

Convex Polycubes: Counting and Growth Constants^{*}

Gill Barequet

Dept. of Computer Science
Technion—Israel Institute of Technology
Haifa 32000, Israel
`barequet@cs.technion.ac.il`

Abstract. A d -dimensional polycube is a connected set of d -dimensional cubes, where connectivity is through $(d-1)$ -dimensional faces. Fixed polycubes are considered distinct if they differ in shape or orientation. In this paper we consider fixed *convex* polycubes according to two definitions of convexity. We investigate algorithms for counting convex polycubes and estimate their asymptotic growth constants.

Keywords: Polyominoes, polycubes, lattice animals.

1 Introduction

A d -dimensional *polycube* of size n is a connected set of n cells on the cubical lattice \mathbb{Z}^d , where connectivity is through $(d-1)$ -dimensional faces. Two *fixed* polycubes are equivalent if one can be transformed into the other by a translation. We consider here only fixed polycubes, so we will omit this adjective throughout the paper. The study of polycubes began in the 1950s in statistical physics [11, 33], where they are usually called *lattice animals*.

Counting polyominoes (polycubes in two dimensions) is a long-standing problem. The number of polyominoes, $A_2(n)$, is currently known up to $n = 56$ [22]. Tabulations of counts of polycubes in higher dimensions appear in the mathematics literature [2, 26, 27] as well as in the statistical-physics literature [16, 18, 29].

The growth constant (asymptotic growth rate) of polyominoes has also attracted much attention. Klarner [23] showed that $\lambda_2 := \lim_{n \rightarrow \infty} \sqrt[n]{A_2(n)}$ exists. The convergence of $A_2(n+1)/A_2(n)$ to λ_2 (as $n \rightarrow \infty$) was proven only three decades years later by Madras [28] using a novel pattern-frequency argument. The best-known lower [8] and upper [24] bounds on λ_2 are 3.9801 and 4.6496,¹ respectively. It is widely assumed (see, e.g., [16, 18]) that $\lambda_2 \approx 4.06$, and the best estimate, $\lambda_2 = 4.0625696 \pm 0.0000005$, is currently due to Jensen [22]. In $d > 2$ dimensions, λ_d , the growth constant of the number of polycubes, also exists [28]. It was proven [6] that $\lambda_d = 2ed - o(d)$; moreover, λ_d was estimated

^{*} This research was supported by Loewengart Research Fund.

¹ In the informal note [5] we showed that $\lambda_2 > 4.0025$.

at $(2d-3)e + O(1/d)$. It was also shown [4] that λ_d^T , the growth constant of *tree* polycubes (animals on the cubical lattice, whose cell-adjacency graphs do not contain cycles) is $2ed - o(d)$ as well, and was estimated at $(2d-3.5)e + O(1/d)$.

Convex polyominoes also attracted some attention in the literature, with no attention given to convex polycubes in more than two dimensions. Generating functions for the sequences enumerating convex polyominoes, as well as counts of these polyominoes, were considered according to the area [10, 14], perimeter [9, 12, 13, 15, 20], and the extents of the axis-parallel bounding boxes of the polyominoes [19].

Let us start with defining convex polycubes.

Definition 1. A polycube is weakly-convex if any axis-parallel line intersects with the polycube in at most one continuous sequence of cubes.

This is a natural definition for polyominoes (in the plane), however, in three or more dimensions it allows convex polycubes with holes (having a positive genus in three dimensions). Consider, for example, the toroidal polycube shown in Figure 1.

In contrast, *strongly-convex* polyominoes are defined in a recursive manner:

Definition 2.

1. All 1-dimensional polycubes are strongly-convex; and
2. A d -dimensional polycube P (for $d > 1$) is strongly-convex if every $(d-1)$ -dimensional slice of P (the intersection of P with a $(d-1)$ -dimensional sublattice spanned by $d-1$ major axes) is a polycube that is strongly-convex.²

The two definitions are equivalent for polyominoes but not for polycubes in $d \geq 3$ dimensions.

In this paper we investigate algorithms for computing the number of weakly and strongly-convex polycubes, and analyze their growth constants. Throughout the paper we denote by $A_d^K(n)$ (resp., $A_d^C(n)$) the number of weakly- (resp., strongly-) convex polycubes of size n in d dimensions. Similarly, we denote by λ_d^K and λ_d^C the growth constants of weakly- and strongly-convex polycubes, respectively. Obviously, we have $A_2^K(n) = A_2^C(n)$ and $\lambda_2^K = \lambda_2^C$.

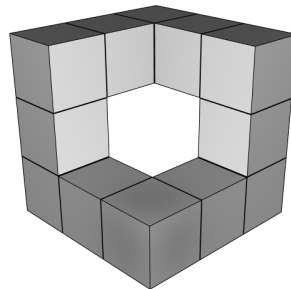


Fig. 1. A 3-dimensional weakly-convex polycube (which is not strongly-convex) with nonzero genus.

² This seemingly awkward wording (“is a polycube that is strongly-convex” instead of “is a strongly-convex polycube”) intends to forbid the case in which a slice is not a polycube at all, say, because it is not connected, but it fulfills the recursive condition. Such a situation occurs, for example, in the polycube shown in Figure 3: The top XY-parallel slice ($\square \sqcup \square$) is not connected and is, therefore, not a polycube.

2 Two Dimensions

As mentioned in the introduction, enumerating all polyominoes is a difficult task. In addition, a formula for the number of convex polyominoes seems to be out of reach, although Bousquet-Mélou and Fédou [10] provided a rather complex generating function for the enumerating sequence. However, a dynamic-programming (DP) procedure can count convex polyominoes quite efficiently.³ The procedure builds polyominoes in layers (horizontal sequences of squares), using a 5-dimensional DP state table. The five dimensions stand for the number of iteration (last two layers suffice), width of the last layer, total area (number of squares), and two boolean indicators stating whether or not the leftmost and/or rightmost cell of the polyomino has already been encountered. The program counts convex polyominoes up to size n in $\Theta(n^4)$ time.

The growth constant of convex polyominoes in the plane, $\lambda_2^C = \lambda_2^K$, is known for almost forty years. Klarner and Rivest [25] showed that this constant exists⁴ and provided almost-matching lower and upper bounds on it, both roughly equal 2.3091. Soon after, Bender [7] provided a precise computation of the same constant.

3 Counting in Higher Dimensions

3.1 Algorithm

Weakly-convex polycubes

It is impractical to generalize the DP procedure to $d \geq 3$ dimensions since it keeps information and counts for all $(d-1)$ -dimensional convex polycubes. Already for $d = 3$ we face two significant problems: The number of 2-dimensional convex polyominoes is exponential with n , and unlike 1-dimensional convex polycubes (which are merely “sticks”), we do not know how to parameterize convex polyominoes efficiently for using them as array indices.

Therefore, we turn to Redelmeier’s algorithm [32] (see Figure 2), the first-published procedure that counts polyominoes *without repetitions*. This is a graph-search procedure which counts all connected subgraphs containing a specific vertex (the “origin”). In fact, the algorithm can count animals on any lattice, in particular, polycubes in high dimensions; one needs only to fix the appropriate lattice graph; see [1] for a discussion of this issue. The algorithm starts with the initial cell (the origin) and “grows” animals sequentially, always taking new cells from the so-called “untried set” U which contains cells neighboring the current

³ An implementation in C of this procedure can be found in <http://www.cs.technion.ac.il/~barequet/polyominoes/convex-dp2d.c>. In the on-line encyclopedia of integer sequences, sequence A067675 [30], one can find a DP program by R.G. Spaans, which seems to be based on a similar idea.

⁴ More precisely, they considered the asymptotics of $\lim_{n \rightarrow \infty} \sqrt[n]{A_2^C(n)}$ and not of $\lim_{n \rightarrow \infty} (A_2^C(n+1)/A_2^C(n))$. Naturally, the existence of the latter limit implies the existence of the former limit (and in this case the two limits are equal).

Initialize the parent to be the empty polycube, and the untried set to contain only the origin. [Mark all cells except the origin as *free*; mark the origin as *reachable*.] The following steps are repeated until the untried set is exhausted.

1. [Let P denote the current polycube.] Remove an arbitrary element $[c]$ from the untried set.
2. Place a cell at this point. [$P := P \cup \{c\}$; Mark c as *occupied*.]
3. Count this new polycube $[P]$.
4. If the size [of P] is less than n :
 - (a) Add *new neighbors* to the untried set. [Mark these cells as *reachable*.]
 - (b) Call this algorithm recursively [i.e., jump to Step 1] with the new parent being the current polycube, and the new untried set being a copy of the current one.
 - (c) Remove the new neighbors from the untried set. [Mark these cells as *free*.]
5. Remove newest cell. [$P := P \setminus \{c\}$; Return to Step 4(c).]

Fig. 2. Redelmeier’s algorithm copied verbatim from [32, p. 196] except that all references to “polyominoes” were changed to “polycubes.” Our clarifications are surrounded by square brackets.

polycube P . The correctness of the algorithm relies crucially on the definition of cells that become “new neighbors” due to adding a cell c to P in Step 4(a): These are cells that (a) did not belong to any polycube in the computation tree that led to P ; *and* (b) become neighbors of P only due to c , not being neighbors of any other cell of P . To enforce these conditions, cells are first marked as “free” (in the initialization phase), turn “reachable” when they become neighbors of P and are therefore put in the untried set in preparation for the recursion (Step 4(a)), turn “occupied” when they are added to P (Step 2), become “free” again when the recursion unfolds (Step 4(c)) in order to allow their re-insertion in sibling branches of the current state of the computation, and are flagged as “done” when they are removed from P (Step 5) in order to forbid their re-insertion when the recursion unfolds to the parent of P . Condition (b) above means that a cell c' is a new neighbor of P when cell c is added to P only if c' is a neighbor of $P \cup \{c\}$ but not of P . To the best of our knowledge, a complete proof of the correctness of this procedure was first given in [2].

Our goal is, then, to modify the algorithm to count only convex polycubes. A naive change to the algorithm would be to carry out the entire procedure, preceding Step 3 by checking that the newly-generated polycube is (weakly or strongly) convex, but this would not take advantage of the fact that the number of convex polycubes is only a vanishing fraction of the number of all polycubes.

Trimming entire subtrees of the computation is not simple: Since the algorithm does not care about the order in which cells are added and/or removed from U , nonconvex polycubes (according to either definition) can be created in the course of building convex polycubes, and so one cannot simply skip the re-

cursive call in Step 4(b) in case the polycube generated in Step 3 is not convex. One may try to solve this problem by implementing U as a queue, and then in case the addition of a new cell c (Step 2) to P violates its convexity, pushing c back to the end of the queue. Unfortunately, this idea fails: There are situations in which, due to the order in which cells were previously processed, no single cell *currently* in U can be added to P .

However, a simple modification to the algorithm makes it suitable for counting *weakly*-convex polycubes: One needs to process the cells in U in lexicographic order, and then precede Step 3 by checking that the newly-generated polycube is weakly-convex. That is, replace Step 1 by

1'. Remove the lexicographically-smallest element c from the untried set.

and Step 3 by


```

3'. If the new polycube  $P := P \cup \{c\}$  is weakly-convex Then
    Count  $P$ .
Else
    Remove  $c$  from  $P$  and mark  $c$  as done.
    Go to Step 1.
End if

```

Theorem 1. *The modified algorithm of Redelmeier (with Steps 1' and 3') counts correctly all weakly-convex polycubes.*

Proof. Step 1' only imposes an order on the processing of cells in the untried set, which is insignificant in the original algorithm. Step 3' simply trims branches of the computation which go through non-weakly-convex polycubes. Therefore, no new polycubes are counted, no polycubes are counted more than once, and we only need to show that no weakly-convex polycubes are missed.

Consider a situation in which Step 3' prohibits the recursive call (because the current polycube is not weakly-convex), which results in erroneously not counting a weakly-convex polycube had the recursion gone down. This can only happen in situations in which the current polycube contains a pattern of the form . The pattern may be much more complex, but it has to contain two axis-aligned but nonconsecutive cells. Assume without loss of generality that these two cells are found along a line parallel to the x -axis, and denote the one with the smaller (resp., larger) x coordinate by c_1 (resp., c_3). To show a contradiction, assume that the newly added cell is either c_1 or c_3 , and the recursive computation tree will supposedly add to the polycube all the empty cells between them (two cells in this example) so as to eventually create a weakly-convex polycube. However, this is not possible. Assume that c_1 is added to the polycube P before c_3 . In this case denote by c_2 the *leftmost* of all empty cells between c_1 and c_3 . Then, c_2 becomes (or is already) a neighbor of P , that is, in the untried set U , and thus c_2 will be added to P before c_3 due to Step 1', which is a contradiction. The case in which c_3 is added to P before c_1 is a bit more involved. When c_3 is

added to P , cell c_1 cannot be in U , otherwise it was added to P before c_3 . In this case denote by c_2 the *rightmost* of all empty cells between c_1 and c_3 . When c_3 is added to P , cell c_2 is not in U for the same reason (otherwise it would be added to P before c_3). However, the addition of c_3 to P results in adding c_2 to U (since c_2 becomes a new neighbor of P), while c_1 is still not in U . Thus, cell c_2 will be added to P before c_1 , which is, again, a contradiction. \square

Strongly-convex polycubes

Unfortunately, the above modifications and the proof of correctness in Theorem 1 do not hold if in the first line of Step 3' we replace “weakly-convex” by “strongly-convex.” Consider, for example, the polycube shown in Figure 3, assuming that it is lying in the positive octant with the lowest possible coordinates. This polycube, P , is not strongly-convex, but adding to it the cell $c = (1, 1, 1)$ would make it strongly-convex, while at the same time c is lexicographically bigger than all other cells in P . Nevertheless, since every strongly-convex polycube is also weakly-convex, the following version of Step 3 will surely count strongly-convex polycube:

```

3''. If the new polycube  $P := P \cup \{c\}$  is weakly-convex Then
    If  $P$  is strongly-convex then count  $P$ .
Else
    Remove  $c$  from  $P$  and mark  $c$  as done.
    Go to Step 1.
End if

```

3.2 Analysis

Weakly-convex polycubes

The running time of Redelmeier’s original algorithm is proportional to the number of counted polycubes, which is known to be asymptotically exponential with the size of the polycubes.⁵ (See [2] for a more accurate analysis which also takes into account the effect of the dimension d .)

The effect of changing Step 1 to Step 1' depends on the implementation of the untried set, but in the worst case it would add a negligible factor of $O(\log(nd) \log d)$ to the running time of a single operation on U , needed to maintain it sorted lexicographically. This is because a polycube can have $O(nd)$ neighboring cells,⁶ and $O(\log d)$ time is needed to access a single cell or to compare two cells.

⁵ In two dimensions, the base of the exponent is the famous Klarner’s constant $\lambda_2 \approx 4.06$.

⁶ The maximum possible number of cells neighboring a d -dimensional polycube of size n , $2dn - 2(n - 1) = 2((d - 1)n + 1) = O(dn)$, is obtained for a “stick”—a sequences of cubes aligned with one of the major axes.

The effect of Step 3' is more interesting. The number of times the weak-convexity check is performed per counted polycube is obviously $O(dn)$, which is also a negligible factor relative to the exponential number of polycubes. Step 3'' only prunes the strongly-convex polycubes out of the weakly-convex ones. Hence, in both versions, the number of traversed polycubes is that of weakly-convex polycubes.

We focus, then, on the efficiency of a single convexity test in Step 3' or 3''. For weakly-convex polycubes, a simple stick-maintenance trick pays off well, especially in high dimensions.

Theorem 2. *One can maintain a data structure which supports the manipulation of weakly-convex polycubes of size n in d dimensions in $\Theta(d)$ time per operation, using $\Theta(dn^{d-1})$ space. Alternatively, one can maintain a data structure which performs each operation in $O(d \log n)$ time, using only $O(dn)$ space.*

Proof. Weak-convexity of a polycube P means that any axis-parallel line intersects P in at most one continuous range of cells. Hence, we maintain a data structure which keeps these ranges of cells of P . In the first variant we maintain intersection ranges for the entire box that contains all possible polycubes. Each such range has $d-1$ fixed coordinates and two coordinates (minimum and maximum) in the remaining axis. By maintaining all these ranges, checking whether the addition of a new cell to P maintains its weak-convexity, and if so, adding it, can be done in $\Theta(d)$ time, *irrespective* of n , the maximum size of P . For each dimension we need to maintain $\Theta(n^{d-1})$ ranges. Therefore, the total consumption of space is $\Theta(dn^{d-1})$.

However, we can trade time for space by maintaining only the $O(dn)$ ranges associated with the actual axis-parallel lines that intersect the polycube at a given time. In this variant the running-time cost of checking and adding a cell to the polyomino will slightly go up to $O(d \log n)$ (the logarithmic factor is due to locating the exact range in a specific dimension), while memory consumption will drop to $O(dn)$. \square

In practice, one can use the second approach but store all the ranges in a hashing table, effectively reducing the expected running time per operation to $\Theta(d)$.

Strongly-convex polycubes

Testing a polycube for strong convexity is more delicate. A polycube is connected by definition, but its slices need not be connected. It follows from the definition that one-dimensional slices of a weakly-convex polycube are connected. A close observation of the recursive condition for strong-convexity reveals, after accumulating the conditions from all levels of the recursion, that the only difference between weak- and strong-convexity is that the latter requires that all slices, of all dimensions, are valid polycubes. We formulate this idea in the following lemma.

Lemma 1. *A polycube P is strongly-convex if and only if P is weakly-convex and all slices of P (of all dimensions) are connected.*

The above lemma gives us an efficient algorithmic tool for checking strong-convexity of polycubes. In a small dimension d , it makes sense to maintain the connectivity graphs of all slices. Although there can be plenty of them, the addition or removal of a *single* cell c affects only a subset of slices (those that contain c) whose cardinality is $\sum_{i=1}^{d-1} \binom{d}{i} = 2^d - 2$, a term which depends on d but not on n . Then, manipulating a single connectivity graph can be done in $O(d \log^c n)$ time [21, 34], for a small constant $c > 0$. In summary:

Theorem 3. *One can maintain data structures for testing the connectivity of all slices (of all dimensions) of a polycube of size n in d dimensions (to support the manipulation of strongly-convex polycubes) in $O(2^d d \log^c n)$ time (for a small constant $c > 0$).*

3.3 Experimental results

We implemented the two variants of the algorithm described above. The counts of 3-dimensional weakly-convex polycubes up to size 13 are 1, 3, 15, 86, 522, 3241, 20256, 126520, 787224, 4873750, 30019640, 184034688, 1123606568 (five new terms with respect to the current list in sequence A191148 [30]). We also counted 3-D strongly-convex polycubes up to size 13: 1, 3, 15, 86, 498, 2797, 14988, 76216, 368040, 1694142, 7471574, 31736892, 130483178 (sequence does not currently exist in [30]).

4 Growth Constant

In the previous section we described variants of Redelmeier’s algorithm for counting weakly- and strongly-convex polycubes. The main factor in the running time of these variants is the number of weakly-convex polycubes, which is roughly exponential with the size of the polycubes. Let us, therefore, estimate the base of this exponent—the growth constant of convex polycubes.

4.1 Weakly-Convex Polycubes

We first consider *weakly*-convex polycubes. Methods similar to those of [4, 6] show rigorously that the order of λ_d^K , the growth constant of weakly-convex polycubes in d dimensions, is $2ed$, as $d \rightarrow \infty$ (details are provided in the full version of the paper), and our goal is to find the constant term in λ_d^K . We estimate λ_d^K at $(2d-3)e$, which is the same as λ_d up to an $O(1/d)$ difference. We say that a polycube is *proper* in d dimensions if the convex hull of the centers of all its cubes is d -dimensional, and, following Lunnon [27] and others, denote by $DX(n, d)$ the number of polycubes of size n that are proper in d dimensions. In addition, we denote by $DK(n, d)$ (resp., $CK(n, d)$) the number of proper (resp.,

all) weakly-convex polycubes of size n in d dimensions. Finally, we define a one-variable function $\text{CK}_n(d) = \text{CK}(n, d)$ for the case in which the value of n is fixed. Our goal, then, is to estimate $\lambda_d^K := \lim_{n \rightarrow \infty} \text{CK}(n+1, d)/\text{CK}(n, d)$.⁷ Here is a high-level description of the computation, which is similar in nature to the one carried out in [6] (for general polycubes) and in [4] (for tree polycubes):

1. Find formulae for $\text{DK}(n, n-1)$ and $\text{DK}(n, n-2)$;
2. Show that for any fixed $n > 0$, $\text{CK}_n(d)$ is a polynomial in d of degree $n-1$;
3. Compute the general terms for the two leading coefficients of $\text{CK}_n(d)$; and
4. Compute the beginning of a formal long polynomial division $\text{CK}_{n+1}(d)/\text{CK}_n(d)$.

Step 1. Formulae for $\text{DK}(n, n-1)$ and $\text{DK}(n, n-2)$ follow from the following observation, which simply tells us that there are no non-weakly-convex polycubes of size n that are proper in $n-1$ or $n-2$ dimensions.

Theorem 4. *All polycubes of size n , which are proper in $n-1$ or $n-2$ dimensions, are also weakly-convex.*

Proof. In order for a polycube P to be non-weakly-convex, two non-adjacent cubes of P need to lie on the same axis-parallel line. The cell combination that violates weak convexity and minimizes k (the difference between the size of the polycube and its dimensionality) is a “U” structure ($\begin{smallmatrix} \square & \square \\ \square & \square \end{smallmatrix}$), a structure consisting of five cubes and spanning only two dimensions. Hence, a polycube of size n , which has a “U” as a substructure (or any other structure which violates weak convexity), cannot span more than $n-3$ dimensions, and, thus, cannot be proper in $n-2$ (let alone, $n-1$) dimensions. \square

The next claim follows immediately.

Corollary 1. *For $k = 1, 2$, we have $\text{DK}(n, n-k) = \text{DX}(n, n-k)$.* \square

Formulae for $\text{DX}(n, n-1)$ and $\text{DX}(n, n-2)$ have been known for many years [17, 18, 31]. (The latter was proven rigorously only recently [6].) By Corollary 1 we know that $\text{DK}(n, n-1)$ and $\text{DK}(n, n-2)$ have the same formulae, respectively. Therefore, $\text{DK}(n, n-1) = \text{DX}(n, n-1) = 2^{n-1}n^{n-3}$ and $\text{DK}(n, n-2) = \text{DX}(n, n-2) = 2^{n-3}n^{n-5}(n-2)(2n^2 - 6n + 9)$.

Step 2. It is easy to prove, as follows, that for any fixed value of n , the function $\text{CK}_n(d)$ is a polynomial in d of degree $n-1$. Consider a variant of Lunnon’s partition formula [4, 6, 27] that is adapted to weakly-convex polycubes:

$$\text{CK}_n(d) = \text{CK}(n, d) = \sum_{i=0}^{\min(n-1, d)} \binom{d}{i} \text{DK}(n, i).$$

Indeed, every polycube of size n in d dimensions is proper in some dimension $0 \leq i \leq d$, and we also have that $\text{DK}(n, i) = 0$ for $i > n-1$ since a polycube of size

⁷ Again, if $\lim_{n \rightarrow \infty} A_d^K(n+1)/A_d^K(n)$ and $\lim_{n \rightarrow \infty} A_d^C(n+1)/A_d^C(n)$ exist, then they are also equal to $\lim_{n \rightarrow \infty} \sqrt[n]{A_d^K(n)}$ and $\lim_{n \rightarrow \infty} \sqrt[n]{A_d^C(n)}$, respectively.

n cannot span more than $n-1$ dimensions. Hence, the summation stops at $i = \min(n-1, d)$. In addition, any such polycube gives rise to $\binom{d}{i}$ different polycubes in the d -dimensional cubical lattice. When n is fixed, the terms $\text{DK}(n, i)$ are constant, and so we are left with the variable term $\binom{d}{i}$. The maximum value that i takes is $n-1$, implying the claim.

Step 3. Knowing the formulae for $\text{DK}(n, n-1)$ and $\text{DK}(n, n-2)$, we are able to compute the two leading coefficients of $\text{CK}_n(d)$, $k_{n,0} = [d^{n-1}]\text{CK}_n(d)$ and $k_{n,1} = [d^{n-2}]\text{CK}_n(d)$.

Theorem 5. $k_{n,0} = 2^{n-1}n^{n-3}/(n-1)!$; $k_{n,1} = -3 \cdot 2^{n-3}n^{n-5}(2n-3)/(n-3)!$.

Proof. The two formulae are obtained by collecting terms from Lunnon's formula.

$$\begin{aligned} k_{n,0} &= \text{DK}(n, n-1)/(n-1)! = 2^{n-1}n^{n-3}/(n-1)!; \quad \text{and} \\ k_{n,1} &= \text{DK}(n, n-1) \frac{\sum_{i=0}^{n-2} (-i)}{(n-1)!} + \text{DK}(n, n-2) \frac{1}{(n-2)!} \\ &= 2^{n-1}n^{n-3} \frac{-1}{2(n-3)!} + 2^{n-3}n^{n-5}(n-2)(2n^2-6n+9) \frac{1}{(n-2)!} \\ &= -3 \cdot 2^{n-3}n^{n-5}(2n-3)/(n-3)!. \end{aligned}$$

In fact, the formulae for the first two coefficients are identical to the formulae for the respective coefficients of the polynomials related to all (not necessarily convex) polycubes [6]. \square

Step 4. As in the previous section, $\text{CK}_n(d) = k_{n,0}d^{n-1} + k_{n,1}d^{n-2} + \dots$. Let us write $\text{CK}_{n+1}(d)/\text{CK}_n(d) = f_1(n)d + f_2(n) + O(1/d)$. A long polynomial division procedure shows that $f_1(n) = 2((n+1)/n)^{n-2}$ and $f_2(n) = -3(n-1)(n+1)^{n-4}(2n^3+6n^2-5n-6)/(2n^n)$. Calculations show that $\lim_{n \rightarrow \infty} f_1(n) = 2e$ and $\lim_{n \rightarrow \infty} f_2(n) = -3e$. Hence, we conclude that $\lambda_d^K \sim (2d-3)e + O(1/d)$, and thus λ_d^K tends to $(2d-3)e$ as $d \rightarrow \infty$.

4.2 Strongly-Convex Polycubes

Methods as in [4, 6] also show rigorously that the order of λ_d^C , the growth constant of strongly-convex polycubes in d dimensions, is $2ed$, as $d \rightarrow \infty$. To reaffirm this conclusion, we follow the same four steps as in the previous section. We denote by $\text{DC}(n, d)$ (resp., $\text{CC}(n, d)$) the number of proper (resp., all) strongly-convex polycubes of size n in d dimensions, and define a one-variable function $\text{CC}_n(d) = \text{CC}(n, d)$ for the case in which the value of n is fixed.

Step 1.

Theorem 6. *All polycubes of size n , which are proper in $n-1$ dimensions, are also strongly-convex.*

Proof. In order for a valid polycube to be non-strongly-convex, a slice of it, which is not a valid polycube, should be made connected by an additional set of cubes in the complementing dimensions. The cardinality of such a set is at least 3: the length of a path in one additional dimension, which makes the entire polycube connected. Using more dimensions will require even more cubes in the path. Hence, the minimum value of k , for which a polycube of size n which is proper in $n-k$ dimensions is not strongly-convex, is 2. Such a structure is shown in Figure 3. This structure is made of 5 cubes and it spans 3 dimensions, and up to rotations and mirroring it is the only non-strongly-convex polycube of size 5. This implies that a polycube of size n , which is proper in $n-1$ dimensions, *must* be strongly-convex. \square

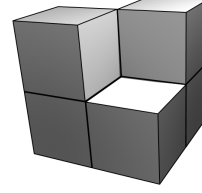


Fig. 3. The 5-cube structure contained in any minimum- k non-strongly-convex polycube.

Corollary 2. $DC(n, n-1) = DX(n, n-1) = 2^{n-1}n^{n-3}$. \square

Thus, we know only the formula for $DC(n, n-1)$ but not for $DC(n, n-2)$. Hence, we perform Steps 2–4 as above but with only one diagonal formula.

Step 2. In the same manner as in the previous section, it is shown that for any fixed value of n , the function $CC_n(d)$ is a polynomial in d of degree $n-1$.

Step 3. Knowing the formula for $DC(n, n-1)$, we are able to compute the leading coefficient of the one-variable function (for a fixed value of n) enumerating polycubes as a function of the dimension, $c_{n,0} = [d^{n-1}]CC_n(d)$.

Theorem 7. $c_{n,0} = 2^{n-1}n^{n-3}/(n-1)!$. \square

Step 4. As above, we write $CC_{n+1}(d)/CC_n(d) = f_1(n)d + O(1)$. The same long polynomial division procedure shows that $f_1(n) = 2((n+1)/n)^{n-2}$ and, again, $\lim_{n \rightarrow \infty} f_1(n) = 2e$. Hence, we conclude that $\lambda_d^C \sim 2de + O(1)$, and thus λ_d^C tends to $(2d+L)e$, for some constant L as $d \rightarrow \infty$. (Obviously, by the computation in the previous section, we have that $L \leq -3$.)

4.3 Discussion

The constant term in the growth constant

Incidentally, $(2d-3)e$, the estimated growth constant of weakly-convex polycubes (up to an $O(1/d)$ term), identifies with the growth constant of all fixed polycubes [6]. Following closely the calculations reveal the reason for this: The ordinal number (i.e., the value of k) of the first diagonal formula that *differs* from the formula for all polycubes is the same as the location of the first term in the $(1/d)$ -expansion of the growth constant in question (namely, the coefficient of

$1/d^{k-2}$), that differs from the respective term in the expansion of λ_d . For weakly-convex polycubes, the first difference occurs at $k = 3$, hence the expansion of λ_d^C differs from that of λ_d starting at the $O(1/d)$ term. For strongly-convex polycubes, since we do not have the second diagonal formula, we know only the leading term $2de$ in the expansion of λ_d^C .

Order of limits

The constant λ_d^K (resp., λ_d^C) involves the limit $n \rightarrow \infty$, whereas the function $\text{CK}_n(d)$ (resp., $\text{CC}_n(d)$) is a polynomial in d . Thus, the correctness of the long-division procedure used above relies on the unproven (but widely used in the literature of statistical physics) assumption that the limits $n \rightarrow \infty$ and $d \rightarrow \infty$ can be exchanged in these cases and yield a converging series for the $1/d$ -expansion. For further details about this issue, the reader is referred to a thorough discussion by Gaunt and Peard [17, p. 7524] and to a review by Whittington and Soteros [35]. Note, however, that the correctness of the leading term ($2ed$) of the expansions does not rely on this assumption.

Fraction of weakly-convex polycubes

The estimation of λ_d^K is identical (up to the constant term) to that of the growth constant of all polycubes. Although both are similar to $(2d - 3)e$ as $d \rightarrow \infty$, the growth constant of d -dimensional weakly-convex polycubes (for a *fixed* d) is strictly less than that of all polycubes. Thus, the fraction of weakly-convex polycubes out of all polycubes vanishes as they grow in size. Obviously, the same holds for strongly-convex polycubes.

References

1. G. ALEKSANDROWICZ AND G. BAREQUET, Counting d -dimensional polycubes and nonrectangular planar polyominoes, *Int. J. of Computational Geometry and Applications*, 19 (2009), 215–229.
2. G. ALEKSANDROWICZ AND G. BAREQUET, Counting polycubes without the dimensionality curse, *Discrete Mathematics*, 309 (2009), 576–4583.
3. G. ALEKSANDROWICZ AND G. BAREQUET, Parallel enumeration of lattice animals, *Proc. 5th Int. Frontiers of Algorithmics Workshop*, Zhejiang, China, *Lecture Notes in Computer Science*, 6681, Springer-Verlag, 90–99, May 2011.
4. G. ALEKSANDROWICZ AND G. BAREQUET, The growth rate of high-dimensional tree polycubes, *Proc. 6th European Conf. on Combinatorics, Graph Theory, and Applications*, Budapest, Hungary, *Electronic Notes in Discrete Mathematics*, 38, 25–30, August–September 2011.
5. G. BAREQUET, M. SHALAH, AND G. ROTE, $\lambda > 4$, Ein Gedi, Israel, March 2014.
6. R. BAREQUET, G. BAREQUET, AND G. ROTE, Formulae and growth rates of high-dimensional polycubes, *Combinatorica*, 30 (2010), 257–275.
7. E.A. BENDER, Convex n -ominoes, *Discrete Mathematics*, 8 (1974), 219–226.
8. G. BAREQUET, M. MOFFIE, A. RIBÓ, AND G. ROTE, Counting polyominoes on twisted cylinders, *Integers: Elec. J. of Comb. Num. Theory*, 6 (2006), A22, 37 pp.
9. M. BOUSQUET-MÉLOU, A method for the enumeration of various classes of column-convex polygons, *Discrete Mathematics*, 154 (1996), 1–25.
10. M. BOUSQUET-MÉLOU AND J.-M. FÉDOU, The generating function of convex polyominoes: The resolution of a q -differential system, *Discrete Mathematics*, 137 (1995), 53–75.
11. S.R. BROADBENT AND J.M. HAMMERSLEY, Percolation processes: I. Crystals and mazes, *Proc. Cambridge Philosophical Society*, 53 (1957), 629–641.
12. S.J. CHANG AND K.Y. LIN, Rigorous results for the number of convex polyominoes on the square and honeycomb lattices, *J. Phys. A: Math. Gen.*, 21 (1988), 2635–2642.
13. A. DEL LUNGO, E. DUCHI, A. FROSINI, AND S. RINALDI, Enumeration of convex polyominoes using the ECO method, *Discrete Mathematics and Theoretical Computer Science*, AB (2003), 103–116.
14. A. DEL LUNGO, E. DUCHI, A. FROSINI, AND S. RINALDI, On the generation and enumeration of some classes of convex polyominoes, *The Electronic J. of Combinatorics*, 11 (2004), 46 pp.
15. M. DELEST AND X. VIENNOT, Algebraic languages and polyominoes enumeration, *Theoret. Computer Science*, 34 (1984), 169–206.
16. D.S. GAUNT, The critical dimension for lattice animals, *J. Phys. A: Math. Gen.*, 13 (1980), L97–L101.
17. D.S. GAUNT AND P.J. PEARD, $1/d$ -expansions for the free energy of weakly embedded site animal models of branched polymers, *J. Phys. A: Math. Gen.*, 33 (2000), 7515–7539.
18. D.S. GAUNT, M.F. SYKES, AND H. RUSKIN, Percolation processes in d -dimensions, *J. Phys. A: Math. Gen.*, 9 (1976), 1899–1911.
19. I.M. GESSEL, On the number of convex polyominoes, *Annales des sciences mathématiques du Québec*, 24 (2000), 63–66.
20. A.J. GUTTMANN AND I.G. ENTING, The number of convex polygons on the square and honeycomb lattices, *J. Phys. A: Math. Gen.*, 21 (1988), L467–L474.

21. M.R. HENZINGER AND V. KING, Randomized fully dynamic graph algorithms with polylogarithmic time per operation, *J. of the ACM*, 46 (1999), 502–516.
22. I. JENSEN, Counting polyominoes: A parallel implementation for cluster computing, *Proc. Int. Conf. on Computational Science*, III (Melbourne, Australia and St. Petersburg, Russia, 2003), *Lecture Notes in Computer Science*, 2659, Springer, 203–212.
23. D.A. KLARNER, Cell growth problems, *Canadian J. of Mathematics*, 19 (1967), 851–863.
24. D.A. KLARNER AND R.L. RIVEST, A procedure for improving the upper bound for the number of n -ominoes, *Canadian J. of Mathematics*, 25 (1973), 585–602.
25. D.A. KLARNER AND R.L. RIVEST, Asymptotic bounds for the number of convex n -ominoes, *Discrete Mathematics*, 8 (1974), 31–40.
26. W.F. LUNNON, Symmetry of cubical and general polyominoes, in: R.C. Read, ed., *Graph Theory and Computing*, Academic Press, New York, NY, 1972, 101–108.
27. W.F. LUNNON, Counting multidimensional polyominoes, *The Computer Journal*, 18 (1975), 366–367.
28. N. MADRAS, A pattern theorem for lattice clusters, *Annals of Combinatorics*, 3 (1999), 357–384.
29. J.L. MARTIN, The impact of large-scale computing on lattice statistics, *J. of Statistical Physics*, 58 (1990), 749–774.
30. The on-line encyclopedia of integer sequences, available at <http://www.oeis.org>.
31. P.J. PEARD AND D.S. GAUNT, $1/d$ -expansions for the free energy of lattice animal models of a self-interacting branched polymer, *J. Phys. A: Math. Gen.*, 28 (1995), 6109–6124.
32. D.H. REDELMEIER, Counting polyominoes: Yet another attack, *Discrete Mathematics*, 36 (1981), 191–203.
33. H.N.V. TEMPERLEY, Combinatorial problems suggested by the statistical mechanics of domains and of rubber-like molecules, *Phys. Review* 2, 103 (1956), 1–16.
34. M. THORUP, Near-optimal fully-dynamic graph connectivity, *Proc. 32nd Ann. ACM Symp. on Theory of computing*, Portland, OR, 343–350, May 2000.
35. S. WHITTINGTON AND C. SOTEROS, Lattice animals: Rigorous results and wild guesses, in: *Disorder in Physical Systems* (G. Grimmett and D. Welsh, eds.), Clarendon Press, Oxford, 1990, 323–335.