The Indecomposable Solutions of Linear Diophantine Equations

Klaus Pommerening
Johannes-Gutenberg-Universität
Mainz, Germany

Version 5, December 2017
last change: December 27, 2017

The subject of this article is the linear Diophantine equation

(E) $$a_1x_1 + \cdots + a_nx_n = 0$$

whose coefficients a_i are integers, $a = (a_1, \ldots, a_n) \in \mathbb{Z}^n$, and whose solutions should be non-negative integers, $x = (x_1, \ldots, x_n) \in \mathbb{N}^n$.

> Note that in this article $\mathbb{N}$ stands for the numbers $\{0, 1, 2, \ldots\}$, and $\mathbb{N}_k$ for $\{k, k+1, \ldots\}$. Think of 0 as being the most natural number.

The solutions form a subsemigroup of $\mathbb{N}^n$ that is finitely generated and has a uniquely determined system of generators consisting of the indecomposable (or minimal non-zero) solutions [18], see also Remark 2 after Theorem 1 below. Thus the general goal is the determination of all indecomposable solutions, with the partial tasks:

(I) *Find bounds for the coordinates of the indecomposable solutions that are as strong as possible.*
(II) *Find an algorithm that constructs all indecomposable solutions and is as efficient as possible.*
(III) *Determine the number of indecomposable solutions, at least give good estimates of this number.*

The article [21] treats the linear congruence in an analogous way.

For applications in combinatorics see [25], [26],[4], and [10], for applications to non-unique factorization in commutative algebra see [9], for applications in logic see [27], for applications in invariant theory see [20]. There are even applications in chemistry [16], electrical engineering [23], and computer science [1].

Notation

We equip $\mathbb{N}^n$ with the (partial) order relation

$$x \leq y \Leftrightarrow y - x \in \mathbb{N}^n$$

and use the norms

$$\|x\|_1 = x_1 + \cdots + x_n, \quad \|x\|_\infty = \max\{|x_1|, \ldots, |x_n|\},$$

for a vector $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$, the notation

$$\operatorname{supp}(x) := \{i = 1, \ldots, n \mid x_i \neq 0\}$$

for its support, and

$$\sigma(x) := \#\operatorname{supp}(x).$$

for the cardinality of its support, called the **width** of x. We abbreviate

$$\lambda(x) := x_1 + \cdots + n \cdot x_n$$

and call it the **weight** of x. Moreover we call

- $\|x\|_1 = |x_1| + \cdots + |x_n|$ the **length** (sometimes [12] also called the degree—this is also the length of the associated multiset [24]),
- $\|x\|_\infty$ the **height**,
- $\|x\|_1 + \sigma(x)$ the **total size** (= length + width)

of x. Clearly in $\mathbb{N}$, for a vector $x \in \mathbb{N}^n$,

$$\sigma(x) = \sum_{x_i \neq 0} 1 \leq \sum_{x_i \neq 0} x_i = \|x\|_1 \leq \sum_{x_i \neq 0} i \cdot x_i = \lambda(x).$$

We denote the canonical unit vectors in $\mathbb{R}^n$ by $e_1, \ldots, e_n$, that is $e_i = (\delta_{ij})_{1 \leq j \leq n}$, or $e_1 = (1, 0, \ldots, 0)$, $e_2 = (0, 1, 0, \ldots, 0)$, ...

Trivial Cases

The case $n = 1$: If the coefficient $a \in \mathbb{Z}$ is zero, the solution set is $\mathbb{N}$, and the unique indecomposable solution is 1. If $a \neq 0$, the solution set consists of 0 only, and the set of indecomposable solutions is empty.

More generally:

Proposition 1 *Let $a \in \mathbb{Z}^n$ and $I_0 = \{i = 1, \ldots, n \mid a_i = 0\}$. Then for* **(E)***:*

(i) *If $I_0 = \{i = 1, \ldots, n\}$, then the indecomposable solutions are the n unit vectors $e_1, \ldots, e_n$.*

(ii) *If* $\#I_0 = n-1$*, that is* $a_i \neq 0$ *for exactly one index* i*, then the indecomposable solutions are the* $n-1$ *unit vectors* e_j *for* $j \neq i$*.*

(iii) *If* $I_0 = \emptyset$ *and all the coefficients* a_i *have the same sign* ($+$ *or* $-$)*, then* 0 *is the only solution in* $\mathbb{N}^n$*.*

(iv) *Every solution* $x \in \mathbb{N}^n$ *directly decomposes into two parts:*

$$(x_i)_{i\in I_0} \in \mathbb{N}^{\#I_0} \quad \textit{arbitrary}$$

and a solution of the remaining Diophantine equation

$$\sum_{i\notin I_0} a_i x_i = 0\,.$$

Proof. Trivial. $\diamond$

Because of (iii) we usually assume that some coefficients in **(E)** are positive and some are negative (and maybe some are zero). Because of (iv) we often assume that all coefficients a_i are non-zero.

The next elementary case to consider is $n = 2$. Here is the result:

Proposition 2 *Let* $a, b \in \mathbb{Z}$*,* $a, b \neq 0$*. Assume* $a > 0$ *and* $b < 0$*. Then the only indecomposable solution of the equation* $ax + by = 0$ *is* $(-\frac{b}{d}, \frac{a}{d})$ *where* $d = \gcd(a, b)$*.*

Proof. Clearly this is a solution. On the other hand if (x, y) is a solution, then using that $\frac{a}{d}$ is coprime with b we conclude that $\frac{a}{d} \mid y$, and likewise $\frac{b}{d} \mid x$, hence $(x, y) \geq (-\frac{b}{d}, \frac{a}{d})$. $\diamond$

1 Outline of a Naive Algorithm

Let $n \in \mathbb{N}_1$, $a = (a_1, \ldots, a_n) \in \mathbb{Z}^n$. In the literature we find several algorithms for constructing all indecomposable solutions $x \in \mathbb{N}^n$ of the linear equation **(E)**, see for example [6, 14, 3, 5, 22, 7, 17, 16, 2]. An obvious algorithm is:

1. Given a finite subset $\mathcal{D} \subseteq \mathbb{N}^n$ that is guaranteed to contain all indecomposable solutions, enumerate all vectors > 0 in $\mathcal{D}$.

2. Test each vector whether it satisfies **(E)** to get the list of all solutions > 0 in $\mathcal{D}$.

3. Eliminate all vectors from the list that are not minimal.

The first goal is to find a suitable set $\mathcal{D}$, see Section 2. Note that a description of $\mathcal{D}$ by inequalities makes the solution of **(E)** accessible to the methods of integer linear programming. Unfortunately linear programming algorithms are not good at finding *all* indecomposable solutions.

A first candidate subset $\mathcal{D} \subseteq \mathbb{N}^n$ is provided by Huet's bound [14] that implies that $\|x\|_\infty \leq M := \|a\|_\infty$ for each indecomposable solution of **(E)**:

$$\mathcal{D} = \{x \in \mathbb{N}^n \mid \|x\|_\infty \leq M\}.$$

For a proof see Corollary 1 of Theorem 1 below. For this subset $\mathcal{D}$ step 1 of the naive algorithms consists of enumerating all integer vectors x in the hypercube $[0, M]^n \subseteq \mathbb{R}^n$. This is done by the Python routine `dlist0` in Appendix A.1. The algorithm is in Appendix A.2. It is called in the form

```
solve_E_0.py <coefficients>
```

For the example:

$$2x_1 - 3x_2 + x_3 + 0x_4 + 4x_5 - 2x_6 = 0 \quad (\text{with } n = 6,\ M = 4)$$

the call

```
solve_E_0.py 2 -3 1 0 4 -2
```

yields the result

```
[0,0,0,0,1,2], [0,0,0,1,0,0], [0,0,2,0,0,1], [0,1,1,0,1,1],
[0,1,3,0,0,0], [0,2,0,0,2,1], [0,2,2,0,1,0], [0,3,1,0,2,0],
[0,4,0,0,3,0], [1,0,0,0,0,1], [1,1,1,0,0,0], [1,2,0,0,1,0],
[3,2,0,0,0,0]
```

This naive algorithm is extremely inefficient. A call with $n = 10$, $M = 5$ already requires an excessive amount of computing power. The obvious approach to get a faster algorithm is to search for tighter bounds for the indecomposable solutions.

2 Bounds for Indecomposable Solutions

A result similar to Tinsley's for the linear congruence, see [21], is Lambert's bound for the indecomposable solutions of the linear Diophantine equation **(E)**. We prove a stronger version, together with a different bound by Sissokho [24], following the approach of [24].

The linear Diophantine equation **(E)** is completely specified by the coefficient vector $a = (a_1, \ldots, a_n) \in \mathbb{Z}^n$ with $n \geq 1$. For this we define a setting consisting of

$$P = \{i \mid a_i > 0\}, \quad N = \{i \mid a_i < 0\}, \quad p = \#P, \quad r = \#N,$$

$$A_\infty = \max\{a_i \mid i \in P\}, \quad B_\infty = \max\{-a_i \mid i \in N\}, \quad M = \max\{A_\infty, B_\infty\} = \|a\|_\infty.$$

Depending on a we define for a vector $x \in \mathbb{N}^n$

$$x^+ = (x_i)_{i \in P} \in \mathbb{N}^p, \quad x^- = (x_i)_{i \in N} \in \mathbb{N}^r,$$

$$\operatorname{supp}^+(x) = \{a_i \mid i \in P,\ x_i \neq 0\}, \quad \operatorname{supp}^-(x) = \{-a_i \mid i \in N,\ x_i \neq 0\},$$

$$\sigma^+(x) = \#\operatorname{supp}^+(x), \quad \sigma^-(x) = \#\operatorname{supp}^-(x), \quad \alpha(x) = \sum_{i\in P} a_i x_i.$$

In the example above we have $a = (2, -3, 1, 0, 4, -2)$, hence

$$P = \{1, 3, 5\}, \quad N = \{2, 6\}, \quad p = 3, \quad r = 2,$$

$$A_\infty = 4, \quad B_\infty = 3, \quad M = 4,$$

$$x^+ = (x_1, x_3, x_5), \quad x^- = (x_2, x_6).$$

For the indecomposable solution $x = (0, 3, 1, 0, 2, 0)$ we have

$$\operatorname{supp}^+(x) = \{a_3, a_5\} = \{1, 4\}, \quad \sigma^+(x) = 2,$$

$$\operatorname{supp}^-(x) = \{a_2\} = \{3\}, \quad \sigma^-(x) = 1,$$

$$\|x^+\|_1 = x_1 + x_3 + x_5 = 3, \quad \|x^-\|_1 = x_2 + x_6 = 3,$$

$$\|x^+\|_1 \cdot \|x^-\|_1 = 9, \quad \alpha(x) = a_1x_1 + a_3x_3 + a_5x_5 = 0 + 1 + 8 = 9.$$

All indecomposable solutions have $\|x^+\|_1 \le 3$. Only the solution $x = (0, 4, 0, 0, 3, 0)$ has $\|x^-\|_1 = 4$, and all the other indecomposable ones have $\|x^-\|_1 \le 3$.

This example illustrates the following theorem:

Theorem 1 *Let $a \in \mathbb{Z}^n$ with $n \ge 1$, and assume that $p \ge 1$ and $r \ge 1$, thus there are positive and negative coefficients in* **(E)**. *Then for all indecomposable solutions $x \in \mathbb{N}^n$ of* **(E)**:

(i) (Lambert)
$$\|x^+\|_1 \le B_\infty \quad \textit{and} \quad \|x^-\|_1 \le A_\infty.$$

(ii) *If $\|x^+\|_1 = B_\infty$, then $\operatorname{supp}^-(x) = \{B_\infty\}$, in particular $\sigma^-(x) = 1$. If $\|x^-\|_1 = A_\infty$, then $\operatorname{supp}^+(x) = \{A_\infty\}$, in particular $\sigma^+(x) = 1$.*

(iii) (Sissokho)
$$\|x^+\|_1 \cdot \|x^-\|_1 \le \alpha(x).$$

Proof. A permutation of the indices doesn't affect the statements of the theorem. Thus we may assume that

$$P = \{1, \ldots, p\} \quad \text{and} \quad N = \{p+1, \ldots, m\}$$

where $m = p + r$. Then $x^+ = (x_1, \ldots, x_p) \in \mathbb{N}^p$, $x^- = (x_{p+1}, \ldots, x_m) \in \mathbb{N}^r$, and $\|x\|_1 = \|x^+\|_1 + \|x^-\|_1 + x_{m+1} + \cdots + x_n$. We may even assume that

$$1 \le a_1 \le \ldots \le a_p \quad \text{and} \quad 1 \le b_1 := -a_{p+1} \le \ldots \le b_r := -a_m,$$

and then $A_\infty = a_p$, $B_\infty = b_r$. The equation **(E)** boils down to

$$a_1x_1 + \cdots + a_px_p = b_1x_{p+1} + \cdots + b_rx_m.$$

We prove all three statements (i), (ii), and (iii) together by induction on $\|x\|_1$ for all coefficient vectors $a \in \mathbb{N}^s$ (with any s) at the same time, and for minimal solutions x.

If $\|x\|_1 = 1$, then x is a unit vector, $x = e_i$ for some index i, and necessarily x_i must have coefficient $a_i = 0$ in **(E)**. Hence $i > m$, $x^+ = 0$, $x^- = 0$, $\sigma^+(x) = \sigma^-(x) = 0$, and assertions (i) and (iii) are trivial. The preconditions in (ii), $\|x^+\|_1 = B_\infty \geq 1$ or $\|x^-\|_1 = A_\infty \geq 1$, are false, hence (ii) is true.

Now we assume that $\|x\|_1 \geq 2$. If x has coefficients $x_i \neq 0$ only for $i > m$, then again there is nothing to prove. Hence we may assume that $x_i \neq 0$ for at least one index $i \in P$, and then necessarily also for at least one index $i \in N$ (or vice versa). Hence both $\|x^+\|_1, \|x^-\|_1 \geq 1$, and necessarily $x_i = 0$ for $i > m$ by the minimality of x.

In the special case of a pair $(i, j) \in P \times N$ of indices with $a_i = -a_j$ and $x_i > 0$, $x_j > 0$ we have the non-zero solution $e_i + e_j \leq x$, hence $= x$, and $\|x^+\|_1 = \|x^-\|_1 = 1$, $\sigma^+(x) = \sigma^-(x) = 1$, thus (i) and (iii) are trivial. The first condition in (ii), $\|x^+\|_1 = B_\infty$, implies $B_\infty = 1$, hence $b_1 = \ldots = b_r = 1$, hence $\mathrm{supp}^-(x) = \{1\} = \{B_\infty\}$, and the first statement of (ii) is true. The analogous reasoning holds for the second statement, so also (ii) is proved in this special case.

Otherwise we have

$$\{a_1, \ldots, a_p\} \cap \{b_1, \ldots, b_r\} = \emptyset.$$

We may assume (without loss of generality) that $x_p, x_m > 0$ and $a_p > b_r$. We consider the modified equation

(E') $$(a_p - b_r)\, u_0 + a_1 u_1 + \cdots + a_p u_p = b_1 v_1 + \cdots + b_r v_r$$

that has the solution $x' = (1, x_1, \ldots, x_{p-1}, x_p - 1, x_{p+1}, \ldots, x_{m-1}, x_m - 1)$, and this solution is minimal > 0:

> Suppose that we have a solution $y \leq x'$ of **(E')**. Then y has the form
>
> $$y = (u_0, u_1, \ldots, u_p, v_1, \ldots, v_r)$$
>
> where $u_0 \leq 1$. The assumption $u_0 = 0$ makes $(u_1, \ldots, u_p, v_1, \ldots, v_r, 0, \ldots, 0)$ a solution of **(E)** and $< x$, hence $= 0$, thus $y = 0$. If $u_0 = 1$, then
>
> $$(a_p - b_r) + a_1 u_1 + \cdots + a_p u_p = b_1 v_1 + \cdots + b_r v_r,$$
>
> $$a_1 u_1 + \cdots + a_p (u_p + 1) = b_1 v_1 + \cdots + b_r (v_r + 1),$$
>
> and from $y \leq x'$ we conclude that $u_i \leq x_i$ for $1 \leq i < p$, $u_p \leq x_p - 1$ or $u_p + 1 \leq x_p$, and $v_i \leq x_{p+i}$ for $1 \leq i < r$, $v_r \leq x_m - 1$ or $v_r + 1 \leq x_m$. The minimality of x enforces that all these inequalities are equalities, and therefore $y = x'$.

Now $\|x'\|_1 = \|x\|_1 - 1$, and by induction the statements (i)–(iii) hold for x' and **(E')**. Moreover $\|x'^+\|_1 = \|x^+\|_1$ and $\|x'^-\|_1 = \|x^-\|_1 - 1$. Hence by (i) and (iii)

(1) $$\|x^+\|_1 \leq b_r, \quad \|x^-\|_1 - 1 \leq a_p,$$

(2) $\|x^+\|_1 \cdot (\|x^-\|_1 - 1) \leq (a_p - b_r) + a_1x_1 + \cdots + a_{p-1}x_{p-1} + a_p(x_p - 1) = \alpha(x) - b_r$.

From Formula (2) and $b_r\|x^-\|_1 \geq b_1x_{p+1} + \cdots + b_rx_m = \alpha(x)$ we get

$$\begin{aligned} \alpha(x) \cdot \|x^-\|_1 - b_r\|x^-\|_1 &\leq \alpha(x) \cdot \|x^-\|_1 - \alpha(x), \\ (\alpha(x) - b_r)\|x^-\|_1 &\leq \alpha(x) \cdot (\|x^-\|_1 - 1), \\ \|x^+\|_1 \cdot (\|x^-\|_1 - 1) \cdot \|x^-\|_1 &\leq \alpha(x) \cdot (\|x^-\|_1 - 1). \end{aligned}$$

If $\|x^-\|_1 = 1$, then $\|x^+\|_1 \cdot \|x^-\|_1 = \|x^+\|_1 \leq \alpha(x)$, so we are done with (iii). Otherwise dividing by $\|x^-\|_1 - 1$ yields (iii).

In Formula (1) we might have $\|x^-\|_1 - 1 = a_p$. Statement (ii) for **(E')** implies $\mathrm{supp}^+(x') = \{a_p\}$, thus $x_i \neq 0$ for $i \in P$ only if $a_i = a_p$, and necessarily the additional coefficient $a_p - b_r$ (with x'-coordinate 1) must be 0, contradicting $a_p > b_r$. Hence $\|x^-\|_1 - 1 \leq a_p - 1$, and the proof of (i) is complete.

For (ii) first assume that $\|x^+\|_1 = B_\infty = b_r$. Then

$$b_rx_{p+1} + \cdots + b_rx_m = b_r \cdot \|x^-\|_1 = \|x^+\|_1 \cdot \|x^-\|_1 \leq \alpha(x) = b_1x_{p+1} + \cdots + b_rx_m.$$

Hence $x_i \neq 0$ for $i \in N$ only if $b_i = b_r$. Since we didn't use the inequality $b_r < a_p$ the analogous reasoning shows that $\|x^-\|_1 = a_p$ implies that $x_i \neq 0$ for $i \in P$ only if $a_i = a_p$. $\diamond$

In particular the linear Diophantine equation **(E)** has only finitely many indecomposable solutions. A coarser bound is:

Corollary 1 (Huet) *For each indecomposable solution x of* **(E)** *we have* $\|x\|_\infty \leq \|a\|_\infty$.

Proof. By the theorem each single coordinate x_i is bounded by A_∞ or by B_∞, hence by $\|a\|_\infty$. $\diamond$

The number of vectors $x \in \mathbb{N}^r$ with $\|x\|_1 \leq q$ equals the number of ways to put up to q balls into r (distinguishable) urns, or the number of ways to put exactly q balls into $r + 1$ urns, that is

$$\binom{q+r}{r}.$$

Thus Lambert's bound gives a coarse upper bound for the number of indecomposable solutions:

$$\begin{aligned} &\binom{B_\infty + p}{p} \text{ possibilities for the partial vector } (x_i)_{i\in P}, \\ &\binom{A_\infty + r}{r} \text{ possibilities for the partial vector } (x_i)_{i\in N}, \end{aligned}$$

plus the $n - p - r$ solutions e_i for $i \in \{1, \ldots, n\} - P - N$. The total is:

Corollary 2 *The number of indecomposable solutions of* **(E)** *is at most*

$$\binom{B_\infty + p}{p} + \binom{A_\infty + r}{r} + n - p - r.$$

Clearly each indecomposable solution x has $\|x\|_1 \leq A_\infty + B_\infty$. Thus we get the coarser bound:

Corollary 3 *Let $C = A_\infty + B_\infty$. Then the number of indecomposable solutions of* **(E)** *is at most*

$$\binom{n + C}{n} = \frac{1}{n!}\, C^n + \mathrm{O}(C^{n-1}).$$

Remarks

1. The usual measure of the size of an integer is bitlength $\ell(C) = 1 + \lfloor \log_2 C \rfloor$. Thus the bound in the corollary is exponential in $\ell(C)$ as well as in n.

2. The qualitative statement of the theorem—the number of indecomposable solutions is finite—didn't use Dickson's lemma [18]. With the weaker bound $\|x\|_\infty < n \cdot \|a\|_\infty$ the result was known already in the 19^{th} century, see [8, Chap. VI, No. 97] or [6].

3 More Efficient Algorithms

Using the bound from Theorem 1 (i) we speed up the naive algorithm from Section 1. We find all indecomposable solutions of **(E)** by the following steps:

1. Set $P = \{i \mid a_i > 0\}$, $N = \{i \mid a_i < 0\}$, $Z = \{i \mid a_i = 0\}$, $p = \#P$, $r = \#N$.

2. Set $A = \max\{a_i \mid i \in P\}$, $B = \max\{-a_j \mid j \in N\}$.

3. Initialize the list `sollist` of solutions by listing the trivial solutions e_i with $i \in Z$.

4. Construct lists

   (a) `xlist` of all vectors $x \in \mathbb{N}^p - \{0\}$ with $x_1 + \cdots + x_p \leq B$,

   (b) `ylist` of all vectors $y \in \mathbb{N}^r - \{0\}$ with $y_1 + \cdots + y_r \leq A$.

5. Construct a candidate list `dlist` by combining each vector in `xlist` with each vector in `ylist`.

6. For each x in `dlist` check **(E)**. In the positive case append x to `sollist`.

7. Reduce `sollist` to its minimal elements.

For Python code see Appendix A.3. It uses the auxiliary routines `dlist1` that lists the integer elements in a simplex, `smaller` that compares two vectors, and `minelts` that eliminates all non-minimal elements from a list of vectors from Appendix A.1.

The program works fine for n and $C = A + B$ up to about 10, but leaves ample prospects for optimization. Here are two sample results:

- $0x_1 + 1x_2 + 2x_3 + 3x_4 + 0x_5 - 1x_6 - 2x_7 - 3x_8$ (with $n = 8$, $C = 6$). Called as

  ```
  solve_E_1.py 0 1 2 3 0 -1 -2 -3
  ```

  the result is the following 15 indecomposable solutions:

  ```
  [1,0,0,0,0,0,0,0], [0,0,0,0,1,0,0,0], [0,0,0,1,0,0,0,1],
  [0,0,0,1,0,1,1,0], [0,0,0,1,0,3,0,0], [0,0,0,2,0,0,3,0],
  [0,0,1,0,0,0,1,0], [0,0,1,0,0,2,0,0], [0,0,2,0,0,1,0,1],
  [0,0,3,0,0,0,0,2], [0,1,0,0,0,1,0,0], [0,1,0,1,0,0,2,0],
  [0,1,1,0,0,0,0,1], [0,2,0,0,0,0,1,0], [0,3,0,0,0,0,0,1]
  ```

- $2x_1 - 3x_2 + x_3 + 4x_4 - 2x_5$ (with $n = 5$, $C = 7$) yields the result

  ```
  [0,0,0,1,2], [0,0,2,0,1], [0,1,1,1,1], [0,1,3,0,0], [0,2,0,2,1],
  [0,2,2,1,0], [0,3,1,2,0], [0,4,0,3,0], [1,0,0,0,1], [1,1,1,0,0],
  [1,2,0,1,0], [3,2,0,0,0]
  ```

Sissokho's bound leads to a further improvement of the algorithm:

1. Set $P = \{i \mid a_i > 0\}$, $N = \{i \mid a_i < 0\}$, $Z = \{i \mid a_i = 0\}$, $p = \#I$, $r = \#J$.
2. Set $A = \max\{a_i \mid i \in P\}$, $B = \max\{-a_j \mid j \in N\}$.
3. Initialize the list `sollist` of solutions by listing the trivial solutions e_i with $i \in Z$.
4. Construct lists
   (a) `xlist` of all vectors $x \in \mathbb{N}^p - \{0\}$ with $\|x\|_1 = x_1 + \cdots + x_p \le B$.
   For each vector x in `xlist` calculate $A' = \lfloor \alpha(x)/\|x\|_1 \rfloor$.
   If $A' < A$, set $A(x) = A'$, otherwise set $A(x) = A$ and store $A(x)$.
   (b) `ylist` of all vectors $y \in \mathbb{N}^r - \{0\}$ with $\|y\|_1 = y_1 + \cdots + y_r \le A$.
   For each vector y in `ylist` store $\|y\|_1$.
5. Construct a candidate list `dlist` by combining each vector x in `xlist` with each vector y in `ylist` that satisfies $\|y\|_1 \le A(x)$.
6. For each z in `dlist` check **(E)**. In the positive case append z to `sollist`.
7. Reduce `sollist` to its minimal elements.

For Python code see Appendix A.4. The improvement allows to increment the parameters n, A, and B roughly by 1 and still have a tolerable runtime.

The following table compares runtimes of the algorithms from Sections 1 (column "naiv"), 3 using Lambert's bound (column "lamb"), and 3 using Sissokho's bound too (column "siss"), where $M = \max\{A, B\} = n$. The entries represent seconds on a typical desktop computer for a typical instance, where "0" means < 0.0005 or immediate output, "0.00" means < 0.005, and "—" means more than 10 minutes or "forget it".

M	naiv	lamb	siss
2	0	0	0
3	0.01	0.00	0.00
4	1.47	0.03	0.01
5	—	0.45	0.11
6	—	7.32	1.17
7	—	152.03	14.32
8	—	—	163.55
9	—	—	—

4 Reduction to Normal Form

Let $a \in \mathbb{Z}^n$ and set $q := \|a\|_\infty = \max\{|a_1|, \ldots, |a_n|\}$. For $r = -q, \ldots, 0, \ldots, q$ let

$$I_r := \{i = 1, \ldots, n \mid a_i = r\}$$

be the set of all indices where the coefficient equals r. Hence

$$\{1, \ldots, n\} = \bigcup_{r=-q}^{q} I_r \,.$$

Note that some of the sets I_r may be empty. Consider the homomorphism of $\mathbb{Z}$-modules

$$\Phi_a \colon \mathbb{Z}^n \longrightarrow \mathbb{Z}^{2q+1}, \quad (x_1, \ldots, x_n) \mapsto (y_{-q}, \ldots, y_{-1}, y_0, y_1, \ldots, y_q),$$

where

$$y_r := \sum_{i \in I_r} x_i \quad \text{for } r = -q, \ldots, 0, \ldots, q \,.$$

Its image is

$$\Phi_a(\mathbb{Z}^n) = \{(y_r)_{-q \le r \le n} \mid y_r = 0 \text{ if } I_r = \emptyset\}.$$

Furthermore we consider the homomorphism (not the α from Theorem 1)

$$\alpha \colon \mathbb{Z}^n \longrightarrow \mathbb{Z}, \quad \alpha(x) = a_1 x_1 + \cdots + a_n x_n.$$

Remarks

1. The semigroup $\mathbb{N}^n \cap \ker \alpha$ is the solution set of **(E)** for the given coefficient vector $a \in \mathbb{N}^n$. The minimal elements > 0 of $\ker \alpha$ are the indecomposable solutions of **(E)**.

2. For $x \in \mathbb{N}^n$ we have $\|\Phi_a(x)\|_1 = \|x\|_1$.

Reduction to a Special Case

As a special case we consider the homomorphism

$$\rho\colon \mathbb{Z}^{2q+1} \longrightarrow \mathbb{Z}, \quad y = (y_{-q}, \ldots, y_q) \mapsto \sum_{r=-q}^{q} r y_r = -q y_{-q} + \cdots + q y_q.$$

Then for $x \in \mathbb{N}^n$

$$\rho(\Phi_a(x)) = \sum_{r=-q}^{q} r \cdot \sum_{i \in I_r} x_i = \sum_{r=-q}^{q} \sum_{i \in I_r} r x_i = \sum_{r=-q}^{q} \sum_{i \in I_r} a_i x_i = \sum_{i=1}^{n} a_i x_i = \alpha(x).$$

This proves the first part of

Lemma 1 (i) $\rho \circ \Phi_a = \alpha$.

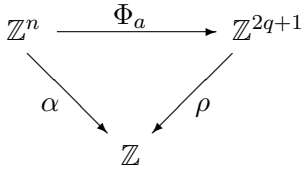
(ii) *The preimage of* $y \in \Phi_a(\mathbb{Z}^n)$ *is* $\Phi_a^{-1}(y) = \{x \mid \sum_{i \in I_r} = y_r \text{ for all } r\}$.

(iii) $\Phi_a(\mathbb{Z}^n) \cap \ker \rho = \Phi_a(\ker \alpha)$.

Proof. (ii) is trivial.

(iii) From (i) we conclude that $\alpha(x) = 0 \Longleftrightarrow \rho(\Phi_a(x)) = 0$. Hence $\Phi_a(\ker \alpha) \subseteq \ker \rho$. On the other hand for $y \in \Phi_a(\mathbb{Z}^n)$ we choose a preimage x by (ii), and $y \in \ker \rho$ implies $x \in \ker \alpha$ by (i). Thus $y \in \Phi_a(\ker \alpha)$. $\diamond$

A commutative diagram visualizes statement (i) of Lemma 1:



Let $\mathcal{M}'_q$ be the set of indecomposable solutions $y = (y_{-q}, \ldots, y_0, \ldots, y_q) \in \mathbb{N}^{2q+1}$ of the special equation

$$(\mathbf{D}'_q) \qquad -q \cdot y_{-q} - \cdots - 1 \cdot y_{-1} + 0 \cdot y_0 + 1 \cdot y_1 + \cdots + q \cdot y_q = 0\,,$$

or in other words, the set of minimal elements > 0 of $\ker \rho$. For each $y \in \mathcal{M}'_q$ choose an arbitrary $x \in \mathbb{N}^n \cap \Phi_a^{-1}(y)$, that is with

$$\sum_{i \in I_r} x_i = y_r \quad \text{for } r = -q, \ldots, 0, \ldots, q\,.$$

Clearly then $0 = \rho(y) = \rho(\Phi_a(x)) = \alpha(x)$, thus x is a solution of **(E)** in $\mathbb{N}^n - \{0\}$, and is minimal. Each minimal solution x is obtained in this way.

Corollary 1 *The set $\mathbb{N}^n \cap \Phi_a^{-1}(\mathcal{M}'_q)$ consists exactly of the indecomposable solutions of* **(E)**.

Applying Proposition 1 (iv) to $(\mathbf{D}'_q)$ we conclude that each $y \in \mathcal{M}'_q$ has one of the forms

- $y_0 = 1$, $y_{-q} = \cdots = y_{-1} = y_1 = \cdots = y_q = 0$,
- $y_0 = 0$, and $(y_{-q}, \ldots, y_{-1}, y_1, \ldots, y_q) \in \mathbb{N}^{2q}$ with $y_{-i} = x_i$ for $1 \leq i \leq q$ is an indecomposable solution of the equation

$$(\mathbf{D}_q) \qquad 1 \cdot x_1 + \cdots + q \cdot x_q = 1 \cdot y_1 + \cdots + q \cdot y_q\,.$$

We denote the set of indecomposable solutions of $(\mathbf{D}_q)$ by $\mathcal{M}_q$.

In summary the equation **(E)** is reduced to the special case where all coefficients a_i are different and non-zero. Note that "duplicate" coefficients might occur in some situations in a natural way: for example when the coefficients are not known a priori but result from intermediary computations of a comprehending algorithm.

Normal Forms

For the general case of **(E)** consider the sets J_+ of values $r > 0$ with $I_r \neq \emptyset$, and J_- of values $r > 0$ with $I_{-r} \neq \emptyset$. Then

$$P = \bigcup_{r \in J_+} I_r \quad \text{and} \quad N = \bigcup_{r \in J_-} I_r.$$

The solution of **(E)** is reduced to the equation

$$(\mathbf{D}_q(J)) \qquad \sum_{r \in J_-} r \cdot z_r = \sum_{r \in J_+} r \cdot y_r\,.$$

Call the equations $(\mathbf{D}_q(J))$ for all sets $J = J_+ \cup (-J_-)$ with subsets $J_+, J_- \subseteq \{1, \ldots, q\}$ the **normal forms** of linear Diophantine equations. Let $\mathcal{M}_q(J)$ be the set of indecomposable solutions of $(\mathbf{D}_q(J))$. Then we have shown (see also [3]):

Proposition 3 *All indecomposable solutions x of* **(E)** *arise in one of the two following ways (where $q = \|a\|_\infty$):*

(i) *For* $i \in I_0$ *set* $x_i = 1$, *and* $x_j = 0$ *for* $j \neq i$.

(ii) *For each* $(z, y) \in \mathcal{M}_q(J)$ *where* $y = (y_r)_{r\in J_+}$ *and* $z = (z_r)_{r\in J_-}$ *choose* $x_i \in \mathbb{N}$ *for* $i \in I_{-q} \cup \cdots \cup I_{-1} \cup I_1 \cup \cdots \cup I_q$ *with*

$$\sum_{i\in I_r} x_i = z_{-r} \quad \textit{for } r = -q, \ldots, -1 \quad \textit{and} \quad \sum_{i\in I_r} x_i = y_r \quad \textit{for } r = 1, \ldots, q.$$

Remark 3 If in (ii) we set $x_+ = (x_i)_{i\in P}$ and $x_- = (x_i)_{i\in N}$, then

$$\|x_+\|_1 = \|y\|_1 \quad \text{and} \quad \|x_-\|_1 = \|z\|_1.$$

Moreover

$$\sum_{i\in P} a_i x_i = \sum_{r\in J_+} r y_r = \sum_{r\in J_-} r \cdot z_r = \sum_{i\in N} (-a_i) x_i.$$

The Standard Equation

For a subset $I \subseteq \{1, \ldots, q\}$ consider the embedding

$$\tau_I : \mathbb{N}^I \longrightarrow \mathbb{N}^q, \quad (x_i)_{i\in I} \mapsto \bar{x},$$

that consists of filling up the positions different from I with zeros, that is

$$\bar{x} = (\bar{x}_1, \ldots, \bar{x}_q) \quad \text{where } \bar{x}_i = \begin{cases} x_i & \text{for } i \in I, \\ 0 & \text{otherwise.} \end{cases}$$

Then clearly (x, y) is a solution of $(\mathbf{D}_q(J))$ if and only if $(\tau_{J_-}(x), \tau_{J_+}(y))$ is a solution of $(\mathbf{D}_q)$, and (x, y) is an indecomposable solution of $(\mathbf{D}_q(J))$ if and only if $(\tau_{J_-}(x), \tau_{J_+}(y))$ is an indecomposable solution of $(\mathbf{D}_q)$. In other words, the natural embedding

$$(\tau_{J_-}, \tau_{J_+}) \colon \mathcal{M}_q(J) \longrightarrow \mathcal{M}_q$$

is a bijection with $\mathcal{M}_q \cap (\tau_{J_-}, \tau_{J_+})(\mathbb{N}^{J_-} \times \mathbb{N}^{J_+})$. Therefore the following procedure gives all indecomposable solutions of $(\mathbf{D}_q(J))$ under the assumption that the complete set $\mathcal{M}_q$ of indecomposable solutions of $(\mathbf{D}_q)$ is known:

- Remove the vectors from $\mathcal{M}_q$ that have at least one non-zero entry at an index not belonging to J_- or J_+.
- From the remaining vectors remove the (zero) components for indices not belonging to J_- or J_+.

This finally reduces the search for the indecomposable solutions of **(E)** to the special case $(\mathbf{D}_q)$, and justifies calling $(\mathbf{D}_q)$ **the standard linear Diophantine equation** for the parameter q.

From a theoretical standpoint the breakdown of the general case of **(E)** to an instance of a well-arranged set of standard cases $(\mathbf{D}_q)$ might seem interesting. Unfortunately this reduction does not result in an essential reduction in complexity. It doesn't make it much easier to find all indecomposable solutions nor does it help much with counting them. But we may hope that results for the special case generalize in a useful way.

5 Examples

We consider the standard equation

$$(\mathbf{D}_q) \qquad x_1 + \cdots + qx_q = y_1 + \cdots + qy_q\,.$$

For convenience (in the present context) we often write vectors $(x, y) \in \mathbb{N}^q \times \mathbb{N}^q$ in the form

$$\begin{pmatrix} x_1 & \dots & x_q \\ y_1 & \dots & y_q \end{pmatrix} \tag{3}$$

and use the linear form (the weight)

$$\lambda\colon \mathbb{Z}^q \longrightarrow \mathbb{Z}, \quad (z_1, \dots, z_q) \mapsto \sum_{i=1}^{q} i \cdot z_i = z_1 + \cdots + qz_q\,.$$

Let us start with the simple example $q = 1$. The Diophantine equation is $x_1 = y_1$. The unique indecomposable solution is

$$\begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

For $q = 2$ the equation is $x_1 + 2x_2 = y_1 + 2y_2$, and we immediately find the indecomposable solutions

$$\begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix}.$$

Hence every other indecomposable solution has at least one of the coordinates x_1 or y_1 zero, likewise one of x_2 or y_2. These conditions only leave the possibilities (see Proposition 2)

$$\begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 0 & 1 \\ 2 & 0 \end{pmatrix}.$$

As a first non-trivial case we take $q = 3$. The following 9 indecomposable solutions have exactly two non-zero coordinates:

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}, \begin{pmatrix} 3 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \begin{pmatrix} 0 & 1 & 0 \\ 2 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 3 & 0 \\ 0 & 0 & 2 \end{pmatrix},$$

$$\begin{pmatrix} 0 & 0 & 1 \\ 3 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 & 2 \\ 0 & 3 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix}.$$

Every other indecomposable solution has at least 3 coordinates $\neq 0$, and a 0 in each column. The possible patterns are (up to interchanging x and y):

$$\begin{pmatrix} * & 0 & 0 \\ 0 & * & * \end{pmatrix}, \begin{pmatrix} 0 & * & 0 \\ * & 0 & * \end{pmatrix}, \begin{pmatrix} 0 & 0 & * \\ * & * & 0 \end{pmatrix}.$$

In the first case we conclude that $x_1 = 2y_2 + 3y_3 \geq 5$, hence

$$(x, y) > \begin{pmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix},$$

thus the first pattern doesn't support an indecomposable solution. In the second case $2x_2 = y_1 + 3y_3 \geq 4$, and in the third case $3x_3 = y_1 + 2y_2 \geq 3$. The remaining possibilities yield the indecomposable solutions

$$\begin{pmatrix} 0 & 2 & 0 \\ 1 & 0 & 1 \end{pmatrix} \text{ and } \begin{pmatrix} 0 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}, \quad \text{and by symmetry} \quad \begin{pmatrix} 1 & 0 & 1 \\ 0 & 2 & 0 \end{pmatrix} \text{ and } \begin{pmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

6 Simple Remarks

The observations in the examples of Section 5 lead to some simple general remarks:

Lemma 2 *Let $q \geq 2$.*

(i) *For $i = 1, \ldots q$ the vector (e_i, e_i) is an indecomposable solution of $(\mathbf{D}_q)$.*

(ii) *All other indecomposable solutions (x, y) have $x_i y_i = 0$ for all i, that is, every column of the matrix (3) contains at most one non-zero entry.*

(iii) *The indecomposable solutions with exactly two non-zero coordinates are the*

$$u_{ij} = \frac{1}{\gcd(i,j)}(je_i, ie_j) \quad \text{for } i, j = 1, \ldots q$$

including the $(e_i, e_i) = u_{ii}$ for $i = 1, \ldots q$.

Proof. (i) is trivial.
(ii) Otherwise $(x, y) \geq (e_i, e_i)$.
(iii) See Proposition 2. $\diamond$

Remark 1 The indecomposable solutions (x, y) of $(\mathbf{D}_{q-1})$ canonically yield indecomposable solutions of $(\mathbf{D}_q)$ by appending a zero as last coordinate to each of x and y. All the other indecomposable solutions of $(\mathbf{D}_q)$ have the form

$$\begin{pmatrix} \ldots & a \\ \ldots & b \end{pmatrix}$$

where at least one of a and b is > 0. They fall into one of three cases:

1. $a = 0$, $b \in \{1, \ldots, q\}$,
2. $b = 0$, $a \in \{1, \ldots, q\}$,
3. $a > 0$ and $b > 0$—there is exactly one indecomposable solution of this type: $(e_q, e_q) = u_{qq}$.

The article [21] treats the standard congruence

$$(\mathbf{C}_q) \qquad x_1 + \cdots + (q-1)x_{q-1} \equiv 0 \pmod{q}.$$

It has a connection with the present context:

Proposition 4 *The indecomposable solutions (x, y) of $(\mathbf{D}_q)$ with $y = ke_q$ $(k > 0)$ are*

(i) (e_q, e_q).

(ii) (x, y) *where* $x = (x', 0)$ *with* $x' \in \mathbb{N}^{q-1}$ *an indecomposable solution of* $(\mathbf{C}_q)$ *and* $\lambda(x') = kq$.

Proof. Let (x, y) be an indecomposable solution of $(\mathbf{D}_q)$ with $y = ke_q$, and $(x, y) \neq (e_q, e_q)$. Then $x_q = 0$ and $\lambda(x) = \lambda(y) = kq \equiv 0 \pmod{q}$, hence $x' = (x_1, \ldots, x_{q-1})$ is a solution of $(\mathbf{C}_q)$ with $\lambda(x') = \lambda(x) = kq$. Let $u \in \mathbb{N}^{q-1}$ be a solution of $(\mathbf{C}_q)$ with $0 < u \leq x'$. Then

$$\lambda(u) = u_1 + \cdots + (q-1) \cdot u_{q-1} \equiv 0 \pmod{q}, \quad \text{hence } = lq \text{ for some } l \in \mathbb{N}.$$

Since $\lambda(u) \leq \lambda(x') = \lambda(x)$ we have $l \leq k$. Thus $((u, 0), le_q)$ is a solution $\leq (x, y)$ of $(\mathbf{D}_q)$. We conclude that $u = x'$, thus x' is indecomposable.

For the reverse direction assume that $x' \in \mathbb{N}^{q-1}$ is an indecomposable solution of $(\mathbf{C}_q)$, and $x'_1 + \cdots + (q-1) \cdot x'_{q-1} = lq$. Then

$$(x, y) = ((x'_1, \ldots, x'_{q-1}, 0), (0, \ldots, 0, l)) \in \mathbb{N}^q \times \mathbb{N}^q$$

is a solution of $(\mathbf{D}_q)$. Let $(u, v) \leq (x, y)$ be a possibly smaller non-zero solution of $(\mathbf{D}_q)$. Then $0 \leq u_i \leq x'_i$ for $i = 1, \ldots, q-1$, $u_q = 0$, and

$$u_1 + \cdots + (q-1) \cdot u_{q-1} = \lambda(u) = \lambda(v) = q \cdot v_q \equiv 0 \pmod{q}.$$

Hence $u' = (u_1, \ldots, u_{q-1})$ is a solution of $(\mathbf{C}_q)$ and $\leq x'$, hence $u' = x'$, $(u, v) = (x, y)$, and $l = k$. $\diamond$

Remark 2 Each solution (x, y) of $(\mathbf{D}_q)$ consists of a pair of partitions of the same number $r = x_1 + \cdots + qx_q = y_1 + \cdots + qy_q$ into pieces of size $\leq q$, namely x_1 pieces of size 1, ..., x_q pieces of size q, same for y. This suggests yet another universal way to construct indecomposable solutions: Fix $r \in \{1, \ldots, q\}$ and set $y_r = 1$, $y_j = 0$ else. Then solve $x_1 + \cdots + qx_q = r$; each solution corresponds to a unique partition of r and yields an indecomposable solution (x, y) of $(\mathbf{D}_q)$. Let P be the partition function (number of partitions of its argument). Then in particular there are exactly $P(q)$ solutions of each of the types (e_q, y) or (x, e_q) where x or y is a partition of q.

Application of Theorem 1 to the standard equation $(\mathbf{D}_q)$ yields:

Proposition 5 *Let $q \geq 2$ and (x, y) be an indecomposable solution of $(\mathbf{D}_q)$. Then:*

(i) $\|x\|_1 \leq q$ *and* $\|y\|_1 \leq q$.

(ii) *If $\|x\|_1 = q$, then $(x, y) = u_{iq}$ for some index $i < q$, coprime with q. Likewise if $\|y\|_1 = q$, then $(x, y) = u_{qj}$ for some index $j < q$, coprime with q.*

(iii) $\|x\|_1 \cdot \|y\|_1 \leq \lambda(x) = \lambda(y)$.

Proof. (i) and (iii) are immediate consequences. So we only need to prove the statement (ii) for $\|y\|_1 = q$. Theorem 1 yields $x_i = 0$ except for $a_i = q$, that is $i = q$. Hence $x = je_q$ for some integer j, $1 \leq j \leq q$. Since (e_q, e_q) is a solution of $(\mathbf{D}_q)$, necessarily $y_q = 0$.

> Otherwise the minimality of (x, y) enforces $(x, y) = (e_q, e_q)$, leading to the contradiction $q = \|y\|_1 = \|e_q\|_1 = 1$.

Now $y = (y', 0)$ where $y' \in \mathbb{N}^{q-1}$ is an indecomposable solution of $(\mathbf{C}_q)$ and $\lambda(y') = \lambda(x) = jq$ by Proposition 4. Since $\|y'\|_1 = \|y\|_1 = q$, Theorem 2 of [21] implies $\|y'\|_1 + \sigma(y') \leq q + 1$, hence $\sigma(y) = \sigma(y') \leq 1$, hence $y = qe_i$ for some index i with $1 \leq i \leq q - 1$, and $qi = \lambda(y) = \lambda(x) = jq$, hence $j = i$, and finally $(x, y) = u_{qj}$, and this is a minimal solution if and only if q and j are coprime. $\diamond$

Corollary 1 *Let (x, y) be an indecomposable solution of $(\mathbf{D}_q)$. Then*

(i) $\lambda(x) \leq q \cdot (q - 1)$.

(ii) *If $\lambda(x) = q \cdot (q - 1)$, then $(x, y) = u_{q-1,q}$ or $u_{q,q-1}$.*

Proof. (i) If $\|x\|_1 = q$, then $(x, y) = u_{iq}$ with $i < q$, and $\lambda(x) = q \cdot i \leq q \cdot (q-1)$. Otherwise $\|x\|_1 \leq q - 1$, and

$$\lambda(x) \;=\; x_1 + \cdots qx_q \;\leq\; q \cdot (x_1 + \cdots + x_q) \;\leq\; q \cdot (q - 1).$$

(ii) If $x_q = y_q = 0$, then $x = (x', 0)$ and $y = (y', 0)$ with $x', y' \in \mathbb{N}^{q-1}$, and (x', y') solves $(\mathbf{D}_{q-1})$. Hence $\lambda(x) = \lambda(x') \leq (q-1)(q-2)$ by (i). Thus we may assume without loss of generality that $y_q > 0$. Then $x_q = 0$ and

$$q \cdot (q - 1) \;=\; \lambda(x) \;=\; x_1 + \cdots + (q - 1)\, x_{q-1} \;\leq\; (q - 1) \cdot \|x\|_1.$$

This implies $\|x\|_1 = q$ and $(x, y) = u_{q-1,q}$. In the same way, if $x_q > 0$ we conclunde that $(x, y) = u_{q,q-1}$. $\diamond$

Remark 3 From Proposition 5 we derive another variant of enumerating all indecomposable solutions of $(\mathbf{D}_q)$. They fall into six cases. For $x = (x_1, \ldots, x_q) \in \mathbb{N}^q$ we denote the truncated vector as $x' = (x_1, \ldots, x_{q-1}) \in \mathbb{N}^{q-1}$. (We assume that $q \geq 2$.)

1. $(x, y) = (e_q, e_q)$. Then $x' = y' = 0 \in \mathbb{N}^{q-1}$, and $x_q = y_q = 1$.

2. $(x, y) = u_{kq} = (qe_k, ke_q)$ where $\gcd(k, q) = 1$. Then $x' = qe_k$, $y' = 0$, $x_q = 0$, $y_q = k$, $\|x'\|_1 = q$, $\|y'\|_1 = 0$.

3. $(x, y) = u_{qk} = (ke_q, qe_k)$ where $\gcd(k, q) = 1$. Then $x' = 0$, $y' = qe_k$, $x_q = k$, $y_q = 0$, $\|x'\|_1 = 0$, $\|y'\|_1 = q$.

4. $((x', 0), (y', 0))$ where (x', y') is an indecomposable solution of $(\mathbf{D}_{q-1})$. Then $x_q = y_q = 0$ and $\|x'\|_1 \leq q - 1$, $\|y'\|_1 \leq q - 1$.

5. $x_q = 0$, $y_q > 0$, $\|x\|_1 \leq q-1$, $\|y\|_1 \leq q-1$. Then $\lambda(x') = \lambda(x) = \lambda(y) = \lambda(y') + qy_q$. This equation determines y_q uniquely as

$$y_q = \frac{\lambda(x') - \lambda(y')}{q}.$$

   A solution for y_q exists if and only if $\lambda(x') > \lambda(y')$ and the right-hand side is an integer.

6. $x_q > 0$, $y_q = 0$, $\|x\|_1 \leq q-1$, $\|y\|_1 \leq q-1$. Then $\lambda(y') = \lambda(y) = \lambda(x) = \lambda(x') + qx_q$. This equation determines x_q uniquely as

$$x_q = \frac{\lambda(y') - \lambda(x')}{q}.$$

   A solution for y_q exists if and only if $\lambda(y') > \lambda(x')$ and the right-hand side is an integer.

7 The Support of an Indecomposable Solution

From the results of Section 6 we immediately derive some lemmas:

Lemma 3 *Every indecomposable solution $(x, y) \in \mathbb{N}^q \times \mathbb{N}^q$ of $(\mathbf{D}_q)$ has $\sigma(x, y) \geq 2$, $\sigma(x) \geq 1$, $\sigma(y) \geq 1$. The only indecomposable solutions with two-element support are the u_{ij} for $1 \leq i, j \leq q$.*

Lemma 4 *If (x, y) is an indecomposable solution of $(\mathbf{D}_q)$ with $\sigma(x, y) \geq 3$ (that is other than u_{ij} for some pair of indices i, j), then $\|x\|_1 \leq q - 1$ and $\|y\|_1 \leq q - 1$.*

Lemma 5 *If (x, y) is an indecomposable solution of $(\mathbf{D}_q)$ other than (e_i, e_i) for some index i, then* $\mathrm{supp}(x)$, $\mathrm{supp}(y) \subseteq \{1, \ldots, q\}$ *are disjoint subsets.*

Lemma 6 *Assume that $q \geq 2$, and that $(x, y) \in \mathbb{N}^{2q}$ is an indecomposable solution of $(\mathbf{D}_q)$. Then its width is*

$$\sigma(x, y) = \sigma(x) + \sigma(y) \leq q.$$

Lemma 7 *Assume that $q \geq 4$, and let $(x, y) \in \mathbb{N}^{2q}$ be an indecomposable solution of $(\mathbf{D}_q)$ with $y_q \geq 1$. Then*

$$\sigma(x) \leq \frac{q}{2}.$$

Proof. If $x_q \geq 1$, then $(x, y) = (e_q, e_q)$, and $\sigma(x) = 1$.

Otherwise we have $x_q = 0$. Assume that $\sigma(x) > \frac{q}{2}$. Then among the $q - 1$ indices $1, \ldots, q - 1$ there is a pair i, $q - i$ with $q - i \neq i$ such that $x_i > 0$ and $x_{q-i} > 0$. This makes $(e_i + e_{q-i}, e_q)$ a solution of $(\mathbf{D}_q)$, and $(e_i + e_{q-i}, e_q) \leq (x, y)$. Hence $x = e_i + e_{q-i}$ and $\sigma(x) \leq 2$, contradiction. $\diamond$

See Section 5 for counterexamples with $q = 1$, 2, or 3.

Lemma 8 *If (x, y) is a solution of $(\mathbf{D}_q)$, then (y, x) is also a solution. If (x, y) is indecomposable, then so is (y, x).*

Call a solution (x, y) of $(\mathbf{D}_q)$ **flat** if all of its non-zero coordinates are 1, or equivalently, if $\|(x, y)\|_1 = \sigma(x, y)$, or $\|(x, y)\|_\infty = 1$.

Lemma 9 *Assume that $(x, y) \in \mathbb{N}^{2q}$ is a flat solution of $(\mathbf{D}_q)$. Then no proper superset $S \supset \mathrm{supp}(x, y)$ can support an indecomposable solution.*

8 An Enhanced Algorithm for the Standard Equation $(\mathbf{D}_q)$

The results and remarks in Sections 6 and 7 lead to noteworthy enhancements of our algorithm for the standard equation.

A preliminary construction of all the indecomposable solution u_{ij} with two-element support makes sense. Then all other solutions satisfy Lemmas 4, 5, and 6.

We look at the support of the first half x of a possible indecomposable solution, and loop over its size s from 1 to q. For each subset $S \subseteq \{1, \ldots, q\}$ of size s we consider all subsets $T \subseteq \{1, \ldots, q\}$ with $T \cap S = \emptyset$. Then we check all vectors (x, y) with supports S and T, all coordinates ≥ 1, $\leq q-1$, and $\|x\|_1 \leq q-1$, $\|y\|_1 \leq q-1$, whether they solve $(\mathbf{D}_q)$. If we find a solution (x, y) we add it to the list of solutions, as well as its "complementary" vector (y, x). Doing so allows us to

- reduce our loop for s to $1, \ldots, \lfloor q/2 \rfloor$,
- assume that $\sigma(x) \leq \sigma(y)$, or in other words restrict the choice of T to sets of size $\#T = t$ with $s \leq t \leq q - s$.

If we detect a flat solution we add its support to a `stoplist` and never consider supersets of it as possible supports (S, T).

Here is the sketch of the algorithm:

1. Initialize the list `solulist` of indecomposable solutions as well as the `stoplist` as empty lists.
2. If $q = 1$ output the one-element `solulist` consisting of $(1, 1)$.
3. For $q \geq 2$ first treat the case $s = t = 1$ separately, i. e. construct the u_{ij}. Loop over $i = 1, \ldots, q$ and over $j = 1, \ldots, q$.
   - Compute $d = \gcd(i, j)$ and set $x_i = j/d$, $y_j = i/d$, $x_k = y_h = 0$ for all other indices.
   - Add (x, y) to `solulist`.

   If $q = 2$ output `solulist` and exit.
4. Loop over $s = 1, \ldots, \lfloor \frac{q}{2} \rfloor$. (At this point $q \geq 3$.)
   - Construct all subsets $S \subseteq \{1, \ldots, q\}$ with s elements. Loop over S. For each S construct all subsets $T \subseteq \{1, \ldots, q\} - S$ with $t \geq s$ elements (for $s = 1$: with $t \geq 2$ elements).
   - Skip (S, T) if it has a subset contained in the `stoplist`.
   - Check whether (S, T) supports a flat solution (x, y), that is whether $\sum_{i \in S} i = \sum_{j \in T} j$. If this is the case, add (x, y) to the `solulist` and add (S, T) to the `stoplist`. Furthermore add (y, x) to the `solulist` and add (T, S) to the `stoplist`.
   - Else (no flat solution): Construct all indecomposable solutions (x, y) supported by (S, T) and add them to `solulist`, as well as (y, x).
5. Reduce `solulist` to its indecomposable elements, and output it.

The next to last step, constructing all indecomposable solutions (x, y) supported by (S, T), is done by letting all coordinates in (S, T) separately run from 1 to $q-1$ with the restrictions $\|x\|_1 \leq q-1$, $\|y\|_1 \leq q-1$, and check whether they make up a solution, that is check whether $\lambda(x) = \lambda(y)$.

Appendix A.5 contains the Python code.

9 A Lower Bound for the Number of Indecomposable Solutions of the Standard Equation

Let $N(q)$ be the number of indecomposable solutions of $(\mathbf{D}_q)$. From the examples in Section 5 we know $N(1) = 1$, $N(2) = 4$, and $N(3) = 13$. The algorithm from Section 3 yields $N(4) = 34$, $N(5) = 99$, $N(6) = 210$, $N(7) = 559$, and $N(8) = 1164$. The enhancements of Section 8 enable the computation of $N(9) = 2531$, $N(10) = 4940$, $N(11) = 10735$. There doesn't seem to be a known closed formula for $N(q)$. Anyway the following tasks make sense:

- Find upper and lower bounds for $N(q)$.
- Determine the asymptotic behaviour of $N(q)$ depending on q.

The explicit numerical results suggest a simple exponential growth, maybe slightly subexponential, see Figure 1.

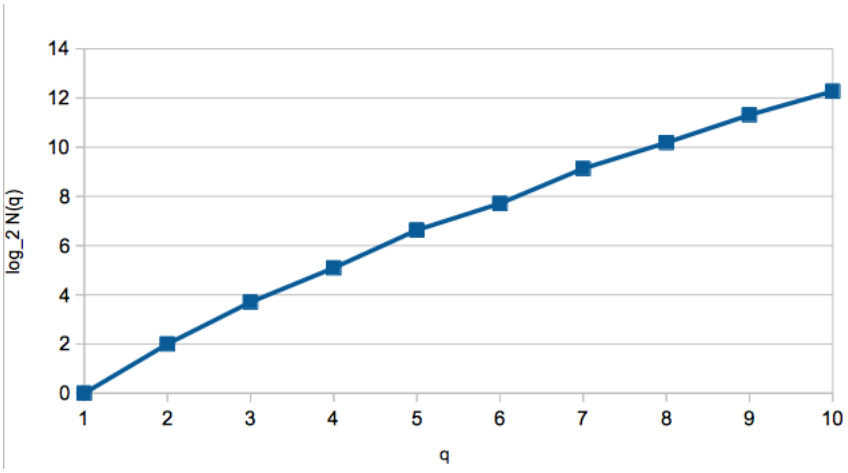


Figure 1: 2-logarithm of the number $N(q)$ of indecomposable solutions of $(\mathbf{D}_q)$

The trivial lower bound $N(q) \geq q^2$ results from the q^2 indecomposable solutions u_{ij}.

A stronger lower bound derives from Remark 2 in Section 6: If we denote the number of partitions of r by $P(r)$, then we easily get

$$N(q) \geq N(q-1) + 2 \cdot P(q),$$

Table 1: Numbers of indecomposable solutions and lower bounds

q	1	2	3	4	5	6	7	8	9	10	11
$N(q)$	1	4	13	34	99	210	559	1164	2531	4940	10735
$P(q)$	1	2	3	5	7	11	15	22	30	42	56
bound	2	6	12	22	36	58	88	132	192	276	388

and by induction prove:

Theorem 2 *The number $N(q)$ of indecomposable solutions of $(\mathbf{D}_q)$ has the lower bound*

$$N(q) \geq 2 \cdot \sum_{r=1}^{q} P(r)$$

for $q \geq 3$ where P is the partition function.

Table 1 gives an impression of the quality of this lower bound: It seems to be far too low.

Now we estimate the asymptotic behaviour. We use the Hardy-Ramanujan bound [11]:

$$P(q) \geq \frac{c}{q} \cdot e^{\pi \cdot \sqrt{\frac{2q}{3}}}$$

for almost all q if $c \in [0, \frac{1}{4\sqrt{3}}[$. In order to apply it to the growth of $\sum P(r)$ we first consider an auxiliary function.

Lemma 10 *Let $f :]0, \infty[\longrightarrow \mathbb{R}$ be the function*

$$f(x) = \frac{1}{x} \cdot e^{a\sqrt{x}} \quad \text{where } a = \pi \cdot \sqrt{\frac{2}{3}}.$$

Then

(i) *f is monotonocally increasing for $x \geq 1$.*

(ii) *In particular*

$$f(r) \geq \int_{r-1}^{r} f(x)\, dx \quad \text{for } r \geq 2$$

Proof. The derivation

$$f'(x) = \frac{x \cdot \frac{a}{2\sqrt{x}} \cdot e^{a\sqrt{x}} - e^{a\sqrt{x}}}{x^2} = \frac{a\sqrt{x} - 2}{x^2} \cdot e^{a\sqrt{x}}$$

has a single zero in $]0, \infty[$ at $x = \frac{4}{a^2} = \frac{6}{\pi^2} < 1$. For $x > \frac{6}{\pi^2}$ we have $f'(x) > 0$, hence f is monotonically increasing. The statement (ii) is an immediate consequence. $\diamond$

The inequality (ii) allows to replace the sum in Theorem 2 by an integral:

$$\sum_{r=1}^{q} p(r) = 1 + \sum_{r=2}^{q} p(r) \;\geq\; 1 + \sum_{r=2}^{q} c \cdot f(r) \geq\; 1 + c \cdot \eta + c \cdot \int_1^q f(x)\, dx$$

where

$$\eta = f(2) - \int_1^2 f(x)\, dx.$$

Collecting the inequalities together we get:

Lemma 11 *For almost all $q \geq 2$ and arbitrary $c \in [0, \frac{1}{4\sqrt{3}}[$*

$$N(q) \;\geq\; 2 + 2c \cdot \eta + 2c \cdot \int_1^q f(x)\, dx.$$

We have no closed formula for this integral but a useful inequality:

Lemma 12 *Let $g{:}]0, \infty[\longrightarrow \mathbb{R}$ be the function*

$$g(x) = \frac{1}{\sqrt{x}} \cdot e^{a\sqrt{x}}.$$

Then for all $x > 0$

$$f(x) \geq \frac{\sqrt{6}}{\pi} \cdot g'(x).$$

Proof. We calculate

$$\begin{aligned} g'(x) &= \frac{\sqrt{x} \cdot \frac{a}{2\sqrt{x}} \cdot e^{a\sqrt{x}} - \frac{1}{2\sqrt{x}} \cdot e^{a\sqrt{x}}}{x} = \frac{e^{a\sqrt{x}}}{2x} \cdot \left(a - \frac{1}{\sqrt{x}}\right) = \frac{1}{2} \cdot f(x) \cdot \left(a - \frac{1}{\sqrt{x}}\right) \\ &\leq \frac{a}{2} \cdot f(x), \end{aligned}$$

and note that $\frac{a}{2} = \frac{\pi}{\sqrt{6}}$. $\diamond$

By the main theorem of calculus we conclude that

$$\begin{aligned} \int_1^q f(x)\, dx \geq \frac{\sqrt{6}}{\pi} \cdot \int_1^q g'(x)\, dx &= \frac{\sqrt{6}}{\pi} \cdot [g(q) - g(1)] \\ &= \frac{\sqrt{6}}{\pi \cdot \sqrt{q}} \cdot e^{\pi\sqrt{\frac{2q}{3}}} - \frac{\sqrt{6}}{\pi} \cdot e^{\pi\sqrt{\frac{2}{3}}}, \end{aligned}$$

hence by Lemma 11

$$\begin{aligned} N(q) &\geq 2 + 2c\eta + 2c \cdot \frac{\sqrt{6}}{\pi \cdot \sqrt{q}} \cdot e^{\pi\sqrt{\frac{2q}{3}}} - 2c \cdot \frac{\sqrt{6}}{\pi} \cdot e^{\pi\sqrt{\frac{2}{3}}} \\ &= \frac{b}{\sqrt{q}} \cdot e^{\pi\sqrt{\frac{2q}{3}}} + h(c) \end{aligned}$$

with

$$b = 2c \cdot \frac{\sqrt{6}}{\pi} \in [0, \frac{1}{\pi\sqrt{2}}[$$

and

$$h(x) = 2 + 2x\eta - 2x \cdot \frac{\sqrt{6}}{\pi} \cdot e^{\pi\sqrt{\frac{2}{3}}}.$$

The function h is linear in x with $h(0) = 2$, and a numeric calculation (using SageMath) gives $h(\frac{1}{4\sqrt{3}}) \geq 0.0049$. Thus $h(x) > 0$ for $0 \leq x \leq \frac{1}{4\sqrt{3}}$.

This finishes the proof of the slightly subexponential bound:

Theorem 3 *The number $N(q)$ of indecomposable solutions of $(\mathbf{D}_q)$ has the lower bound*

$$N(q) \geq \frac{b}{\sqrt{q}} \cdot e^{\pi\sqrt{\frac{2q}{3}}} \quad \textit{for almost all } q \in \mathbb{N}_1$$

for arbitrary $b \in [0, \frac{1}{\pi\sqrt{2}}[$.

10 An Upper Bound for the Number of Indecomposable Solutions of the Standard Equation

Theorem 1 yields an upper bound for $N(q)$. This is slightly enhanced by Remark 3 in Section 6 via an auxiliary result:

Proposition 6 *Let $x', y' \in \mathbb{N}^{q-1}$ be given with $\|x'\|_1 \leq q-1$ and $\|y'\|_1 \leq q-1$. Then there exists at most one pair $(x_q, y_q) \in \mathbb{N}^2$ such that $((x', x_q), (y', y_q))$ is an indecomposable solution of $(\mathbf{D}_q)$.*

Proof. In each of the six cases of the remark the values of x_q and y_q are uniquely determined. $\diamond$

There is a total of $\binom{2q-2}{q-1}$ points $x' \in \mathbb{N}^{q-1}$ whose coordinate sum $\|x'\|_1$ is $\leq q-1$. Thus there are $\binom{2q-2}{q-1}^2$ points $(x', y') \in \mathbb{N}^{2q-2}$ with $\|x'\|_1 \leq q-1$ and $\|y'\|_1 \leq q-1$. Since (for $q \geq 2$) among these are also the points $(e_k, 0)$ and $(0, e_k)$ for $1 \leq k \leq q-1$ that don't lead to solutions of $(\mathbf{D}_q)$ as in Proposition 6, we don't need to separately count the few missing indecomposable solutions u_{kq} and u_{qk}. The result (valid also for $q = 1$) is:

Theorem 4 *The number of indecomposable solutions of $(\mathbf{D}_q)$ is bounded by*

$$N(q) \leq \binom{2q-2}{q-1}^2.$$

This yields the bounds

$$N(1) \le 1, \quad N(2) \le 4, \quad N(3) \le 36, \quad N(4) \le 400, \quad N(5) < 4900.$$

Clearly the bound in Theorem 4 is much too large: We counted *all* integer points in the product of two simplices, not only the solutions of $(\mathbf{D}_q)$, not to mention only the indecomposable ones!

From the well-known inequality for the middle binomial coefficients (see for example [19])

$$\binom{2n}{n} < \frac{4^n}{\sqrt{\pi n}} \quad \text{for all } n \ge 1$$

we derive a bound for the asymptotic behaviour of $N(q)$ (assume $q \ge 2$):

$$N(q) \;\le\; \binom{2q-2}{q-2}^2 \;<\; \frac{16^{q-1}}{\pi(q-1)} \;=\; \frac{q}{q-1} \cdot \frac{1}{\pi} \cdot \frac{16^{q-1}}{q} \;<\; \frac{16^{q-1}}{q}.$$

Corollary 1 *The number of indecomposable solutions of* $(\mathbf{D}_q)$ *is bounded by*

$$N(q) < \frac{16^{q-1}}{q} = \frac{1}{q} \cdot e^{(q-1)\log 16}.$$

Even this bound, although far from the true value, exhibits a slightly subexponential growth.

Conjecture. Theorem 1 should improve for $(\mathbf{D}_q)$ in the following way: Let $(x, y) \in \mathbb{N}^{2q}$ be an indecomposable solution of $(\mathbf{D}_q)$. Then $\|x\|_1 + \sigma(x) \le q + 1$ and $\|y\|_1 + \sigma(y) \le q + 1$. This conjecture is empirically true for q up to 8. For an analogous result on linear congruences compare Theorem 2 of [21].

A Python Code

A.1 Auxiliary Routines

```
def dlist0(n, m):
  """Construct the list of integer vectors > 0 of dimension n and
     height <= m."""
  if n == 0:
    return []
  auxlist = [[]]
  for r in range(n):
    outlist = []
    for y in auxlist:
      for t in range(m+1):
        x = y + [t]
        outlist.append(x)
    auxlist = [] + outlist
  return outlist

def dlist1(n,m):
  """Construct the list of integer vectors (x_1,...,x_n) of dim n
     in the simplex x_1+...+x_n <= m, all x_i >= 0"""
  if n == 0:
    return []
  auxlist = [[]]
  for r in range(n):
    outlist = []
    for y in auxlist:
      s = sum(y)
      for t in range(m+1-s):
        x = y + [t]
        outlist.append(x)
    auxlist = [] + outlist
  return outlist
```

```
def dlist2(n,m):
  """generate list of integer vectors (x_1,...,x_n) of dim n
     in the simplex x_1+...+x_n <= m, all x_i >= 1"""
  if n == 0:
    return []
  auxlist = [[]]
  for r in range(n):
    outlist = []
    for y in auxlist:
      s = sum(y)
      for t in range(1,m+1-s):
        x = y + [t]
        outlist.append(x)
    auxlist = [] + outlist
  return outlist

def smaller(lista,listb):
  """Compare two integer lists in componentwise (partial) order of Z^n."""
  ll = len(lista)
  unequal = False
  if ll != len(listb):
    return False
  for i in range(ll):
    if lista[i] > listb[i]:
      return False
    elif (not(unequal) and (lista[i] < listb[i])):
      unequal = True
  if unequal:
    return True
  else:
    return False
```

```
def minelts(vlist):
  """Delete all entries from vlist that are properly larger than another
  entry."""
  i = 0
# Loop over i
  ll = len(vlist)
  while i < ll:
    t = vlist[i]
    for j in range(ll-1,i,-1):
      if smaller(t,vlist[j]):
        del vlist[j]
    for j in range(i-1,-1,-1):
      if smaller(t,vlist[j]):
        del vlist[j]
        i = i-1
    i = i+1
    ll = len(vlist)
  return vlist

def chkdio(alist, xlist):
  """Check if alist[0]*xlist[0] + ... + alist[l-1]*ilist[l-1] == 0."""
  l = len(alist)
  if len(xlist) != l:
    return False
  sum = 0
  for i in range(l):
    sum = sum + alist[i]*xlist[i]
  if sum == 0:
    return True
  else:
    return False
```

```
def stddio(xyvec):
  """Check if 1*xyvec[0] + ... + q*xyvec[q-1] = 1*xyvec[q] + ... + q*xyvec[2q-1]
     where q = len(xyvec/2)."""
  qq = len(xyvec)//2
  xsum = 0
  ysum = 0
  for i in range(qq):
    xsum = xsum + (i+1)*xyvec[i]
  for j in range(qq):
    ysum = ysum + (j+1)*xyvec[qq+j]
  if xsum == ysum:
    return True
  else:
    return False

def subset(l1,l2):
  """Check whether l1 is a subset of l2."""
  for elt in l1:
    if elt not in l2:
      return False
  return True

def subsets(n,k):
  """Generate list of all k-element subsets of {1,...,n}."""
  setlist = []
  NN = 2**n
  for i in range(NN):
    bitlist = baserep(i,2)
    while len(bitlist) < n:
      bitlist.insert(0,0)
    set = []
    for j in range(n):
      if bitlist[j] == 1:
        set.append(j+1)
    if len(set) == k:
      setlist.insert(0,set)
  return setlist
```

```
def eEuclid(a,b):
  """Compute the gcd d of two integers a and b together with
  integer coefficients x and y such that d = ax + by.
  Ouput the triple [d,x,y]."""
# Initialization
  if a < 0:
    r0 = -a
    v = -1    # keep sign
  else:
    r0 = a
    v = 1
  if b < 0:
    r1 = -b
    w = -1    # keep sign
  else:
    r1 = b
    w = 1
  x0 = 1
  x1 = 0
  y0 = 0
  y1 = 1
# Extended division chain
  while r1 > 0:
    q = r0//r1
    r = r0 - q * r1
    x = x0 - q* x1
    y = y0 - q * y1
# Here we have r0 = |a|*x0+|b|*y0, r1 = |a|*x1+|b|*y1, r = |a|*x+|b|*y.
    r0 = r1
    r1 = r
    x0 = x1
    x1 = x
    y0 = y1
    y1 = y
# Finalization
  d = r0
  x = v * x0
  y = w * y0
  return [d,x,y]
```

```
def D2q(q):
  """Generate list of all solutions with 2-element support."""
  outlist = []
  nullvec = [0]*q
  for i in range(q):
    for j in range(q):
      x = nullvec[0:]
      y = nullvec[0:]
      d = eEuclid(i+1,j+1)[0]
      x[i] = (j+1)//d
      y[j] = (i+1)//d
      x.extend(y)
      outlist.append(x)
  return outlist
```

A.2 Solving (E)—Naive Algorithm

See Section 1.

```
### Process command line parameters
coeff = sys.argv[1:]
ll = len(coeff)            # = dimension
for i in range(ll):
  coeff[i] = int(coeff[i]) # strings to numbers
A = 0                      # max of pos coeff
B = 0                      # -min of neg coeff
for a in coeff:
  if a > A:
    A = a
  elif a < -B:
    B = -a
M = A
if B > A:
  M = B
sollist = []
dlist = dlist0(ll,M)
nullvec = [0]*ll
dlist.remove(nullvec)
for xlist in dlist:
  if chkdio(coeff,xlist):
    sollist.append(xlist)
redlist = minelts(sollist)
print(redlist)
```

A.3 Solving (E) More Efficiently

See Section 3.

```
### Process command line parameters
coeff = sys.argv[1:]
ll = len(coeff)  # = dimension
A = 0       # max of positive coefficients
B = 0       # -min of negative coefficients
posit = []  # list of indices of positive coefficients
negat = []  # list of indices of negative coefficients
zeros = []  # list of indices of zero coefficients
for i in range(ll):
  coeff[i] = int(coeff[i]) # strings to numbers
  if coeff[i] > 0:
    posit.append(i)
    if coeff[i] > A:
      A = coeff[i]
  elif coeff[i] < 0:
    negat.append(i)
    if coeff[i] < -B:
       B = -coeff[i]
  else:
    zeros.append(i)
lx = len(posit)
ly = len(negat)
lz = len(zeros)

### Register the trivial solutions
sollist = []
nullvec = [0]*ll
for iz in zeros:      # generate unit vector
  zsol = nullvec[:]
  zsol[iz] = 1
  sollist.append(zsol)
```

```
### Construct all candidate vectors in the critical domain D
xlist = dlist1(lx,B)
xnullvec = [0]*lx
if xlist != []:
  xlist.remove(xnullvec)
ylist = dlist1(ly,A)
ynullvec = [0]*ly
if ylist != []:
  ylist.remove(ynullvec)
dlist = []           # combine x- and y-vectors
for x in xlist:
  vec = nullvec[:]
  for i in range(lx):
    vec[posit[i]] = x[i]
  for y in ylist:
    xy = vec[:]
    for j in range(ly):
      xy[negat[j]] = y[j]
    dlist.append(xy)

### Single out the solutions, then the minimal solutions
for xyz in dlist:
  if chkdio(coeff,xyz):
    sollist.append(xyz)
redlist = minelts(sollist)
print(redlist)
```

A.4 Improved Algorithm Using Sissokho's Bound

See Section 3

```
### Process command line parameters
coeff = sys.argv[1:]
ll = len(coeff)
A = 0       # max of positive coefficients
B = 0       # -min of negative coefficients
posit = []  # list of indices of positive coefficients
negat = []  # list of indices of negative coefficients
zeros = []  # list of indices of zero coefficients
for i in range(ll):
  coeff[i] = int(coeff[i]) # strings to numbers
  if coeff[i] > 0:
    posit.append(i)
    if coeff[i] > A:
      A = coeff[i]
  elif coeff[i] < 0:
    negat.append(i)
    if coeff[i] < -B:
       B = -coeff[i]
  else:
    zeros.append(i)
lx = len(posit)
ly = len(negat)
lz = len(zeros)

### Register the trivial solutions
sollist = []
nullvec = [0]*ll
for iz in zeros:      # generate unit vector
  zsol = nullvec[:]
  zsol[iz] = 1
  sollist.append(zsol)
```

```
### Construct all candidate vectors in the critical domain D
xlist = dlist1(lx,B)
xnullvec = [0]*lx
if xlist != []:
  xlist.remove(xnullvec)
Alist = []            # list of corresponding A-values
for x in xlist:
  xsum = sum(x)
  asum = 0
  for i in range(lx):
    asum = asum + coeff[posit[i]] * x[i]
  asum = asum//xsum
  if asum < A:
    Alist.append(asum)
  else:
    Alist.append(A)

ylist = dlist1(ly,A)
ynullvec = [0]*ly
if ylist != []:
  ylist.remove(ynullvec)
Nlist = []             # list of corresponding 1-norms
for y in ylist:
  ysum = sum(y)
  Nlist.append(ysum)

dlist = []           # combine x- and y-vectors
for ix in range(len(xlist)):
  x = xlist[ix]
  Ax = Alist[ix]
  vec = nullvec[:]
  for i in range(lx):
    vec[posit[i]] = x[i]
  for iy in range(len(ylist)):
    y = ylist[iy]
    Ny = Nlist[iy]
    if Ny <= Ax:
      xy = vec[:]
      for j in range(ly):
        xy[negat[j]] = y[j]
      dlist.append(xy)
```

```
### Single out the solutions, then the minimal solutions
for xyz in dlist:
  if chkdio(coeff,xyz):
    sollist.append(xyz)
redlist = minelts(sollist)
print(len(redlist), "indecomposable solutions:")
print(redlist)
```

A.5 Solving the Standard Equation

See Section 8

```
### Process command line parameters
q = int(sys.argv[1])
solulist = []
stoplist = []
nullvec = [0]*q
if q == 1:
  solulist.append([1,1])
  print(solulist)
  exit()

# At this point q >= 2
# First compute the solutions with 2-element support.
twolst = D2q(q)
solulist.extend(twolst)
if q == 2:
  print(solulist)
  exit()

# At this point q >= 3
qhalf = q//2
for s in range(1,qhalf+1):
  sublist = subsets(q,s)
  for sset in sublist:
    t0 = s
    if t0 == 1:
      t0 = 2
    for t in range(t0,q-s+1):
      sublist1 = subsets(q,t)
      sublist2 = []
      for tset in sublist1:
        meet = False
        for el in sset:
          if el in tset:
            meet = True
        if not(meet):
          sublist2.append(tset)
          supp = sset[:]
          for tt in tset:
            supp.append(tt+q)
          include = True
```

```
for stopset in stoplist:
  if subset(stopset,supp):
    include = False
if include:
  ssum = sum(sset)
  tsum = sum(tset)
  if ssum == tsum:    # Flat solution detected
    xy = [0]*(2*q)
    for index in supp:
      xy[index-1] = 1
    if not(xy in solulist):
      solulist.append(xy)
    stoplist.append(supp)
    suppcpl = tset[:]
    for ss in sset:
      suppcpl.append(ss+q)
    yx = [0]*(2*q)
    for index in suppcpl:
      yx[index-1] = 1
    if not(yx in solulist):
      solulist.append(yx)
    stoplist.append(suppcpl)
  else: # Now construct solutions supported by (sset, tset).
    xlist = dlist2(len(sset),q-1)
    ylist = dlist2(len(tset),q-1)
    for z in xlist:
      x = nullvec[:]
      zc = z[:]
      for ind in sset:
        t = zc.pop(0)
        x[ind-1] = t
      for w in ylist:
        y = nullvec[:]
        wc = w[:]
        for ind in tset:
          u = wc.pop(0)
          y[ind-1] = u
        xy = x[:]
        xy.extend(y)
        sol = stddio(xy)
        if sol:
          if not(xy in solulist):
            solulist.append(xy)
```

```
                    yx = y[:]
                    yx.extend(x)
                    if not(yx in solulist):
                      solulist.append(yx)
redlist = minelts(solulist)
print(len(redlist), "indecomposable solutions:")
print(redlist)
```

References

[1] M. Adi, C. Kirchner: AC-unification race: the system solving approach, implementation and benchmarks. J. Symbolic Comput. 14 (1992) 51–70.

[2] W. Bruns, B. Ichim: Normaliz: Algorithms for affine monoids and rational cones. J. Algebra 324 (2010), 1098–1113.

[3] M. Clausen, A. Fortenbacher: Efficient solution of linear Diophantine equations. J. Symbolic Comput. 8 (1989), 201–216.

[4] P. Diaconis, R. Graham, B. Sturmfels: Primitive partition identities. In: D. Miklós, V. T. Sós, T. Szönyi (eds.): *Combinatorics, Paul Erdős is Eighty*, Volume 2. Bolyai Society, Budapest 1995, 1–20.

[5] E. Domenjoud: Solving systems of linear Diophantine equations: an algebraic approach. UNIF'90, International Workshop on Unification, Leeds UK July 1990.

[6] E. B. Elliott: On linear homogeneous Diophantine equations. Quart. J. Pure Appl. Math.34 (1903), 248–377.

[7] M. Filgueiras, A. P. Tomás: A fast method for finding the basis of non-negative solutions to a linear Diophantine equation. J. Symbolic Comput., 19 (1995), 507–526.

[8] J. H. Grace, A. Young: *The Algebra of Invariants.* Cambridge University Press 1903.

[9] A. Geroldinger, F. Halter-Koch: Non-unique factorizations: a survey. In: J. W. Brewer, S. Glaz, W. Heinzer, B. Olberding (Eds.): *Multiplicative Ideal Theory in Commutative Algebra.* Springer, New York 2006, 207–226.

[10] E. Q. Halleck: *Magic square subclasses as linear Diophantine systems.* PhD Thesis, Univ. of California, San Diego 2000.

[11] G. H. Hardy, E. M. Wright: *Zahlentheorie.* Oldenbourg, München 1958. = *Introduction to the Theory of Numbers.* Oxford Univ. Press 1938, 1954.

[12] J. C. Harris, D. L. Wehlau: Non-negative integer linear congruences. Indag. Math. 17 (2006), 37–44.

[13] M. Henk, R. Weismantel: On minimal solutions of linear Diophantine equations. Beitr. Alg. Geom. 41 (2000), 49–55.

[14] G. Huet: An algorithm to generate the basis of solutions to homogeneous linear Diophantine equations. Information Processing Letters 7 (1978), 144–147.

[15] J. L. Lambert: Une borne pour les générateurs des solutions entières positives d'une équation diophantienne linéaire. C.R. Acad. Sci. Paris Série I, 305 (1987), 39–40.

[16] D. Papp, B. Vizvári: Effective solution of linear Diophantine equation systems with an application in chemistry. Rutcor Research Report 28-2004.

[17] D. V. Pasechnik: On computing Hilbert bases via the Elliott-MacMahon algorithm. Theor. Comp. Sc. 263 (2001), 37–46.

[18] K. Pommerening: A remark on subsemigroups (Dickson's lemma). Online: `http://www.staff.uni-mainz.de/pommeren/MathMisc/Dickson.pdf`

[19] K. Pommerening: Stirling's formula. Online: `http://www.staff.uni-mainz.de/pommeren/MathMisc/Stirling.pdf`

[20] K. Pommerening: Some remarks on the complexity of invariant algebras. Online: `http://www.staff.uni-mainz.de/pommeren/MathMisc/ComplInv.pdf`

[21] K. Pommerening: The indecomposable solutions of linear congruences. Online: `http://www.staff.uni-mainz.de/pommeren/MathMisc/LinCong.pdf`

[22] L. Pottier: Minimal solution of linear Diophantine systems: Bounds and algorithms. In: Proc. 4th Intern. Conf. on Rewriting Techn. and Appl., Como, Italy (1991), 162–173.

[23] A. S. Rybakov: Positive integer solutions of systems of linear equations and polynomials of small weight divisible by $(1 - x)^r$. Discrete Math. Appl. 24 (2014), 175–183.

[24] P. A. Sissokho: A note on minimal zero-sum sequences over $\mathbb{Z}$. Acta Arithmetica 166 (2014), 279–288.

[25] R. Stanley: Linear homogeneous Diophantine equations and and magic labelings of graphs. Duke Math. J. 40 (1973), 607–632.

[26] R. Stanley: Linear Diophantine equations and local cohomology. Invent. math. 68 (1982), 175–193.

[27] M. E. Stickel: A complete unification algorithm for associative-commutative functions. Proc. 4th Internat. Joint Conf. on Artificial Intelligence, Tblissi 1975.