Using multiobjective optimization to map the entropy region

László Csirmaz *†‡

Abstract

Mapping the structure of the entropy region of at least four jointly distributed random variables is an important open problem. Even partial knowledge about this region has far reaching consequences in other areas in mathematics, like information theory, cryptography, probability theory and combinatorics. Presently, the only known method of exploring the entropy region is, or equivalent to, the one of Zhang and Yeung from 1998. Using some non-trivial properties of the entropy function, their method is transformed to solving high dimensional linear multiobjective optimization problems.

Benson's outer approximation algorithm is a fundamental tool for solving such optimization problems. An improved version of Benson's algorithm is presented, which requires solving one scalar linear program in each iteration rather than two or three as in previous versions. During the algorithm design, special care is taken for numerical stability. The implemented algorithm is used to verify previous statements about the entropy region, as well as to explore it further. Experimental results demonstrate the viability of the improved Benson's algorithm for determining the extremal set of medium-sized numerically ill-posed optimization problems. With larger problem sizes, two limitations of Benson's algorithm is observed: the inefficiency of the scalar LP solver, and the unexpectedly large number of intermediate vertices.

Keywords: multiobjective programming, effective solutions, entropy region, Benson's algorithm, entropy inequality.
AMC numbers: 90C60, 90C05, 94A17, 90C29

1 Introduction

Exploring the 15 dimensional entropy region formed by the entropies of the family of non-empty subsets of four random variables is an intriguing research problem. The entropy function maps the nonempty subsets of a finite set of jointly distributed random variables into the Shannon entropies of the marginal distributions. The range of the entropy function is the *entropy region*; it is a subset of a high-dimensional Euclidean space indexed by the non-empty subsets of the random variables. Inequalities that hold for the points of the region are called *information theoretic*. The entropy region is bounded by hyperplanes corresponding to the well-known Shannon information inequalities.

Presently, the only available method which goes beyond the standard Shannon inequalities is, or equivalent to, the one of Zhang and Yeung from 1998. The method starts with a description of "copy steps," which determine a (usually very) high dimensional linearly constrained region. The projection of this polytope onto the 15 dimensional space of the original entropies contains the entropy region, and, quite frequently, its facets yield new (linear) entropy inequalities. Using some non-trivial properties of the entropy region, this problem is transformed into the problem of finding all extremal (non-dominated) vertices of a 10-dimensional projection of a high dimensional polytope – a problem which lies in the realm of *linear multiobjective optimization*.

*Central European University and Rényi Institute, Budapest
†e-mail: csirmaz@renyi.hu
‡Supported by TAMOP-4.2.2.C-11/1/KONV-2012-0001 and the Lendulet program

Benson's outer approximation algorithm is a fundamental tool for solving multiobjective linear optimization problems. Compared to the original version and its refinements, we introduce a modification which leads to a significant improvement. The improvement is based on the observation that the scalar LP instance, which is used to decide whether an objective point is on the boundary of the projection or not, can also provide a separating facet when the point is outside the facet. In all earlier published versions of the algorithm, a separate LP instance was used to find such a facet.[1] This improvement is of independent interest as it applies to all versions of Benson's algorithm.

The algorithm presented in this paper is used successfully to check earlier results on the entropy region. It also generates hundreds of new entropy inequalities, and is essential in formulating a general conjecture about the limits of the Zhang–Yeung method. The experiments indicate the shortcomings of the implemented variant of the algorithm, and raise an interesting theoretical question about the structure of high dimensional polytopes.

1.1 The entropy region

The Shannon-inequalities bound the $2^N - 1$-dimensional entropy region; this bounding polytope is known as the *Shannon bound*. If $N = 2$, then the entropy region and the Shannon bound coincide; if $N = 3$, then the Shannon bound is the closure of the entropy region, and there are missing points on the boundary. (In fact, the boundary looks like a fractal, its exact structure is unknown.) In the case of $N \geq 4$, the Shannon bound strictly exceeds the closure of the entropy region [26].

To map the structure of the entropy region for $N \geq 4$ is an intriguing open problem. Even partial knowledge about this region has important consequences in several mathematical and engineering disciplines. N. Pippenger argued in [24] that linear information inequalities encode the fundamental laws of Information Theory, which determine the limits of information transmission and data compression. In communication networks, the capacity region of any multi-source network coding can be expressed in terms of the entropy region; see the thorough review on network coding in [3]. Information inequalities have a direct impact on the converse setting with multiple receivers [26]. In cryptography, such inequalities are used to establish bounds on the complexity of secret sharing schemes [4]. In probability theory, the implication problem of conditional independence among subvectors of a random vector can be rephrased as the investigation of the lower dimensional faces of the entropy region [22, 25]. Guessing number of games on directed graphs are related to network coding, where new bounds on the entropy region provide sharper bounds [2]. Information theoretic inequalities surface in additive combinatorics [17], and are intimately related to Kolmogorov complexity [18], determinant inequalities and group-theoretic inequalities [8].

The very first information theoretic inequality which showed that the entropy region is strictly contained in the Shannon bound was found by Zhang and Yeung in 1998 [28]. Since then, many other inequalities have been found based on their idea [11, 19]. So far this is the only technique at our disposal: all other proposed techniques were shown to be equivalent to the Zhang-Yeung method [16].

1.2 Mapping the entropy region is an optimization problem

Section 2 outlines why the technique of Zhang and Yeung sketched above is equivalent to solving a multiobjective linear optimization problem, with the main focus on describing the general form of these problems. Using some non-trivial properties of the entropy region, in the case of $N = 4$ random variables the *objective space* of the optimization can be reduced to be 10 dimensional. The exact details of how to generate the optimization problem are given in the Appendix. Solving these linear optimization problems is especially challenging as

a) the size of the problem grows exponentially, and becomes prohibitively large very soon;

[1] A. H. Hamel, A. Löhne and B. Rudloff in [14] have observed the same improvement independently.

b) while the linear constraints form a sparse matrix, there are many non-trivial linear combinations among them; consequently

c) the whole system is numerically ill-posed.

When $N \geq 5$, the corresponding optimization problems have less structure, are two order of magnitude larger, and even in the simplest case, no existing optimization technique seems to be able to handle them.

1.3 Benson's algorithm revisited

Benson's outer approximation algorithm [5] works in the low dimensional *objective* space, which in the $N = 4$ case has 10 dimensions, rather in the much larger (several hundred dimensional) problem space. It was a natural choice to use Benson's algorithm for solving the optimization problem described in Section 2. In Section 3 we describe an improved version of Benson's algorithm. The material of this section is of independent interest, as the improvement applies to all variants of Benson's algorithm as well. The original version [5] and other published variants [7, 12] use two scalar LP instances in each iteration step, while our version requires solving a single LP instance of the same size in each iteration. In typical applications of Benson's algorithm, the computation time outside the LP solver is negligible, thus the total running time of the algorithm is reduced considerably. The same improvement can be applied to the "dual" optimization problem as defined in [15]; thus both the primal and the dual problem require solving smaller number of scalar LP instances. The same improvement of Benson's algorithm has been observed independently and put into a wider context by Hamel, Löhne, and Rudloff in [14].

After sketching the general idea behind the improved version, we prove its correctness in Section 3.2. The termination of the algorithm is immediate from the facts that the extremal polytope has finitely many facets and finitely many vertices, and that each iteration generates either a new vertex or a new facet of the extremal polytope.

The modified algorithm is detailed in Section 3.4. It uses the *double description method* [13] for vertex enumeration. Section 3.5 discusses the modifications of the simplex-based LP solver we employed, which improves efficiency and numerical stability.

1.4 The results

Section 4 describes the experimental results. The presented algorithm handles successfully all 133 copy strings described in [11]. For each of those strings, all extremal solutions of the corresponding multiobjective linear optimization problem is generated. The 10 dimensional extremal solutions correspond to the "strongest" entropy inequalities that this copy string could yield.

Copy strings leading to larger optimization problems are also considered. As a result, the total number of known computer-generated information-theoretic entropy inequalities grows from 214 in [11] to more than 480.

Our algorithm runs successfully on a couple of significantly larger problems, where the symmetry of the problem allows us to reduce the dimension of the objective space (the dimension of the extremal polytope) from 10 to three. Results achieved here are essential in formulating and proving a general result about the limits of the Zhang–Yeung method [10].

Section 5 concludes the paper where we also discuss the shortcomings of the described variant of Benson's algorithm. The double description method, which is used to enumerate the intermediate vertices and facets, seems to be the bottleneck when the extremal polytope has several thousand vertices. The question how many extra vertices the intermediate polytopes might have compared to the final number of vertices and facets were investigated in the context of vertex enumeration algorithms [1]. When the dimension of the objective space is three, this increase can be linear, which is, apart form the constant, is the worst amount one can expect. In higher dimensions the increase can be as large as $s^{\lfloor\sqrt{d}\rfloor}$, see [6].

2 How mapping the entropy region leads to multiobjective optimization

As the entropy is non-negative, the entropy region is a subset of the non-negative orthant of the $2^N - 1$-dimensional Euclidean space. Its closure (in the usual Euclidean topology) is traditionally denoted by $\bar{\Gamma}^*_N$ [26]. It is known that $\bar{\Gamma}^*_N$ is a convex, closed, pointed cone, and the entropy region misses only boundary points, see [20].

The region $\bar{\Gamma}^*_N$ is bounded by linear facets corresponding to the so-called Shannon entropy inequalities. If $N \leq 3$, $\bar{\Gamma}^*_N$ is exactly the polytope determined by these Shannon inequalities, while for $N \geq 4$, it is a proper subset. Hyperplanes that cut into the Shannon polytope and contain the entropy region on one side are called *non-Shannon (entropy) inequalities*. The first such inequality was found by Zhang and Yeung in 1998 [28]. Since then, many other inequalities have been found. For a thorough discussion and new results, see [11, 16, 19, 28]. The Zhang–Yeung method can be outlined as follows. The process starts with four jointly distributed random variables: ξ_1, ξ_2, ξ_3, and ξ_4. They are split into two groups. An independent copy of the first group is created over the second group, and these new *auxiliary* random variables are added to the pool of existing (random) variables. Due to this conditional independence, several linear equations hold for their entropies. This copy step is then repeated as described by the *copy string*. The process yields an extension of the original set of random variables and a list of linear dependencies among their entropies. Then all Shannon inequalities for the entropies of this extended set are collected, and all linear dependencies are added. As the Shannon inequalities are linear, this results in a large set of (homogeneous) linear inequalities among the entropies of the subsets of the original and the auxiliary random variables. Finally, it is checked whether this set of homogeneous linear inequalities has any (new) consequences on the fifteen entropies of the original four variables ξ_1, ξ_2, ξ_3, and ξ_4.

Consider this system of linear inequalities written as

$$P'^T y + A'^T z \geq 0, \tag{1}$$

where $y \in \mathbb{R}^{15}$ corresponds to the entropies of the original four variables, $z \in \mathbb{R}^k$ is a vector of additional variables arising from the auxiliary random variables, and $A' \in \mathbb{R}^{k \times n}$ and $P' \in \mathbb{R}^{15 \times n}$ are the matrices of the corresponding coefficients. (1) is equivalent to

$$\text{for all } x \in \mathbb{R}^n, \text{ if } x \geq 0 \text{ then} x^T P'^T y + x^T A'^T z \geq 0.$$

The aim is to determine the coefficients $x^T P'^T$ under the constraint that the coefficients $x^T A'^T$ are zero, that is, to evaluate the pointed polyhedral cone

$$\mathcal{C}' = \{P'x : \ A'x = 0, \ x \geq 0\}.$$

Of course, it is sufficient to determine the extremal rays of $\mathcal{C}'$, because non-extremal rays produce inequalities which can be obtained as non-negative combinations of other inequalities.

Using information-theoretic considerations as discussed in [9, 21, 22, 27], it is more convenient to look at $\mathcal{C}'$ in a different coordinate system. Let us consider the cone

$$\mathcal{C} = U\mathcal{C}' = \{UP'x : \ A'x = 0, \ x \geq 0\},$$

where U is a 15×15 unimodular matrix (for details, please see the Appendix). There are many advantages of considering $\mathcal{C}$ instead of $\mathcal{C}'$. First, we can set one U-coordinate – the so-called Ingleton coordinate – to 1, cutting the pointed cone $\mathcal{C}$ to a polytope $\mathcal{P}$. Rather than searching for the extreme rays in $\mathcal{C}$, now we can search for the vertices of $\mathcal{P}$.

Second, the other 14 U-coordinates are all non-negative entropy expressions. Consequently, the coordinates of $y \in \mathcal{P}$ in these directions are necessarily non-negative, that is, $\mathcal{P}$ lies in the non-negative orthant of $\mathbb{R}^{14}$. Furthermore, if $y \in \mathcal{P}$ and $y' \geq y$ coordinatewise, then $y' \in \mathcal{P}$. Thus, the vertices of $\mathcal{P}$ are exactly the *non-dominated*, or extremal points of $\mathcal{P}$, where $y \in \mathcal{P}$ is non-dominated if no $y' \leq y$, different from y, is in $\mathcal{P}$.

Third, the polytope $\mathcal{P}$ is known to be the direct product of a 10-dimensional polytope $\mathcal{Q}$ and the non-negative orthant of $\mathbb{R}^4$, see [9]. Therefore, the non-dominated vertices of $\mathcal{P}$ are the non-dominated vertices of $\mathcal{Q}$ with four zero coordinates added. One can get the points of $\mathcal{Q}$ directly by merging these five additional constraints on x (that the Ingleton coordinate in $UP'x$ should be 1, and the four additional coordinates in $UP'x$ should be zero) to the original constraints $A'x = 0$.

The problem of finding the minimal set of entropy inequalities which generate (via non-negative linear combinations) all other entropy inequalities resulting from a given copy string has thus been transformed into the following multiobjective linear optimization problem.

Optimization Problem *Given the $m \times n$ matrix A, the $p \times n$ matrix P with $p = 10$ (both generated from the copy string), find the non-dominating vertices of the p-dimensional polytope*

$$\mathcal{Q} = \{Px : Ax = b,\, x \geq 0\}, \tag{2}$$

knowing that $\mathcal{Q}$ is in the non-negative orthant of $\mathbb{R}^p$, and the column vector b contains 1 in the Ingleton row, and zero elsewhere.

Indeed, given the linear constraints $Ax = b$, $x \geq 0$ in the n-dimensional problem space $\mathbb{R}^n$, one has to simultaneously minimize the p linear objectives given by the matrix Px. The solution of the optimization problem is the complete list of the non-dominated points of $\mathcal{Q}$. This list gives the coefficients of the minimal set of entropy inequalities which generates all consequences of the copy string.

3 Improved variant of Benson's outer algorithm

Benson's algorithm [5] solves the optimization problem defined in Section 2 working in the p-dimensional objective space, the range of Px, rather than in its much larger domain $\mathbb{R}^n$. The outline of this algorithm is as follows. It starts from a polytope S_0 containing $\mathcal{Q}$. In each iteration the algorithm maintains a convex bounding polytope S_n by listing all of its vertices and all of its facets. If all vertices of S_n are on the boundary of $\mathcal{Q}$, then the algorithm terminates, as the extremal (non-dominated) vertices of $\mathcal{Q}$ are among the vertices of S_n. Otherwise, we select a vertex of S_n that is not in $\mathcal{Q}$, connect it to an internal point of $\mathcal{Q}$, and find the intersection of this line with the boundary of $\mathcal{Q}$. Let the intersection point be $\hat{y}_n$. Then, we find a facet of $\mathcal{Q}$ which is adjacent to $\hat{y}_n$. We add this facet to S_n to get S_{n+1}, determine the new vertices of S_{n+1}, and iterate the method. The algorithm always terminates after finitely many iterations.

Several improvements have been suggested to the original algorithm: see, among others, [7, 12]. The first paper discusses how S_0 is chosen, which may have a heavy impact on the performance of the algorithm. Other papers suggest improvements related to the steps where we need to find the new vertices of S_{n+1} and decide whether a new vertex is on the boundary of $\mathcal{Q}$.

The improvement of Benson's algorithm this paper describes is achieved by merging the steps of finding a boundary point and finding the adjacent facet of $\mathcal{Q}$. We note that the same improvement was found independently in [14].

3.1 Non-dominated points

The transpose of matrix M is denoted by M^T. Vectors are usually denoted by small letters, and are considered single column matrices. For two vectors x and y of the same dimension, xy denotes their inner product, which is the same as the matrix product $x^T y$.

A is an $m \times n$ matrix mapping the *problem space* $\mathbb{R}^n$ to $\mathbb{R}^m$, and P is a $p \times n$ matrix mapping the problem space into the *objective space* $\mathbb{R}^p$. Let $\mathcal{A}$ be the polytope

$$\mathcal{A} = \{x \in \mathbb{R}^n : Ax = b,\, x \geq 0\,\},$$

and

$$\mathcal{Q} = \{Px :\, x \in \mathcal{A}\}.$$

For better clarity, y will denote points of the objective space $\mathbb{R}^p$, while points in the problem space will be denoted by x.

We want to generate the set of non-dominated vertices of $\mathcal{Q}$. Instead of $\mathcal{Q}$ we consider another polytope, $\mathcal{Q}^+$, which has the same set of non-dominated points, but is easier to handle [5]. This polytope is defined as

$$\mathcal{Q}^+ = \{y \in \mathbb{R}^p : y \geq y' \text{ for some } y' \in \mathcal{Q}\}. \tag{3}$$

In fact, $\mathcal{Q}^+$ is the Minkowski sum of $\mathcal{Q}$ and the non-negative orthant of $\mathbb{R}^p$, thus it is a convex closed polytope. The following facts can be found, e.g., in [12, Proposition 4.3].

Proposition 1. *a) The non-dominated points of $\mathcal{Q}^+$ and $\mathcal{Q}$ are the same.*
b) The weakly non-dominated points of $\mathcal{Q}^+$ are exactly its boundary points. □

3.2 Finding a boundary point and a supporting hyperplane at the same time

The crucial step in Benson's algorithm is to find a boundary point of $\mathcal{Q}^+$, and then to find the supporting hyperplane at that point. In the original version, it is achieved by solving two appropriately chosen scalar LP problems [5, 12, 15]. In our version, we need to solve only one scalar LP instance to do both jobs.

To describe the procedure, let $q \in \mathbb{R}^p$ be an internal point of $\mathcal{Q}^+$, and let $d \in \mathbb{R}^p$ be some direction in the p-dimensional space. Consider the ray starting at q with the direction $-d$, that is, points of the form $q - \lambda d \in \mathbb{R}^p$ where $\lambda \geq 0$ is a non-negative real number. If not the whole ray is in $\mathcal{Q}^+$, then there is a $\hat{\lambda} > 0$ so that $q - \lambda d \in \mathcal{Q}^+$ if and only if $\lambda \leq \hat{\lambda}$. By the next proposition, this threshold $\hat{\lambda}$ can be found by solving a scalar LP problem.

Proposition 2. *Suppose $q \in \mathcal{Q}^+$ is an internal point, and not the whole ray $\{q - \lambda d : \lambda \geq 0\}$ is in $\mathcal{Q}^+$. Let $\hat{\lambda}$ be the solution of the LP problem*

$$\max_{\lambda,x}\{\lambda : q - \lambda d \geq Px,\ Ax = b,\ x \geq 0\}. \qquad \mathrm{P}(q,d)$$

Then $\hat{y} = q - \hat{\lambda} d$ is a boundary point of $\mathcal{Q}^+$. Let moreover $(\hat{u}, \hat{v})$ be a place where the dual problem

$$\min_{u,v}\{bu + qv : A^T u + P^T v \geq 0,\ dv = 1,\ v \geq 0\} \qquad \mathrm{D}(q,d)$$

takes the same extremal value. Then $\{y \in \mathbb{R}^p : y\hat{v} = \hat{y}\hat{v}\}$ is a supporting hyperplane to $\mathcal{Q}^+$ at $\hat{y}$.

Proof. The first part of the proposition is clear: $q - \lambda d$ is in $\mathcal{Q}^+$ if it is $\geq Px$ for some $x \geq 0$ with $Ax = b$. The LP problem $\mathrm{P}(q,d)$ simply searches for the largest λ with this property. From this, it also follows that $\hat{y}$ is a boundary point of $\mathcal{Q}^+$: there are points arbitrarily close to $\hat{y}$ which are not in $\mathcal{Q}^+$.

The dual problem $\mathrm{D}(q,d)$ has the same optimal value as the primal one, thus

$$\min_{u,v}\{bu + qv : A^T u + P^T v \geq 0,\ dv = 1,\ v \geq 0\} = \hat{\lambda}. \tag{4}$$

Fixing the $\hat{v}$ part of the solution, the minimum is still $\hat{\lambda}$ as u runs over its domain $\mathbb{R}^m$, and therefore

$$\max_u\{-bu : A^T(-u) \leq P^T\hat{v}\} = q\hat{v} - \hat{\lambda}.$$

Consequently, its dual also has the same optimal value:

$$\min_x\{x(P^T\hat{v}) : Ax = b,\ x \geq 0\} = q\hat{v} - \hat{\lambda}.$$

Now $\hat{v} \geq 0$ by (4), and an arbitrary point y of $\mathcal{Q}^+$ can be written as $z + Px$ where $z \geq 0$, $z \in \mathbb{R}^p$ and $x \geq 0$, $Ax = b$, $x \in \mathbb{R}^n$, and so

$$(z + Px)^T\hat{v} = z\hat{v} + x(P^T\hat{v}) \geq 0 + q\hat{v} - \hat{\lambda}.$$

On the other hand, $d\hat{v} = 1$ by (4) again, thus

$$q\hat{v} - \hat{\lambda} = q\hat{v} - \hat{\lambda}(d\hat{v}) = (q - \hat{\lambda}d)\hat{v} = \hat{y}\hat{v}.$$

This means that for any point y of $\mathcal{Q}^+$, we have $y\hat{v} \geq \hat{y}\hat{v}$, furthermore $\hat{y}$ is in $\mathcal{Q}^+$. Thus $\{y \in \mathbb{R}^p : y\hat{v} = \hat{y}\hat{v}\}$ is a supporting hyperplane to $\mathcal{Q}^+$, as was claimed. □

3.3 Initial bounding polytope

Following the ideas of Burton and Ozlen in [7], we extend the objective space by *positive ideal* elements. As they showed, this extension does not restrict the applicability of the algorithm, but simplifies the intermediate polytopes.

First of all, we know that points of $\mathcal{Q}$ have non-negative coordinates. Thus $\mathcal{Q}^+$ is included in the non-negative orthant of $\mathbb{R}^p$, that is, it is part of the "ideal" p-dimensional simplex with the the origin and the p "positive endpoints" of the coordinate axes of $\mathbb{R}^p$ as its vertices.

During the algorithm we will be dealing with two kinds of "objective" points: ordinary (finite) points $y = \langle y_1, \ldots, y_p \rangle$ lying in the non-negative orthant (that is, $y_i \geq 0$), and *ideal points* at the positive endpoints of the rays $m = \langle m_1, \ldots, m_p \rangle$ in the non-negative orthant (that is, $m_i \geq 0$ for all i). Using homogeneous coordinates as suggested in [7], we can handle both types uniformly: ordinary points have coordinates $(y, 1) = \langle y_1, \ldots, y_p, 1 \rangle$, while ideal points can be conveniently written as $(m, 0)$, indicating that rays are invariant under multiplication by a (positive) scalar. A *hyperplane* is a $p+1$-tuple $(w, d) = \langle w_1, \ldots, w_p, d \rangle$. A point (z, j) is on a hyperplane if $(z, j)^T \cdot (w, d) = zw + jd = 0$, and it is on its *non-negative side* if $(z, j)^T \cdot (w, d) \geq 0$. All ideal points are on the non-negative side of the hyperplane (w, d) if and only if $w_i \geq 0$ for each i. The line connecting the points $(y, 1)$ and $(m, 0)$ intersects the hyperplane (w, d) in the (ordinary) point $(y + \lambda m, 1)$, where the scalar λ is determined by the condition

$$(y + \lambda m, 1)^T \cdot (w, d) = y^T w + \lambda m^T w + d = 0,$$

where $m^T w \neq 0$, as otherwise $(m, 0)$ is on the hyperplane.

Benson's algorithm starts with an *internal point* $q \in \mathcal{Q}^+$, and an *initial bounding polytope* $S_0 \supseteq \mathcal{Q}^+$. In the course of the algorithm, q will be connected to the vertices of the polytope S_n, and these lines will ideally intersect the boundary of $\mathcal{Q}^+$ in the relative interior of some $p-1$-dimensional facet. If this happens, then the supporting hyperplane to $\mathcal{Q}^+$ at the intersection point is uniquely determined, and is, in fact, a facet of $\mathcal{Q}^+$. This preferable event occurs if the internal point q is in *general position*, that is, not in any hyperplane determined by any p points among the union of the vertices of $\mathcal{Q}^+$ and S_0, …, S_n. As points of the objective space not in general position have measure zero, choosing q randomly from a set with positive measure will ensure that the above event will happen with probability 1. We will discuss the choice of q in detail later.

The choice of the initial surrounding polytope S_0 is quite natural. It is the p-dimensional ideal simplex with vertices $(0, \ldots, 0, 1)$ (the origin), and the ideal points $(e_i, 0)$, where the coordinates of the ray e_i are zero except for the i-th coordinate. p facets of this simplex are the coordinate hyperplanes with the equation $(e_i, 0)$. The $p+1$-st facet is the *ideal plane* containing all ideal points. As $\mathcal{Q}^+$ is not empty, this facet is, in fact, part of $\mathcal{Q}^+$, thus the ideal points $(e_i, 0)$ are *boundary points of* $\mathcal{Q}^+$. Moreover, it follows from the remark after the proof of Proposition 1 that ideal points are on the non-negative side of any supporting hyperplane of $\mathcal{Q}^+$.

3.4 Details of Benson's algorithm

Benson's algorithm constructs a sequence S_n of bounding polytopes. We discussed in Section 3.3 how to select the initial polytope S_0. We also have an internal point $q \in \mathcal{Q}^+$ *in general position*; we will return later to how a point in general position can be generated. With each polytope S_n we also maintain two sets: a set of hyperplanes such that the intersections of the non-negative

sides of these hyperplanes is exactly S_n, and the list of vertices of S_n, indicating whether each vertex is known to be a boundary point of $\mathcal{Q}^+$ or not.

Suppose we have obtained S_n, $n \geq 0$, and we proceed to generate S_{n+1}.

1. Let us look at the vertices of S_n, and select one which is not marked as a boundary point of $\mathcal{Q}^+$. If no such vertex can be found, then the algorithm terminates: according to Proposition 1, these vertices are the non-dominated vertices of $\mathcal{Q}^+$, thus the non-dominated vertices of $\mathcal{Q}$.

2. Let y be the vertex selected. Then, the boundary point of $\mathcal{Q}^+$ needs to be found on the line segment y—q by solving the scalar LP problem $\mathrm{P}(q, q-y)$. The solution is denoted by $\hat{\lambda}$. According to Proposition 2, if $\hat{\lambda} = 1$, then $q - (q - y) = y$ is on the boundary of $\mathcal{Q}^+$. If so, it is marked as such, and continue the algorithm at step 1.

3. Otherwise, $0 < \hat{\lambda} < 1$. Let us compute $\hat{y} = q - \hat{\lambda}(y - q)$, which point is on the boundary of $\mathcal{Q}^+$. By the second part of Proposition 2, if $(\hat{u}, \hat{v})$ is the solution to the dual problem $\mathrm{D}(q, q-y)$, then the hyperplane $h = (\hat{v}, -\hat{y}^T\hat{v})$ written in homogeneous coordinates is a supporting hyperplane to $\mathcal{Q}^+$ at $\hat{y}$, and $\mathcal{Q}^+$ is on its non-negative side. Also, as q is in general position, h is a facet of $\mathcal{Q}^+$. We can therefore add h to the hyperplanes of S_n to create the polytope S_{n+1}.

4. Next, the vertices of S_{n+1} need to be computed using the double description method, see [13] as follows. Vertices of S_n on the non-negative side of h remain vertices of S_{n+1}. All other vertices of S_{n+1} are the intersection points of the relative interiors of some edges (one dimensional face) of S_n and h.

Please note that in Step 4, all considered edges of S_n intersect h at different vertices of S_{n+1}, so there is no need to check for equality between them. This leaves us to determine whether a pair of vertices (v_1, v_2) of S_n is an edge of S_n. To this end, we use the following observation from [7].

Proposition 3. *Vertices v_1 and v_2 of S_n are connected by an edge if, and only if, every other vertex is missed by some facet of S_n containing both v_1 and v_2.* □

Proposition 3 suggests the following fast combinatorial test: if one considers all facets of S_n adjacent to both v_1 and v_2, and takes the intersection of the adjacency lists of these faces, then if this intersection contains v_1 and v_2 only, they form an edge of S_n, otherwise, they do not. To perform this test, we also need to maintain the adjacency lists of vertices and facets:

5. Adjust the adjacency lists of vertices and facets of S_{n+1} as follows:

   (a) create an adjacency list for h with all (old and new) vertices of S_{n+1} which are on it;

   (b) if the vertex v of S_n is on h, then add h to the adjacency lists of v;

   (c) if v is the intersection of h and the S_n edge v_1—v_2, then the adjacency list of v should consist of h and all facets of S_n adjacent to both v_1 and v_2; moreover v should be added as an adjacent vertex to these facets.

While the algorithm works with a mixture of ordinary and ideal points, fortunately, the initial ideal points are on the boundary of the polytope $\mathcal{Q}^+$, and because of this, the algorithm introduces no other ideal points. Ideal points may occur in steps 4 and 5 only, when calculating the intersection of the edge v_1 — v_2 of S_n with h; here v_1 or v_2, but not both may be one of the ideal points. Also, some of the ideal vertices might be adjacent to the new facet h, thus they may appear on the adjacency list of h (and *vice versa*).

The easiest way of finding a *random internal point* $q \in \mathcal{Q}^+$ is to get any feasible $x \in \mathcal{A}$ (that is, $x \geq 0$, $Ax = b$), and then increasing all coordinates of Px by some positive random amount. To find such a feasible x, solving an LP problem is required, but we can improve the algorithm further if we postpone finding q until it is actually needed, which is when we determine the first

new facet of S_1 in Step 2. As the only vertex of S_0 that is not a boundary point of $\mathcal{Q}^+$ is the origin, we can direct a ray from the origin in a random direction $d > 0$, find the intersection point of the ray and $\mathcal{Q}^+$ by solving the LP problem $\mathrm{P}(0, -d)$, and then set q on that ray at some random distance behind the intersection point. This way, we not only get the internal point q, but also the supporting hyperplane at the intersection of the segment 0—q and $\mathcal{Q}^+$.

If $\mathcal{Q}^+$ is not empty, then any ray $0 + \lambda d$ with $d > 0$ cuts into it. Therefore, for example, we can choose d so that every coordinate is a uniform random value between 1 and 2. If the LP problem $\mathrm{P}(0, -d)$ has no solution, then $\mathcal{Q}^+$ is empty; otherwise, let $\hat{\lambda} > 0$ be the solution, r be a uniform random value between 1 and 2, and we can set $q = (r + \hat{\lambda})d$, having $\hat{y} = \hat{\lambda}d$ as the point at the boundary of $\mathcal{Q}^+$.

3.5 The scalar LP problems

The scalar LP problems which are to be solved repeatedly during the algorithm have very similar structures. Figure 1 shows their general form. We indicated that there is only one row in matrix

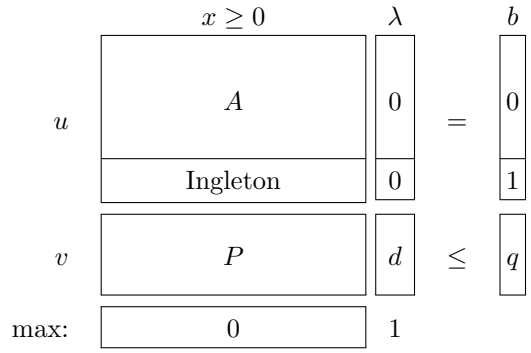

Figure 1: The structure of the $\mathrm{P}(q, d)$ problem

A where b is not zero. The internal point q is fixed throughout the algorithm; only direction d changes in each iteration. The uniformity of the LP problems can be used to speed up the initializations required by the LP solver.

3.6 Tweaks and modifications

Due to the nature of the original combinatorial problem, the optimization problem is ill-posed, and special care needs to be taken to maintain numerical stability. We dismissed using integer arithmetic as being prohibitively expensive both in the LP solver and during computing the vertices of the approximating polytopes. Instead, the LP solver and vertex computation were carefully modified to achieve better numerical stability.

The applied scalar LP solver is a standard but finely tuned simplex method. During the execution of the simplex method, one walks through the vertices of a large dimensional polytope while maintaining different descriptions of the polytope. Each vertex during the walk is determined by a set of the original facets (equations) whose intersection the vertex is. In simplex terminology, this set of facets is known as a *base*. Knowing the base is sufficient to regenerate the internal state at any step directly from the original facet equations. Therefore, after a predetermined number of steps we do two things: we save the actual base, and recalculate the description of the polytope. When we discover any problem caused by accumulating numerical errors, we return to the last saved base, and continue the method from that point on. We decided to save the base after each 60 steps, which seemed to be a good choice for the size of the problems to be solved.

As described in Section 3.4, Benson's algorithm advances by computing the vertices of the new approximating polytope S_{n+1} from the vertices (and facets) of S_n, and from a new facet of

$\mathcal{Q}^+$. The input for this computation comes from the LP solver, which gives the equation of the facet. In our case, all facets had rational coefficients with small denominators, and so they could be represented either exactly or with extremely small error. When computing the new vertices of S_{n+1}, we lose some of this exactness as the new vertices are computed from the intersections of an old edge and a new facet, and the angle between the edge and the facet can be very small. When this happens repeatedly, the coordinates of a vertex can be very far from their exact values. Therefore, rather than carrying these errors forward, we decided to compute the coordinates of each new vertex directly from the facets they are incident to.

Last, but not least, the modified Benson's algorithm requires a solution $\hat{v}$ of the dual LP problem $\mathrm{D}(q, d)$ only when the optimum $\hat{\lambda}$ is not one. This means that we can abort the LP solver as soon as it finds a feasible solution with $\lambda = 1$, further improving the speed of the algorithm.

4 Experimental results

The modified Benson's algorithm as described in the previous section has been successfully used to investigate the entropy region $\bar{\Gamma}_N^*$ by generating hundreds of new entropy inequalities. We show some of the results for three different data sets. In the first two of the sets, the objective space had 10 dimensions, while in the third one, this dimension was reduced to 3 using the internal symmetry of the original problem.

The algorithm was coded in C with an embedded scalar LP solver, a school-book simplex method with the tweaks discussed in Section 3.6. The compiled code was run on a dedicated personal computer with a 1GHz AMD Athlon 64 X2 Dual Core processor and 4 gigabytes of RAM. In fact, the program used only a single core (thus two instances could be run simultaneously on the dual-core machine without affecting the running time), and the combined memory usage was never above 2 gigabytes.

4.1 Results for $p = 10$

The paper by Dougherty et al [11] lists all extremal non-Shannon entropy inequalities which are consequences of at most three copy steps using no more than four auxiliary variables. They used 133 different copy strings to determine 214 new entropy inequalities. The modified Benson algorithm, as outlined in Section 3, was used to generate all non-dominated vertices of the polyhedrons determined by these copy strings. The results confirmed their report, and did not find any new entropy inequalities which were not consequences of the ones on their list.

Table 1 gives some representative data. *Copy strings* are explained in Appendix A.3. *Size* is the size of matrix A (columns and rows), followed by the number of vertices and facets. *Time* is the running time of the algorithm in hours, minutes and seconds. The running time is included here to indicate how it changes with the size of the problem, and does not indicate any comparison with other implementations of Benson's algorithm. The typical number of the non-dominated vertices is around a couple of hundred, with the only exception shown in the last row of Table 1, where the number of non-dominated vertices is 2506. Also, it might be worth mentioning that in this case, one of the intermediate polytopes had more than 22,000 vertices, which is an about 10-fold increase.

As the size of matrix A does not vary too much from case to case, we expected the running time of the algorithm to depend only on the number of iterations, that is, on the sum of the number of vertices and facets. The plot in Figure 2 confirms this expectation, indicating a linear dependence. The variance is due to the fact that occasionally, the LP solver failed and had to resume the work from an earlier stage, as it was discussed in Section 3.6.

There appears to be no easy way to predict the number of iterations, and thus the expected running time. Similar copy strings require a widely varying number of iterations, and have a varying number of non-dominated vertices. Let us also mention that in each of the 133 cases, the polytope $\mathcal{Q}^+$ factors to an 8-dimensional polytope and the non-negative quadrant of $\mathbb{R}^2$. The

Copy string	Size	Vertices	Facets	Time
r=c:ab;s=r:ac;t=r:ad	561×80	5	20	0:01
rs=cd:ab;t=r:ad;u=s:adt	1509×172	40	132	6:19
rs=cd:ab;t=a:bcs;u=(cs):abrt	1569×178	47	76	6:51
rs=cd:ab;t=a:bcs;u=b:adst	1512×178	177	261	17:40
rs=cd:ab;t=a:bcs;u=t:acr	1532×178	85	134	18:27
rs=cd:ab;t=(cr):ab;u=t:acs	1522×172	181	245	22:58
r=c:ab;st=cd:abr;u=a:bcrt	1346×161	209	436	29:18
rs=cd:ab;t=a:bcs;u=c:abrst	1369×166	355	591	38:59
rs=cd:ab;t=a:bcs;u=c:abrt	1511×178	363	599	1:04:32
rs=cd:ab;t=a:bcs;u=s:abcdt	1369×166	355	591	1:07:01
rs=cd:ab;t=a:bcs;u=(at):bcs	1555×177	484	676	1:39:30
rs=cd:ab;t=a:bcs;u=a:bcst	1509×177	880	1238	4:30:26
rs=cd:ab;t=a:bcs;u=a:bdrt	1513×177	2506	2708	5:11:25

Table 1: Representative results for the Dougherty et al list ($p = 10$)

fact that the objective space is practically 8 dimensional rather than 10 might have a significant impact on the observed speed of the algorithm.

The implemented algorithm was used to check consequences of copy strings beyond the ones considered in [11]. This resulted in increasing the total number of known computer-generated entropy inequalities from 214 in [11] to more than 480. New inequalities with all coefficients below 100 are listed in Appendix B. Some representative cases are shown in Table 2. As the copy

Copy string	Size	Vertices	Facets	Time
rs=cd:ab;tu=cr:ab;v=(cs):abtu	4055×370	19	58	1:10:10
rs=ad:bc;tu=ar:bc;v=r:abst	4009×370	40	103	3:24:37
rs=cd:ab;t=(cr):ab;uv=cs:abt	3891×358	30	102	3:34:31
rs=cd:ab;tu=cr:ab;v=t:adr	3963×362	167	235	9:20:19
rs=cd:ab;tu=dr:ab;v=b:adsu	4007×370	318	356	13:20:08
rs=cd:ab;tv=dr:ab;u=a:bcrt	4007×370	318	356	14:34:42
rs=cd:ab;tu=cs:ab;v=a:bcrt	4007×370	297	648	22:02:39
rs=cd:ab;t=a:bcs;uv=bt:acr	3913×362	779	1269	37:15:33
rs=cd:ab;tu=cr:ab;v=a:bcstu	3987×362	4510	7966	427:43:30
rs=cd:ab;tu=cs:ab;v=a:bcrtu	3893×362	10387	13397	716:36:32

Table 2: Some extended copy strings with $p = 10$

string becomes more involved, the generated problem becomes larger, and we have also observed a significant increase in the number of vertices of the intermediate polytopes. This unexpected increase makes the double description method highly inefficient, and it actually becomes the bottleneck in the algorithm. In most of the cases listed in Table 2, we had to set the bound on the number of facets and vertices to 2^{16}, as smaller values were exhausted very fast. This in turn meant that the size of the incidence matrix of vertices and facets was $2^{16} \times 2^{16}$, and that executing the combinatorial test of Proposition 3 for each pair v_1 and v_2 of several thousand vertices took a considerable amount of time, and became comparable to the time taken by the LP solver. In about 20% of the extended cases investigated, even this limit was too small, and the algorithm aborted.

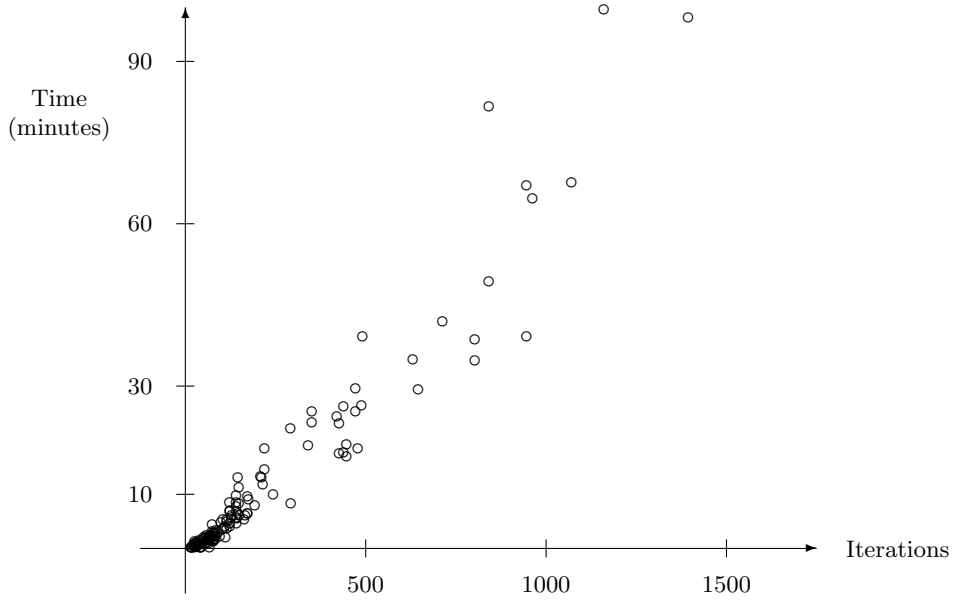

Figure 2: Running time vs. problem size

4.2 Results for $p = 3$

Table 3 shows statistics for a set of even larger problems. In these problems `cd` is copied repeatedly over all previously introduced variables, as is described by the copy string

`rs=cd:ab; tu=cd:abrs; vw=cd:abrstu; xy=cd:abrstuvw`.

The optimization problem is identified by the total number of random variables employed during the copy steps; this number is 12 for the above copy string. Relying on the high symmetry in these problems and using some careful preprocessing, we were able to reduce the problem size

Random variables	Size	Vertices	Facets	Running time
10	692×99	11	13	0:01
12	1298×150	26	24	0:55
14	2175×233	53	43	9:38
16	3373×338	100	78	2:18:45
18	4942×474	171	129	6:55:40
20	5772×635	278	208	23:20:17

Table 3: More random variables with $p = 3$

significantly: 20 random variables have more than 10^6 entropies, and more than $4.8 \cdot 10^7$ Shannon inequalities; these numbers have been reduced to 635, and 5772, respectively. The symmetry in the problems also made it possible to reduce the dimensions of the objective space from 10 to 3.

Problems listed in this section had their running time determined almost exclusively by the LP solver. Solving the numerous numerically ill-posed scalar LP problems of such sizes appears to be at the limit of the simplex-based LP solver implemented in our software. It would be an interesting research problem to investigate which LP solvers can handle problems of this size efficiently and with sufficient accuracy.

5 Conclusion

All presently known methods exploring the entropy region are equivalent to the one of Zhang and Yeung from 1998 [28]. In case of four random variables, using non-trivial properties of the entropy region, the method was transformed into the problem of finding all non-dominated vertices of a 10-dimensional projection of a high dimensional polytope. A modified variant of Benson's algorithm was used to solve this linear multiobjective optimization problem. Our variant uses one scalar LP instance in each iteration instead of two, practically doubling the speed of the algorithm.

The implemented algorithm was used successfully to check the results on the 133 copy strings of [11], and confirm that no entropy inequality has been missed. Copy strings leading to larger optimization problems were also investigated. This resulted in more than doubling the number of known computer-generated entropy inequalities.

The algorithm was also run on some extremely symmetrical, but significantly larger data sets, where the objective space could be reduced to three dimensions. The results achieved were essential in forming a general conjecture about the limits of the Zhang–Yeung method [10].

A compelling theoretical problem arose as the experimental results were evaluated. It has been observed that some of the intermediate polytopes S_n (bounded by a certain subset of the facets of $\mathcal{Q}^+$) had significantly larger number of vertices than $\mathcal{Q}^+$ itself. It is worth noting that while this phenomenon might occur when the objective space is three dimensional, the number of intermediate vertices cannot grow too large. Indeed, according to the Euler formula, the number of vertices of any convex 3-dimensional polytope is bounded by $2f-4$, where f is the number of its facets. The situation changes dramatically when the dimension p of the objective space increases. David Bremner in [6] constructs a p-dimensional polytope with f facets (f can be arbitrary large) such that removing *any* facet increases the number of vertices proportional to $f^{-1+\sqrt{p}}$.

The shortcomings of using the double description method have been observed earlier [23], and alternative algorithms have been suggested to generate the extremal rays of pointed cones. In our case, however, these algorithms could not be used directly, as Benson's algorithm generates facets *and* non-dominated vertices simultaneously, and so we do not know all the facets in advance. The procedure outlined in Section 3.2 can be considered as an "oracle call," which, given any point in the objective space, returns whether the point is outside the polytope $\mathcal{Q}^+$, and if yes, also provides a facet of $\mathcal{Q}^+$ that separates the polytope from this point. The challenge then becomes to devise an algorithm which calls the oracle, and finds all vertices of $\mathcal{Q}^+$ in time and space *linear* in the final number of vertices, plus the number of facets.

Investigating the entropy region of *five random variables* instead of four appears to be significantly harder. The very first obstacle is the lack of our understanding even of the Shannon region of this 31-dimensional space. There does not seem to be any suitable coordinate transformation similar to the unimodular matrix U which would simplify the structure of some cross-sections of this region as U did in the four-variable case.

Acknowledgments

The author would like to acknowledge the help received during the numerous insightful, fruitful, and enjoyable discussions with Frantisek Matúš on the entropy function, matroids, and on the ultimate question of everything.

Appendix A

A.1 Shannon inequalities

Recall that given a discrete random variable x with possible values $\{a_1, a_2, \ldots, a_n\}$ and probability distribution $\{p(a_i)\}_{i=1}^n$, the Shannon entropy of x is defined as $\mathbf{H}(x) = -\sum_{i=1}^n p(a_i) \log p(a_i)$ which is a measure of the average uncertainty associated with x. Let $\langle x_i : i \in I \rangle$ be a collection of random variables. For $A \subseteq I$, we let $x_A = \langle x_i : i \in A \rangle$, and $\mathbf{H}(x_A)$ be the entropy of x_A equipped

with the marginal distribution. Thus the entropy function $\mathbf{H}$ associated with collection $\langle x_i : i \in I \rangle$ maps the non-empty subsets of I to non-negative real numbers. The *Shannon inequalities* say that this $\mathbf{H}$ is a monotone and submodular function, that is,

$$0 \leq \mathbf{H}(x_A) \leq \mathbf{H}(x_B) \quad \text{when } A \subseteq B, \tag{5}$$

and

$$\mathbf{H}(x_{A \cup B}) + \mathbf{H}(x_{A \cap B}) \leq \mathbf{H}(x_A) + \mathbf{H}(x_B), \tag{6}$$

for all subsets A, B of I. There are redundant inequalities among the Shannon inequalities. For example, the following smaller collection implies all others: consider the inequalities from (5) where $B = I$, and A is missing only one element of I; and the inequalities from (6) where both A and B has exactly one element not in $A \cap B$.

A.2 Independent copy of random variables

We split a set of random variables into two disjoint groups $\langle x_i : i \in I \rangle$ and $\langle y_j : j \in J \rangle$, and create $\langle x'_i : i \in I \rangle$ as an *independent copy* of $\langle x_i \rangle$ over $\langle y_j \rangle$. It means that $\langle x'_i \rangle$ and $\langle x_i \rangle$ have the same set of possible values, and

$$\begin{aligned} &\text{Prob}\left(\langle x'_i{=}a'_i \rangle, \langle x_i{=}a_i \rangle, \langle y_j{=}b_j \rangle\right) = \\ &= \frac{\text{Prob}\left(\langle x_i{=}a'_i \rangle, \langle y_j{=}b_j \rangle\right) \cdot \text{Prob}\left(\langle x_i{=}a_i \rangle, \langle y_j{=}b_j \rangle\right)}{\text{Prob}\left(\langle y_j{=}b_j \rangle\right)}, \end{aligned}$$

expressing that $\langle x'_i \rangle$ and $\langle x_i \rangle$ are independent over $\langle y_j \rangle$. The entropy of certain subsets of $\langle x'_i, x_i, y_j \rangle$ can be computed from the entropy of other subsets as follows. Let $A, B \subseteq I$ and $C \subseteq J$. Then,

$$\mathbf{H}(x'_A x_B y_C) = \mathbf{H}(x'_B x_A y_C),$$

which is due to the complete symmetry between $\langle x'_i \rangle$ and $\langle x_i \rangle$. The fact that x'_I and x_I are independent over y_J translates into the following entropy equality:

$$\mathbf{H}(x'_A x_B y_J) = \mathbf{H}(x'_A y_J) + \mathbf{H}(x_B y_J) - \mathbf{H}(y_J)$$

for all subsets $A, B \subseteq I$.

A.3 Copy strings

The process starts by fixing four random variables `a`, `b`, `c`, and `d` with some joint distribution. Split them into two parts, create an independent copy of the first part over the second, add the newly created random variables to the group, and then repeat this process. To save on the number of variables created, in each step certain newly generated variables can be discarded, or two or more new variables can be merged into a single one. This process is described by a *copy string*, which has the following form:

```
rs=cd:ab; t=(cr):ab; u=t:acs
```

This string describes three iterations which are separated by semicolons. In the first step we create an independent copy of `cd` over `ab`, and name the two new variables by `rs` such that `r` is a copy of `c`, and `s` is a copy of `d`. After this step we have six variables `abcdrs` with some joint distribution. In the next step we make an independent copy of `cdrs` over `ab`, merge the copies of `c` and `r` to a single variable, name it `t`, and add it to the pool. In the last step create an independent copy of `bdrt` over `acd`, keep the copy of `t`, name it `u`, and discard the other three newly created variables. As the result, we get the eight random variables `abcdrstu`.

	a	b	c	d	ab	ac	ad	bc	bd	cd	abc	abd	acd	bcd	$abcd$
Ingleton	-1	-1	0	0	1	1	1	1	1	-1	-1	-1	0	0	0
	0	0	-1	0	0	1	0	1	0	0	-1	0	0	0	0
	0	0	0	-1	0	0	1	0	1	0	0	-1	0	0	0
	0	-1	0	0	1	0	0	1	0	0	-1	0	0	0	0
	-1	0	0	0	1	1	0	0	0	0	-1	0	0	0	0
	0	-1	0	0	1	0	0	0	1	0	0	-1	0	0	0
	-1	0	0	0	1	0	1	0	0	0	0	-1	0	0	0
	-1	0	0	0	0	1	1	0	0	0	0	0	-1	0	0
	0	-1	0	0	0	0	0	1	1	0	0	0	0	-1	0
	0	0	1	1	0	0	0	0	0	-1	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	-1	0	0	1	1	-1
z	0	0	0	0	0	0	0	0	0	0	0	0	0	-1	1
z	0	0	0	0	0	0	0	0	0	0	0	0	-1	0	1
z	0	0	0	0	0	0	0	0	0	0	0	-1	0	0	1
z	0	0	0	0	0	0	0	0	0	0	-1	0	0	0	1

Table 4: The unimodular matrix

A.4 A unimodular matrix

It is advantageous to look at the 15 entropies of the four random variables in another coordinate system. The new coordinates can be computed using the unimodular matrix shown in Table 4. Columns represent the entropies of the subsets of the four random variables a, b, c and d, as indicated in the top row. The value of the "Ingleton row" should be set to 1, and rows marked by the letter "z" vanish for all extremal vertices, thus they should be set to 0.

Appendix B

This section lists new entropy inequalities which were found during the experiments described in Section 4 and have all coefficients less than 100. Each entry in the list contains nine integers representing the coefficients c_0, c_1, …, c_8 for the non-Shannon information inequality of the form

$$\begin{aligned} &c_0\big(\mathbf{I}(c,d) - \mathbf{I}(a,b) + \mathbf{I}(a,b\,|\,c) + \mathbf{I}(a,b\,|\,d)\big) + \\ &+ c_1\mathbf{I}(a,b\,|\,c) + c_2\mathbf{I}(a,b\,|\,d) + \\ &+ c_3\mathbf{I}(a,c\,|\,b) + c_4\mathbf{I}(b,c\,|\,a) + c_5\mathbf{I}(a,d\,|\,b) + c_6\mathbf{I}(b,d\,|\,a) + \\ &+ c_7\mathbf{I}(c,d\,|\,a) + c_8\mathbf{I}(c,d\,|\,b) \geq 0. \end{aligned}$$

Here $\mathbf{I}(A,B) = \mathbf{H}(A)+\mathbf{H}(B)-\mathbf{H}(AB)$ is the mutual information, $\mathbf{I}(A,B\,|\,C) = \mathbf{H}(AC)+\mathbf{H}(BC)-\mathbf{H}(ABC)-\mathbf{H}(C)$ is the conditional mutual information. The expression after c_0 is the Ingleton value. Following the list of coefficients is the applied copy string.

1) 2 1 0 3 1 0 0 3 0 rs=bd:ac;tu=cd:abs;v=b:acrst
2) 3 4 0 4 1 1 1 0 4 rs=ad:bc;tu=cd:abs;v=a:bcrst
3) 4 2 0 5 4 0 2 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
4) 4 2 0 5 4 2 0 0 0 rs=cd:ab;t=a:bcs;uv=bt:acr
5) 4 3 1 2 0 10 5 0 0 rs=cd:ab;tu=cr:ab;v=(rtu):ad
6) 4 4 0 5 2 0 0 0 2 rs=ad:bc;t=c:abs;uv=bt:acr
7) 4 5 0 4 2 0 0 0 2 rs=ad:bc;t=c:abs;uv=bt:acr
8) 4 5 1 3 2 2 0 6 0 rs=bd:ac;tu=cd:abs;v=b:acrst
9) 5 1 0 6 6 6 5 0 0 rs=cd:ab;tu=dr:ab;v=b:adstu
10) 5 1 0 9 8 2 2 0 0 rs=cd:ab;tu=dr:ab;v=b:adst
11) 5 3 0 6 5 1 0 0 0 rs=cd:ab;t=a:bcs;uv=bt:acr
12) 5 4 2 3 0 9 4 0 0 rs=cd:ab;t=(cr):ab;uv=rt:ad
13) 5 5 0 17 1 7 7 0 0 rs=cd:ab;t=a:bcr;uv=ds:abt
14) 6 1 0 7 7 10 10 0 0 rs=cd:ab;tu=cs:ab;v=(ds):abtu
15) 6 1 0 8 8 7 7 0 0 rs=cd:ab;tu=cs:ab;v=(ds):abtu
16) 6 1 0 12 10 3 5 0 0 rs=cd:ab;tu=cs:ab;v=b:acrtu
17) 6 1 0 12 9 4 6 0 0 rs=cd:ab;tu=cs:ab;v=b:acrtu
18) 6 1 0 13 12 2 3 0 0 rs=cd:ab;tu=cr:ab;v=a:bcstu
19) 6 1 0 14 12 3 2 0 0 rs=cd:ab;tu=cr:ab;v=b:acstu
20) 6 1 0 14 11 2 3 0 0 rs=cd:ab;tu=cr:ab;v=a:bcstu
21) 6 1 0 16 16 1 1 0 0 rs=cd:ab;tu=cr:ab;v=(cr):abtu
22) 6 2 2 18 7 3 0 0 0 rs=cd:ab;tu=cr:ab;v=a:bcstu
23) 6 2 2 19 7 2 0 0 0 rs=cd:ab;tu=cr:ab;v=a:bcstu
24) 6 3 0 8 4 2 4 0 1 rs=cd:ab;t=b:acs;uv=at:bcr
25) 6 3 2 3 0 12 7 0 0 rs=cd:ab;tu=cr:ab;v=(rtu):ad
26) 6 4 0 8 3 3 4 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
27) 6 5 2 26 3 2 0 0 0 rs=ac:bd;tu=ar:bd;v=a:bcrtu
28) 6 7 0 4 4 1 1 7 0 rs=bd:ac;tu=cd:abs;v=b:acrst
29) 6 7 2 5 0 12 5 0 0 r=c:ab;st=cd:abr;uv=cr:at
30) 7 1 0 9 9 13 13 0 0 rs=cd:ab;t=(cs):ab;uv=ds:abt
31) 7 1 0 10 10 9 9 0 0 rs=cd:ab;tu=cs:ab;v=(ds):abtu
32) 7 1 0 15 15 3 3 0 0 rs=cd:ab;t=(cr):ab;uv=cs:abt
33) 7 1 0 20 20 2 2 0 0 rs=cd:ab;tu=cr:ab;v=(cr):abtu
34) 7 7 2 28 3 2 0 0 0 rs=ac:bd;tu=ar:bd;v=a:bcrtu
35) 8 1 0 14 14 7 7 0 0 rs=cd:ab;t=(cr):ab;uv=cs:abt
36) 8 1 0 16 16 5 5 0 0 rs=cd:ab;t=(cr):ab;uv=cs:abt
37) 8 1 0 18 18 4 4 0 0 rs=cd:ab;t=(cr):ab;uv=cs:abt
38) 8 4 0 9 5 4 7 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
39) 8 4 0 10 5 4 6 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
40) 8 6 0 10 4 3 4 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
41) 8 9 3 7 0 15 6 0 0 r=c:ab;st=cd:abr;uv=cr:at
42) 9 1 0 17 17 8 8 0 0 rs=cd:ab;t=(cr):ab;uv=cs:abt

43) 9 1 0 19 19 6 6 0 0 rs=cd:ab;t=(cr):ab;uv=cs:abt
44) 9 1 0 22 22 5 5 0 0 rs=cd:ab;t=(cr):ab;uv=cs:abt
45) 9 3 0 16 7 11 12 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
46) 9 3 0 16 7 13 10 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
47) 9 6 0 11 5 3 7 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
48) 9 6 0 12 5 3 5 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
49) 9 6 6 12 5 6 0 0 0 rs=cd:ab;t=(cr):ab;uv=rt:ad
50) 9 7 0 13 4 4 5 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
51) 9 9 0 11 5 0 0 0 3 rs=ad:bc;t=c:abs;uv=bt:acr
52) 9 9 4 8 0 15 6 0 0 r=c:ab;st=cd:abr;uv=cr:at
53) 10 1 0 20 20 10 10 0 0 rs=cd:ab;t=(cr):ab;uv=cs:abt
54) 10 1 0 23 23 7 7 0 0 rs=cd:ab;t=(cr):ab;uv=cs:abt
55) 10 2 0 23 16 3 6 0 0 rs=cd:ab;tu=cr:ab;v=a:bcstu
56) 10 5 0 11 7 4 5 0 1 rs=cd:ab;t=b:acs;uv=at:bcr
57) 10 6 0 11 10 5 0 0 0 rs=cd:ab;t=a:bcs;uv=bt:acr
58) 10 6 0 12 9 0 5 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
59) 10 6 0 13 6 3 9 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
60) 10 7 0 14 5 4 8 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
61) 10 10 0 11 6 0 0 0 5 rs=ad:bc;t=c:abs;uv=bt:acr
62) 10 12 0 9 6 0 0 0 5 rs=ad:bc;t=c:abs;uv=bt:acr
63) 11 6 1 12 12 4 0 0 0 rs=cd:ab;t=a:bcs;uv=bt:acr
64) 11 6 1 14 10 4 0 0 0 rs=cd:ab;t=a:bcs;uv=bt:acr
65) 11 7 0 12 11 0 4 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
66) 11 7 0 12 11 4 0 0 0 rs=cd:ab;t=a:bcs;uv=bt:acr
67) 11 7 0 13 6 5 7 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
68) 11 7 0 16 6 4 10 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
69) 11 7 0 27 5 10 10 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
70) 11 8 0 14 5 6 7 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
71) 11 9 0 12 7 0 0 13 0 rs=bd:ac;t=c:abs;u=b:acrst;
v=b:acrstu
72) 12 7 0 14 7 5 11 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
73) 12 8 0 31 5 13 10 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
74) 12 8 0 31 5 12 12 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
75) 12 9 0 29 4 18 16 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
76) 12 9 0 29 4 14 23 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
77) 12 9 0 29 4 17 17 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
78) 12 9 0 34 4 18 11 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
79) 12 9 0 34 4 17 12 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
80) 12 9 0 34 4 14 18 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
81) 12 9 0 36 4 18 10 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
82) 12 9 0 36 4 14 17 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
83) 12 10 5 8 0 21 9 0 0 rs=cd:ab;t=(cr):ab;uv=rt:ad
84) 12 12 9 12 0 15 5 0 0 rs=cd:ab;t=(cr):ab;uv=rt:ad
85) 12 13 0 9 9 0 0 17 0 rs=bd:ac;t=c:abs;u=c:abrt;
v=c:abrtu
86) 13 6 4 50 11 9 0 0 0 rs=cd:ab;tu=cr:ab;v=a:bcstu
87) 13 6 4 53 11 6 0 0 0 rs=cd:ab;tu=cr:ab;v=a:bcstu
88) 13 8 0 15 7 7 9 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
89) 13 8 0 16 7 7 8 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
90) 13 8 0 18 7 6 10 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
91) 13 8 0 32 6 12 14 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
92) 13 9 0 37 5 16 10 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
93) 13 9 0 37 5 14 14 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
94) 14 8 0 16 8 7 11 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
95) 14 8 1 15 15 5 0 0 0 rs=cd:ab;t=a:bcs;uv=bt:acr
96) 14 8 1 17 13 5 0 0 0 rs=cd:ab;t=a:bcs;uv=bt:acr
97) 14 9 0 17 8 5 8 0 1 rs=cd:ab;t=b:acs;uv=at:bcr
98) 14 9 0 34 6 14 16 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
99) 14 10 0 17 7 6 9 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
100) 14 10 0 18 7 6 7 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
101) 14 10 0 43 5 19 11 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
102) 14 10 0 43 5 16 17 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
103) 14 16 7 14 0 22 8 0 0 r=c:ab;st=cd:abr;uv=cr:at
104) 15 9 0 18 14 0 6 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
105) 15 10 0 41 6 21 11 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
106) 15 10 0 42 6 24 10 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
107) 15 11 0 17 9 3 10 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
108) 15 11 0 20 7 7 10 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
109) 16 10 0 18 9 7 11 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
110) 16 10 0 20 9 6 14 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
111) 16 11 0 18 10 4 9 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
112) 16 11 0 38 6 18 27 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
113) 16 11 0 44 6 18 21 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
114) 16 11 0 44 6 21 15 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
115) 16 11 0 46 6 18 20 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
116) 16 11 0 46 6 21 14 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
117) 16 11 0 47 6 24 12 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
118) 16 12 0 24 7 8 11 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
119) 16 19 9 16 0 24 8 0 0 r=c:ab;st=cd:abr;uv=cr:at
120) 17 10 0 23 10 6 14 0 1 rs=cd:ab;t=b:acs;uv=at:bcr
121) 17 10 0 23 10 6 15 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
122) 17 11 0 20 9 8 12 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
123) 17 12 0 26 8 8 13 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
124) 17 12 0 42 6 20 31 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
125) 17 12 0 42 6 24 23 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
126) 17 12 0 42 6 25 22 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
127) 17 12 0 48 6 20 25 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
128) 17 12 0 48 6 24 17 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
129) 17 12 0 48 6 25 16 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
130) 17 12 0 52 6 25 14 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
131) 17 12 0 52 6 24 15 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
132) 17 12 0 52 6 20 23 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
133) 18 6 1 47 27 3 0 0 0 rs=cd:ab;tu=cr:ab;v=a:bcstu
134) 18 8 5 66 16 12 0 0 0 rs=cd:ab;tu=cr:ab;v=a:bcstu
135) 18 8 5 70 16 8 0 0 0 rs=cd:ab;tu=cr:ab;v=a:bcstu
136) 18 9 3 22 19 5 0 0 0 rs=cd:ab;t=a:bcs;uv=bt:acr
137) 18 9 3 23 18 5 0 0 0 rs=cd:ab;t=a:bcs;uv=bt:acr
138) 18 9 4 24 17 5 0 0 0 rs=cd:ab;t=a:bcs;uv=bt:acr
139) 18 10 0 21 18 0 8 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
140) 18 10 0 21 18 8 0 0 0 rs=cd:ab;t=a:bcs;uv=bt:acr
141) 18 11 0 22 10 8 14 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
142) 18 12 0 26 9 8 17 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
143) 18 12 0 30 9 8 15 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
144) 18 18 0 21 10 0 0 0 8 rs=ad:bc;t=c:abs;uv=bt:acr
145) 18 21 0 18 10 0 0 0 8 rs=ad:bc;t=c:abs;uv=bt:acr
146) 19 11 0 21 20 0 7 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
147) 19 11 0 22 19 7 0 0 0 rs=cd:ab;t=a:bcs;uv=bt:acr
148) 19 11 0 23 11 9 12 0 1 rs=cd:ab;t=b:acs;uv=at:bcr
149) 19 12 0 22 10 10 14 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
150) 19 12 0 25 10 9 15 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
151) 19 13 0 45 7 29 23 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
152) 19 13 0 49 7 29 19 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
153) 19 13 0 52 7 30 17 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
154) 19 13 0 56 7 30 15 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
155) 20 4 0 51 34 5 8 0 0 rs=cd:ab;tu=cr:ab;v=a:bcstu
156) 20 12 1 22 20 0 8 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
157) 20 12 1 23 19 8 0 0 0 rs=cd:ab;t=a:bcs;uv=bt:acr
158) 20 13 0 24 11 8 13 0 1 rs=cd:ab;t=b:acs;uv=at:bcr

159) 20 13 0 25 11 8 11 0 1 rs=cd:ab;t=b:acs;uv=at:bcr
160) 20 16 0 18 17 2 7 0 0 rs=cd:ab;t=b:acs;u=b:acst;
v=b:acstu
161) 21 11 2 24 23 0 7 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
162) 21 12 0 24 12 11 15 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
163) 21 12 0 25 12 10 19 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
164) 21 12 0 26 12 10 16 0 1 rs=cd:ab;t=b:acs;uv=at:bcr
165) 21 12 0 26 12 10 17 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
166) 21 13 0 29 12 7 18 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
167) 21 14 0 26 11 9 15 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
168) 21 14 0 48 8 34 24 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
169) 21 14 0 50 8 34 22 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
170) 21 14 0 56 8 36 18 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
171) 21 14 0 60 8 36 16 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
172) 22 4 0 53 38 7 10 0 0 rs=cd:ab;tu=cr:ab;v=a:bcstu
173) 22 11 0 24 14 11 17 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
174) 22 13 0 24 13 9 19 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
175) 22 13 0 27 13 8 19 0 1 rs=cd:ab;t=b:acs;uv=at:bcr
176) 22 14 0 25 12 10 16 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
177) 22 15 0 28 11 11 12 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
178) 22 23 0 19 14 0 0 27 0 rs=bd:ac;t=c:abs;u=c:abrt;
v=c:abrtu
179) 22 26 9 22 0 39 15 0 0 r=c:ab;st=cd:abr;uv=cr:at
180) 23 13 0 25 14 10 18 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
181) 23 13 0 27 23 0 9 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
182) 23 13 0 30 14 8 22 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
183) 23 13 0 30 14 8 20 0 2 rs=cd:ab;t=b:acs;uv=at:bcr
184) 23 14 0 27 13 11 14 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
185) 23 15 0 57 10 23 20 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
186) 23 15 0 57 10 22 22 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
187) 23 17 0 27 14 4 13 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
188) 23 17 0 71 8 34 17 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
189) 23 24 0 20 15 0 0 25 0 rs=bd:ac;t=c:abs;u=c:abrt;
v=c:abrtu
190) 24 13 3 28 25 7 0 0 0 rs=cd:ab;t=a:bcs;uv=bt:acr
191) 24 13 3 29 24 7 0 0 0 rs=cd:ab;t=a:bcs;uv=bt:acr
192) 24 14 1 28 23 0 10 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
193) 24 16 0 31 12 12 17 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
194) 24 19 0 77 8 36 17 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
195) 24 27 15 24 0 33 11 0 0 r=c:ab;st=cd:abr;uv=cr:at
196) 25 14 1 29 25 0 10 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
197) 25 16 0 36 14 8 21 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
198) 26 15 0 31 15 12 19 0 1 rs=cd:ab;t=b:acs;uv=at:bcr
199) 26 16 0 31 14 13 21 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
200) 26 18 0 32 13 12 17 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
201) 26 19 0 61 9 40 33 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
202) 26 19 0 80 9 39 20 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
203) 27 15 0 32 17 10 18 0 2 rs=cd:ab;t=b:acs;uv=at:bcr
204) 27 15 0 32 16 12 26 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
205) 27 16 0 33 15 13 22 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
206) 27 24 15 24 0 39 16 0 0 r=c:ab;st=cd:abr;uv=cr:at
207) 27 29 0 23 17 0 0 31 0 rs=bd:ac;t=c:abs;u=c:abrt;
v=c:abrtu
208) 28 16 1 31 29 0 11 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
209) 28 16 1 33 27 11 0 0 0 rs=cd:ab;t=a:bcs;uv=bt:acr
210) 28 19 0 35 14 14 17 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
211) 28 20 0 38 13 14 21 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
212) 28 32 13 28 0 45 17 0 0 r=c:ab;st=cd:abr;uv=cr:at
213) 29 19 0 70 12 30 36 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
214) 30 8 0 81 48 3 6 0 0 rs=cd:ab;tu=cr:ab;v=a:bcstu
215) 30 23 0 91 10 44 23 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
216) 31 16 0 36 19 16 24 0 1 rs=cd:ab;t=b:acs;uv=at:bcr
217) 31 17 0 34 19 14 24 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
218) 31 17 0 37 20 11 19 0 3 rs=cd:ab;t=b:acs;uv=at:bcr
219) 31 18 0 38 18 14 20 0 2 rs=cd:ab;t=b:acs;uv=at:bcr
220) 31 24 0 97 11 53 19 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
221) 32 18 1 37 32 0 13 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
222) 32 21 0 39 17 14 21 0 1 rs=cd:ab;t=b:acs;uv=at:bcr
223) 32 22 0 70 12 55 36 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
224) 33 21 0 43 18 13 28 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
225) 33 25 0 97 11 46 29 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
226) 33 25 0 97 11 47 28 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
227) 33 25 0 99 11 47 27 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
228) 33 25 0 99 11 46 28 0 0 rs=cd:ab;tu=cs:ab;v=a:bcrtu
229) 34 20 0 44 19 16 25 0 2 rs=cd:ab;t=b:acs;uv=at:bcr
230) 34 20 0 50 19 16 25 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
231) 34 21 0 42 19 15 20 0 2 rs=cd:ab;t=b:acs;uv=at:bcr
232) 35 22 0 45 19 15 28 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
233) 35 37 0 30 23 0 0 37 0 rs=bd:ac;t=c:abs;u=c:abrt;
v=c:abrtu
234) 36 21 0 47 21 14 29 0 3 rs=cd:ab;t=b:acs;uv=at:bcr
235) 36 26 0 42 21 9 21 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
236) 38 23 0 44 21 19 26 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
237) 38 23 0 45 23 12 29 0 3 rs=cd:ab;t=b:acs;uv=at:bcr
238) 38 24 0 46 21 18 21 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
239) 40 21 0 44 25 19 30 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
240) 40 23 0 47 25 14 27 0 3 rs=cd:ab;t=b:acs;uv=at:bcr
241) 40 23 0 51 23 18 29 0 3 rs=cd:ab;t=b:acs;uv=at:bcr
242) 40 24 0 47 22 20 31 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
243) 40 25 0 44 24 14 31 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
244) 40 25 0 49 22 18 25 0 2 rs=cd:ab;t=b:acs;uv=at:bcr
245) 40 31 0 40 32 3 15 0 0 rs=cd:ab;t=b:adr;u=b:adrt;
v=b:adrtu
246) 41 22 0 45 26 18 29 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
247) 42 21 8 32 29 14 18 0 0 rs=cd:ab;t=a:bcs;u=a:bcst;
v=a:bcstu
248) 42 45 21 42 0 63 25 0 0 r=c:ab;st=cd:abr;uv=cr:at
249) 42 45 24 42 0 60 22 0 0 r=c:ab;st=cd:abr;uv=cr:at
250) 45 31 0 43 37 8 12 0 0 rs=cd:ab;t=b:adr;u=b:adrt;
v=b:adrtu
251) 46 28 0 62 25 22 33 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
252) 48 27 0 57 31 16 29 0 5 rs=cd:ab;t=b:acs;uv=at:bcr
253) 54 48 27 48 0 81 35 0 0 r=c:ab;st=cd:abr;uv=cr:at
254) 55 34 0 65 30 27 36 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
255) 58 44 0 78 27 26 27 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
256) 61 36 0 71 34 31 45 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
257) 62 36 0 77 35 30 47 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
258) 66 44 0 86 33 34 41 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
259) 67 42 0 79 36 33 46 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
260) 70 35 12 54 49 22 32 0 0 rs=cd:ab;t=a:bcs;u=a:bcst;
v=a:bcstu
261) 72 39 0 85 47 26 43 0 5 rs=cd:ab;t=b:acs;uv=at:bcr
262) 74 44 0 91 41 36 57 0 0 rs=cd:ab;t=b:acs;uv=at:bcr
263) 80 55 5 70 62 14 29 0 0 rs=cd:ab;t=b:adr;u=b:adrt;
v=b:adrtu
264) 80 57 0 92 48 18 51 0 0 rs=cd:ab;t=b:acs;uv=at:bcr

References

[1] D. Avis, D. Bremner, and R. Seidel, *How good are convex hull algorithms?*, Comput. Geom. **7** (1997), no 5-6, pp. 265–301

[2] R. Baber, D. Christofides, A.N. Dang, S. Riis, and E.R. Vaughan, Multiple unicasts, graph guessing games, and non-Shannon inequalities, *Proc. NetCod 2013*, Calgary, pp. 1–6.

[3] R. Bassoli, H. Marques, J .Rodriguez, K.W. Shum and R. Tafazolli, Network Coding Theory: A Survey. *IEEE Communications Surveys & Tutorials*, vol 15, no 4 (2013), pp. 1950–1978.

[4] A. Beimel, Secret-Sharing Schemes: A Survey, *Coding and Cryptology*. LNCS vol. 6639 (2011) pp. 11–46.

[5] H. P. Benson, An outer approximation algorithm for generating all efficient extreme points in the outcome set of a multiple objective linear program, *J. Global Optim* vol 13 (1998), vol 1, pp. 1–24.

[6] D. Bremner. *On the complexity of vertex and facet enumeration for convex polytopes.* PhD thesis, School of Computer Science, McGill University, 1997

[7] B. A. Burton, M. Ozlen, *Projective geometry and the outer approximation algorithm for multiobjective linear programming*, ArXiv:1006.3085, June 2010.

[8] T. H. Chan, Recent progresses in characterising information inequalities, *Entropy* vol 13 (2011), pp. 379–401.

[9] T. H. Chan, Balanced information inequalities, *IEEE Trans. Inform. Theory* vol 49 (2003), pp. 3261–3267.

[10] L. Csirmaz, Book inequalities, *IEEE Trans. Inform. Theory*, vol 60 (2014) pp, 6811–6818.

[11] R. Dougherty, C. Freiling, K. Zeger, *Non-Shannon information inequalities in four random variables*, ArXiv:1104.3602, April 2011.

[12] M. Ehrgott, A. Löhne, L. Shao, A dual variant of Benson's outer approximation algorithm for multiple objective linear programming, *J. Glob. Optim* vol 52 (2012), pp. 757–778.

[13] K. Fukuda, A. Prodon, *Double description method revisited*, Combinatorics and Computer Science (Brest, 1995) *LNCS* vol 1120 (1996), Springer, Berlin, 1996, pp. 91-111.

[14] A. H. Hamel, A. Löhne, B. Rudloff, Benson type algorithms for linear vector optimization and applications, *J. Glob. Optim* vol 59 (2013) pp. 811–836.

[15] F. Heyde, A. Löhne, *Geometric duality in multiple objective linear programming*, *SIAM Journal on Optimization* vol 19(2), (2008), pp. 836–845.

[16] T. Kaced, Equivalence of two proof techniques for non-Shannon-type inequalities, In: *Proceedings of the 2013 IEEE International Symposium on Information Theory*, Istambul, 2013; pp. 236–240.

[17] M. Madiman, A.W. Marcus and P. Tetali, Information-theoretic inequalities in additive combinatorics, *IEEE ITW 2010* pp. 1–4.

[18] K. Makarychev, Yu. Makarychev, A. Romashchenko and N. Vereshchagin, A new class of non-Shannon-type inequalities for entropies, *Communications in Information and Systems* vol 2(2) (2002), pp. 147–166.

[19] F. Matus, Infinitely many information inequalities, *Proceedings ISIT*, June 24–29, 2007, Nice, France, pp. 41–47.

[20] F. Matus, Two constructions on limits of entropy functions, *IEEE Trans. Inform. Theory*, vol 53(1) (2007) pp. 320–330.

[21] F. Matus, Personal communication, 2012.

[22] F. Matus and M. Studeny, Conditional independencies among four random variables I, *Combinatorics, Probability and Computing*, no 4, (1995) pp. 269-278.

[23] W. B. McRae, E. R. Davidson, An algorithm for the extreme rays of a pointed convex polyhedral cone, *SIAM J. Comput.*, vol 2(4) (1973) pp. 281–293.

[24] N. Pippenger, *What are the laws of information theory*, 1986 Special Problems on Communication and Computation Conference, Palo Alt, California, Sept 3-5 1986.

[25] M. Studený, *Probabilistic Conditional Independence Structures*, Springer, New York. (2005)

[26] R.W. Yeung (2002) *A First Course in Information Theory*, Kluwer Academic/Plenum Publishers, New York.

[27] J. MacLaren Walsh, S. Weber, Relationships among bounds for the region of entropic vectors in four variables, in *2010 Allerton Conference on Communication, Control, and Computing*, (2010)

[28] Z. Zhang, R. W. Yeung, On characterization of entropy function via information inequalities, *Proc IEEE Trans. on Inform. Theory*, vol 44(4) (1998) pp. 1440–1452.