Convex Split-Face Polygon Cuts

October 14, 2022
Kate Novak

Tools

For more compact, finite summations, define a function S to be

$$S(x_1, ..., x_N) = \sum_{k=1}^{N} x_k \tag{1}$$

Introduction

There is a formula which counts the number of ways you can cut a convex polygon between all its faces. Given a convex polygon with N faces, this is

$$S(N-1, ..., 0) = \frac{N(N-1)}{2} \tag{2}$$

This captures the idea that from one face, you can cut to $N-1$ other faces, from the next you can cut to $N-2$ other faces if you're careful not to double count, and so on until at the last face where no new cuts can be made.

What if the polygon's faces were further divided into some number of co-linear segments? How can we count the number of ways to cut it?

Equations

Suppose there is a convex polygon with N faces, each face labeled by k ranging over 1, 2, ..., N. Let x_k be the number of segments present in face k. Let $C(x_1, ..., x_N)$ be the function we want to find which counts the number of cuts possible for this polygon. For the simple case where each x_k is equal to 1, it's the familiar formula

$$C(1, ..., 1) = S(N-1, ..., 0) = \frac{N(N-1)}{2} \tag{3}$$

In the general case, the function C satisfies some useful relations. First, since the polygon in question has no preferred orientation and may be rotated, the arguments to the count function may also be "rotated"

$$C(x_1, ..., x_N) = C(x_2, ..., x_N, x_1) \tag{4}$$

Second, extending the cut process described for the simple case in the introduction, if we have a known count, then dividing a face one additional time produces one new cut from each other faces' segments

$$C((x_1 + 1), ..., x_N) = C(x_1, ..., x_N) + S(0, x_2, ..., x_N) \tag{5}$$

When $x_1 > 1$, this can be expressed in another way

$$C(x_1, ..., x_N) = C((x_1 - 1), ..., x_N) + S(0, x_2, ..., x_N) \tag{6}$$

This can be applied repeatedly until the first parameter of C is reduced to 1

$$\begin{aligned} C(x_1, ..., x_N) &= C((x_1 - 1), ..., x_N) + S(0, x_2, ..., x_N) \\ &= C((x_1 - 2), ..., x_N) + 2S(0, x_2, ..., x_N) \\ &= C(1, x_2, ..., x_N) + (x_1 - 1)S(0, x_2, ..., x_N) \end{aligned} \tag{7}$$

Using the two rules together, each parameter to C can be reduced to 1 and a fully reduced form can be derived

$$\begin{aligned} C(x_1, ..., x_N) &= C(1, x_2, ..., x_N) \\ &\quad + (x_1 - 1)S(0, x_2, ..., x_N) \\ &= C(1, 1, x_3, ..., x_N) \\ &\quad + (x_1 - 1)S(0, x_2, ..., x_N) \\ &\quad + (x_2 - 1)S(1, 0, x_3, ..., x_N) \\ &= C(1, ..., 1) + \sum_{k=1}^{N}(x_k - 1)S(1, ..., 0, x_{k+1}, ..., x_n) \\ &= C(1, ..., 1) + \sum_{k=1}^{N}(x_k - 1)\left[(k - 1) + S(x_{k+1}, ..., x_n)\right] \end{aligned} \tag{8}$$

This derivation process is useful when calculating these counts on paper. The abbreviations can be expanded for the final result

$$C(x_1, ..., x_k) = \frac{N(N-1)}{2} + \sum_{k=1}^{N}(x_k - 1)\left[(k - 1) + \sum_{j=k+1}^{N} x_j\right] \tag{9}$$

Computationally, the nested summations do not necessarily mean the calculation will take $O(N^2)$ additions since the inner summation gets shorter, so its value is reduced by x_k on each iteration of the outer summation.