Potential gardens derivation and post-hoc analysis

# Derive LU, LC, Slope, and Sunlight layers for all case study cities

Can use the writeups for each city to guide derivation.

1. [Dortmund](https://docs.google.com/document/d/1P3FBj-uuGyviD8tYvpNZh8eToblGUd38ITeChutXnek/edit?usp=drive_link)
2. [Detroit](https://docs.google.com/document/d/1bWgMKp40i6dT949iprVoX0UrYW69Ixmz6f5D3drVk5s/edit?usp=drive_link)
3. [Gorzow](https://docs.google.com/document/d/1_HM8awFJ_533D_hPiNg5ZcFH-3PKK7GevHX4vKZEOew/edit?usp=drive_link)
4. [London](https://docs.google.com/document/d/1-sfoYeWW3Zbjyn0M7Q8kZFfHDF5Lxsvu_fJRvIYdQco/edit?usp=drive_link)
5. [NYC](https://docs.google.com/document/d/15VKDEastyWHeRrqB4UomDVLdxRRdk3bcbcmwVumfbKw/edit?usp=drive_link)
6. [Paris](https://docs.google.com/document/d/1g980vclgZj19BdYI_zNpFSxCRZOPfYOTzrcrlkTnuMM/edit?usp=drive_link)

# Calculate suitability rasters

IF-THEN, same across cities. Run for each scenario independently, since the categories overlap between scenarios.

| Scenario | UA type | Coding |
| --- | --- | --- |
| Base | Individual | ("LU@1" = 10 OR "LU@1" = 11) \* ( "Sunlight@1" >= 4 ) \* ("Slope binary@1" = 1) \* ( "LC@1" = 1 OR "LC@1" = 2 OR "LC@1" = 4 ) |
| Collective | ( "LU@1" = 12 OR "LU@1" = 13 OR "LU@1" = 21 OR "LU@1" = 24 OR "LU@1" = 34 OR "LU@1" = 35 OR "LU@1" = 70 ) \* ( "Sunlight@1" >= 4 ) \* ("Slope binary@1" = 1) \* ( "LC@1" = 1 OR "LC@1" = 2 OR "LC@1" = 4 ) |
| Farm | ( "LU@1" = 22 OR "LU@1" = 23) \* ( "Sunlight@1" >= 4 ) \* ("Slope binary@1" = 1) \* ( "LC@1" = 1 OR "LC@1" = 2 OR "LC@1" = 4 ) |
| Rooftop | ( "Sunlight@1" >= 4 ) \* ("Slope binary@1" = 1) \* ( "LC@1" = 3 ) |
| 2a: Ignore sunlight | Individual | ("LU@1" = 10 OR "LU@1" = 11) \* ("Slope binary@1" = 1) \* ( "LC@1" = 1 OR "LC@1" = 2 OR "LC@1" = 4 ) |
| Collective | ( "LU@1" = 12 OR "LU@1" = 13 OR "LU@1" = 21 OR "LU@1" = 24 OR "LU@1" = 34 OR "LU@1" = 35 OR "LU@1" = 70 ) \* ("Slope binary@1" = 1) \* ( "LC@1" = 1 OR "LC@1" = 2 OR "LC@1" = 4 ) |
| Farm | ( "LU@1" = 22 OR "LU@1" = 23) \* ("Slope binary@1" = 1) \* ( "LC@1" = 1 OR "LC@1" = 2 OR "LC@1" = 4 ) |
| Rooftop | ("Slope binary@1" = 1) \* ( "LC@1" = 3 ) |
| 2b: Ignore slope on ground | Individual | ("LU@1" = 10 OR "LU@1" = 11) \* ( "Sunlight@1" >= 4 ) \* ( "LC@1" = 1 OR "LC@1" = 2 OR "LC@1" = 4 ) |
| Collective | ( "LU@1" = 12 OR "LU@1" = 13 OR "LU@1" = 21 OR "LU@1" = 24 OR "LU@1" = 34 OR "LU@1" = 35 OR "LU@1" = 70 ) \* ( "Sunlight@1" >= 4 ) \* ( "LC@1" = 1 OR "LC@1" = 2 OR "LC@1" = 4 ) |
| Farm | ( "LU@1" = 22 OR "LU@1" = 23) \* ( "Sunlight@1" >= 4 ) \* ( "LC@1" = 1 OR "LC@1" = 2 OR "LC@1" = 4 ) |
| Rooftop | ( "Sunlight@1" >= 4 ) \* ( "LC@1" = 3 ) |
| 2c. Remove tree-covered spaces | Individual | ("LU@1" = 10 OR "LU@1" = 11) \* ( "Sunlight@1" >= 4 ) \* ("Slope binary@1" = 1) \* ( "LC@1" = 1 OR "LC@1" = 2) |
| Collective | ( "LU@1" = 12 OR "LU@1" = 13 OR "LU@1" = 21 OR "LU@1" = 24 OR "LU@1" = 34 OR "LU@1" = 35 OR "LU@1" = 70 ) \* ( "Sunlight@1" >= 4 ) \* ("Slope binary@1" = 1) \* ( "LC@1" = 1 OR "LC@1" = 2) |
| Farm | ( "LU@1" = 22 OR "LU@1" = 23) \* ( "Sunlight@1" >= 4 ) \* ("Slope binary@1" = 1) \* ( "LC@1" = 1 OR "LC@1" = 2) |
| Rooftop | ( "Sunlight@1" >= 4 ) \* ("Slope binary@1" = 1) \* ( "LC@1" = 3 ) |
| Only parking | All | ( "LU@1" = 41) \* ( "Sunlight@1" >= 4 ) \* ("Slope binary@1" = 1) \* ( "LC@1" = 1 OR "LC@1" = 2 OR "LC@1" = 4 ) |
| Convert % of parks | Collective | ( "LU@1" = 31) \* ( "Sunlight@1" >= 4 ) \* ("Slope binary@1" = 1) \* ( "LC@1" = 1 OR "LC@1" = 2 OR "LC@1" = 4 ) |

Worth noting that % of rooftops is also a sensitivity analysis, but is done as post-hoc analysis.

# Aggregate gardens and create garden polygons

Once we know where the gardens are, we need to assess how they relate to residential parcels. We don’t really need the gardens themselves in parcels, but we do need to know what are really gardens and what are just a smattering of points. So we do have to run the zonal statistics still for each parcel, so we can get a feeling for where the real gardens are. Here are the layers we use for polygons in each city:

* Dortmund: Nutzung
* Gorzow: dzialki
* London: UKMap
* NYC: PLUTO
* Paris: Parcelle Cadastrale (could’ve used MOS+)

Nutzung and MOS+ are a bit different from PLUTO and others, because these are not property lines but where land use changes. So it’s like we’ve dissolved the shapes already. Most of these still run with the same graphical modeler - just tell the computer where to find all the rasters, then load the shapefile, and send it to work. It should run fairly quickly on the smaller cities, maybe a day or so on Paris. For London, you need to be sure to change the universal error-handling setting if you want to use the graphical modeler, because of the different shapes and holes and islands.

Once we know how much area is in each polygon, we can create a new attribute that just assigns it a type under each scenario - 0, 1, 2, 3, or 4. 4 just indicates it’s only on the roof and we don’t know yet what the farm or garden type is - we correct this in the R code.

BaseType =

CASE

WHEN IndBase\_su >= 10 THEN 1

WHEN CollBase\_s >= 100 THEN 2

WHEN FarmBase\_s >= 100 THEN 3

WHEN RoofBase\_s >= 100 THEN 4

ELSE 0

END

SlpType =

CASE

WHEN IndSlp\_sum >= 10 THEN 1

WHEN CollSlp\_su >= 100 THEN 2

WHEN FarmSlp\_su >= 100 THEN 3

WHEN RoofSlp\_su >= 100 THEN 4

ELSE 0

END

SunType =

CASE

WHEN IndSun\_sum >= 10 THEN 1

WHEN CollSun\_su >= 100 THEN 2

WHEN FarmSun\_su >= 100 THEN 3

WHEN RoofSun\_su >= 100 THEN 4

ELSE 0

END

TreeType =

CASE

WHEN IndTree\_su >= 10 THEN 1

WHEN CollTree\_s >= 100 THEN 2

WHEN FarmTree\_s >= 100 THEN 3

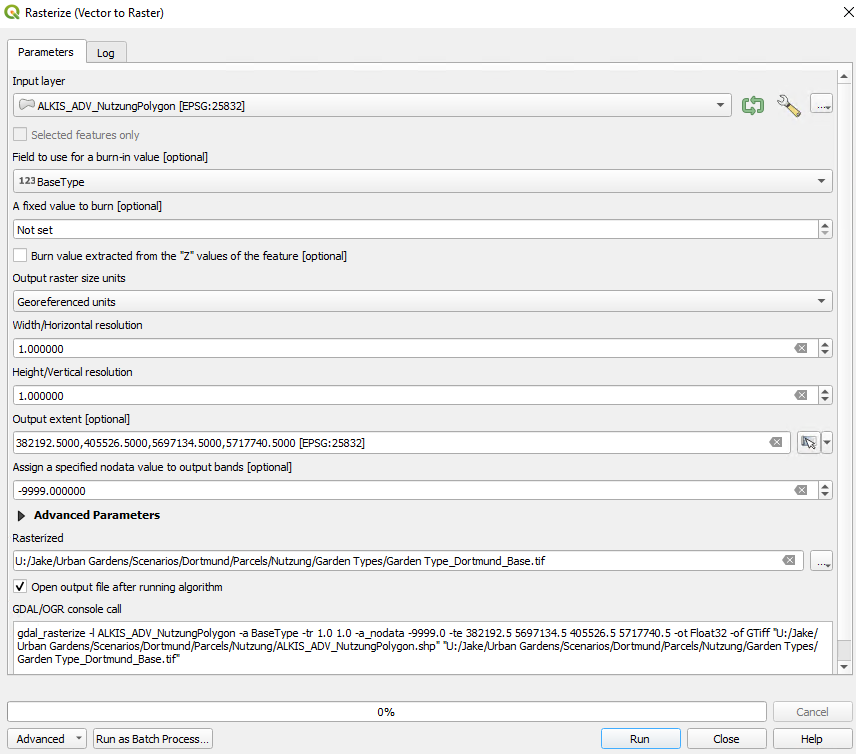
WHEN RoofTree\_s >= 100 THEN 4

ELSE 0

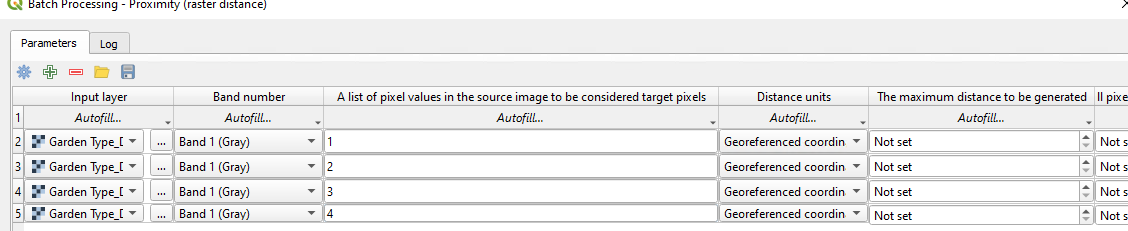
END

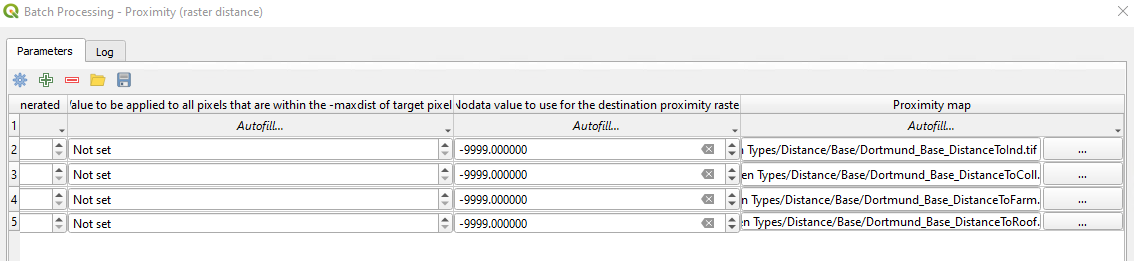
# Proximity analysis

Once the above is done, we can rasterize the new garden types (one raster per scenario with values between 1 and 4) and calculate distance from each residential parcel. To do this, we develop a distance map to each garden type (for each scenario) and then run zonal stats for every residential polygon on this to identify the max (or min?) in each polygon. For ease, it might be helpful to reimport one of the midpoint layers at this point to make sure the new rasters align with the old ones, in case we need to do any comparison at some point.



Once we have this raster, we can do the distance map for each scenario. Unfortunately, this means having 16 rasters per city again, then once again running zonal statistics. We can use the target value feature in the Proximity function to generate the distance maps.





For the next step, we need a map of residential plots. In Dortmund and London, we can pull back out our Land Use polygon file for the residential map. In NYC, we can basically do this, but it’s missing some of the parking lot and park data we added - that’s ok, we don’t need those and can just pull the PLUTO codes directly. For Paris and Gorzow, we need to pull our final land use codes into the parcels - this is simple zonal statistics using the majority. As usual, I recommend the graphical modeler to make it easy to save it to the existing file, especially if you did as I did and made a copy of the parcel file at the beginning of this stage. Once you have all those parcel files ready, do the following:

For Gorzow, London, Dortmund, and Paris:

* Select by attribute value and extract all that equal 10, 11, 12, 13, or 21.
* In some cases, it’s faster this way, in others it might be easier simply to Extract by Expression - e.g., for London FEWM\_LU = 10 OR FEWM\_LU = 11 OR FEWM\_LU = 12 OR FEWM\_LU = 13 OR FEWM\_LU = 21

For NYC:

* Select by attribute value for the residential PLUTO codes: LandUse = 01, 02, 03, or 04

In all cases:

* Export this selection as a new file - or, in the case of Gorzow and Paris where you already had to make a new file, just delete the non-residential parcels by inverting the selection and hitting delete.
* Run zonal statistics for all distance scenarios - calculate the mean. Once again, it’s best to use the graphical modeler so it all gets saved in the original file. You can load the old program, just save it as a new model and change the zonal stats employed to be mean (if you are using a very fast computer, you could add min and max for interesting info, but it’s not necessary and adds time), then delete the parking and park scenarios - they’re not relevant in this particular analysis.

Once we know this, it’s a pretty simple matter of finding the average value across residential parcels. I do this summarization in R, shown in the same Markdown document as the flows analysis discussed below, made available [here](https://drive.google.com/file/d/1PgPcfv_L9yF70GTTKdmKQVwzo4-6UyCH/view?usp=drive_link).

# Non-proximity analysis: summarize rasters - total area, across scenarios - followed my flows analysis

This can be done in R or QGIS. I did it in R, because I felt like the matrix algebra was more streamlined in that format. If you use Python in QGIS, it could be equally streamlined that way. For now, though, the entire process, including the proximity analysis mentioned above, is available in [this](https://drive.google.com/file/d/1PgPcfv_L9yF70GTTKdmKQVwzo4-6UyCH/view?usp=drive_link) R Markdown document.