Deception ESSF TSP

This document describes the different tests being used for the travelling salesman problem. Given a set of helicopter drop points, there are multiple ways we could choose clhs points. I’m skeptical about using slices, with one slice per drop point, because the initial slices create a huge spread to try and fill the variable space. I think a better strategy is to create the cost layer based on all drop points, so the clhs knows the full range of low cost areas. Of course, this strategy creates some issues because the points are not equally distributed between the drop sites. The tests below demonstrate this problem and some possible solutions. All tests assume we want 50 transects, and that a transect takes 45 min.

***Some notes (WHM).***

***The intention to the TSP application is to further constraint cLHS placed field transect sampling to improve data collection efficiency and to allow a reasonable estimation of actual time costs for a given sample plan.***

***For the TSP to work optimally it will have limited impact on the cLHS sampling which already applies a cost constraint. It is intended to help cluster sample sites to ensure that desired number of sites/day can be realistically collected. This not only improves overall efficiency of sampling but will also allow cost control for administration of ‘per transect” contracts.***

***An evaluation of sample plan coverage of landscape variable space should be carried out to ensure that that a sample plan is adequate.***

1. Heli\_All\_OrigCost: This is a straightforward clhs + TSP with the four covariate layers plus lat and long in the clhs. There is no time restriction on the TSP optimisation. Obviously, this is not a practical setup, because some distances are huge. *THIS IS THE ROUGH STARTING POINT AND CAN BE DROPPED*
2. Heli\_NoLatLong: Same as #1, but this time lat and long not included in clhs. The clustering of points looks much better, but note how some sites (esp ESSF3) only have two sample points. This means that the TSP isn’t able to fit all the sites without having an unreasonable amount of time. Also something odd happening with the TSP for ESSFw3 and ESSFw4 – I think due to having an excessive time limit. *I THINK NO INCLUDING LAT LONG IS OK AND CAN BE ELIMATED FROM FUTURE LAYOUT OF TRANSECTS AROUND A SITE. HOWEVER, WE MAY WANT TO INCLUDE THIS IS THE DETERMINATION OF STARTING LOCATIONS FOR DAILY SITE CLUSTERS*
3. Heli\_TimeLimit: Same clhs points as #2, but now I’ve set an 8 hour time limit for each circuit (this includes ~~45min~~ 1HR (to allow for sites away from the start point) and I’ve allowed it to create two circuits from one drop point if necessary. This in general looks better, but there are some inefficiencies where a whole trip is made for one transect (ESSF4)[*i THINK THIS MAY BE REASONABLE AS IT IS LIKELY HIGHLIGHTING THAT THAT LOCATION HAS BETTER COVERAGE OF THE VARIABLE SPACE – IT MAY INDICATE THE LIMITED SET OF AVAILBLE ZERO COST STARTING POINTS IN THIS TEST EXAMPLE*.
4. Heli\_DropAllowed: To try and deal with the previous problem, I added functionality to the TSP optimisation which allows it to drop a point if necessary to fit in the time limit (dropping a point comes with a penalty – currently all set the same, but could be different for each point). Now only one circuit from each drop is allowed. I like the output of this run; in a lot of cases it’s exactly the same as #3, but it’s dropped three points to allow it to fit the time limit.
5. Heli\_SlicedBuffered: Dropping points isn’t ideal, because it makes it no longer be a latin hypercube (although in reality it’s probably not a huge deal). To create points more evenly distributed around the drop sites, I created a sliced design, but limited the points to within a 4 km radius of the drop site. The resulting routing seems pretty sensible. The problem with this strategy is that limiting the point space requires limiting the covariate space, so the clhs is only based on the 4km circles around the drop points. Note to self: I played around with adjusting the GlobalArcCostCoefficient (which sets how the longest path is combined with the overall length); I decreased it substantially from 100 to 10, which seems to give better routes.
6. Heli\_Included\_Unsliced: I’ve now added the previously sampled points in the ESSF (although two of the points aren’t actually ESSF). Here, I don’t think the slicing strategy used in #5 is an option, because the buffers around the drop points would exclude a lot of the previously sampled points. To fit in 10 days, the optimisation has had to drop two points. Including the previous samples has decreased the number of points at the top right portion, and added more in the middle, which has created some long days with 7 transects (7:55 with no breaks).
7. Heli\_Included\_MultiDay: Same sample locations as #6, but now allowing two circuits from each drop site. Since there are two dropsites with very few points, these could perhaps be added onto a shorter day.
8. Heli\_Included\_ShortDay: In a sampling strategy with a morning and afternoon drop point, we could set a shorter time limit (in this case 4 hours) and combine two circuits per day. This sampling plan only has 25 transect locations, but could get completed in 5 days. However, this strategy doesn’t seem to work well with the clhs points given the drop locations, because the clhs points are so clustered in the centre. Perhaps a redistribution of heli drop points could make this strategy work.