Drawing Isoglosses Algorithmically

Bryce Wiedenbeck
bryce@cs.swarthmore.edu

Kit La Touche
kit@cs.swarthmore.edu

Abstract

In this paper, we apply algorithms for defining regions from sets of points to the problem of drawing *isoglosses*, the boundaries between dialect regions. We discuss the justifications for our method, and alternative models that could be constructed from this data. We evaluate the resultant model by comparison to the traditional method of drawing isoglosses, by hand.

1 Introduction

In the linguistic subfield of dialectology, an important activity is the drawing of isoglosses, or boundaries between dialect areas. It is often difficult to pin down the meaning of these terms, as within a region people come and go, and bring their speech with them, but broadly speaking, an isogloss is a boundary between where people speak like *this* and where people speak like *that*. Typically, an isogloss will not be a sharp line, but will be an area of overlap between speakers of one type and the other.

Unfortunately, isoglosses are typically drawn by hand, as an approximate dividing line. This leads to two problems: one, they should not be thought of or represented as lines, but rather as approximate transition zones, and two, they could be better drawn, we think, by algorithm than by eyeballing. As the noted sociolinguist William Labov writes,

> Every dialect geographer yearns for an automatic method for drawing dialect boundaries which would insulate this procedure from the preconceived notions of the analyst. No satisfactory program has yet been written. (Labov et al., 2005)

We have therefore attempted, as something like a proof-of-concept, to redraw the isoglosses for certain dialect differences in American English. We have taken as our data the results of the telephone survey of speakers across the contiguous USA done for the Atlas of North American English (Labov et al., 2005). To this data, we have applied a number of algorithms from the field of computational geometry, and hope that the result will better represent the boundaries between dialect regions.

One way we expect that our method will improve on a hand-drawn line is by clearly showing areas that are thoroughly mixed. It can be tempting, when drawing by hand, to mark an area as primarily speaking one way, and drawing your line as though that is the case. It may well be the case, however, that in such situations, the region is much more evenly mixed than it appears to the eye, and should probably not be marked decidedly in either direction. We suspect that an algorithmic approach will see these cases more clearly.

The clearest way to test whether our method improves on a hand-drawn line would be to see if this model has greater predictive power, such that if we were to randomly make telephone calls to people, their proximity to our lines would be a better indicator of the features of their dialect. However, doing so is outside the scope of the project. Other changes to our method that would make it more explicitly a predictive model, such as predicting the value for a point based on the inverse-distance weighted average of the n-nearest neighbors,[1] while interesting, would also be outside the scope of this project. Such an approach would really be a machine learning task, and not a cartographic task.

As such, our evaluation is limited to observing in a qualitative way the differences between our method and the hand-drawn method. We expect that if there is no significant difference, this will at least provide a way of automating the creation of a machine-readable form, assuming data points are available. If there is a significant difference, then

[1] As suggested by George Dahl.

perhaps more investigation into the predictive powers of the two models is warranted.

2 The data

The data from the Atlas of North American English (Labov et al., 2005) is in the form of approximately 600 points, identified geographically by ZIP code, which we then converted to latitude and longitude coordinates. At each point, there was an indication of whether the speaker at that point makes or does not make each of a set of possible linguistic distinctions. The data is generally denser around the north east and Great Lakes regions, but this is in part due to greater population density in these areas. Of course, more datapoints would always be desirable, but these data are still useful.

A dialect area can be considered an area of overlap between polygons from different feature sets, where the area of overlap consists of most of the area of the parent polygons. Thus, it is an area where, at least with respect to the features under consideration, people speak the same way, and that way is distinct from the surrounding area. The scale of this can vary, of course: a city may form a distinct dialect area within a state, if it has sufficiently different speech, but there may also be dialect areas within that city, which are each more like each other than they are like the speech outside the city, but still differ from each other.

The specific features which we mapped were the following: whether the speaker distinguishes ɑ and ɔ, whether the speaker distinguishes ʍ and w, and whether the speaker distinguishes ɪ and ɛ before a nasal, such as n or m.[2] These are a set of easily explicable features, with the first and third being generally considered to be characteristic of large US dialects. For each feature, following the format in the Atlas, a speaker could make the distinction, not make the distinction, or be unclear — that is, the interviewer was unable to tell whether they reliably made or did not make the distinction. In all cases, we used the interviewer's judgment, rather than the speaker's self-reporting.

The scale of our project is across the contiguous USA, so many smaller-scale regional variations don't show up. This is dependent on a number of parameters in the algorithm, and the size and granularity of the dataset. For example the data set and our techniques might allow us to note the speech of western Pennsylvania and West Virginia as distinct from the area around it, but would not create a distinct region for the dialect of San Francisco and its countryside.

3 Methods

Our goal of identifying dialect regions based on individual phoneme information required us to first locate areas where speakers pronounced a phoneme similarly. For most phonemic variables in the Atlas of North American English, the speakers with similar pronounciation are not located all in one area. This meant that for each feature setting, we had to first break the data points up into several regional groupings, which we did using k-means clustering. Once we had clustered the data for a particular feature setting we found boundaries for the resulting regions by computing the convex hulls of the data points in each cluster. This gave us several polygons describing regions in which a particular feature had the same setting, so to find dialect regions we overlaid these polygon sets and found regions of intersection. Each of these three steps is described in detail below.

3.1 Clustering

To break the data points from a data set for a particular feature setting down into smaller groups, we used k-means, a clustering algorithm proposed by MacQueen (1967). The algorithm starts with k centroids that can be specified as input or randomly selected from the data points. Each data point is then assigned to the nearest centroid, and each centroid is adjusted to minimize the distance to all of its points. Then the data points are again assigned to the closest centroid, and the centroids are recomputed, and so on. The algorithm terminates when an iteration occurs in which no data points switched centroids. The theoretical worst-case time complexity of the k-means algorithm is superpolynomial (Arthur and Vassilvitskii, 2006), but in practice it runs quite quickly.

We considered a number of alternative clustering algorithms including faster heuristic-based or

[2]These symbols are explained in Section 7.

approximate k-means implementations. We also considered other more adaptive clustering algorithms, like growing k-means or growing neural gas (Daszykowski et al., 2002). The main advantage to such techniques is that they would not require the number of clusters to be specified in advance, but instead start with very few centroids and add more as needed. We decided against such methods after considering the potential applications of our work. First, if a user of our algorithm, say a linguist creating dialect maps, is unfamiliar with the details of the implementation, it might be easier for him to specify starting-point centroids than to tune the parameters that control the termination point of growing k-means or growing neural gas. Additionally, because our data come from an atlas with hand-drawn isoglosses, we have available to us reasonably good starting-point centroids to use in our testing.

3.2 Convex Hulls

After forming clusters, we computed the convex hull of each set of clustered points, using a simple convex hull algorithm, the Jarvis March. The algorithm uses a "gift-wrapping" technique, starting from the leftmost point in the data set, and at each subusequent step, scanning through all the remaining points to find the one that makes the widest angle with the previous hull boundary-segment. This runs in in O(nh) time, because each of h loops scans all of the input points (de Berg et al., 1997). Non output-sensitive, O($n \log n$), convex hull algorithms also exist and are useful when h is close to n, resulting in n^2 complexity for Jarvis, but in our application, h tends to be quite small relative to h, so the Jarvis March performs marginally better.

When inspecting the first applications of our convex-hull implementation to real data, we noticed that the hulls were often significantly affected by a few outlier points that weren't particularly close to any of the cluster centroids but our implementation of k-means required that they be assigned to some cluster. For example one data set had three clear clusters in the northeast, the midwest, and the deep south, plus a few extraneous data points in California. These California points caused the midwest-hull to stretch halfway across the country (see figure 1).

To fix this, we tried two different methods. The

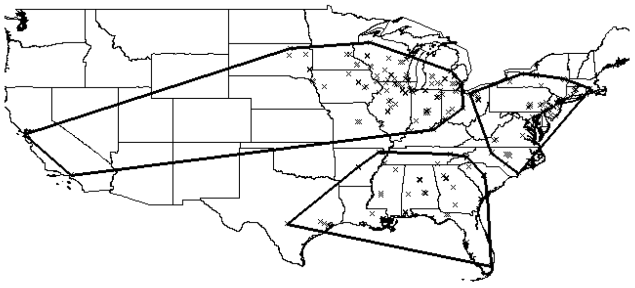

Figure 1: Convex hulls of point clusters for regions where ɑ and ɔ are distinguished, without oulier removal.

first was onion-peeling of the convex hulls, [3] which would involve throwing out the points on the convex hull and recomputing the convex hull of the remaining points (perhaps more than once). We found this method ineffective because removing all outliers would sometimes require more than one onion-peeling step, in which case some more-compact clusters would be whittled away and no longer accurately represent the data. We then tried, and were satisfied with, calculating the outliers more explicitly. For each cluster, we computed the standard deviation of the distance from each point to the centroid and removed all points more than some number of standard deviations[4] away before computing the convex hull.

3.3 Overlay

Finally, given the hulls for each cluster in a particular map, we overlaid them to find the common area between, for example, the speakers who both make the ɑ/ɔ distinction, and drop syllable-final r. By overlaying hulls from different features, we could identify regions of significant overlap, indicating dialect regions, and by overlaying hulls from different settings of the same feature, we can determine border areas where common sets of linguistic features cannot be readily described.

[3]Suggested by Andrew Danner.

[4]We found that 3 standard deviations produced good results.

Overlaying these regions requires first detecting whether two polygons intersect. A method to do this in O($\log n$) time was proposed by Chazelle and Dobkin (1980), but is exceedingly difficult to implement, so we opted instead for an O(n^2) algorithm based on the leftTurn and rightTurn primitives we had already implemented for computing the intersection. The idea is that if two polygons don't intersect, then there exists a dividing line between them, and one such line must be a side of one of the polygons. Therefore we can walk around the outside of one polygon and determine for each side, whether all points in the other polygon lie on the opposite side of that line from the rest of its polygon. If so, the polyogns don't intersect; if no such line is found after walking around both polygons, then the polygons intersect.

Once we know that two polygons intersect, we use a rotating calipers implementation provided by Mary Wooters and George Dahl to compute bridge points between them (Toussaint, 1983). Each of these bridge points corresponds to a point of intersection between the two hulls at the other end of the "sail polygon" (see figure 2), which we locate using the stepDown procedure outlined in Toussaint (1985). Once we have located these intersection points, finding the convex intersection of the two polygons is simply a matter of walking around one polygon and then the other, switching at each intersection point. Both rotating calipers and stepDown run in time linear in the size of the polygons, as does the final output step, so the whole overlay step runs in worst-case O($k^2 * h$) time, where k is the number of k-means centroids (and therefore the number of polygons), and h is the number of points on the polyogns being combined. In practice, k and h are well under $n^{\frac{1}{3}}$, so this works out to be no worse than linear in the size of the overall data set.

To visualize the data, we used GRASS,[5] mapping our datapoints to latitude and longitude coordinates.

4 Results

The regions drawn by this method seem plausible given our knowledge of the data sets we used. However, a rigorous test of the predictive power of this

[5]GRASS is an Open Source geographic information system, available at `http://grass.osgeo.org/`.

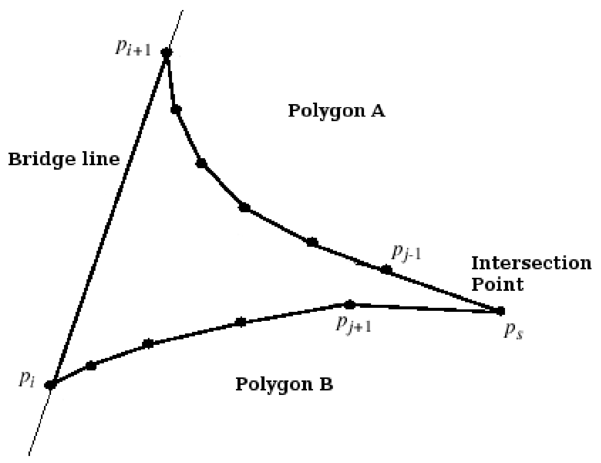

Figure 2: The stepDown function iterates over a "sail polygon" to find the intersection between two polygons given a bridge-line between them.

model is not particularly feasible without more data.

One major difference is that this method produces only convex polygons as dialect regions, and also does not cover all the inhabited territory on the map. Both of these facts imply something about how our regions are to be interpreted: for a given feature, the regions of no overlap with other settings for that feature are areas of high-confidence that there is one dialect, areas of overlap are areas of confidence that the speech is mixed there, and areas outside of polygons are areas of insufficient data.

There were some moments when it was clear how to hand-evaluate the algorithm. For example, before switching to standard-deviation hulls, we attempted onion-skin hulls. This resulted in a hull that stretched across the US from Wisconsin to southern California, even after peeling off the first layer. This was clearly an artifact of the processing, and not reflective of a here-to-fore unobserved swath of speakers who distinguish ɑ and ɔ; the greater part of the polygon, as it stretched across the country, was devoid of datapoints, having them all clustered at the Wisconsin end.

5 Future Work

The clearest route for future work would be to develop a metric with which to test the quality of a dialect map. The purpose of such a map is presumably as a predictive tool; as such, gathering larger amounts of data, for distinct training and testing,

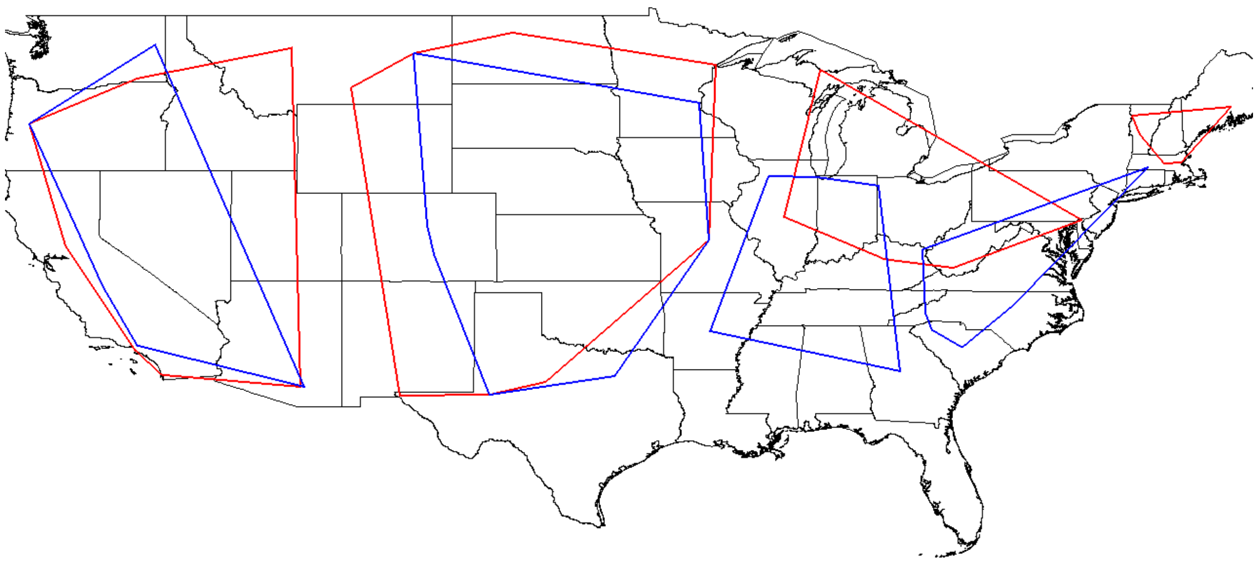

Figure 3: Overlaid convex hulls of clustered data. The red are areas where ɑ and ɔ are merged. The blue are areas where ɪ and ɛ are merged before nasals.

would be ideal. However, collecting such data remains time-consuming and difficult.

It has been suggested that an alternative approach to this problem would be to treat it as a machine learning task. One could imagine easily a learning system that would, given training data of geographic points classed by setting of linguistic features, would produce the likeliest settings for linguistic features, given testing data of just geographic points. While such a system would be interesting, and possibly very informative, it would not lend itself immediately to map-making, and was outside the scope of this project.

6 Conclusions

It is difficult to evaluate the success of our algorithm both because the Atlas is not conducive to a quantitative comparison, and because testing our algorithm against real-world data is impractical. The algorithm seems to produce reasonable results, but a large part of our ability to evaluate this is based, circularly, on the maps available in the Atlas. Clearly, both more data and a better means of evaluation would be desirable.

7 Linguistic Appendix

A number of phonetic symbols have been used in this paper. As familiarity with them is not assumed, we will explain them here.

- ɑ: for those that distinguish this from the following vowel, it is the vowel in t**o**t.
- ɔ: for those that distinguish this from the preceding vowel, it is the vowel in t**augh**t.

- w: this is the first consonant of **w**itch.
- ʍ: this is the first consonant of **wh**ich, though many speakers do not use this, instead merging with w, above.
- ɪ: as in p**i**n.
- ɛ: as in p**e**n. Some US dialects do not distinguish ɪ and ɛ before nasal consonants, such as n or m.

These features were selected because they were available in the Atlas, and easily explicable. A thorough effort to algorithmically create dialect maps would use many more features.

References

David Arthur and Sergei Vassilvitskii. 2006. How slow is the k-means method? In *SCG '06: Proceedings of the twenty-second annual symposium on Computational geometry*, pages 144–153, New York, NY, USA. ACM.

Bernard Chazelle and David P. Dobkin. 1980. Detection is easier than computation (extended abstract). In *STOC '80: Proceedings of the twelfth annual ACM symposium on Theory of computing*, pages 146–153, New York, NY, USA. ACM.

M. Daszykowski, B. Walczak, and D. L. Massart. 2002. On the optimal partioning of data with k-means, growing k-means, neural gas and growing neural gas. *Journal of Chemical Information and Modelling*, 42(6):1378–1389.

Mark de Berg, Marc van Kreveld, Mark Overmas, and Otfried Schwartzkopf. 1997. *Computational Geometry*. Springer.

William Labov, Sharon Ash, and Charles Boberg. 2005. *The Atlas of North American English*. Mouton de Gruyter.

J. B. MacQueen. 1967. Some methods for classification and analysis of multivariate observations. In L. M. Le Cam and J. Neyman, editors, *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press.

Godfried Toussaint. 1983. Solving geometric problems with the rotating calipers.

Godfried T. Toussaint. 1985. A simple linear algorithm for intersecting convex polygons. *The Visual Computer*, 1(2):118–123.