Shake-and-bake algorithms for generating uniform points on the boundary of bounded polyhedra

C.G.E. Boender, R.J. Caron, A.H.G. Rinnooy Kan, J.F. McDonald,

H. Edwin Romeijn, Robert L. Smith, J. Telgen, and, A.C.F. Vorst

Technical Report 89-24

July 1989

Shake-and-bake algorithms for generating uniform points on the boundary of bounded polyhedra

C.G.E. Boender[1] R.J. Caron[2] J.F. McDonald[2]
A.H.G. Rinnooy Kan[1] H.E. Romeijn[1] R.L. Smith[3] J. Telgen[4]
A.C.F. Vorst[5]

July 28, 1989

Abstract

We present a class of shake-and-bake algorithms for generating (asymptotically) uniform points on the boundary of full-dimensional bounded polyhedra. We also report chi-square goodness-of-fit tests, and the results of simulations for some elementary testproblems.

[1]Department of Operations Research, Erasmus University Rotterdam, P.O. Box 1738, 3000 DR Rotterdam, The Netherlands.

[2]Department of Mathematics and Statistics, University of Windsor, 401 Sunset Avenue, Windsor, Ontario N9B 3P4, Canada.

[3]Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, Michigan 48109-2117, U.S.A.

[4]Department of Applied Mathematics, Twente University, P.O. Box 217, 7500 AE Enschede, The Netherlands.

[5]Department of Mathematical Economics and Department of Mathematics, Erasmus University Rotterdam, P.O. Box 1738, 3000 DR Rotterdam, The Netherlands.

1 Introduction

In this paper we study the so-called *shake-and-bake* algorithms which, as far as we know, provide the only practical way of generating (asymptotically) uniform points on the boundary $\overline{S}$ of a full-dimensional polytope S. The fact that the points are uniformly distributed means that, in the long run, all subsets of $\overline{S}$ of equal size will be visited equally often, and subsets with positive measure will be visited with probability one. Part of the material which we present in this paper can be found in more detail in the technical reports Boender et al. [1988a,b]. For the so-called hit-and-run algorithms which generate asymptotically uniform points on the interior of S we refer to Smith [1984] and Berbee et al. [1987].

The relevance of generating points on $\overline{S}$, and the name shake-and-bake for this class of algorithms were mentioned earlier in Smith & Telgen [1981] in the context of detecting necessary constraints. Note that the shake-and-bake algorithms can also be used for the problem of optimizing functions which attain their optimal value on $\overline{S}$, such as, for example, the minimization of a concave function over S (see e.g. Patel & Smith [1983]).

Smith [1982] suggested the first shake-and-bake algorithm, referred to here as *original SB*, which generates a sequence of points which are asymptotically uniformly distributed on $\overline{S}$. Given an *iteration point* $x^0 \in \overline{S}$, a random search vector v is generated from the uniform distribution on the surface of the unit hypersphere with centre x^0. The intersection point y^0 of the line passing through x^0 with direction vector v with $\overline{S}$ is accepted as a *move point* (i.e. $x^1 = y^0$) with probability $\cos\phi_{x^0}/(\cos\phi_{x^0} + \cos\phi_{y^0})$, where ϕ_{x^0} and ϕ_{y^0} are the (acute) angles of the search vector v with the inward unit normals to $\overline{S}$ at x^0 and y^0, respectively; else $x^1 = x^0$. The intersection point y^0 is referred to as *hit point*. Hence, if the hit point y^0 is accepted as a move point, then the next iteration point x^1 is equal to y^0, else the next iteration point x^1 is equal to the current one x^0.

In the next SB algorithm, *limping SB*, the *move probability* is equal to $\cos\phi_{x^0}$, while the search vector v is drawn from the same distribution as for original SB. The limiting distribution of the sequence of iteration points, as well as the sequence of move points, generated by limping SB is proven to be uniform on $\overline{S}$. The third algorithm, *running SB*, chooses the search vector from a distribution such that the hit point is *always* accepted as a move point, while maintaining the important property that the distribution of the iteration points is asymptotically uniform on $\overline{S}$. (See figures 1 and 2.)

In this paper we give a proof for the uniform limiting distribution of the iteration points which applies to a complete class of shake-and-bake algorithms, including the three algorithms mentioned above.

The outline of this paper is as follows: in section 2 we will describe the general class of shake-and-bake algorithms and discuss the above special cases in more detail. Section 3 contains a proof that the limiting distribution of the sequence of iteration points generated by the algorithms is uniform on the boundary of S. Results of the chi-square goodness-of-fit test are given in section 4, and in section 5 we present some experimental results.

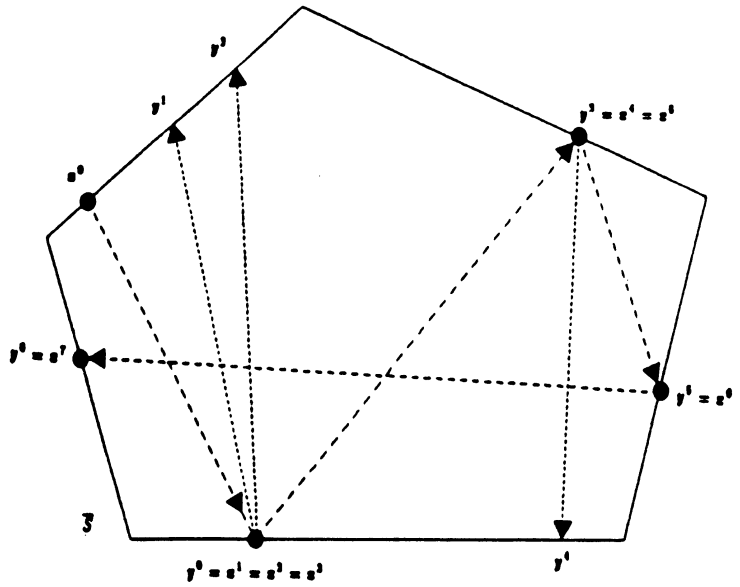

Figure 1: *Shake-and-bake algorithms original/limping SB*

Iteration 0: Hitpoint y^0 accepted → iteration point $x^1 := y^0$, x^1 is a move point.
Iteration 1: Hitpoint y^1 rejected → iteration point $x^2 := x^1$.
Iteration 2: Hitpoint y^2 rejected → iteration point $x^3 := x^2$.
Iteration 3: Hitpoint y^3 accepted → iteration point $x^4 := y^3$, x^4 is a move point.
Iteration 4: Hitpoint y^4 rejected → iteration point $x^5 := x^4$.
Iteration 5: Hitpoint y^5 accepted → iteration point $x^6 := y^5$, x^6 is a move point.
Iteration 6: Hitpoint y^6 accepted → iteration point $x^7 := y^6$, x^7 is a move point.

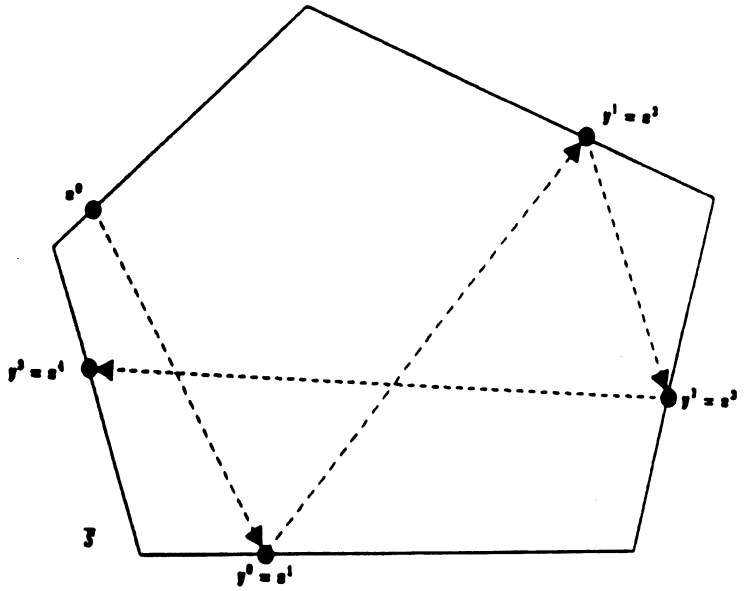

Figure 2: *Shake-and-bake algorithm running SB*

2 The shake-and-bake algorithms

2.1 Introduction

Consider a feasible region $S \subset R^d$ $(d \geq 2)$ defined by the following system of linear inequalities:

$$a_i'w \leq b_i \qquad (i = 1, \ldots, m) \tag{1}$$

where we have normalized the coefficients of the inequalities by choosing $\|a_i\| = 1$. Assume that S is bounded, nonempty and of full dimension. Then S is a polytope that containts interior points, i.e. points for which the inequalities (1) are all satisfied as strict inequalities. We assume, without loss of generality, that all of the constraints are nonredundant, i.e. none of them can be dropped from the system (1) without changing S.

Let $\overline{S}^0$ be the set of points in S for which exactly one constraint is binding, that is,

$$\overline{S}^0 = \bigcup_{i=1}^{m} \{w : a_i'w = b_i; a_j'w < b_j, j \neq i\}.$$

The shake-and-bake algorithms are based on a search from some point $x \in \overline{S}^0$ in a feasible direction v, i.e. if constraint k is binding at x, then $a_k'v < 0$. The intersection point y with the constraint hit first in the direction v is referred to as a *hit point* and becomes a *move point* with probability $\beta(y|x)$.

The hit point y is computed as follows. Let $\overline{V}_i$ be the bounding hyperplane of the half-space

$$V_i := \{w : a_i'w \leq b_i\} \qquad (i = 1, \ldots, m).$$

To determine y we compute all intersection points of the straight line passing through x with direction vector v, denoted by

$$x + \lambda v \qquad \lambda \in R$$

with the hyperplanes $\overline{V}_i$, $(i = 1, \ldots, m)$. It is easy to show that the intersection points correspond to the following values of λ:

$$\lambda_i = \frac{b_i - a_i'x}{a_i'v} \qquad (i = 1, \ldots, m).$$

Clearly the intersection point corresponding to the smallest positive value of λ is the hitpoint, which, for the algorithms described below, is an element of $\overline{S}^0$ with probability one.

2.2 The class of shake-and-bake algorithms

The class of shake-and-bake algorithms for polyhedral sets can be described as follows (see figure 1):

Step 0: Find some point $x^0 \in \overline{S}^0$. Define k as the index corresponding to the constraint that is binding at x^0. Set $n := 0$.

Step 1: Generate a direction vector v as follows: Select a random point z from an absolutely continuous probability distribution over the intersection of V_k with the surface of the d-dimensional unit hypersphere with centre x^n. Set $v := z - x^n$.

Step 2: Determine the hit point y^n as follows:

$$\begin{aligned} \lambda_i &:= \frac{b_i - a_i' x^n}{a_i' v} \qquad (i = 1, \ldots, m) \\ r &:= \arg \min_{1 \le i \le m} \{\lambda_i | \lambda_i > 0\} \\ y^n &:= x^n + \lambda_r v \end{aligned}$$

Go to *Step 4* with probability $\beta(y^n|x^n)$.

Step 3: Set $x^{n+1} := x^n$, i.e. the next iteration point is equal to the current one. Go to *Step 5*.

Step 4: Set $x^{n+1} := y^n$ and $k := r$, i.e. y^n becomes a move point.

Step 5: Set $n := n + 1$ and go to *Step 1*.

Note that from the description of the algorithms it can be concluded that the sequence of iteration points defines a *Markov chain* with a *stationary transition density function* and *continuous state space* $\overline{S}^0$, which will be useful in the subsequent analysis of the algorithms.

Because of our assumptions on the set S, there is a one-to-one correspondence between a feasible direction from a particular point in $\overline{S}^0$ and a hit point. We will define our class of algorithms by imposing conditions on the distribution of the search directions and on the move probability function $\beta(y|x)$.

Without loss of generality we write the density of the absolutely continuous component of the distribution of hit points in the form:

$$\rho(y|x) = f(x,y) \frac{b_y - a_y' x}{\|x - y\|^d}$$

where $\overline{V}_y := \{w : a_y' w = b_y\}$ is the hyperplane that is binding at $y \in \overline{S}^0$. Note that $b_y - a_y' x$ is equal to the distance from x to $\overline{V}_y$. Also note that $(b_y - a_y' x)/\|x - y\| = \cos \phi_y$, where $\cos \phi_y$ is as described in section 1. The move probability function will be written in the form

$$\beta(y|x) = g(x,y) \frac{b_y - a_y' y}{\|x - y\|}$$

where $\overline{V}_x$ is the hyperplane that is binding at $x \in \overline{S}^0$. Denote the density of the absolutely continuous component of the distribution of the ν-step transition probability of *moving* from $x \in \overline{S}^0$ to a neighborhood of $y \in \overline{S}^0$ by $p^{(\nu)}(y|x)$. The 1-step transition density function $p(y|x) = p^{(1)}(y|x)$ is then given by:

$$\begin{aligned} p(y|x) &= \beta(y|x)\,\rho(y|x) \\ &= h(x,y)\,\frac{(b_x - a_x'y)(b_y - a_y'x)}{\|x-y\|^{d+1}} \end{aligned}$$

where

$$h(x,y) = f(x,y)\,g(x,y)$$

The class of shake-and-bake algorithms we discuss in this paper is defined by imposing the following conditions on the function h:

1. $h(x,y)$ is symmetric in x and y, i.e. $h(x,y) = h(y,x)$ for all $x, y \in \overline{S}^0$.
2. $h(x,y)$ is uniformly bounded from below by zero, i.e. there exists a constant $\delta_h > 0$ such that $h(x,y) \geq \delta_h$ for all $x, y \in \overline{S}^0$, $x \neq y$.

Of course we also require $0 \leq \beta(y|x) \leq 1$ for all $x, y \in \overline{S}^0$. It is easily shown that the first condition implies that $p^{(\nu)}(y|x) = p^{(\nu)}(x|y)$ for all $x, y \in \overline{S}^0$, where $\nu = 1, 2, \ldots$ The class of shake-and-bake algorithms which satisfy conditions 1 and 2 will be referred to as *SB algorithms*.

2.3 Some special cases

In this section we will describe three specific SB algorithms.

Original SB

In this algorithm, the direction vector v has a uniform distribution over the feasible directions. The distribution of hit points is given by

$$\rho_O(y|x) = \frac{2\,(b_y - a_y'x)}{C_d\,\|x-y\|^d}$$

where

$$C_d = \frac{2\pi^{d/2}}{\Gamma\left(\frac{d}{2}\right)}$$

is the surface area of a d-dimensional hypersphere with unit radius.

The move probability function is given by

$$\beta_O(y|x) = \frac{(b_x - a_x'y)}{(b_x - a_x'y) + (b_y - a_y'x)}.$$

So for this algorithm we have

$$h_O(x,y) = \frac{2\,\|x-y\|}{C_d\,[(b_x - a_x'y) + (b_y - a_y'x)]} \geq \delta_{h_O}$$

with

$$\delta_{h_O} = \frac{1}{C_d} > 0$$

and the 1-step transition density function is equal to:

$$p_O(y|x) = \frac{2\,(b_x - a_x'y)(b_y - a_y'x)}{C_d\,\|x-y\|^d\,[(b_x - a_x'y) + (b_y - a_y'x)]}.$$

Limping SB

In this algorithm the direction vector has the same distribution as for original SB, i.e. $\rho_L(y|x) = \rho_O(y|x)$. The move probability function is given by

$$\beta_L(y|x) = \frac{b_x - a_x'y}{\|x-y\|} = -a_x'v.$$

Note that $\beta_L(y|x)$ does not depend explicitly on y. Thus, the hit point need not be computed unless it is accepted as a move point. For this algorithm we have

$$h_L(x,y) = \frac{2}{C_d}$$

so that

$$\delta_{h_L} = \frac{2}{C_d} > 0$$

and

$$p_L(y|x) = \frac{2\,(b_x - a_x'y)(b_y - a_y'x)}{C_d\,\|x-y\|^{d+1}}.$$

Running SB

A drawback to original and limping SB is the fact that not every hit point is a move point. This means that, in general, several random directions may have to be considered before a move point is generated. It also means that the rate of convergence of the iteration points to being *independently* and uniformly distributed will be slow, since successive iteration points will be highly correlated. The algorithm running SB solves this problem by taking

$$\beta_R(y|x) = 1.$$

The distribution of the direction vector is then chosen with the goal of obtaining a uniform limiting distribution. In particular, the random direction vector v is obtained as follows: Draw a point u from a uniform distribution on the (relative) interior of the intersection of the d-dimensional unit hypersphere centered at the origin and the hyperplane $\{w : a_x'w = 0\}$. The search vector v is defined as the vector with Euclidean norm 1 whose projection on the hyperplane $\{w : a_x'w = 0\}$ is equal to u, under the condition that $a_y'v < 0$. We will describe this in more detail below. This choice of distribution of the direction vector leads to the following transition density function:

$$p_R(y|x) = \frac{(b_x - a_x'y)(b_y - a_y'x)}{B_{d-1}\,\|x - y\|^{d+1}}$$

where

$$B_d = \frac{\pi^{d/2}}{\Gamma\left(\frac{d}{2}+1\right)}$$

is the volume of a d-dimensional hypersphere with unit radius. Note that for this algorithm

$$h_R(x,y) = \frac{1}{B_{d-1}}$$

so that

$$\delta_{h_R} = \frac{1}{B_{d-1}} > 0.$$

In this algorithm, we need to generate a point u from the uniform distribution on the (relative) interior of a $(d-1)$-dimensional unit hypersphere contained in some hyperplane, say $\{w : c'w = 0\}$ (with $\|c\| = 1$). This point can be obtained in the following way: First, draw a point $\tilde{u}$ from the uniform distribution on the surface of a d-dimensional unit hypersphere centered at the origin. The point $(I - cc')\tilde{u}/\|(I - cc')\tilde{u}\|$, which is the projection of $\tilde{u}$ on the hyperplane $\{w : c'w = 0\}$, rescaled to have norm equal to 1, is uniformly distributed on the surface of the $(d-1)$-dimensional unit hypersphere centered at the origin contained in that hyperplane. This point is now rescaled to have norm r, where r is chosen such that the resulting point u has the required distribution, which means that r^{d-1} has to be drawn from the uniform distribution on $(0,1]$.
So we have:

$$u = \frac{r(I - cc')\tilde{u}}{\|(I - cc')\tilde{u}\|} = \frac{r(I - cc')\tilde{u}}{\sqrt{1 - (c'\tilde{u})^2}}$$

and the corresponding direction vector is given by:

$$v = u - \sqrt{1-r^2}\,c = \frac{r}{\sqrt{1-(c'\tilde{u})^2}}\,\tilde{u} - \left(\frac{r\,(c'\tilde{u})}{\sqrt{1-(c'\tilde{u})^2}} + \sqrt{1-r^2}\right) c.$$

2.4 Limping and running SB reexamined

Given that the present iteration point is x, the conditional probability that the next hit point is also a move point is given by

$$P(x) = \int_{\overline{S}^0} \beta(y|x)\,\rho(y|x)\,dy = \int_{\overline{S}^0} p(y|x)\,dy.$$

Since for all x, $p(y|x) > 0$ for all y in a subset of $\overline{S}^0$ of measure greater than 0, it follows that for all SB algorithms $P(x) > 0$ for all x. Since the expected value of a random variable having a geometric distribution with parameter q is equal to $(1-q)/q$, the expected number of iterations required to generate a move point from x is equal to $1/P(x)$, so that this expected value is always finite.

For any shake-and-bake algorithm, say A, with transition density function $p(y|x)$ we can define another algorithm $\tilde{A}$ generating only the move points corresponding to A. (Obviously, when $P(x) = 1$ for all x (e.g. when A = running SB) the two algorithms are the same.) The transition density function $\tilde{p}(y|x)$ of algorithm $\tilde{A}$ can be expressed in terms of $p(y|x)$ and $P(x)$:

$$\begin{aligned}\tilde{p}(y|x) &= p(y|x) + (1-P(x))\,p(y|x) + (1-P(x))^2\,p(y|x) + \ldots \\ &= \sum_{i=0}^{\infty}(1-P(x))^i\,p(y|x) \\ &= \frac{1}{P(x)}\,p(y|x).\end{aligned}$$

It follows that $\tilde{h}(x,y) = h(x,y)/P(x)$. Of course, if the function $\tilde{h}$ satisfies conditions 1 and 2 in section 2.2 then algorithm $\tilde{A}$ is an SB algorithm. If A is an SB algorithm, then so is $\tilde{A}$ if and only if $P(x)$ is independent of x.

For limping SB the move probability $P_L(x)$ is equal to:

$$P_L(x) = \frac{2B_{d-1}}{C_d}.$$

This move probability is independent of x, so the algorithm generating the move points of limping SB is also an SB algorithm. Comparing the transition density functions of limping and running SB we observe that limping $\widetilde{\text{SB}}$ = running SB (see figures 1 and 2).

2.5 Computational efficiency

To determine a next iteration point all shake-and-bake algorithms have to generate a search vector. If the acceptance probability function of an algorithm depends explicitly on y, then the hit point corresponding to the search vector has to be computed. If this is not the case, then the hit point only needs to be computed if it is accepted as a move point.

For the three algorithms described above, we have that the generation of a search vector requires $O(d)$ time. Due to the $2d$ multiplications for each inequality of S the computation of the intersection points requires $O(md)$ time. This implies that for original and running SB the computation of an iteration point requires $O(md)$ time. For limping SB the acceptance probability does not depend explicitly on y, so the expected computation time per iteration point depends on the probability that a hit point is also a move point. Since $1/P_L(x) \approx \sqrt{\pi(d+1)/2}$ for large d, we have that this probability is $O(1/\sqrt{d})$, so the expected computation time per iteration point is $O(m\sqrt{d})$.

The foregoing analysis seems to suggest that, in some sense, limping SB is "better" than original or running SB. However, there will be a difference among SB algorithms in the rate of convergence to the uniform distribution. As noted before, the convergence rate of algorithms for which not every iteration point is a move point (e.g. original and limping SB) may well be much lower than for algorithms generating only move points, such as running SB. This issue will be addressed in some more detail in the next section.

3 The uniform limiting distribution

In this section we prove that for all SB algorithms the random sequence $\left\{X^i\right\}_{i=0}^{\infty}$ of iteration points converges to the uniform distribution on $\overline{S}^0$, independently of the starting point in the set $\overline{S}^0$.

We use the following theorem:

Theorem 1 *If*
(a) there exists a scalar $\delta > 0$ and a $\nu \geq 1$ such that $p^{(\nu)}(y|x) \geq \delta$ for all $x, y \in \overline{S}^0$, and
(b) $p(y|x) = p(x|y)$ for all $x, y \in \overline{S}^0$,
then the sequence $\left\{X^i\right\}_{i=0}^{\infty}$ of points has a uniform limiting distribution on $\overline{S}^0$.

Proof
It follows from (Doob [1953], p. 197) that (a) implies the existence of an asymptotic stationary distribution. Analogous to the proof given in (Smith [1984], p. 1300) it follows that (b) implies that the uniform distribution is the unique stationary distribution. □

As was noted in section 2.2, condition (b) is satisfied. What remains is to find a $\delta > 0$ and a $\nu \geq 1$ such that condition (a) is satisfied. Theorem 2 proves that (a) is satisfied with $\nu = 4$.

Theorem 2 *There exists a scalar $\delta > 0$ such that $p^{(4)}(y|x) \geq \delta$ for all $x, y \in \overline{S}^0$.*

The proof will be given after the following two lemmas.

Lemma 1 *There exists an $\hat{\epsilon}_0 > 0$ such that for every $x \in \overline{S}^0$ there exists an index j such that $b_j - a_j' x > \hat{\epsilon}_0$.*

Proof
Suppose such an $\hat{\epsilon}_0$ does not exist. This means that for all $\epsilon > 0$ there exists an $x_\epsilon \in \overline{S}^0$ such that $b_j - a_j' x_\epsilon \leq \epsilon$ for all j. So for all $n = 1, 2, \ldots$ there exists an $x_n \in \overline{S}^0$ such that $b_j - a_j' x_n \leq \frac{1}{n}$ for all j. Since $\overline{S}$ is compact, we know that the sequence $\{x_n\}_{n=1}^{\infty}$ has a limiting point $x_0 \in \overline{S}$ for which $b_j - a_j' x_0 = 0$ for all j. Now suppose y is an interior point of S, i.e. $b_j - a_j' y > 0$ for all j. Then $x_0 + \alpha(y - x_0)$ is an interior point for every $\alpha > 0$ since $b_j - a_j'(x_0 + \alpha(y - x_0)) = \alpha(b_j - a_j' y) > 0$ for all j. This implies that S is unbounded. Contradiction. □

Lemma 2 *For every i there exists an $\hat{\epsilon}_i > 0$ such that the set $\overline{V}_i^{\hat{\epsilon}_i} := \{x \in \overline{V}_i : b_j - a_j' x > \hat{\epsilon}_i \forall j \neq i\}$ has positive $(d-1)$-dimensional Lebesgue measure. (See figure 3.)*

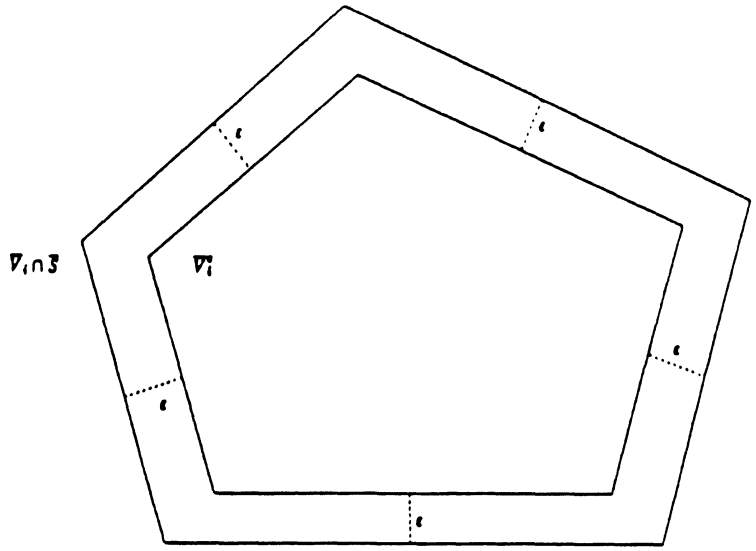

Figure 3: $\overline{V_i^\epsilon}$

Proof

Choose an arbitrary constraint i. We know that constraint i is nonredundant, so there exists an x_0 for which

$$b_i - a_i' x_0 = \mu_0 < 0$$

$$b_j - a_j' x_0 \geq 0 \qquad (j \neq i).$$

Suppose x_1 is an interior point of S, so

$$b_i - a_i' x_1 = \mu_1 > 0$$

$$b_j - a_j' x_1 > 0 \qquad (j \neq i).$$

Choose

$$x^* = \frac{\mu_1 x_0 - \mu_0 x_1}{\mu_1 - \mu_0}$$

which is a convex combination of x_0 and x_1. So we have

$$b_i - a_i' x^* = b_i - \frac{\mu_0}{\mu_1 - \mu_0}(\mu_1 - b_i) + \frac{\mu_1}{\mu_1 - \mu_0}(\mu_0 - b_i) = 0$$

$$b_j - a_j' x^* > 0 \qquad (j \neq i).$$

So $b_i - a_i' x^* = 0$ and $b_j - a_j' x^* \geq 2\hat{\epsilon}_i$ for all $j \neq i$, where

$$\hat{\epsilon}_i := \min_{j \neq i}(b_j - a_j' x^*)/2 > 0.$$

For all $z \in \overline{V}_i$ for which $\|z - x^*\| < \hat{\epsilon}_i$ we have

$$b_i - a_i' z = 0$$

$$b_j - a_j' z = b_j - a_j' x^* + a_j'(x^* - z) > 2\hat{\varepsilon}_i - \hat{\varepsilon}_i = \hat{\varepsilon}_i \qquad (j \neq i)$$

which implies that $z \in \overline{V}_i^{\hat{\varepsilon}_i}$. Hence, the set $\overline{V}_i^{\hat{\varepsilon}_i}$ has positive $(d-1)$-dimensional Lebesgue measure. □

Corollary
If we define $\hat{\varepsilon} := \min\{\hat{\varepsilon}_0, \min_i \hat{\varepsilon}_i\} > 0$ then for all $0 < \varepsilon < \hat{\varepsilon}$ and for all i the set $\overline{V}_i^{\varepsilon}$ has positive $(d-1)$-dimensional Lebesgue measure. **Corollary**

If we define $\hat{\varepsilon} := \min\{\hat{\varepsilon}_0, \min_i \hat{\varepsilon}_i\} > 0$ then for all $0 < \varepsilon < \hat{\varepsilon}$ and for all i the set $\overline{V}_i^{\varepsilon}$ has positive $(d-1)$-dimensional Lebesgue measure.

Proof of Theorem 2
For all $x, y \in \overline{S}^0$

$$p^{(4)}(y|x) \geq \int_{\overline{S}^0} \int_{\overline{S}^0} \int_{\overline{S}^0} p(z_1|x)\, p(z_2|z_1)\, p(z_3|z_2)\, p(y|z_3)\, dz_3\, dz_2\, dz_1.$$

Suppose $x \in \overline{V}_i$ and $y \in \overline{V}_j$. Choose some $0 < \varepsilon < \hat{\varepsilon}$. It follows from lemma 1 that there exists a $k \neq i$ and an $l \neq j$ such that $b_k - a_k' x > \varepsilon$ and $b_l - a_l' y > \varepsilon$. Choose some index $t \neq k, l$ (see figure 4). Define r_S as the maximal distance between two points in $\overline{S}$:

$$r_S := \max_{\eta_1, \eta_2 \in \overline{S}} \|\eta_1 - \eta_2\|.$$

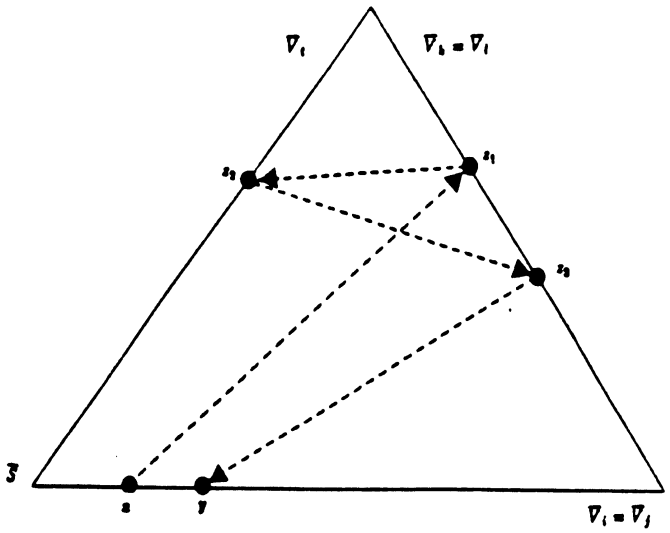

Figure 4: *From x to y in 4 steps*

Then:

1. $$p(z_1|x) = h(x,z_1)\frac{(b_i - a_i'z_1)(b_k - a_k'x)}{\|x - z_1\|^{d+1}} > \delta_h \frac{\varepsilon^2}{r_S^{d+1}}$$

   for all $z_1 \in \overline{V}_k$ satisfying $b_i - a_i'z_1 > \varepsilon$.

2. $$p(y|z_3) = h(z_3,y)\frac{(b_l - a_l'y)(b_j - a_j'z_3)}{\|z_3 - y\|^{d+1}} > \delta_h \frac{\varepsilon^2}{r_S^{d+1}}$$

   for all $z_3 \in \overline{V}_l$ satisfying $b_j - a_j'z_3 > \varepsilon$.

3. $$p(z_2|z_1) = h(z_1,z_2)\frac{(b_k - a_k'z_2)(b_t - a_t'z_1)}{\|z_1 - z_2\|^{d+1}} > \delta_h \frac{\varepsilon^2}{r_S^{d+1}}$$

   for all $z_1 \in \overline{V}_k$ satisfying $b_t - a_t'z_1 > \varepsilon$ and for all $z_2 \in \overline{V}_t$ satisfying $b_k - a_k'z_2 > \varepsilon$.

4. $$p(z_3|z_2) = h(z_2,z_3)\frac{(b_t - a_t'z_3)(b_l - a_l'z_2)}{\|z_2 - z_3\|^{d+1}} > \delta_h \frac{\varepsilon^2}{r_S^{d+1}}$$

   for all $z_2 \in \overline{V}_t$ satisfying $b_l - a_l'z_2 > \varepsilon$ and for all $z_3 \in \overline{V}_l$ satisfying $b_t - a_t'z_3 > \varepsilon$.

So we have

$$\begin{aligned} p^{(4)}(y|x) &\geq \int_{\overline{V}_k^\varepsilon}\int_{\overline{V}_t^\varepsilon}\int_{\overline{V}_l^\varepsilon} \delta_h^4 \frac{\varepsilon^8}{r_S^{4d+4}}\, dz_3\, dz_2\, dz_1 \\ &\geq \delta_h^4 \frac{\varepsilon^8}{r_S^{4d+4}} \cdot m_{d-1}(\overline{V}_k^\varepsilon) \cdot m_{d-1}(\overline{V}_t^\varepsilon) \cdot m_{d-1}(\overline{V}_l^\varepsilon) \\ &\geq \delta_h^4 \frac{\varepsilon^8}{r_S^{4d+4}} \cdot \left(\min_i m_{d-1}(\overline{V}_i^\varepsilon)\right)^3 =: \delta \end{aligned}$$

where $m_d(\cdot)$ denotes the d-dimensional Lebesgue measure of a set. Using $\varepsilon > 0$, $r_S < \infty$, $\delta_h > 0$, and the corollary of lemma 2 we know that $\delta > 0$. □

The value of δ can be used to compare the convergence rate of the SB algorithms using the following theorem from Doob [1953]:

Theorem 3 *If there exists a $\delta > 0$ and a $\nu \geq 1$ such that $p^{(\nu)}(y|x) \geq \delta$ for all $x, y \in \overline{S}^0$, then*

$$\left| \Pr\left\{ X^m \in A | X^0 = x^0 \right\} - \frac{m_{d-1}(A)}{m_{d-1}(\overline{S})} \right| \leq \left(1 - \delta m_{d-1}(\overline{S})\right)^{\frac{m}{\nu}-1}$$

for all $A \subset \overline{S}^0$ with positive $(d-1)$-dimensional Lebesgue measure, and for all $x^0 \in \overline{S}^0$.

Clearly, a larger value for δ corresponds with a faster rate of convergence to the uniform distribution. We now rewrite the expression for δ given above as follows:

$$\delta = \delta_h^4 \cdot \delta_S$$

where

$$\delta_S = \frac{\varepsilon^8}{r_S^{4d+4}} \cdot \left(\min_i m_{d-1}(\overline{V}_i^s) \right)^3$$

only depends on the region S, and *not* on the particular algorithm used. This means that we can compare the convergence rates of the SB algorithms by comparing the values of δ_h. Recall from section 2 that

$$\delta_{h_O} = \frac{1}{C_d}, \quad \delta_{h_L} = \frac{2}{C_d}, \text{ and } \delta_{h_R} = \frac{1}{B_{d-1}}.$$

It is easily seen that, for $d \geq 2$, the following holds:

$$\delta_{h_O} < \delta_{h_L} < \delta_{h_R}.$$

Moreover, as the function $h_R(x, y)$ is a constant, and the move probability equals one for this algorithm, we have

$$\delta_h \leq \delta_{h_R}$$

for all SB algorithms, since $\int_{\overline{S}} p(y|x)\, dy \leq 1$ for all SB algorithms. Thus, taking into account that we are discussing lower bounds, we may conjecture that the sequence of iteration points generated by running SB converges faster to the uniform distribution than the sequence of points generated by limping SB, which in turn converges faster to the uniform distribution than the sequence generated by original SB. Furthermore,

running SB has the fastest rate of convergence of all SB algorithms. It should however be noted that for a specific region S it might be possible to obtain a better lower bound than the one given in theorem 2, specifically for algorithms having a non-constant function h associated with them (like for instance original SB). Therefore, we should be careful in drawing strong conclusions from only the comparison of values for δ_h, when h is not equal to a constant, at least until some empirical results are taken into account.

4 Chi-square tests

In this section we will compare the performance of the three algorithms original, limping and running SB, measured by the rate of convergence to the uniform distribution. The three algorithms were run on two hyperrectangles, of dimension 5 and 10 respectively, of the following form:

$$0 \leq x_i \leq i \qquad (i = 1, \ldots, d).$$

We have chosen to test the algorithms on hyperrectangles because the area of the facets of these polytopes can be calculated analytically.

Any two iteration points are correlated, but the serial correlation between two points X^i and X^{i+N} goes to zero as N goes to infinity. So, to avoid serial correlation effects, we only sample one of every N iteration points, for different values of N. If N is large enough we can assume that the random variables $X^N, X^{2N}, \ldots$ are i.i.d. uniform. In that case we can use a chi-square goodness-of-fit test to test uniformity. We divided the hyperrectangles into a number of cells, according to the formula of Mann & Wald (see Mann & Wald [1942]). Each facet was divided into a number of exactly equal-sized cells, and the number of cells per facet was chosen such that the sizes of cells in different facets were almost equal.

For the first testproblem (of dimension $d = 5$) we chose $N = 10$, and we generated samples of different lengths n. For every sample size, we generated 5 different samples and performed the chi-square test at a significance level of 5%. The number of times a chi-square test was rejected is reported in table 1.

n	original SB	limping SB	running SB
200	2	2	0
400	1	1	1
600	0	1	1
800	0	1	0
1000	1	2	0

Table 1: $d = 5, N = 10$

For the second testproblem (of dimension $d = 10$) we used $N = 10, 25$ and 50. Again, the number of times a chi-square test was rejected is reported. See table 2.

An observation drawn from these tables is that in almost all cases the number of rejections of the null hypothesis is minimal for running SB. This means that the convergence of the distribution of the points generated by running SB to the uniform

n	original SB			limping SB			running SB		
N	10	25	50	10	25	50	10	25	50
200	4	0	0	5	2	1	2	0	0
400	3	1	1	3	1	1	2	1	1
600	5	3	1	5	1	1	1	1	1
800	4	3	1	5	2	2	2	1	1
1000	4	2	0	4	5	0	3	2	1

Table 2: $d = 10$

distribution is faster than the convergence of points generated by original or limping SB. This confirms our theoretical analysis of convergence rates in section 3. The convergence rate of original and limping SB appears to be almost equal. Numerical experiments (Caron & McDonald [1987]) have suggested that the average number of iteration points required to generate a move point grows faster (as a function of the dimension d) for limping SB than for original SB. Thus, the value of N necessary to eliminate serial correlation effects will also grow faster with growing d for limping SB. This could mean that, for growing d, original SB will perform better than limping SB, at least for smaller values of N. Thus the performance of original SB as opposed to limping SB still remains an open question, as our experiments fail to confirm the conjecture of section 3.

To summarize, we can conclude that running SB outperforms both limping SB and original SB.

5 Experimental results

We conclude the paper with some additional experimental results of the algorithm running SB. We let the figures speak for themselves.

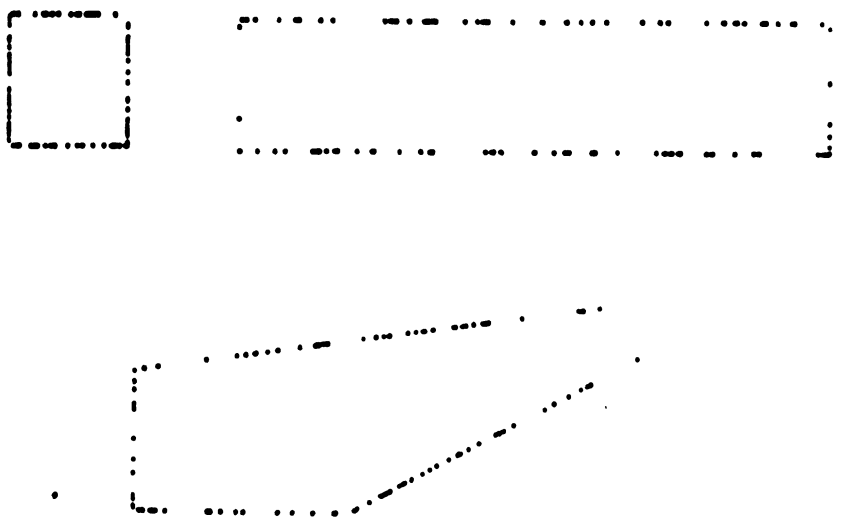

Figure 5: *100 iteration points*

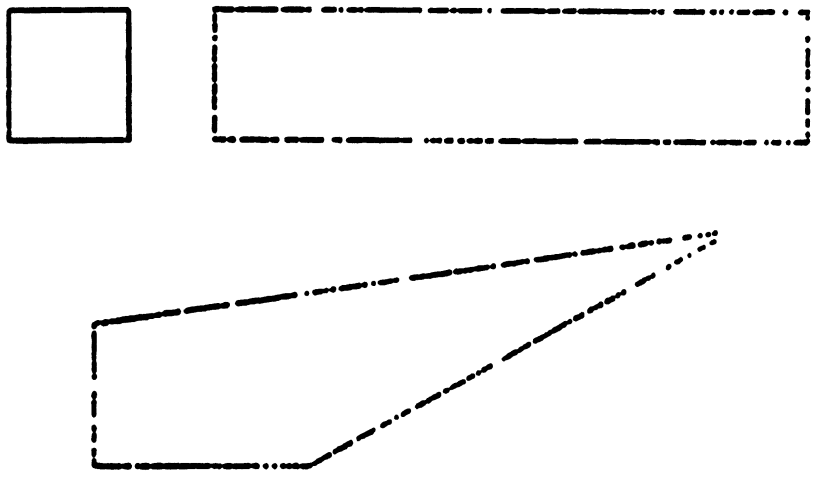

Figure 6: *500 iteration points*

References

[1] H.C.P. Berbee, C.G.E. Boender, A.H.G. Rinnooy Kan, C.L. Scheffer, R.L. Smith, and J. Telgen. Hit-and-run algorithms for the identification of nonredundant linear inequalities. *Mathematical Programming*, 37:184–207, 1987.

[2] C.G.E. Boender, R.J. Caron, J.F. McDonald, R.L. Smith, and J. Telgen. *A class of shake-and-bake algorithms for generating random points on the surface of a bounded region.* Technical Report WMR 88-08, Department of Mathematics and Statistics, University of Windsor, 1988a.

[3] C.G.E. Boender, A.H.G. Rinnooy Kan, H.E. Romeijn, and A.C.F. Vorst. *Shake-and-bake algorithms for generating uniform points on the boundary of bounded polyhedra.* Technical Report 8826/A, Econometric Institute, Erasmus University Rotterdam, 1988b.

[4] R.J. Caron and J.F. McDonald. Private communication, 1987.

[5] J.L. Doob. *Stochastic Processes.* Wiley, New York, 1953.

[6] H.B. Mann and A. Wald. On the choice of the number of class intervals in the application of the chi-square test. *The Annals of Mathematical Statistics*, 13:306–317, 1942.

[7] N.R. Patel and R.L. Smith. The asymptotic extreme value distribution of the sample minimum of a concave function under linear constraints. *Operations Research*, 31:789–794, 1983.

[8] R.L. Smith. Private communication, 1982.

[9] R.L. Smith. Efficient Monte Carlo procedures for generating random feasible points uniformly over bounded regions. *Operations Research*, 32:1296–1308, 1984.

[10] R.L. Smith and J. Telgen. *Random methods for identifying nonredundant constraints.* Technical Report 81-4, Department of Industrial and Operations Engineering, College of Engineering, The University of Michigan, Ann Arbor, 1981.