Principles of automated zone design methods

Hello, I'm David Martin and in this series of short videos I'm explaining some of those methods that we can use for automated zone design. And in this video were going to look at some of the principles behind automated zone design procedures.

We are going to think a little about building blocks that we can use, the nature of design criteria and the way in which iterative calculations can be used to achieve an optimal design. The problem that are trying to solve in automated zone design is to find a means of achieving a set of zones that meet specific design criteria. For example, we might want zones that are not too big, not too small, are sensible shapes on the map, or which have similar populations and so inevitably this will always be a trade-off between a number of competing objectives. Faced with a problem of that kind, a natural human inclination is to draw lines on the map in order to subdivide a larger area. An alternative approach is to put together many small areas as if completing a jigsaw, and the computational approaches which I'm going to describe here are much closer to that second analogy.

So, the procedures which I'm illustrating here are examples of methods which can be traced back to the Automated Zoning Procedure introduced by Stan Openshaw in 1977. What we are doing here is taking a set of small building block zones and aggregating them into groups which from now on I'm going to call tracts. What we would do is to create an initial random aggregation of these tracts a bit like putting together a small area of a jigsaw, in in a provisional sense and then compute a set of statistics relating to the size, shape and the other design criteria which we may have for the final configuration. And then we try swapping some of the building blocks, very much like taking one of the jigsaw pieces off one small block, putting it onto a neighbouring block and then recalculating the statistics which describe our configuration and if that results in an improvement to the overall solution we may decide to keep that swap, whereas if it clearly results in a poorer solution we may put it back and try a different swap. And so we have an iterative process which is based on a highly computational method which goes and tries very, very, very many small swaps, evaluating their effect and gradually building towards a more optimal solution than the random one we started with. In computational turns there are various devices available to stop these kinds of algorithms from getting stuck in local suboptima, using methods such a simulated annealing or tabu searching, which are referenced in the literature.

In diagrammatic terms, we can see here that what we're doing is taking set a small building blocks, evaluating a set of possible solutions against some notion of which might be best and then adopting that best solution to create the set of tracts. Now it's worth saying a little more at this point about the building blocks that we might be using. Building block zones might come from many different sources and in different applications they may need to be purpose-built or they may be already in existence and they need to be small relative to the output zones, in the same way that jigsaw pieces would be small in relation to the elements of the picture. Really, we would expect that in most contexts these are going to be zones themselves which have come from a geographic information system - although we do not need to be an expert in using geographic information systems to make use of automated zone design.

Another important consideration is that all the statistical information which might be relevant to our design criteria has to be available for each of the building block zones, so if population size, for example, is an important consideration, we must know the population of each of the

building blocks and there will be research situations in which this requires the use of confidential data in a secure setting in order to design zones for data which can subsequently be published once the zonation is complete - and that's precisely how it works in the context of a population census and the output geography from it.

So, here we take a look at some simple building blocks. In this case, I've used the GIS to generate a set of Thiessen or Voroni polygons around each of a set of address locations. The addresses are shown in blue. The artificial polygons they're space-filling polygons here, are in red and there are some black lines which are features in this case the centrelines of the roads to which we've constrained the address polygons.

In this case we may then take some additional information such as the postcodes of the addresses and dissolve the boundaries between each adjacent address which has got the same postcode and the result, as we can see here, is that a set of small polygons containing all the addresses with the same postcode. In this instance note that the boundaries don't precisely follow the elements of the background mapping because I didn't supply them. If we wanted those boundary features to follow every stream, every parcel of land, then that information would need to be used in the design of the building blocks. Nevertheless, what I have here is a set of small building block polygons from which we could begin to construct a geography at a higher level. So, considerations of these building blocks: well importantly, the eventual outputs zone boundaries are going to be drawn from the building block boundaries, so the outputs can't be smoother or more realistic or better aligned to real-world features than those of the building blocks themselves. So, for example, if it's important in a particular zone design that the street geography is reflected in the output, street geography will need to be reflected in the construction of the building blocks. And, in general, as the number of building blocks increases there will be more permutations possible, we will get longer computation times, but probably we'll find that there are more solutions which meet our design criteria well. Conversely, if the number of building blocks is quite small and they're quite large relative to the outputs it won't take very long to compute the solution but it might not actually meet the criteria very well because we've got a lot less capacity to move things around and trade off the competing criteria.

The zone design criteria themselves can be thought of as both those which are hard constraints: things which must be met, and soft constraints: which we can't meet exactly but where we're trying to maximize some continuous function. Examples of the hard constraints might be that every zone must contain more than one hundred people and more than forty households because otherwise the zone would not meet some basic confidentiality threshold. Similarly a geographical constraint would be that all of the zones we're creating are not permitted to cross a local authority or a region boundary. Those are absolutes, and we can set up the zone design problem such that we will not produce solutions of any kind which violate those rules. But the soft constraints are much harder because, for example, we might say that we want zones which containe as close as possible to a hundred and twenty five households and it will not be possible to meet that exactly. We will find, perhaps, some large residential buildings themselves contain more than a hundred and twenty-five households or the area to be zones may not be actually a neat multiple of a hundred and twenty-five so were are only ever going to get close to that particular target.

Similarly, we might find that for certain purposes, a constraint might be specified of the firm that every zone should be as compact in shape as possible, and there are different ways that can be measured. It is very difficult to find any configuration of real-world features which are

highly compact. The most compact shape's going to be something like hexagons – that's going to be very unrealistic in most geographical applications using real-world data. Clearly, in order to implement the zone design we need a measurement metric for each of those criteria, so every one of the criteria needs to have some way that can be repeatedly recalculated as we step through that iterative movement of the building blocks and comparison of the swaps. So clearly we could simply disallow any zones with a population coming out smaller than the threshold value and say that's not a permissible move. We could solve using all the building blocks within a local authority and then we will be confident that none of those zones have crossed the local authority boundary. But for those soft constraints such as meeting a target population size, we might use a measure such as minimization of the sum of squared differences from the target size. That would make zones which are a long way from the target very unattractive in the solution and we would always seek to minimize that aggregate squared distance across the map.

Similarly, if we wanted to work on compactness of geographical areas we could, for example, seek to minimize the sum of the a statistic such as perimeter squared by area. A circle will be the shape with the smallest perimeter squared by area, and therefore we have a circularity index which means the shapes more like circles will be more attractive in the solution. So having specified a set of building blocks, and a set of design constraints, we can effectively run to an initial random aggregation of the building blocks and at the very first iteration we measure each of those constraints and then we try to go into the swapping process and evaluate each possible swap to see how our design constraints have been improved or perhaps worsened by that potential swap. Broadly speaking, we will progress by keeping the improving moves and rejecting the non-improving moves and eventually will create such tracked output areas which are the best achievable outcome, given the number of iterations which we've adopted. There isn't any right answer to how far we should keep going, but what we'll often find in these kinds of problems is that we get a rapid improvement in the solution which then gradually trades off until very, very many moves don't make any further improvement.

We have an example here of how it works using the same small map of building blocks that we looked at just now. And here, I have an initial random aggregation of those building blocks into potential tracts. There are just three of them in this map and the positions of the addresses and the streets are still maintained. So we have a blue potential tract on the left, green one on the right and a small yellow one. And we might surmise here that the yellow one's probably going to be rather small if the objective was to have zones of equal population size. So, there's our initial random aggregation and as we begin swapping we highlight one particular building block and experiment by swapping it: in this case, from the blue zone into the green zone and the statistics which are describing the solution are then re-evaluated. What we're very likely to find in this circumstance is that the blue zone has got smaller – maybe that's what we wanted – but the green zone is probably getting too large, and in addition those shapes are very irregular and if circularity, for example, were a metric we were using, this would look quite an unattractive solution and therefore we may decide at this point that's not a good move and we'll go back, we'll reject it, back to the start. And then we'll pick up a new potential swap, in this case a move from blue to yellow, and then we re-evaluate, we recalculate the criteria statistics and here we may see that we've improved both the shapes of the blue and the yellow: blue was too big, we've made it smaller; yellow was too small, we've made it bigger on population terms and this actually might be quite an attractive move in which case we would retain it and then proceed to another swap somewhere else in the map. So we're gradually incrementing towards a more optimal solution with each iteration

So, in this sense we see that the building blocks need to be available and they have a big impact on the final solution. Placement of their boundaries and having all the necessary data are preconditions for us to begin with the zonation. Building blocks may be needing to be constructed, for example from addresses or from small geographical – units postcodes, street blocks – which are appropriate to the research need. But, providing we have the data, we can put those building blocks into the automated design process. We also are going to have to be very clear what are our intended criteria, and this usually involves a process of reflection. If there are users of the research, it may involve discussing with them the characteristics of an optimal geography: whether or not it's important that the zones to be created nest within some higher units such as local government geography; whether they're built from some particular elements such as the postal system and whether they are aligned with things such as roads, rivers which can be seen in the physical geography. And when we have those sets of information ready to commence with, then we can use automated zone design in order to progress towards our zonation solution.