NYC Scenario Derivation

This notebook identifies the source data for NYC tradeoffs analysis and explains how a location allocation scenario is generated. Before doing this, you need to create a layer detailing the sites that could host future gardens. This is detailed in the [Facility derivation](https://docs.google.com/document/d/1YoSsH2eBXz2hcYNuirhtfQKsxvzvLAsXEUbuWgMfpyY/edit?usp=sharing) file.

Scenarios are derived by creating a combination of facilities and demand points data that reflect the overarching research question. If you are interested in assessing the impacts of transportation on these scenarios, you could also vary the network map, but that is beyond the scope of this exercise.

Scenarios build from: 1. The multi-criteria layers generated to describe available land in NYC; and 2. The household dataset, which can then be overlain with additional criteria layers to create weights.

Our scenarios of interest are:

1. Sustainability planning: Focused on food waste and building energy conserved
2. Resilience planning: Focused on stormwater and heat remediation, as well as local food production
3. Justice planning: Focused on equitable access, including income and racial equity indicators

These translate to layers with the following weights:

| **Scenario** | **Demand points composition** | **Facilities composition** |
| --- | --- | --- |
| Sustainability | Households, weighted by likely # of residents (incorporating vacancy) | Feasible sites, weighted by: 1. Amount of food waste within X meters; 2. Rooftop surface area covered |
| Resilience | Feasible sites, weighted by: 1. Total rooftop area within X meters; 2. Heat island intensity; 3. Total food production |
| Justice | Feasible sites, weighted by green space availability in the area and the # of low income residents as well as # of non-white residents (incorporating vacancy) |

For each of these, we need to know basic garden characteristics, like their capacity to use compost, their capacity to produce food, and their use of irrigation. This will remain the same across scenarios. So, this translates into the step-by-step derivation below.

# Sustainability layers derivation

Sustainability scenarios focus on providing access to the most people, recycling food waste, and conserving building energy.

To accomplish this, you need to know the number of households, their likely occupancy, and the rooftop area of buildings per parcel. In addition to garden data, you also need to be able to convert households to food waste generation. This is likely to vary by city, and the more local the data, the better. The data sources for these are as follows:

| Used in | Variable | Layer name | Scale | Basic contents | Source |
| --- | --- | --- | --- | --- | --- |
| Demands points | Households | PLUTO | Parcel level | Varied - includes whether or not a site is residential | NYC Planning |
| Demands points | # of residents per household | Decennial census Table H9. Household Size | Block group | Avg household size | US Census Bureau, NHGIS |
| Demand points | Vacancy rate | ACS B25002. Occupancy Status | Block group | Total, occupied, and vacant households at census block group level | US Census Bureau, NHGIS |
| Facilities | Facilities base layer | Facilities - Base - PLUTO format | Parcel level | Includes a variety of PLUTO data + the possible area of each of six types of gardens/farms | Derived |
| Facilities | Usable rooftop area of buildings | Land Cover | 1m raster | Derived - identifies rooftops along with concrete, grass/dirt, and trees | Derived |
| Facilities | Food waste per person | – | – | Food waste per person - aspatial, all of NYC | … |

## Demand points

It’s a fairly simple derivation to get the residential buildings and their residential units:

1. Create a Points on Surface file from PLUTO (instead of centroids, since centroids can escape the bounds of a polygon and we want our points to be on the surface).
2. Filter to only residential land uses (LandUse = 01, 02, 03, 04). Use Select By Expression ( "LandUse" = 01 OR "LandUse" = 02 OR "LandUse" = 03 OR "LandUse" = 04), then right click and export only the Selected Features. In the process, remove most of the columns, we don’t need them. Just make sure to keep the UnitsRes variable. I also keep the identifying variables out of habit as well as some of the floor area and zoning information. Like I said, purely a habit.
3. Pull in residents per unit from NHGIS (Average Household Size of Occupied Housing Units by Tenure available from NHGIS for ACS) at the census block group level - also make sure to grab data on vacancy for the calculation (Table H1, Occupancy Status, Decennial Census or ACS B25002. Occupancy Status).
   1. To do this, you load the census block group shapefile, load the NHGIS csv’s, and use Join in properties to connect the data. I recommend clipping the shapefile to the bounds of the five boroughs with a select by expression and export selected combination again. You can select them by focusing on the five counties – COUNTYFP = 005, 047, 061, 081, 085. Load the csv’s and use the join feature in Properties to join the data to the shapes with the GISJOIN field.
   2. Once this is loaded at the census block group level, use a Join Attributes by Location to get this at the household level.
4. You should be able to estimate the number of residents by multiplying the number of units by the residents per unit by vacancy. You can get the weight of this demand point by multiplying that by the total number of units at a property. This can be broken down into a couple calculations:
   1. OccupRate = "U7G002" / "U7G001" (make sure the output is a decimal)
   2. OcupPerRes = OccupRate \* "AMUUE001" (again, decimal output)
   3. DmndWeight = "UnitsRes" \* "OcupPerRes" (decimal output)

## Facilities layer

The facilities layer derivation is a bit more complicated. So we start with the feasible sites on rooftops and on the ground. We know this at the parcel level, where we have the total available rooftop area and ground area for each parcel (categorized by land use, etc. into community, individual, or farm space). This has been converted from a wide dataset into a long dataset which sorts the different types of sites. See the [Facilities derivation](https://docs.google.com/document/d/1YoSsH2eBXz2hcYNuirhtfQKsxvzvLAsXEUbuWgMfpyY/edit?usp=sharing) for more information.

### Food waste

Since the facilities layer will already contain the area of roof covered, we don’t need to stress about adding that. On the other hand, we do need to calculate the food waste generated within a certain distance of each facility and compare that to the amount of food waste a facility can absorb. These two calculations are separate but related. We can identify the amount of food waste absorbed with just information about the site and some basic coefficients. Here’s the basic derivation:

1. Collective gardens in FEW-meter an average of 15.97 kg of compost/m2 productive area.
2. Conversion from food waste to compost is something like [30% yield](https://www.sciencedirect.com/science/article/pii/S0959652620312671) on a mass basis.
3. So food waste absorption (using field calculator) = ( 15.97 / 0.3) kg/m2 \* AreaOfPossibleFarm = kg food waste absorbed.

This formula recreated for all three types of sites:

1. Collective garden absorption: 15.97 / 0.3 \* AreaOfPossibleFarm
   1. We also treat the existing gardens as collective gardens numerically.
2. Individual garden absorption: 0.424 / 0.3 \* AreaOfPossibleFarm
3. Urban farm absorption: 3.86 / 0.3\* AreaOfPossibleFarm

This translates to this field calculation: if( "FacType" = 'Collective Garden', 15.97 / 0.3 \* "Area", if( "FacType" = 'Existing', 15.97 / 0.3 \* "Area", if( "FacType" = 'Individual Garden', 0.424 / 0.3 \* "Area", if( "FacType" = 'Farm', 3.86 / 0.3 \* "Area", 0))))

We then need to compare this to another attribute - the total amount of food waste generated by the neighbors of a garden. Although it would technically be more accurate to consider neighbors from an actual access perspective, it would also be very computationally difficult - so we assume Euclidean distances for this particular part. In other words, we can assign a food waste value to every household. Then we can buffer each facility by a certain distance, use Join Attributes by Location (summary) to add up all food waste, and compare this value to the potential absorption. Here’s the basics of this derivation:

1. Food waste generation is calculated by combining the demand weight values derived above (total occupancy per site) with the food waste per capita values from an [NRDC study](https://www.nrdc.org/sites/default/files/food-waste-city-level-report.pdf).
   1. It may actually also be possible to get a little fancier with the calculations of food waste, since this study also assessed existing composting habits, etc. Not doing that for this work.
   2. NYers were found to waste 3.2 lbs/per person/week (1.4515 kg - pg 20). This can be combined with household size data and used to generate weekly and annual values per household at the census block level.
2. So the detailed steps to generate this layer:
   1. Load the demand layer - it doesn’t hurt anything to clutter it with the field calculated compost value, so we’ll just do it within the demand centroid layer.
   2. DmndWeight \* 1.4515 \* 52 = food waste per parcel per annum in kg
3. To pull this into the facilities layer, we can use a simple buffer and zonal statistics. First, run the buffer function on the Facilities layer - 440m, or ¼ mile for delivery of the food waste. You’ll want to run this with 20 segments, and round ends with round joins - this group of settings is most amenable to unbuffering later on. Once we have a buffered file, Rasterize the Demand file with Food Waste Per Parcel as the burn in value. Finally, run Zonal Statistics with the buffered polygons and the new raster file - only need the sum, really, but keeping the count might also be useful at some point.
4. Next, you need to figure out what food waste value you’re actually interested in - absorbed or produced. The field calculation should look like this: if("FdWstAbsrd" > "FdWst\_sum", "FdWst\_sum", "FdWstAbsrd")

### Final cleaning

Now we have a polygon layer with all the basic data we need. We need to convert this into a point layer with a simplified indicator for the facility weight. To do this, we first need to unbuffer this layer. We take the computer layer and unbuffer it by 440 m (buffer of -440 – make sure it is set to the same settings as before, since this gives better approximations of the original shape). Next, we convert this into a “point on surface.” We do this instead of centroids in case centroids end up outside the polygon in many cases. Once we have the point file, we need to develop the final layer weight. To do this, we’ll take a percentile of the two variables (compost and roof space) and average them. For me, this is again easiest to accomplish in R, though as before it could be done in Python if you’re so inclined. Once it comes out of R, it’s ready to go straight into the Location Allocation program. Here’s the code for R:

########### Sustainability final cleaning ############

## Import the point file extracted from QGIS.

Sustainability\_raw <- st\_read("U:/Jake/Urban Gardens/Location allocation/NYC/Facilities Layers/SRJ layers/Sustainability/Facilities - FdWst Points.shp")

head(Sustainability\_raw)

# Convert weight variables to percentiles.

Sustainability\_raw %<>% mutate(FdWstFin = ifelse(FdWstAbsrd > FdWst\_sum, FdWst\_sum, FdWstAbsrd))

Sustainability\_percentiles <- Sustainability\_raw %>% mutate(

RoofPerc = ntile(RoofArea, 100),

FdWstPerc = ntile(FdWstFin, 100)

) %>% select(-IndRoof\_su,-FarmGrnd\_s,-CommGrnd\_s,-IndGrnd\_su,-CommRoof\_s,-FarmRoof\_s,-WebAddr,-GardenOrg,-YrOfOrigin,-Purpose,-layer,-path,-FdWst\_coun)

Sustainability\_facilities\_final <- Sustainability\_percentiles %>% mutate(SusWeight = ((RoofPerc + FdWstPerc) / 2))

# Resilience layers derivation

Resilience scenarios focus on providing access to the most people, diverting stormwater runoff, offsetting the urban heat island effect, and producing local food.

To accomplish this, you need to know the number of households, their likely occupancy, the rooftop area of buildings per parcel, and the relative heat island intensity in the immediate vicinity of each parcel. The data sources for these are as follows:

| Used in | Variable | Layer name | Scale | Basic contents | Source |
| --- | --- | --- | --- | --- | --- |
| Demands points | Households | PLUTO | Parcel level | Varied - includes whether or not a site is residential | NYC Planning |
| Demands points | # of residents per household | Decennial census Table H9. Household Size | Block group | Varied - includes average household size | US Census Bureau, NHGIS |
| Demand points | Vacancy rate | ACS B25002. Occupancy Status | Block group | Total, occupied, and vacant households at census block group level | US Census Bureau, NHGIS |
| Facilities | Facilities base layer | Facilities - Base - PLUTO format | Parcel level | Includes a variety of PLUTO data + the possible area of each of six types of gardens/farms | Derived |
| Facilities | Rooftop area of buildings | Building footprint | Building | Varied - includes shapes so we can calculate proximate area. | NYC Planning |
| Facilities | Temperature | NYC Heat | 1m raster | Avg LST | NYC Planning |

## Demand points

This is exactly the same layer as sustainability, so we can just use that.

## Facilities layer

### Heat data

The heat data we use for NYC comes from the city’s own analysis of heat island effects based on land surface temperature. There is a substantial debate on the suitability of LST as an indicator of heat island effects, but it’s a good enough proxy for the experience of the city that we can use it for this modeling exercise. See below for some notes on alternatives. You can find the data [here](https://github.com/NewYorkCityCouncil/heat_map/tree/main). It’s a raster, so we can just use zonal statistics to summarize the heat effect in each parcel. Later on, we’ll run a percentile to normalize this, but for now the raw variable is all we need. Personal preference on what indicator to use, I initially used mean, median, st-dev, max and min to get a feel for it, though I only ended up using mean.

Measuring heat in general - notes from Sara Meerow:

The easiest and certainly the most common is probably using remotely sensed Land surface temperature data. There are various sources for that (e.g. google earth engine). There are some existing preprocessed datasets you could potentially just adopt, for example this one: <https://asu.elsevierpure.com/en/publications/a-spatially-explicit-surface-urban-heat-island-database-for-the-u>

Or you could use a city-specific source like this one: <https://council.nyc.gov/data/heat/>

Of course there are lots of critiques of LST for heat, some people are trying to develop better models for more relevant indicators like mean radiant temperature, this one could be a possibility, Xiaojiang has created datasets for a lot of US cities including NYC already: <https://xiaojianggis.github.io/heatexpo/>

### Stormwater data

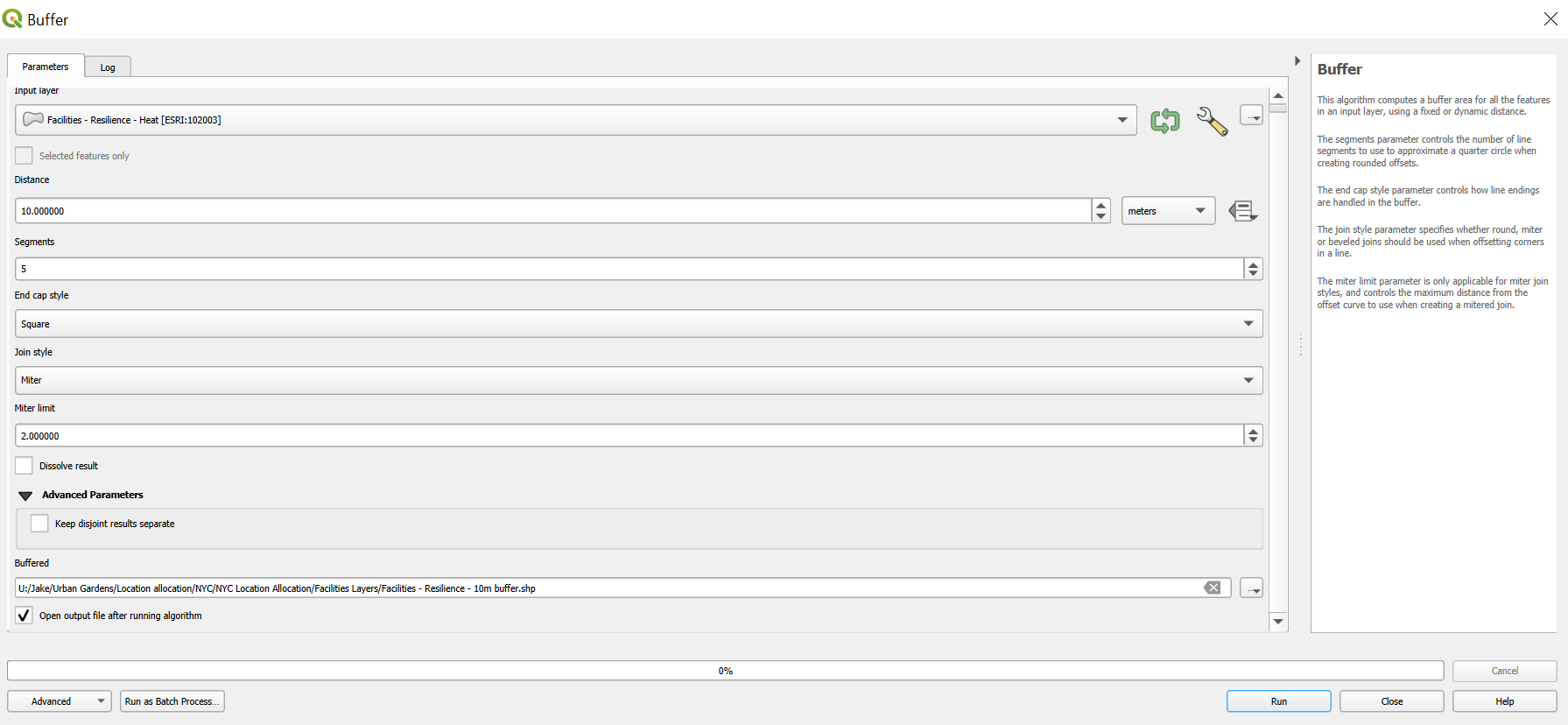
The stormwater measure relies on functionally the same process as the compost work above. We need to compare production with absorption and take the lower number. Absorption just requires knowing how much water is used at each type of facility, production just requires knowing the roof area around a garden and the amount of rain in a year.

Collective gardens in FEW-meter used between 32.68 and 514.06 L of water/m2, with an average of 226.87 L/m2. So we just calculate an attribute by multiplying area by 226.87 L/m2 \* garden area (m2). This leaves us with L absorbed over the course of a year. Translated for each type of garden, this is:

1. Collective garden absorption: 226.87 \* AreaOfPossibleFarm
2. Individual garden absorption: 55.71 \* AreaOfPossibleFarm
3. Urban farm absorption: 733.73 \* AreaOfPossibleFarm

This translates to this field calculation: if( "FacType" = 'Collective Garden', 226.87 \* "Area", if( "FacType" = 'Existing', 226.87 \* "Area", if( "FacType" = 'Individual Garden', 55.71 \* "Area", if( "FacType" = 'Farm', 733.73 \* "Area", 0))))

We don’t necessarily need to calculate the actual rainfall for each building in NYC - it’s going to be the same per square meter of roof. Instead, we just need to aggregate the roof area surrounding the garden. Since the default building layer comes as a projected CRS, we have to transform this to 102003, then use the $area function in the field calculator to calculate an actually informative area column. To do this, we take the facilities layer where we just calculated the absorption potential, and we buffer it by ten meters.



Instead of rasterizing everything in the demand file, we keep our polygons and use Join Attributes by Location (Summary) to document the total roof area of the surrounding buildings. This is both necessary and feasible in this case, because we need to know where the edges of the buildings are, and we have a much smaller area to cover, so the computation actually runs. Trying to do this with the quarter mile buffer takes a very long time.It will save you time and trouble if you go ahead and select the wrench next to both inputs on this function and change the error handling to Ignore - this is just because some of the geometry of the buildings layer is not perfect and there is overlap in the buffered facilities file.   


### Final cleaning

Once we have this value, we need to unbuffer and generate a file of points, just as we did for sustainability. Once again, use the Point on Surface tool instead of Centroids for the same odd shapes reason. Once we have this point file, we can once again use R to develop a single weight indicator based on percentiles. Here’s the code:

########### Resilience final cleaning ############

## Import the point file extracted from QGIS.

Resilience\_raw <- st\_read("U:/Jake/Urban Gardens/Location allocation/NYC/Facilities Layers/SRJ layers/Resilience/Facilities - HW Points.shp")

head(Resilience\_raw)

# Convert rooftop area to rain liters of water used and determine whether rainfall or water diverted is greater.

# Rainfall during the growing season is 0.89084191 m.

Resilience\_raw %<>% mutate(Rainfall = BuiltArea\_ \* 0.89084191,

WaterAbsrd = WaterAbsrd / 1000, # convert water use to m3 as well

WaterSaved = ifelse(WaterAbsrd > Rainfall, Rainfall, WaterAbsrd), # compare values and take the lower

FoodProduced = ifelse(FacType == "Collective Garden", 2.07375 \* Area, # Add food produced as well

ifelse(FacType == "Individual Garden", 1.702909091 \* Area,

ifelse(FacType == "Farm", 2.617777778 \* Area, 0)))

)

# Convert weight variables to percentiles.

Resilience\_percentiles <- Resilience\_raw %>% mutate(

FoodPerc = ntile(FoodProduced, 100),

WaterPerc = ntile(WaterSaved, 100),

HeatPerc = ntile(Heat\_mean, 100)

) %>% select(-IndRoof\_su,-FarmGrnd\_s,-CommGrnd\_s,-IndGrnd\_su,-CommRoof\_s,-FarmRoof\_s,-WebAddr,-GardenOrg,-YrOfOrigin,-Purpose,-FdWstAbsrd,-layer,-path)

Resilience\_facilities\_final <- Resilience\_percentiles %>% mutate(ResWeight = ((FoodPerc + WaterPerc + HeatPerc) / 3))

# Justice layers derivation

Justice scenarios focus on providing access to the highest number of low-income, non-white households.

To accomplish this, you need to know the number of households, their likely occupancy, the racial composition of the neighborhood, and the amount of green space in the immediate vicinity of each parcel. The data sources for these are as follows:

| Used in | Variable | Layer name | Scale | Basic contents | Source |
| --- | --- | --- | --- | --- | --- |
| Demands points | Households | PLUTO | Parcel level | Varied - includes whether or not a site is residential | NYC Planning |
| Demands points | # of residents per household | Decennial census Table H9. Household Size | Block group | Varied - includes average household size | US Census Bureau, NHGIS |
| Demand points | Vacancy rate | ACS B25002. Occupancy Status | Block group | Total, occupied, and vacant households at census block group level | US Census Bureau, NHGIS |
| Demands points | Racial composition | ACS Table B02001. Race | Block group | Various racial indicators - easy to calculate non-white. | US Census Bureau, NHGIS |
| Demands points | Economic composition | ACS Table C17002. Ratio of Income to Poverty Level in the Past 12 Months | Block group | Income levels relative to poverty rate | US Census Bureau, NHGIS |
| Facilities | Facilities base layer | Facilities - Base - PLUTO format | Parcel level | Includes a variety of PLUTO data + the possible area of each of six types of gardens/farms | Derived |
| Facilities | Land cover | LC | 1m resolution raster | Derived as part of facility siting, identifies green space in NYC | Derived, see [Facilities layer derivation](https://docs.google.com/document/d/1YoSsH2eBXz2hcYNuirhtfQKsxvzvLAsXEUbuWgMfpyY/edit?usp=sharing) |

## Notes on overall procedure

1. Demand: Weighted by justice indicators - Done.
2. Facilities: Weighted only by green space indicators and the relative social impact of each type of facility 1, 0.5, 0.5 or 1, 0.5, 0?

## Demand points

Same demand points as the other layers.

## Facilities points

We consider the green space access in the area around the potential garden, as well as the race and poverty rates.

Green Space

We’ll start with green space, so we’ll run zonal statistics on a modified land cover raster to detect percent green space (take the mean because this will be binary, give you a proportion) in the neighborhood. Take the original land cover raster and run the following raster calculation: if( "LC@1" = 2 OR "LC@1" = 4, 1, 0 ). Once we have this binary layer, we need to buffer the facilities and run zonal statistics. In this case, a reasonable buffer would seem to be the walking distance metric we’re using, which is ¼ mile or 440 meters. We should have a 440m buffer file sitting around from earlier, or you can create a new one if you lost that one. We can run both sum and mean to get the total area and the percent area.

### Poverty and Communities of Color

We will also prioritize gardens within walking distance of Communities of Color and communities with higher poverty rates. To get started, we’ll join race and poverty data from NHGIS to census block groups (also from NHGIS), then calculate rates in each block group. More details on this can be found at NHGIS’ [website](https://www.nhgis.org/gis-files). Put simply, you download the race and poverty tables from the most recent ACS, and also download the NYC Census Blocks file (presumably you’ll be using the 2020 file). Once you have those files, pull them all (including the csv’s) into QGIS. Use GISJOIN to join the data you’re interested in from the Census Block Groups property menu. To identify which fields you need, you use the accompanying text files which provide the attribute metadata.

You may have to convert the poverty data to decimal data from string first. I’m not sure why it imports it as string, but it does. to\_real should do the trick followed by the simple field calculations.

* Poverty rate: HalfPov = to\_real(“p\_AMZME002), Pov = to\_real(“p\_AMZME003), TotalPop = to\_real(“p\_AMZME001)
  + PovRate = (HalfPov + Pov) / TotalPop \* 100
* Non-white pct: (1-( "r\_AON5E002" / "r\_AON5E001" )) \* 100

Once you have these values ready, you can rasterize and run zonal statistics - once again, model builder will help avoid multiple files.

### Final cleaning

You actually don’t have to unbuffer and generate points for these if you don’t want to, since the id’s are what we use to join in R. Ultimately up to you. Once you’re ready to move things to R, the following code should do the trick:

########### Justice final cleaning ############

## Import the point file extracted from QGIS.

Justice\_raw <- st\_read("U:/Jake/Urban Gardens/Location allocation/NYC/Facilities Layers/SRJ layers/Justice/Facilities - GS NW PR.shp")

# Convert weight variables to percentiles.

Justice\_percentiles <- Justice\_raw %>% mutate(

GS\_need = 1 - GS\_mean,

GSPerc = ntile(GS\_need, 100),

PovPerc = ntile(PR\_mean, 100),

NWPerc = ntile(NW\_mean, 100)

) %>% select(-IndRoof\_su,-FarmGrnd\_s,-CommGrnd\_s,-IndGrnd\_su,-CommRoof\_s,-FarmRoof\_s,-WebAddr,-GardenOrg,-YrOfOrigin,-Purpose,-FdWstAbsrd,-layer,-path)

Justice\_facilities\_final <- Justice\_percentiles %>% mutate(JusWeight = GSPerc) # ((GSPerc + JustPerc) / 2))

# Post-Cleaning - Selecting the facilities of interest

Since we need a facilities layer for each location allocation, we use R to pull out the Sustainability, Resilience, and Justice facilities of interest. It’s also convenient at this point to join all the layers so we have the SRJ data in each layer.

########## Combining layers and tagging the top 300 in each category. ###########

## First, we join aspatial versions of the resilience and justice data to join with a column (ids).

SRJFacilities <- Sustainability\_facilities\_final %>% left\_join(Resilience\_facilities\_final %>% as\_tibble() %>% select(-geometry)) %>% left\_join(Justice\_facilities\_final %>% as\_tibble() %>% select(-geometry))

ExistingFacilities <- SRJFacilities %>% filter(existing == 1)

SRJFacilities %<>% filter(existing != 1)

## Explore what the cutoffs might look like - how many rows at certain values of weights?

# nrow(SRJFacilities %>% filter(SusWeight >= 97))

# nrow(SRJFacilities %>% filter(ResWeight >= 95))

# nrow(SRJFacilities %>% filter(JusWeight >= 90))

## Pull the top 300 rows - this is actually not likely to be exactly 300, since it also pulls all tied rows.

TopSusFacilities <- SRJFacilities %>% slice\_max(SusWeight, n = 300)

TopResFacilities <- SRJFacilities %>% slice\_max(ResWeight, n = 300)

TopJusFacilities <- SRJFacilities %>% slice\_max(JusWeight, n = 300)

# Pull the existing gardens back in for optimization.

TopSusFacilities %<>% rbind(ExistingFacilities)

TopResFacilities %<>% rbind(ExistingFacilities)

TopJusFacilities %<>% rbind(ExistingFacilities)

# Write them to file.

st\_write(TopSusFacilities, "U:/Jake/Urban Gardens/Location allocation/NYC/Facilities Layers/SRJ layers/Sustainability/Sustainability - Top Facilities.shp")

st\_write(TopResFacilities, "U:/Jake/Urban Gardens/Location allocation/NYC/Facilities Layers/SRJ layers/Resilience/Resilience - Top Facilities.shp")

st\_write(TopJusFacilities, "U:/Jake/Urban Gardens/Location allocation/NYC/Facilities Layers/SRJ layers/Justice/Justice - Top Facilities.shp")