# Ambient Social Frontier code notes April 2022

Repo: https://github.com/DanOlner/socialfrontier\_sim

..

* spatialOptim\_w\_cpp.R: main script with most of the work in. Like most of my code, not very tidy! You'll see a lot of repetition as code from earlier sections copied later for trying different things.
  + Sections/lines/code worth looking at:
    - DI (line 17): setting up simple grid, targetting/finding DI via C++
    - OPTIMISING ACD GIVEN A TARGET DI (139): uses optimiseAverageAbsoluteContiguousDifference C++ function, either targeting min or max depending on maximise flag (max=T is maximise). Turns out it's usually easier to find a good min just by sorting cells by size.
    - SET PROPORTION, TARGET DI, GET SPREAD OF ACD VALUES (200): Some additions to above:
      * targetDissimilarityIndex keepproportions argument: if cells have a fixed pop, make sure it stays fixed.
      * 270: getAverageAbsoluteContiguousDifference
      * 282: getRepeatedAACDfromPermutedCells – does the permuting, uses the previous many times, permutes cell locations, returns all the AACDs in a vector
    - 300-525 is just the parameter sweep code in the figure I sent you, prob not useful
    - A lot of the code after this is trying all the bzillion different tweaks to the basic AACD calculation that Nema and I tried, chasing after Gwilym's two pretend cities being either side of the permute mean. (See 576, SMALLER TOY EXAMPLE BASED ON GWILYM'S LITTLE CITY PLOTS, for code for Gwilym's example cities; 3243 PREPPING 16X16 SIM CITIES makes 16\*16 version, twice the size)
      * One useful part: there's a cutoff argument for AACD values in any of the C++ functions. If the abs contig difference between two neighbouring cells is below the cutoff, it's not entered into the average calc at all.
    - getNeighbourIndexAACD (used on e.g. line 632): gets AACD from a neighbour list produced using spdep (see lines before 632), so can work with any geography at all, not just grid.
    - REAL GEOGRAPHY WITH CUTOFF (1338): uses getNeighbourIndexAACD with actual geographies including for Sheffield.
      * Here (1423 on) there's an optimiseGetNeighbourIndexAACD C++ function used for doing the same min/max hill climb thing for any geography.
  + frontierlength.R:
    - Used for the google doc writeup. That short doc explains both AACD and how frontier length is found (the latter only for grid squares, however.)
    - Main thing this script is doing: working with 16x16 grid, creating a network object of frontier components (i.e. each differently coloured thing in the frontier length image in the google doc) using a neighbour contiguity matrix