The only normal people are the ones you don't know very well.

— Alfred Adler

I have captured the signal, and am presently triangulating the vectors,
and compressing the data down, in order to express it as a function of my hand.
[Points.] ***They're over therrrrrrrre!***

— Prof. John Frink, "Wild Barts Can't be Broken", *The Simpsons* (1999)

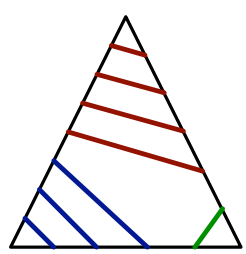
13 Normal Curves and Compression

In the late 1920s, Helmuth Kneser [3] introduced the theory of ***normal surfaces*** to prove a certain decomposition theorem about three-dimensional manifolds. Normal surfaces were further developed by Wolfgang Haken [2] a few decades later to answer certain algorithmic questions about 3-manifolds. Since Haken's work, dozens of refinements, extensions, and applications of normal surface theory have been developed; a comprehensive survey is beyond the scope of this course (and the expertise of the instructor!). In this lecture, I will describe some algorithmic results related to ***normal curves***, which are just like normal surfaces, only one dimension lower; I will return to (one of) Haken's algorithmic applications in the next lecture.

13.1 Normal Curves and Normal Coordinates

Fix a 2-manifold M, possibly with boundary. Let T be a ***triangulation*** of M: a cellularly embedded graph in which every face has three sides. Tn particular, each boundary cycle of M is covered by a cycle in T. A ***simple cycle*** in M is *the image* of a continuous injective map from S^1 to M. A ***simple arc*** in M is the image of a continuous injective map $\alpha\colon [0,1] \to M$ whose endpoints $\alpha(0)$ and $\alpha(1)$ lie on the boundary of M. A ***curve*** is the union of a finite number of pairwise-disjoint simple cycles and arcs.

Two curves γ and δ are ***isotopic*** (relative to ∂M) if γ can be continuously deformed into δ without moving any point on the boundary of M or introducing any intersections.[1] A curve is ***normal*** with respect to T if every intersection with an edge of T is transverse, and the intersection of the curve with any triangle is a finite set of ***elementary segments***: simple paths whose endpoints lie on distinct sides of the triangle. The endpoints of the elementary segments partition the edges of T into ***ports***.



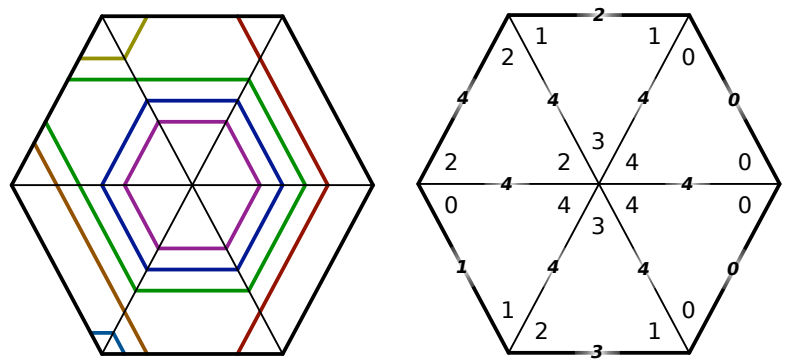
Nine elementary segments in a triangle.

Every triangle in T can contain three different types of elementary segments, each 'cutting off' one of corners of the triangle. For any corner x of any triangle A, let $\gamma(A,x)$ denote the number of elementary segments in $\gamma \cap A$ that separate x from the other two corners of A. Suppose T has t triangles. The vector of $3t$ non-negative integers $\gamma(A,x)$ are the ***normal coordinates*** of γ. We denote the normal coordinate vector of any normal curve γ by $\langle\gamma\rangle$. Not every vector in $\mathbb{N}^{3t}$ is a normal coordinate vector of a curve; for

[1] More formally, an isotopy from γ to δ is a continuous function of the form $h\colon [0,1] \times \gamma \to M$, such that (1) $h(0,\gamma) = \gamma$, (2) $h(1,\gamma) = \delta$, (3) $h(t,\gamma)$ is a curve for all $t \in [0,1]$, and (4) $h(t,x) = x$ for all $t \in [0,1]$ and $x \in \gamma \cap \partial M$.

any edge uv separating two triangles A and B in T, the normal coordinates of any curve γ must satisfy the equation $\gamma(A,u)+\gamma(A,v)=\gamma(B,u)+\gamma(B,v)$.

Suppose T has m edges. For any edge e in T, let $\gamma(e)$ denote the number of times γ intersects e; in terms of normal coordinates, we have $\gamma(xy)=\gamma(A,x)+\gamma(A,y)$ for any triangle A adjacent to edge xy. The vector of m non-negative integers $\gamma(e)$ are the ***edge coordinates*** of γ. Every edge coordinate vector satisfies two constraints: the sum of the edge coordinates for every triangle must be even, and the edge coordinates for each triangle must satisfy the triangle inequality $\gamma(xy)+\gamma(yz)\leq\gamma(xz)$. Given the edge coordinates of a normal curve, we can easily reconstruct its normal coordinates; for any triangle xyz in T, we have $\gamma(xyz,x)=(\gamma(xy)+\gamma(xz)-\gamma(yz))/2$. The parity constraints and triangle inequalities guarantee that the resulting normal coordinates are non-negative integers.



Left: A normal curve in a triangulation of the disk. Right: Its normal and edge coordinates

An isotopy is ***normal*** with respect to T if it moves the points on any edge of T only along that edge. Both the normal coordinates and the edge coordinates of any normal curve precisely characterize the normal isotopy class of that curve.

We can also represent any normal curve γ by the ***crossing sequences*** of is component arcs and cycles. However, this explicit representation can be *exponentially* larger than the coordinate representation of the same curve. The crossing sequence of γ has length

$$X := \sum_{xy}\gamma(x,y);$$

whereas, the number of bits required to store the normal coordinate vector $\langle\gamma\rangle$ is only

$$\|\gamma\| := \sum_{xy}\lceil\lg(\gamma(x,y)+1)\rceil = O(m\log(X/m)).$$

Given this extreme disparity, it is natural to ask whether interesting properties of curves can be computed from their coordinate representations in time that is polynomial in the input complexity $\|\gamma\|$. For example, how quickly can we determine whether a given coordinate vector describes a *connected* curve? Can we compute the number of components of $M\setminus\gamma$ quickly? Given two coordinate vectors, can we quickly decide whether they can represent disjoint curves, or isotopic curves?

13.2 Word Equations and Straight Line Programs

The first polynomial-time algorithms for many of these problems were published by Schaefer, Sedgwick, and Štefankovič [8, 9]. Their algorithms manipulate strings representing the sequence of crossings along each edge of the triangulation. However, these strings are never computed explicitly, because they might

have exponential length; instead, each string is represented in a certain compressed form whose length is guaranteed to be polynomial.

Let Σ be a finite alphabet. A ***straight-line program*** is a context-free grammar in Chomsky normal form that generates a single word in Σ^*. More explicitly, a straight-line program $\mathcal{A}$ is a sequence of n assignments, where the ith assignment in Γ is either $A_i \leftarrow x$ for some symbol $x \in \Sigma$, or $A_i \leftarrow A_j \cdot A_k$ for some indices $1 \le j, k < i$, where $\cdot$ denotes concatenation.

Every straight-line program $\mathcal{A}$ generates a unique word $\boldsymbol{w(\mathcal{A})}$ in Σ^*. More generally, for any index i, let $\boldsymbol{w(A_i)}$ denote the word in Σ^* generated by the first i assignments. Specifically, if the ith assignment is $A_i \leftarrow x$, then $w(A_i) = x$, and if the ith assignment is $A_i \leftarrow A_j \cdot A_k$, then $w(A_i) = w(A_j) \cdot w(A_k)$. For example, if $\mathcal{A}$ is the straight line program

$$A_1 \leftarrow \mathtt{0}; \quad A_2 \leftarrow \mathtt{1}; \quad A_3 \leftarrow A_1 \cdot A_2; \quad A_4 \leftarrow A_2 \cdot A_3; \quad A_5 \leftarrow A_3 \cdot A_4; \quad A_6 \leftarrow A_4 \cdot A_5; \quad A_7 \leftarrow A_5 \cdot A_6,$$

then $w(\mathcal{A}) = w(A_7) = \mathtt{0110110101101}$. Straight-line programs are closely related to the Lempel-Ziv(-Welsh) family of compression algorithms.

We need the following algorithmic result:

Lemma 13.1 (Miyazaki, Shinohara, and Takeda [6]). *Given a straight line program $\mathcal{A}$ of length n and a word $p \in \Sigma^m$, we can compute the number of occurrences and the index of the first occurrence (if any) of p in the string $w(\mathcal{A})$, in time polynomial in n and m.*

In all the applications of this lemma in this lecture, the pattern word p is only a single character; in this case, Lemma 13.1 is solved by a straightforward dynamic programming algorithm in $O(n)$ time.

Now let Θ be an alphabet of ***variables*** disjoint from Σ. An ***assignment*** to Θ is a function $\alpha \colon \Theta \to \Sigma^*$. Any assignment α induces a unique ***morphism*** $\alpha^* \colon (\Sigma \cup \Theta)^* \to \Sigma^*$, by defining $\alpha^*(w) = w$ for any string $w \in \Sigma^*$, and $\alpha^*(u \cdot v) = \alpha^*(u) \cdot \alpha^*(v)$ for any strings $u, v \in (\Sigma \cup \Theta)^*$. A ***word equation*** is an equation of the form $u = v$, where u and v are strings in $(\Sigma \cup \Theta)^*$; a ***solution*** to the word equation is an assignment $\alpha \colon \Theta \to \Sigma^*$ such that $\alpha^*(u) = \alpha^*(v)$. Any number of word equations can easily be combined into a single equation, and the solution to a word equation can be exponentially longer than the equation itself; consider the following equation over the alphabets $\Sigma = \{\mathtt{0}, \mathtt{1}, \diamond\}$ and $\Theta = \{A, B, C, D, E, F, G\}$:

$$A\diamond B\diamond C\diamond D\diamond E\diamond F\diamond G = \mathtt{0}\diamond\mathtt{1}\diamond AB\diamond BC\diamond CD\diamond DE\diamond EF$$

A ***word equation with lengths*** is a word equation $u = v$ together with a length function $f : \Theta \to \mathbb{N}$. Any solution α to such an equation must satisfy the constraint $\alpha(\theta) \in \Sigma^{f(\theta)}$ for all $\theta \in \Theta$.

Theorem 13.2 (Plandowski and Rytter [7]). *For any ordering of Σ and any system of word equations over $\Sigma \cup \Theta$ with lengths $f : \Theta \to \mathbb{N}$, a straight-line program encoding the lexicographically smallest solution (or a proof that there is no solution) can be computed in polynomial time.*

13.3 Solving Normal Curve Problems

Given the normal coordinates or edge coordinates of a curve γ, we construct a system of word equations with lengths whose solution encodes the components of γ, as follows. Our variable alphabet Θ contains two variables A_{xt} and A_{tx} for each corner x of each triangle t, and variables A_{xy} and A_{yx} for each edge xy. Our system contains two equations $A_{xy} = A_{xt}A_{ty}$ and $A_{yx} = A_{yt}A_{tx}$ for each edge xy. The length function assigns $f(A_{xy}) = f(A_{yx}) = \gamma(x, y)$ for every edge xy and $f(A_{tx}) = f(A_{xt}) = \gamma(t, x)$ for every corner x of every triangle t.

Any solution to this system of word equations associates a symbol with each component of γ; different components may be associated with the same symbol. Suppose we associate a symbol with

each component of γ. Each corner variable A_{xt} represent the sequence of elementary segments cutting off corner x in triangle t, in order from the corner outward; each segment is represented by the symbol of the component of γ that contains it. Each corner variable A_{tx} is just the reverse of A_{tx}. Each edge variable A_{xy} represents the sequence of components of γ crossed by edge xy, in order from x to y; again, A_{yx} is the reversal of A_{xy}.

13.3.1 Testing Connectivity

We have no way to assign each component of γ a *different* label, first because we don't know what the components are, and second because there may be an exponential number of them. However, we can label *some* components by adding additional equations to our system.

Suppose we want to determine whether a curve γ is connected. Let $\Sigma = \{\mathtt{0}, \mathtt{1}\}$, with the obvious ordering. We add a variable B and a constraint $A_{xy} = \mathtt{1}B$ and then compute the lexicographically smallest solution α to the resulting system. This solution assigns $\mathtt{1}$ to the component γ_1 of γ that crosses xy closest to x, and $\mathtt{0}$ to every other component of γ. In particular, if γ is connected, every variable is assigned a string in $\mathtt{1}^*$. Moreover, by counting the $\mathtt{1}$'s in each string $\alpha(A_{xy})$ and $\alpha(A_{tx})$, we can compute the edge and normal coordinates of one component of γ.

Theorem 13.3. *Given the normal coordinate vector $\langle\gamma\rangle$, we can determine whether γ is connected, and compute the normal coordinates of one component of γ, in polynomial time.*

A different(?) algorithm for testing connectivity of normal curves was independently described by Agol, Hass, and Thurston [1]. Intuitively, their algorithm labels the N intersection points between a normal curve γ and the edges of T with the integers 1 through N, so that intersection points along any edges are labeled consecutively. The elementary arcs at each corner of each triangle establish correspondences between equal-length intervals $[a_i \mathinner{..} b_i]$ and $[c_i \mathinner{..} d_i]$ of the range $[1 \mathinner{..} N]$. The transitive closure of these correspondences defines an equivalence relation, whose equivalence classes (which Agol *et al.* call ***orbits***) correspond to the components of γ. Agol *et al.* describe an algorithm to compute the number of orbits induces by *any* set of interval-pairings over the range $[1 \mathinner{..} N]$, whose running time is polynomial in the number of pairings and $\log N$.

13.3.2 Counting Components

With a little more work, we can extend this algorithm to count the number of components of any normal curve γ, by considering one normal-isotopy class of components at a time.

Let γ_1 be the component of γ that crosses some edge xy closest to endpoint x. Let γ_i be the component of γ containing the ith intersection point along xy. A simple modification of Theorem 13.3 computes the normal coordinates $\langle\gamma_k\rangle$ in polynomial time. Any two normal-isotopic components of γ bound a disk or an annulus with no other vertices inside; moreover, any other component of γ inside that disc or annulus is also in that normal isotopy class. Thus, the components normal-isotopic to γ_1 intersect xy consecutively; they are $\gamma_1, \gamma_2, \ldots, \gamma_k$ for some integer k. We can find k using binary search; this requires $O(\log \gamma(x, y))$ calls to Theorem 13.3.

The normal coordinate vectors $\langle\gamma_1\rangle, \langle\gamma_2\rangle, \ldots \langle\gamma_k\rangle$ are all equal. Thus, $\langle\gamma\rangle - k \cdot \langle\gamma_1\rangle$ is the normal coordinate vector of $\gamma' = \gamma \setminus \bigcup_{i=1}^{k} \gamma_i$. We can now recursively compute the number of components of γ'. The following lemma guarantees that this recursive algorithm ends after only a small number of iterations.

Lemma 13.4. *The components of any normal curve fall into at most $O(t)$ normal isotopy classes.*

Proof: Suppose the underlying Σ is orientable and has genus g and b boundaries; the analysis of the non-orientable case is similar. Suppose also that the triangulation T has n vertices, m edges, t triangles, and B boundary vertices (and boundary edges). Let $\chi = n - m + t + b = 2 - 2g - b$ denote the Euler characteristic of Σ. A standard double-counting argument with Euler's formula implies that $t = 2n + 2g + 2b - 4 - B \geq n + 2g + 2b - 4$.

We separately consider three overlapping classes of connected normal curves: separating curves, noncontractible cycles, and noncontractible arcs; an arc is contractible if it is isotopic to a boundary path. Because these classes overlap, our analysis will not be tight.

Let $\mathcal{S}$ be a maximal set of pairwise-disjoint *separating* cycles and arcs in distinct normal isotopy classes. Any two curves in $\mathcal{S}$ partition the surface into exactly three components, each containing at least one vertex of T; otherwise, the two curves would be normal-isotopic. It follows that $\mathcal{S}$ has exactly $n - 1$ elements.

Let $\mathcal{C}$ be a maximal set of pairwise-disjoint *noncontractible cycles* in distinct isotopy (not just normal isotopy) classes. Cutting the surface along any cycle leaves the Euler characteristic of the surface unchanged. Each component of $\Sigma \setminus \mathcal{C}$ is either a pair of pants (a sphere minus three disks) or an annulus. A pair of pants has Euler characteristic -1, and each annulus contains exactly one boundary cycle of Σ. Thus, $\Sigma \setminus \mathcal{C}$ consists of $-\chi(\Sigma) = 2g + b - 2$ pairs of pants and b annuli. It follows that $|\mathcal{C}| = (3(2g + b - 2) + b)/2 = 3g + 2b - 3$.

Finally, let $\mathcal{A}$ be a maximal set of pairwise-disjoint *noncontractible arcs* in distinct isotopy (not just normal isotopy) classes. Each component of $\Sigma \setminus \mathcal{A}$ is a disk bounded by exactly three arcs in $\mathcal{A}$ and three boundary arcs. Contracting each boundary cycle of Σ to a point transforms $\mathcal{A}$ into a b-vertex triangulation of a surface of genus g. Thus, Euler's formula implies that $\mathcal{A} = 3b + 3g - 6$.

We conclude that the components of any normal curve fall into at most $|\mathcal{S}| + |\mathcal{C}| + |\mathcal{A}| = n + 6g + 5b - 7 \leq 3t + 3$ distinct normal isotopy classes. □

Theorem 13.5. *Given the normal coordinate vector $\langle\gamma\rangle$, we can compute the normal coordinates of each normal-isotopy class of components of γ, as well as the number of components of γ in each class, in polynomial time.*

13.3.3 Testing Contractibility

Finally, suppose we want to determine whether some component of a normal curve is contractible. Without loss of generality, we can assume that our input is the normal coordinate vector $\langle\gamma\rangle$ of a normal cycle or arc γ. Recall that γ is contractible if and only if $M \setminus \gamma$ has two components, one of which has Euler characteristic 1. We can test these two conditions as follows:

For each edge xy, we define two variables P_{xy} and P_{yx}, representing the sequence *ports* along edge xy. For each corner x of each triangle t, we also define variables P_{tx} and P_{xt}, which are the sequence of components of $t \setminus \gamma$ that do not touch the side of t opposite x. Finally, we let P_t be the unique component of $t \setminus \gamma$ that touches all three sides of t. These variables satisfy the equations $P_{xy} = P_{xt}p_tP_{ty}$ and $P_{yx} = P_{yt}p_tP_{tx}$ for every edge xy of every triangle t. We also have length functions $f(P_{xy}) = f(P_{yx}) = \gamma(x, y) + 1$; $f(P_{tx}) = f(P_{xt}) = \gamma(x, t)$; and $f(P_t) = 1$.

The surface $M \setminus \gamma$ has at most two components, which we can isolate as follows. We add an equation $P_{xy} = 1Q$ to the system of word equations, and then compute the lexicographically smallest solution over the alphabet $\{0, 1\}$. If the solution is a set of strings in 1^*, then $M \setminus \gamma$ is connected, which implies that γ is nonseparating, and therefore noncontractible. Otherwise, γ is separating, and each component of $M \setminus \gamma$ has a unique label; call these components M_0 and M_1 according to their labels.

The ports and elementary segments defined by T an γ define a graph T_γ. We can extract the Euler characteristic of M_1 from the strings P_{xy}, P_{xt}, and P_t. Specifically, M_1 has $2A$ vertices, $A + 2B + 3C$ edges,

and $B + C$ faces, and therefore has Euler characteristic $A - B - 2C$, where

$$A := \sum_{xy} \#1(P_{xy}), \qquad B := \sum_{t,x} \#1(P_{xt}), \quad \text{and} \quad C := \sum_{t} \#1(P_t).$$

We conclude that γ is contractible if and only if either $\chi(M_1) = 1$ or $\chi(M_1) = \chi(M) - 1$.

Theorem 13.6. *Given the normal coordinate vector $\langle\gamma\rangle$ of a normal curve, we can determine whether γ is contractible in polynomial time.*

A similar algorithm can be used to test whether two components of a normal curve are isotopic (even if they are not normal-isotopic).

Theorem 13.7. *Given the normal coordinate vectors $\langle\gamma_1\rangle$ and $\langle\gamma_2\rangle$ of two disjoint normal curves, we can determine whether γ_1 and γ_2 are isotopic in polynomial time.*

It is also possible to determine in polynomial time whether two *intersecting* normal curves are isotopic, but the algorithm is more complicated. This and several other related group-theoretic results have been developed by Lohrey and Schleimer [4, 5, 10].

References

[1] I. Agol, J. Hass, and W. P. Thurston. The computational complexity of knot genus and spanning area. *Trans. Amer. Math. Soc.* 358(9):3821–3850, 2006.

[2] W. Haken. Theorie der Normalflächen: Ein Isotopiekriterium für den Kreisknoten. *Acta Mathematica* 105:245–375, 1961.

[3] H. Kneser. Geschlossene Flächen in dreidimensionalen Mannigfaltigkeiten. *Jahresbericht Deutscher Math.-Verein.* 38:248–260, 1929.

[4] M. Lohrey. Word problems and membership problems on compressed words. *SIAM J. Comput.* 35(5):1210–1240, 2006.

[5] M. Lohrey and S. Schleimer. Efficient computation in groups via compression. *Proc. 2nd Int. Symp. Comput. Sci. in Russia*, 249–258, 2007. Lecture Notes Comput. Sci. 4649, Springer-Verlag. ⟨http://www.informatik.uni-leipzig.de/~lohrey/08-CWP-long.pdf⟩.

[6] M. Miyazaki, A. Shinohara, and M. Takeda. An improved pattern matching algorithm for strings in terms of straight-line programs. *J. Discrete Algorithms* 1(1):187–204, 2000.

[7] W. Plandowski and W. Rytter. Application of Lempel-Ziv encodings to the solution of word equations. *Proc. nth Int. Conf. Automata Lang. Prog.*, 731–742, 1998. Lecture Notes Comput. Sci. 1443, Springer-Verlag.

[8] M. Schaefer, E. Sedgwick, and D. Štefankovič. Algorithms for normal curves and surfaces. *Proc. 8th Int. Conf. Comput. Combin.*, 370–380, 2002. Lecture Notes Comput. Sci. 2387, Springer-Verlag.

[9] M. Schaefer, E. Sedgwick, and D. Štefankovič. Computing Dehn twists and geometric intersection numbers in polynomial time. *Proc. 20th Canadian Conf. Comput. Geom.*, 111–114, 2008. Full version: Technical Report 05–009, Comput. Sci. Dept., DePaul Univ., April 2005.

[10] S. Schleimer. Polynomial-time word problems. *Comm. Math. Helv.* 83(4):741–765, 2008.