder.* Herbert Lang, Bern, 1978.

[26] Prékopa, A. On logarithmic concave measures and functions. *Acta Sci. Math. Hungarica 34* (1973), 335–343.

[27] Shephard, G. C. An elementary proof of Gram's theorem for convex polytopes. *Canad. J. Math. 19* (1967), 1214–1217.

[28] Stefanescu, S., and Vaduva, I. On computer generation of random vectors by transformations of uniformly distributed vectors. *Computing 39* (1987), 141–153.

[29] Wakefield, J. C., Gelfand, A. E., and Smith, A. F. M. Efficient generation of random variates via the ratio-of-uniforms method. *Statist. Comput. 1* (1991), 129–133.

[30] Ziegler, G. M. *Lectures on Polytopes*, vol. 152 of *Graduate Texts in Mathematics.* Springer-Verlag, New York, 1995.