London Scaling - Midpoint Layer Derivation

This notebook identifies the source data for London scaling and explains pre-processing to output the four basic layers on which scenarios are built:

1. Simplified Land Cover
2. Simplified Land Use
3. Slope
4. Sunlight availability

To replicate this work, first open a blank project in QGIS. I have tested most of this on QGIS 3.18 and 3.24, though anything past 3.00 should suffice. When possible, I will include both a verbal explanation and a screenshot for the procedure described - this should make it easier to replicate the process in varying versions where the syntax or appearance may change slightly.

The goal of this notebook is to translate the input layers into the four basic layers described above. Unfortunately, at the time of writing, this method is already out-of-date. During the development of this project, Geomni (the company that produced UKMap, used here) was acquired by Verisk, and the product will no longer be updated after 2022. The version used in this document is the final version of the dataset. The good news is that this can now be replicated for cheaper, because they aren’t charging as much for an out-of-date dataset, but the bad news is that you won’t be able to use these instructions to produce an updated version of this project unless someone else takes up the reins of the UKMap product. The good news is that most of these data layers are still available as part of the [Ordinance Survey datasets](https://www.ordnancesurvey.co.uk/education) - and these are actually mostly free at UK institutions. We had to buy the polished, proprietary version because our analysis process was based in the US.

* UKMap - [metadata](https://data.cityofnewyork.us/Environment/Land-Cover-Raster-Data-2017-6in-Resolution/he6d-2qns)
* London DSM - [metadata](https://environment.data.gov.uk/dataset/df4e3ec3-315e-48aa-aaaf-b5ae74d7b2bb) and [download](https://environment.data.gov.uk/DefraDataDownload/?Mode=survey)
* London DTM - [metadata](https://environment.data.gov.uk/dataset/13787b9a-26a4-4775-8523-806d13af58fc) and [download](https://environment.data.gov.uk/DefraDataDownload/?Mode=survey)
* London Wards boundary file - [download and metadata](https://data.london.gov.uk/dataset/statistical-gis-boundary-files-london)
* Linke turbidity data - [download and metadata](http://www.soda-pro.com/help/general-knowledge/linke-turbidity-factor)

At the end of this analysis, we will have transformed these inputs into four aligned rasters at 1m resolution. All functions will be conducted in EPSG 27700 (British National Grid) because it works well with solar irradiance actions and is what all the UK govt files are delivered in. Based on those rasters, we will be able to develop a variety of scenarios that capture the possible areas of expansion for urban agriculture in London.

If you’re interested in automating any of this with Python, you can find many of the commands in the NYC data guide, available [here](https://docs.google.com/document/u/0/d/15VKDEastyWHeRrqB4UomDVLdxRRdk3bcbcmwVumfbKw/edit).

# Land Cover Layer Derivation

This layer will identify open ground areas and rooftops. These open areas and rooftops will then be filtered by other qualifications from the other layers (e.g. slope). The final layer produced via this procedure will have the following codes:

* Impervious - 1
* Grass or dirt - 2
* Roof - 3
* Tree - 4
* Otherwise occupied - 0 (e.g., monument, water, railroad, road)

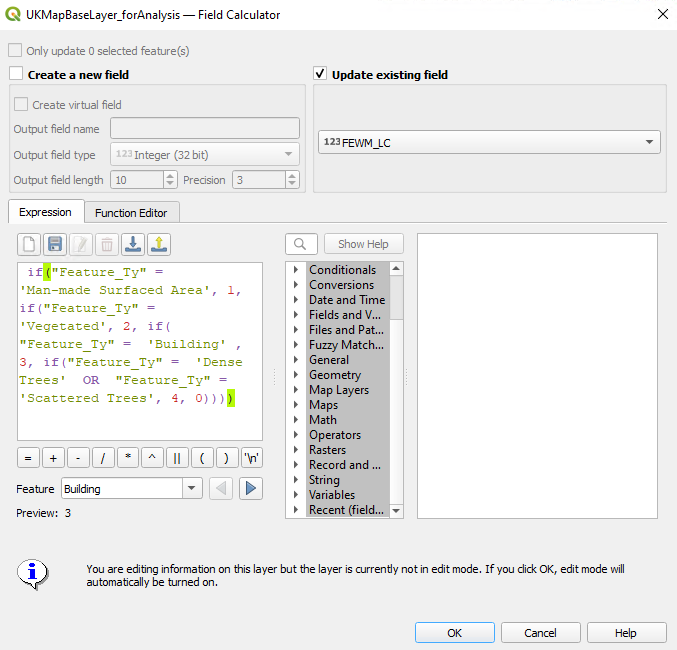
Overview: We have a land cover dataset that needs to be extracted from the rest of the UKMap and simplified to follow this coding structure.

Because UKMap is an all-inclusive dataset, this is much easier than in NYC or other places. We can simply pull the Feature Type Code into a raster (and overlay it with trees from the overlay dataset). To build the base map, we’ll use the Base geodatabase in UKMap. The easiest way to import this is by dragging the geopackage onto the QGIS platform and selecting the layers you want - the base map and the overlay. You won’t need the points file for this.

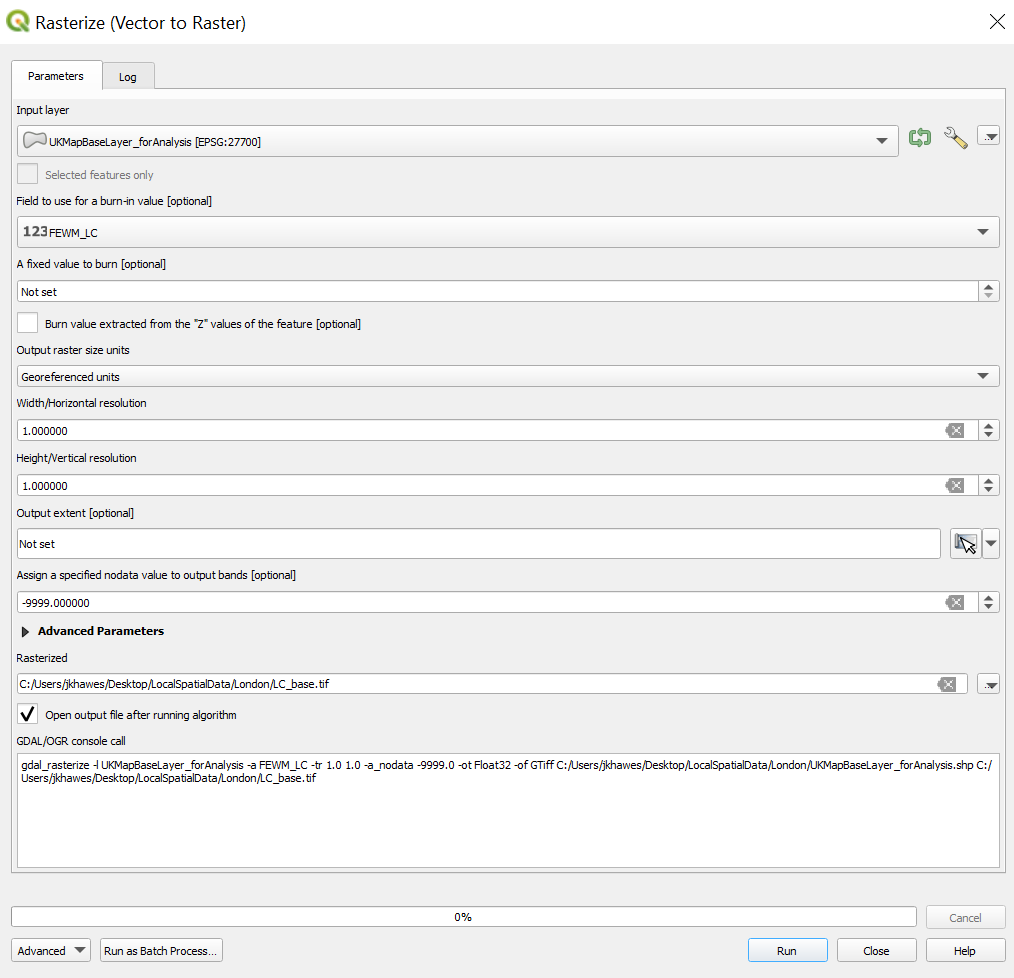
*I strongly recommend making a copy of the UKMap database files before doing this.* They are very large datasets and not particularly easy to reset to base if something goes wrong - which isn’t particularly unlikely when Q is parsing this much data.

## Convert Base Tables Feature\_Type to numeric, then rasterize

So the Feature\_Type, despite being shown as numbers in the codebook, is actually strings in the final attribute table. So we need to do a field calculation to get from there to the numbers that we can burn into a raster. Fairly straightforward raster calculation where we match building with Roof (3), vegetated with Grassy (2), trees with trees (4), impervious with Impervious (3), and anything else with Other (0). In case this proves useful, the screenshot below also shows all the unique values in the feature\_type field. The code for copy-paste is: if("Feature\_Ty" = 'Man-made Surfaced Area', 1, if("Feature\_Ty" = 'Vegetated', 2, if( "Feature\_Ty" = 'Building' , 3, if("Feature\_Ty" = 'Dense Trees' OR "Feature\_Ty" = 'Scattered Trees', 4, 0))))

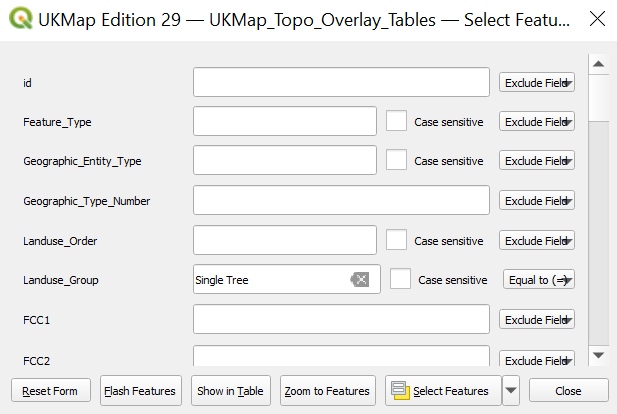


Before you can rasterize the file, you need to save it. This takes a very long time - on my standard desktop, this is on the order of an entire evening. On our high-powered simulation computer, more like 10 minutes. If you interrupt it halfway through, it will act like it saved but the column will be blank. Once it’s saved, use the rasterize function, selecting FEWM\_LC (or whatever you named it) as the burn-in value. I recommend -9999 as the no-data value followed by r.null.

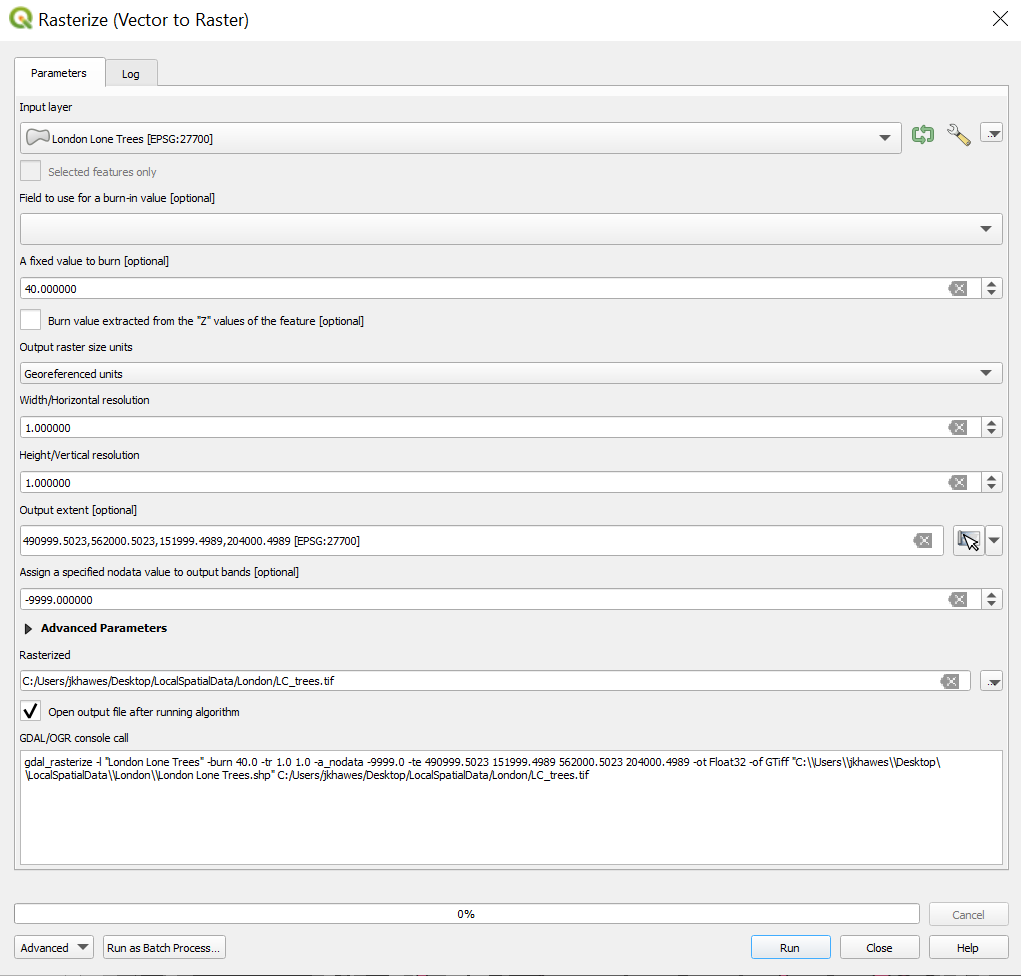


## Create an overly file of only trees, then rasterize

We can use Select Features by Value to extract the trees from the overlay file. Most of the features in the overlay file *are* trees, but we can filter out the few things that aren’t. The function is simple - just select all the entries that have Single Tree as their land use group:



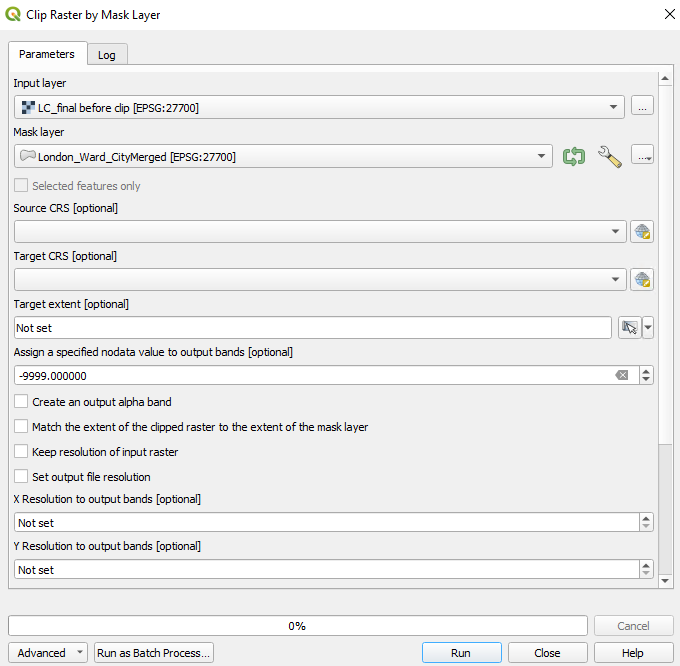
Once this is selected, we Save Selected Features As by right-clicking on the layer and going to Export. Save the trees file, then pull it back into QGIS. Rasterize the file, making sure to align it to the extent of the base map raster. Burn in a value of 40, which we can use in the overlay and raster calculation. Set the no data value to -9999. Once this is done, use r.null to replace the NoData cells with 0.



## Overlay raster files and clip

Once you have both rasters in QGIS, you should be able to use simple raster calculator work to overlay them. First, add the rasters together. Then, set anything above 39 to 4. We should now have a usable land cover file with the 0-4 coding structure indicated above. Remember that all the forthcoming rasters must be aligned to this one so we can eventually add them all together.

The last thing we need to do before we can leave this be is just to trim it down to the administrative bounds of the city. Using the London Wards file found [here](https://data.london.gov.uk/dataset/statistical-gis-boundary-files-london), you can use the Clip Raster by Mask Layer tool and make everything outside the administrative boundary null.



# Land Use and Ownership Layer Derivation This layer will describe the land use at a parcel level. These will be used primarily to sort the different types of gardens - e.g., it makes much more sense to assume an individual garden in a single-family backyard than a community garden. The final layer produced via this procedure will have the usual land use codes I developed for all cities. You can find the equivalency chart [here](https://docs.google.com/spreadsheets/d/176HUxNceFgc3QG77QbuSUSxrKWQ1HY6M/edit?usp=sharing&ouid=102554365591872343075&rtpof=true&sd=true).

For this, we will only need to use the UKMap data.

## Recode Land Use with Field Calculator

The UKMap data uses a complex land use scheme based on the UK national system. The most robust way to recode it is to do it at the FCC4 level, which means recoding the entire FCC group code. The most up-to-date recoding table can always be found [here](https://docs.google.com/spreadsheets/d/176HUxNceFgc3QG77QbuSUSxrKWQ1HY6M/edit#gid=1403094731).

First, we can start by creating a LandUseGrp attribute, which allows us to access the full FCC code instead of the levels separately. The field calculator code for this is: concat( "FCC1" , "FCC2" , "FCC3" , "FCC4" )

Next, we can calculate the recoding. I found the easiest way to do this to be running every LandUseGrp individually, which makes for a very long field calculator command. I generated this using concatenate in Google Sheets to automatically generate the code - see the recoding table for details. There are endless ways to write and execute this command, but not all of them are created equal. For me, using if statements endlessly crashed QGIS before I ever got to run the command - something about so many opening and closing parentheses sends the program into a tailspin. So I use the Case When structure for this and it seems to work quite well.

Command:

CASE

WHEN "LandUseGrp" = 11121 THEN 22

WHEN "LandUseGrp" = 11124 THEN 22

WHEN "LandUseGrp" = 11210 THEN 90

WHEN "LandUseGrp" = 11211 THEN 90

WHEN "LandUseGrp" = 11511 THEN 90

WHEN "LandUseGrp" = 11610 THEN 90

WHEN "LandUseGrp" = 11620 THEN 90

WHEN "LandUseGrp" = 11621 THEN 90

WHEN "LandUseGrp" = 11623 THEN 90

WHEN "LandUseGrp" = 11628 THEN 90

WHEN "LandUseGrp" = 11811 THEN 90

WHEN "LandUseGrp" = 11824 THEN 33

WHEN "LandUseGrp" = 11831 THEN 90

WHEN "LandUseGrp" = 11840 THEN 33

WHEN "LandUseGrp" = 11910 THEN 90

WHEN "LandUseGrp" = 11911 THEN 80

WHEN "LandUseGrp" = 12110 THEN 34

WHEN "LandUseGrp" = 12121 THEN 34

WHEN "LandUseGrp" = 12122 THEN 34

WHEN "LandUseGrp" = 12123 THEN 34

WHEN "LandUseGrp" = 12124 THEN 34

WHEN "LandUseGrp" = 12130 THEN 34

WHEN "LandUseGrp" = 12133 THEN 34

WHEN "LandUseGrp" = 12134 THEN 34

WHEN "LandUseGrp" = 12135 THEN 34

WHEN "LandUseGrp" = 12136 THEN 34

WHEN "LandUseGrp" = 12137 THEN 34

WHEN "LandUseGrp" = 12138 THEN 34

WHEN "LandUseGrp" = 12139 THEN 34

WHEN "LandUseGrp" = 121312 THEN 34

WHEN "LandUseGrp" = 12141 THEN 34

WHEN "LandUseGrp" = 12142 THEN 34

WHEN "LandUseGrp" = 12211 THEN 34

WHEN "LandUseGrp" = 12310 THEN 39

WHEN "LandUseGrp" = 12312 THEN 39

WHEN "LandUseGrp" = 12420 THEN 34

WHEN "LandUseGrp" = 12510 THEN 36

WHEN "LandUseGrp" = 12512 THEN 36

WHEN "LandUseGrp" = 12513 THEN 36

WHEN "LandUseGrp" = 12521 THEN 39

WHEN "LandUseGrp" = 12411 THEN 13

WHEN "LandUseGrp" = 12413 THEN 13

WHEN "LandUseGrp" = 12522 THEN 39

WHEN "LandUseGrp" = 12523 THEN 39

WHEN "LandUseGrp" = 12524 THEN 39

WHEN "LandUseGrp" = 12525 THEN 39

WHEN "LandUseGrp" = 12610 THEN 22

WHEN "LandUseGrp" = 12621 THEN 24

WHEN "LandUseGrp" = 12622 THEN 39

WHEN "LandUseGrp" = 12623 THEN 39

WHEN "LandUseGrp" = 12710 THEN 24

WHEN "LandUseGrp" = 12800 THEN 39

WHEN "LandUseGrp" = 12812 THEN 39

WHEN "LandUseGrp" = 13110 THEN 39

WHEN "LandUseGrp" = 14110 THEN 35

WHEN "LandUseGrp" = 14120 THEN 35

WHEN "LandUseGrp" = 14130 THEN 35

WHEN "LandUseGrp" = 14140 THEN 35

WHEN "LandUseGrp" = 14142 THEN 35

WHEN "LandUseGrp" = 14151 THEN 35

WHEN "LandUseGrp" = 14152 THEN 35

WHEN "LandUseGrp" = 14162 THEN 35

WHEN "LandUseGrp" = 14165 THEN 35

WHEN "LandUseGrp" = 14167 THEN 35

WHEN "LandUseGrp" = 14211 THEN 37

WHEN "LandUseGrp" = 14212 THEN 37

WHEN "LandUseGrp" = 14221 THEN 31

WHEN "LandUseGrp" = 14222 THEN 37

WHEN "LandUseGrp" = 15111 THEN 32

WHEN "LandUseGrp" = 15112 THEN 31

WHEN "LandUseGrp" = 15113 THEN 31

WHEN "LandUseGrp" = 15114 THEN 31

WHEN "LandUseGrp" = 15115 THEN 31

WHEN "LandUseGrp" = 15116 THEN 31

WHEN "LandUseGrp" = 15118 THEN 31

WHEN "LandUseGrp" = 15121 THEN 37

WHEN "LandUseGrp" = 15122 THEN 37

WHEN "LandUseGrp" = 15123 THEN 37

WHEN "LandUseGrp" = 15124 THEN 37

WHEN "LandUseGrp" = 15130 THEN 22

WHEN "LandUseGrp" = 15131 THEN 22

WHEN "LandUseGrp" = 15136 THEN 22

WHEN "LandUseGrp" = 15139 THEN 39

WHEN "LandUseGrp" = 15141 THEN 22

WHEN "LandUseGrp" = 15144 THEN 39

WHEN "LandUseGrp" = 15148 THEN 39

WHEN "LandUseGrp" = 15211 THEN 39

WHEN "LandUseGrp" = 15221 THEN 39

WHEN "LandUseGrp" = 15230 THEN 39

WHEN "LandUseGrp" = 15231 THEN 39

WHEN "LandUseGrp" = 15310 THEN 26

WHEN "LandUseGrp" = 15311 THEN 26

WHEN "LandUseGrp" = 15313 THEN 26

WHEN "LandUseGrp" = 15318 THEN 26

WHEN "LandUseGrp" = 15321 THEN 26

WHEN "LandUseGrp" = 15322 THEN 26

WHEN "LandUseGrp" = 15324 THEN 26

WHEN "LandUseGrp" = 15325 THEN 26

WHEN "LandUseGrp" = 15326 THEN 26

WHEN "LandUseGrp" = 15327 THEN 26

WHEN "LandUseGrp" = 15331 THEN 26

WHEN "LandUseGrp" = 15332 THEN 26

WHEN "LandUseGrp" = 15341 THEN 26

WHEN "LandUseGrp" = 15343 THEN 26

WHEN "LandUseGrp" = 15344 THEN 26

WHEN "LandUseGrp" = 15345 THEN 26

WHEN "LandUseGrp" = 15354 THEN 26

WHEN "LandUseGrp" = 15363 THEN 26

WHEN "LandUseGrp" = 15370 THEN 26

WHEN "LandUseGrp" = 15383 THEN 26

WHEN "LandUseGrp" = 15392 THEN 26

WHEN "LandUseGrp" = 15394 THEN 26

WHEN "LandUseGrp" = 15396 THEN 26

WHEN "LandUseGrp" = 153100 THEN 26

WHEN "LandUseGrp" = 15411 THEN 26

WHEN "LandUseGrp" = 15420 THEN 83

WHEN "LandUseGrp" = 15421 THEN 83

WHEN "LandUseGrp" = 15431 THEN 83

WHEN "LandUseGrp" = 15511 THEN 14

WHEN "LandUseGrp" = 15512 THEN 14

WHEN "LandUseGrp" = 15513 THEN 14

WHEN "LandUseGrp" = 15514 THEN 14

WHEN "LandUseGrp" = 161300 THEN 22

WHEN "LandUseGrp" = 161400 THEN 23

WHEN "LandUseGrp" = 161500 THEN 23

WHEN "LandUseGrp" = 17100 THEN 23

WHEN "LandUseGrp" = 18110 THEN 22

WHEN "LandUseGrp" = 18111 THEN 39

WHEN "LandUseGrp" = 18112 THEN 39

WHEN "LandUseGrp" = 19111 THEN 12

WHEN "LandUseGrp" = 19120 THEN 12

WHEN "LandUseGrp" = 19121 THEN 39

WHEN "LandUseGrp" = 19132 THEN 14

WHEN "LandUseGrp" = 19133 THEN 14

WHEN "LandUseGrp" = 19210 THEN 10

WHEN "LandUseGrp" = 19219 THEN 10

WHEN "LandUseGrp" = 192110 THEN 10

WHEN "LandUseGrp" = 20100 THEN 22

WHEN "LandUseGrp" = 20210 THEN 22

WHEN "LandUseGrp" = 20211 THEN 22

WHEN "LandUseGrp" = 20212 THEN 22

WHEN "LandUseGrp" = 20213 THEN 22

WHEN "LandUseGrp" = 20214 THEN 22

WHEN "LandUseGrp" = 20220 THEN 22

WHEN "LandUseGrp" = 20221 THEN 22

WHEN "LandUseGrp" = 20222 THEN 22

WHEN "LandUseGrp" = 20223 THEN 22

WHEN "LandUseGrp" = 20224 THEN 22

WHEN "LandUseGrp" = 20225 THEN 22

WHEN "LandUseGrp" = 20226 THEN 22

WHEN "LandUseGrp" = 20227 THEN 22

WHEN "LandUseGrp" = 20228 THEN 22

WHEN "LandUseGrp" = 20230 THEN 22

WHEN "LandUseGrp" = 20231 THEN 22

WHEN "LandUseGrp" = 20232 THEN 22

WHEN "LandUseGrp" = 20233 THEN 22

WHEN "LandUseGrp" = 20234 THEN 22

WHEN "LandUseGrp" = 20235 THEN 22

WHEN "LandUseGrp" = 20236 THEN 22

WHEN "LandUseGrp" = 20237 THEN 22

WHEN "LandUseGrp" = 20238 THEN 22

WHEN "LandUseGrp" = 20240 THEN 22

WHEN "LandUseGrp" = 20241 THEN 22

WHEN "LandUseGrp" = 20242 THEN 22

WHEN "LandUseGrp" = 20243 THEN 22

WHEN "LandUseGrp" = 20244 THEN 22

WHEN "LandUseGrp" = 20245 THEN 22

WHEN "LandUseGrp" = 20246 THEN 22

WHEN "LandUseGrp" = 20248 THEN 22

WHEN "LandUseGrp" = 20250 THEN 22

WHEN "LandUseGrp" = 20251 THEN 22

WHEN "LandUseGrp" = 20252 THEN 22

WHEN "LandUseGrp" = 20253 THEN 22

WHEN "LandUseGrp" = 20254 THEN 22

WHEN "LandUseGrp" = 20255 THEN 22

WHEN "LandUseGrp" = 20256 THEN 22

WHEN "LandUseGrp" = 20257 THEN 22

WHEN "LandUseGrp" = 20258 THEN 22

WHEN "LandUseGrp" = 20201 THEN 22

WHEN "LandUseGrp" = 20202 THEN 22

WHEN "LandUseGrp" = 20203 THEN 22

WHEN "LandUseGrp" = 20204 THEN 22

WHEN "LandUseGrp" = 20205 THEN 22

WHEN "LandUseGrp" = 20206 THEN 22

WHEN "LandUseGrp" = 20207 THEN 22

WHEN "LandUseGrp" = 20208 THEN 22

WHEN "LandUseGrp" = 21210 THEN 22

WHEN "LandUseGrp" = 22111 THEN 81

WHEN "LandUseGrp" = 22112 THEN 81

WHEN "LandUseGrp" = 22114 THEN 81

WHEN "LandUseGrp" = 22121 THEN 81

WHEN "LandUseGrp" = 22131 THEN 81

WHEN "LandUseGrp" = 22142 THEN 42

WHEN "LandUseGrp" = 22150 THEN 42

WHEN "LandUseGrp" = 221510 THEN 42

WHEN "LandUseGrp" = 221511 THEN 42

WHEN "LandUseGrp" = 221512 THEN 42

WHEN "LandUseGrp" = 221513 THEN 42

WHEN "LandUseGrp" = 221514 THEN 33

WHEN "LandUseGrp" = 221515 THEN 42

WHEN "LandUseGrp" = 221516 THEN 41

WHEN "LandUseGrp" = 221517 THEN 81

WHEN "LandUseGrp" = 221518 THEN 42

WHEN "LandUseGrp" = 221519 THEN 81

WHEN "LandUseGrp" = 221520 THEN 42

WHEN "LandUseGrp" = 221521 THEN 41

WHEN "LandUseGrp" = 221522 THEN 42

WHEN "LandUseGrp" = 221523 THEN 42

WHEN "LandUseGrp" = 221524 THEN 42

WHEN "LandUseGrp" = 221525 THEN 42

WHEN "LandUseGrp" = 221526 THEN 42

WHEN "LandUseGrp" = 221527 THEN 42

WHEN "LandUseGrp" = 221528 THEN 42

WHEN "LandUseGrp" = 221529 THEN 42

WHEN "LandUseGrp" = 221530 THEN 42

WHEN "LandUseGrp" = 221531 THEN 42

WHEN "LandUseGrp" = 221532 THEN 42

WHEN "LandUseGrp" = 221533 THEN 42

WHEN "LandUseGrp" = 221534 THEN 42

WHEN "LandUseGrp" = 221535 THEN 41

WHEN "LandUseGrp" = 221536 THEN 42

WHEN "LandUseGrp" = 221537 THEN 42

WHEN "LandUseGrp" = 221538 THEN 42

WHEN "LandUseGrp" = 221539 THEN 42

WHEN "LandUseGrp" = 22160 THEN 42

WHEN "LandUseGrp" = 22165 THEN 42

WHEN "LandUseGrp" = 221610 THEN 42

WHEN "LandUseGrp" = 221612 THEN 42

WHEN "LandUseGrp" = 221614 THEN 42

WHEN "LandUseGrp" = 22213 THEN 42

WHEN "LandUseGrp" = 22214 THEN 42

WHEN "LandUseGrp" = 22217 THEN 42

WHEN "LandUseGrp" = 22219 THEN 42

WHEN "LandUseGrp" = 22221 THEN 42

WHEN "LandUseGrp" = 22222 THEN 42

WHEN "LandUseGrp" = 22223 THEN 42

WHEN "LandUseGrp" = 22224 THEN 42

WHEN "LandUseGrp" = 22232 THEN 42

WHEN "LandUseGrp" = 22233 THEN 41

WHEN "LandUseGrp" = 22234 THEN 41

WHEN "LandUseGrp" = 22237 THEN 41

WHEN "LandUseGrp" = 22238 THEN 41

WHEN "LandUseGrp" = 22311 THEN 83

WHEN "LandUseGrp" = 22312 THEN 83

WHEN "LandUseGrp" = 22412 THEN 83

WHEN "LandUseGrp" = 22413 THEN 83

WHEN "LandUseGrp" = 22414 THEN 83

WHEN "LandUseGrp" = 22522 THEN 42

WHEN "LandUseGrp" = 22611 THEN 42

WHEN "LandUseGrp" = 22711 THEN 42

WHEN "LandUseGrp" = 22712 THEN 42

WHEN "LandUseGrp" = 22713 THEN 42

WHEN "LandUseGrp" = 22714 THEN 42

WHEN "LandUseGrp" = 22715 THEN 42

WHEN "LandUseGrp" = 22716 THEN 42

WHEN "LandUseGrp" = 22717 THEN 42

WHEN "LandUseGrp" = 22718 THEN 42

WHEN "LandUseGrp" = 22719 THEN 42

WHEN "LandUseGrp" = 227110 THEN 42

WHEN "LandUseGrp" = 227111 THEN 42

WHEN "LandUseGrp" = 227112 THEN 42

WHEN "LandUseGrp" = 227113 THEN 42

WHEN "LandUseGrp" = 227114 THEN 42

WHEN "LandUseGrp" = 227115 THEN 42

WHEN "LandUseGrp" = 227116 THEN 81

WHEN "LandUseGrp" = 227117 THEN 42

WHEN "LandUseGrp" = 22810 THEN 42

WHEN "LandUseGrp" = 22820 THEN 81

WHEN "LandUseGrp" = 22830 THEN 42

WHEN "LandUseGrp" = 23111 THEN 33

WHEN "LandUseGrp" = 23112 THEN 33

WHEN "LandUseGrp" = 23113 THEN 33

WHEN "LandUseGrp" = 23114 THEN 33

WHEN "LandUseGrp" = 23115 THEN 33

WHEN "LandUseGrp" = 23116 THEN 83

WHEN "LandUseGrp" = 23117 THEN 33

WHEN "LandUseGrp" = 23120 THEN 70

WHEN "LandUseGrp" = 221613 THEN 42

WHEN "LandUseGrp" = 23126 THEN 22

WHEN "LandUseGrp" = 23134 THEN 83

WHEN "LandUseGrp" = 23135 THEN 83

WHEN "LandUseGrp" = 23136 THEN 22

WHEN "LandUseGrp" = 23210 THEN 70

WHEN "LandUseGrp" = 24110 THEN 42

WHEN "LandUseGrp" = 24210 THEN 42

WHEN "LandUseGrp" = 24213 THEN 42

WHEN "LandUseGrp" = 24214 THEN 42

WHEN "LandUseGrp" = 24221 THEN 42

WHEN "LandUseGrp" = 24222 THEN 42

WHEN "LandUseGrp" = 24311 THEN 83

WHEN "LandUseGrp" = 24312 THEN 42

WHEN "LandUseGrp" = 24313 THEN 42

WHEN "LandUseGrp" = 24321 THEN 42

WHEN "LandUseGrp" = 24322 THEN 42

WHEN "LandUseGrp" = 24340 THEN 42

WHEN "LandUseGrp" = 24410 THEN 42

WHEN "LandUseGrp" = 24412 THEN 42

WHEN "LandUseGrp" = 24422 THEN 42

WHEN "LandUseGrp" = 24511 THEN 42

WHEN "LandUseGrp" = 24512 THEN 42

WHEN "LandUseGrp" = 24513 THEN 42

WHEN "LandUseGrp" = 24712 THEN 39

WHEN "LandUseGrp" = 24721 THEN 32

WHEN "LandUseGrp" = 24722 THEN 39

WHEN "LandUseGrp" = 24811 THEN 39

WHEN "LandUseGrp" = 24822 THEN 42

WHEN "LandUseGrp" = 24824 THEN 42

WHEN "LandUseGrp" = 24831 THEN 42

WHEN "LandUseGrp" = 24832 THEN 42

WHEN "LandUseGrp" = 24841 THEN 42

WHEN "LandUseGrp" = 24842 THEN 42

WHEN "LandUseGrp" = 24843 THEN 42

WHEN "LandUseGrp" = 24851 THEN 42

WHEN "LandUseGrp" = 25123 THEN 22

WHEN "LandUseGrp" = 55140 THEN 33

WHEN "LandUseGrp" = 61000 THEN 0

WHEN "LandUseGrp" = 62000 THEN 0

WHEN "LandUseGrp" = 63000 THEN 0

WHEN "LandUseGrp" = 99000 THEN 0

WHEN "LandUseGrp" = 221611 THEN 42

WHEN "LandUseGrp" = 12112 THEN 34

WHEN "LandUseGrp" = 12114 THEN 34

WHEN "LandUseGrp" = 121311 THEN 34

WHEN "LandUseGrp" = 12412 THEN 13

WHEN "LandUseGrp" = 15134 THEN 31

WHEN "LandUseGrp" = 20247 THEN 22

WHEN "LandUseGrp" = 14168 THEN 35

ELSE 0

END

## Rasterize and clip

Once this is ready, you can rasterize it. Best practice would be to set the extent to the land cover layer, although this won’t be strictly necessary, since they’re derived from the same shapefile - and we’re going to align later anyway.



The last step is the same as land cover - trim the file down to the administrative bounds of the city. Using the London Wards file found [here](https://data.london.gov.uk/dataset/statistical-gis-boundary-files-london), you can use the Clip Raster by Mask Layer tool and make everything outside the administrative boundary null.

# Binary Slope Layer Derivation

This layer will describe slope at 1m resolution. The final layer produced via this procedure will have the following codes:

* Ineligible, over 15% grade-- 0
* Flat ground – 1
* Flat roof – 2

Overview: We can use the LiDAR-derived DSM and DTM to identify flat ground and flat roofs throughout the city. We have to do these two things separately, since ground level varies across London and the DSM is reported in feet above sea level.

We will use the following data sets:

1. London DSM - [metadata](https://environment.data.gov.uk/dataset/df4e3ec3-315e-48aa-aaaf-b5ae74d7b2bb) and [download](https://environment.data.gov.uk/DefraDataDownload/?Mode=survey)
2. London DTM - [metadata](https://environment.data.gov.uk/dataset/13787b9a-26a4-4775-8523-806d13af58fc) and [download](https://environment.data.gov.uk/DefraDataDownload/?Mode=survey)
3. London Wards boundary file - [metadata](https://data.london.gov.uk/dataset/statistical-gis-boundary-files-london)

We will begin by preparing city-wide DTM and DSM layers, then we will proceed with the flat ground analysis. We will then identify flat roofs. Finally, we will mask buildings from the ground layer and add in the buildings results. This takes a while because of the format of download and size of the files.

## Download DEFRA data products

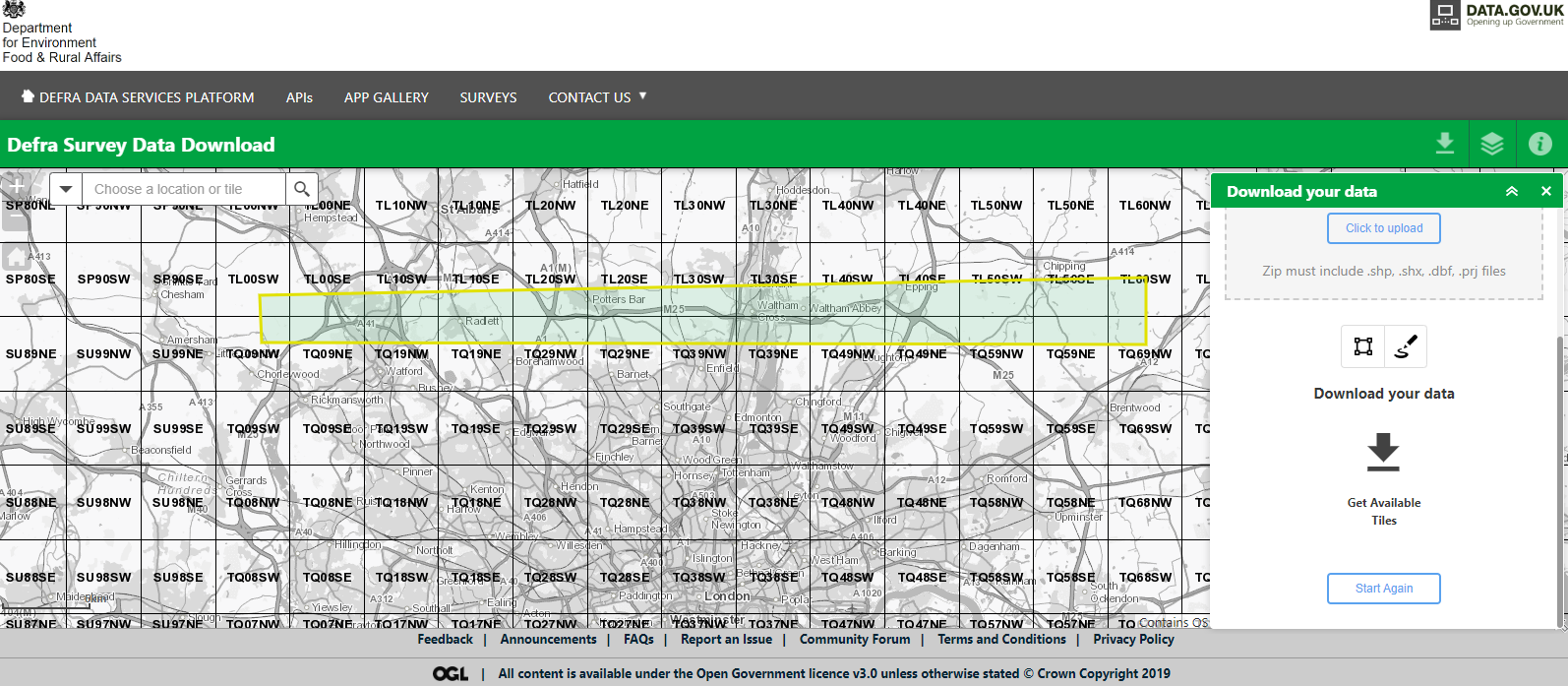
DEFRA produces a suite of products useful for this analysis, including a number of LiDAR-derived products. For the slope derivation, we can use the DTMs and DSMs made available through the online data download. Fair warning that this site took a very long time to load for me even on campus internet. Once it loads though, it’s pretty fast. There are [size limitations on the online portal](https://support.environment.data.gov.uk/hc/en-gb/articles/360009247332-Can-I-download-large-areas-of-LIDAR-data-), which means you have two options for getting the data (also explained at that link).

1. You can use the grid around London and download the 5km x 5km data in chunks. Since it usually lets you download 26 grid cells at once, it really isn’t that hard to do this by hand. In some cases, you can only do 13 or so at once - I’m not sure why. If you want to try to automate the download process with shapefiles, you can get a map of all of London’s wards from the city’s [data portal](https://data.london.gov.uk/dataset/statistical-gis-boundary-files-london). Using this, you can build a grid. Since the downloads aren’t all the same size, I don’t recommend this.
2. You can reach out to a representative at DEFRA to collate the data for you and make it available for download.

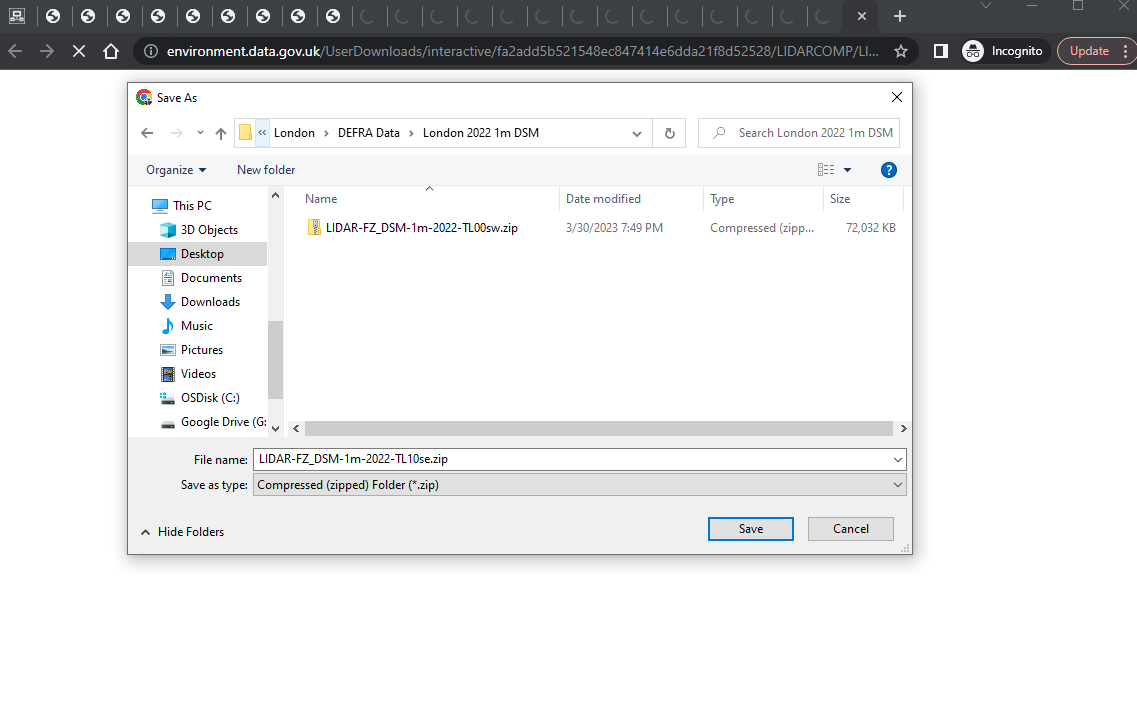
I recommend doing this by hand. You can capture the city in rows to make sure you don’t miss anything. London stretches as far west as the M25 and as far east as Brentwood (slightly outside the M25). It goes from Potter’s Bar in the north to Catesham in the south. Drawing long boxes like shown below makes fairly quick work of the download process. I recommend downloading the following products while you’re at this so you don’t have to do it all twice:

1. LIDAR Composite DTM (DTM 1m)
2. LIDAR Composite DSM (DSM 1m)

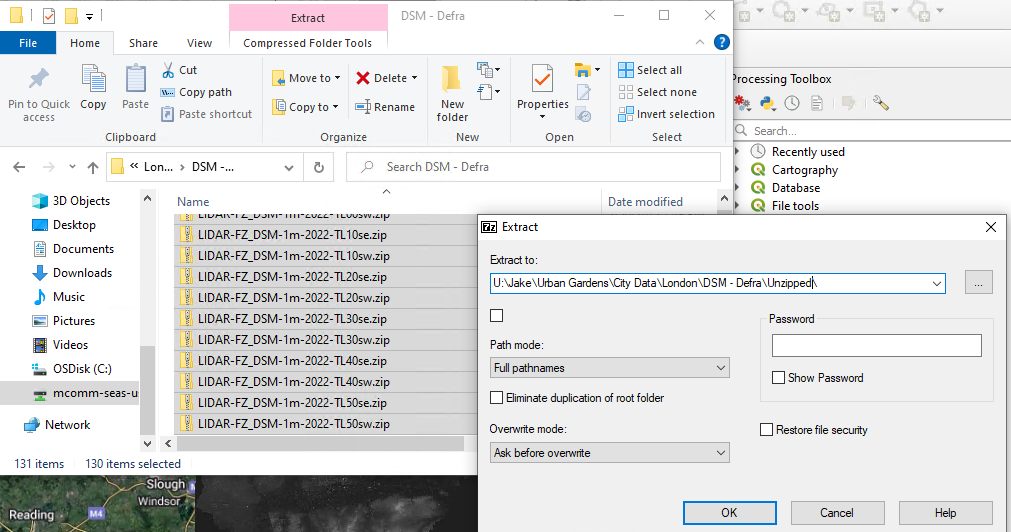
These composite files have the best coverage, and you should be able to download the most recent files in batches. At time of writing, this was 2022 data.



Once you’ve drawn the box, you can click Get Available Tiles and it will load the 26 results (assuming you adopt the 13x2 approach I have). The next bit does require a bit of clicking, but with the new Download All button (added in 2022), it’s not so bad. You hit Download All, then make sure popups are enabled (hit it again if you have to), then it will open 26 download windows and you just have to hit save for each zip file. I used an incognito tab for this, since some users reported trouble with cookies and the Download All button - I’m not sure if this is explicitly necessary. One last note about the downloads - if you don’t have your Chrome configured to ask where to save downloads, everything just drops into your downloads folder, which is actually substantially easier for this, since you can then just cut and paste them into the right directory all at once. It takes a minute to load all the download pages, but it requires a lot less clicks.

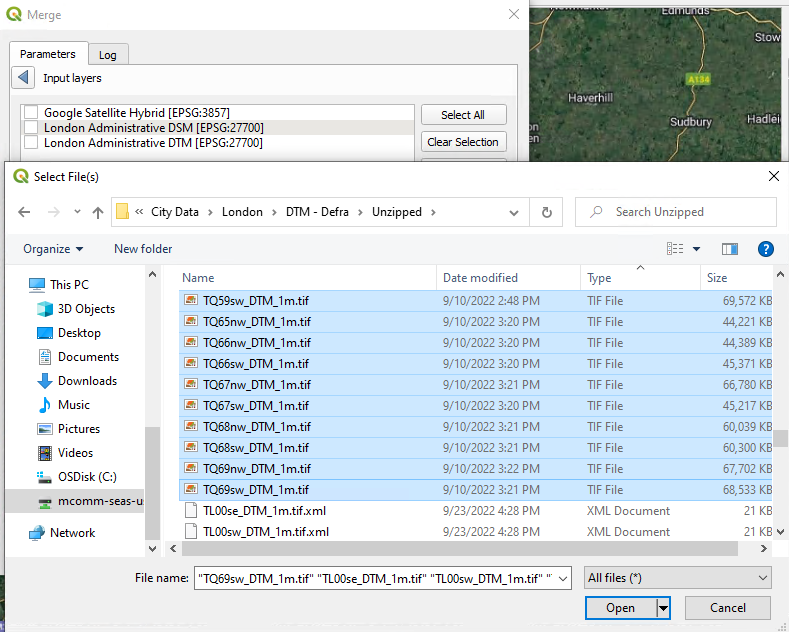


Once all the files are downloaded, they need to be unzipped. It is possible to unzip them in bulk and place them all in one folder using 7Zip. First, create a folder called Unzipped. Then, highlight all the zip files, right click, and move to 7-Zip. Click on Extract Files. Select the folder you created and let it run.

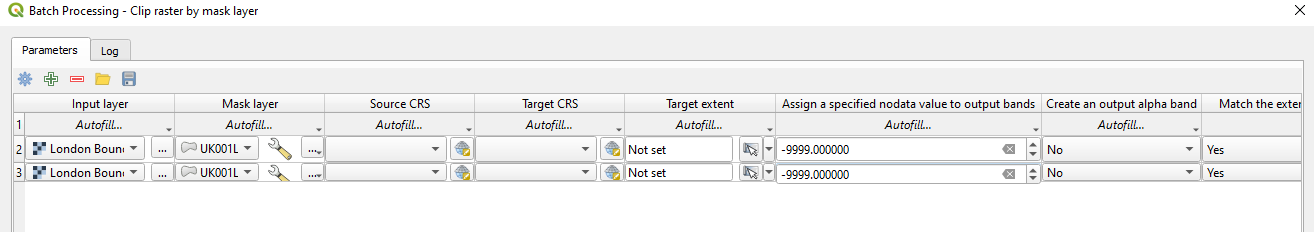


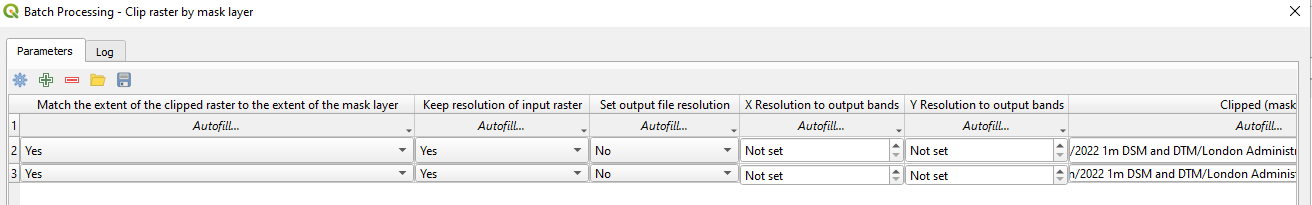
## Merge DSM and DTM files and clip to administrative boundary

Since these files are downloaded as individual tiles, they will need to be merged to be useful in city-wide analysis. Having them all in one folder like shown above is particularly helpful for this stage, since the merge tool can pull an entire directory. Under Raster >> Miscellaneous, the Merge tool will take a bunch of input layers and create on merged raster layer. You do not have to load the files into QGIS to merge them. Since they’re all in one folder, you can go in to the Merge tool, click Add Layer, then click Add Files. Go to the directory where you unzipped everything to and sort by file type so you don’t have to worry about the tfw files. Take all the tif files, select them, and add them to the merge. Click “OK” otherwise the list of inputs does not always save. Save the file as something like “London Bounding Box DTM” so you know this is the whole merged file before being clipped. Run the program.



You’ll need to do this with the DTM and DSM separately. Once they’re ready, you can clip those files to the administrative boundary to save some computational time - batch processing works for this. Using the London Wards file found [here](https://data.london.gov.uk/dataset/statistical-gis-boundary-files-london), you can use the Clip Raster by Mask Layer tool and make everything outside the administrative boundary null.



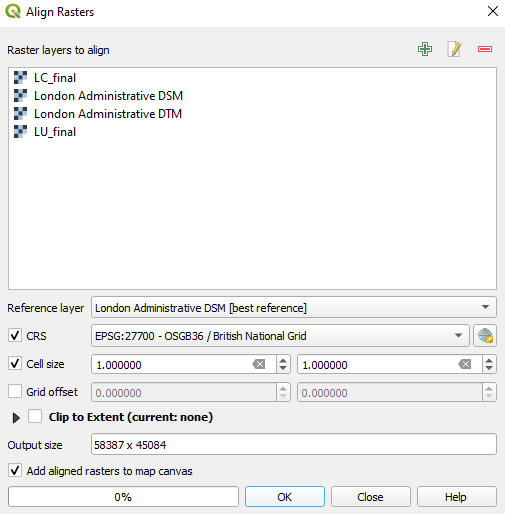


The output should look something like this:

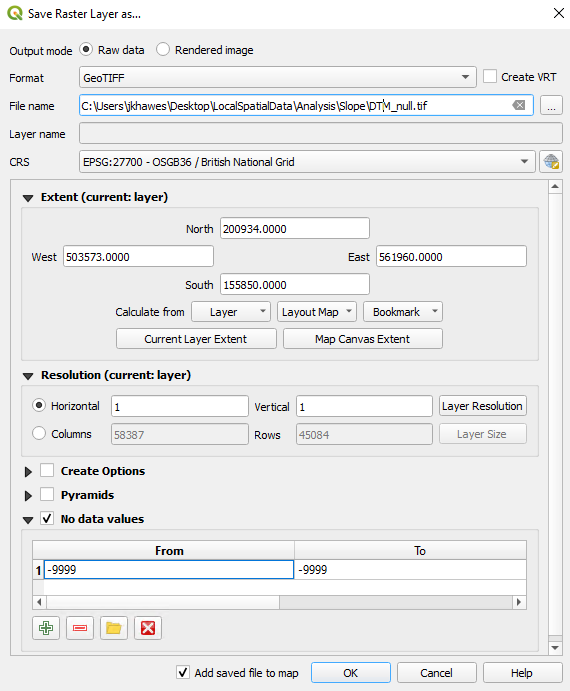


## Align with Land Cover and Land Use final layers

Next we need to make sure our slope layers align with the layers we created previously. Since the land use and land cover are already aligned, we don’t strictly need to align them. But the DSM is actually the best alignment reference - probably because it started out as a raster - so it works best just to run all four midpoint layers and get our aligned layers at this point.



After aligning the files, you will likely have to remind the computer that -9999 is a null value and not an actual value (this would obviously throw off the slope pretty dramatically at the city boundary). At first glance, this is actually kind of a pain because the files are too large for GRASS and you can’t use the BIGTIFF option in QGIS, so we can’t use r.null. HOWEVER, the good news is that there is a way to accomplish this with a built-in QGIS function. Just right-click on the layer, then export it. We’ll use mostly default settings, but we’ll check the box for No data values at the bottom. When we do this, it allows us to input -9999 as a no data value, and the system keeps track of it from here on. Don’t forget to run this for both the DTM and DSM.

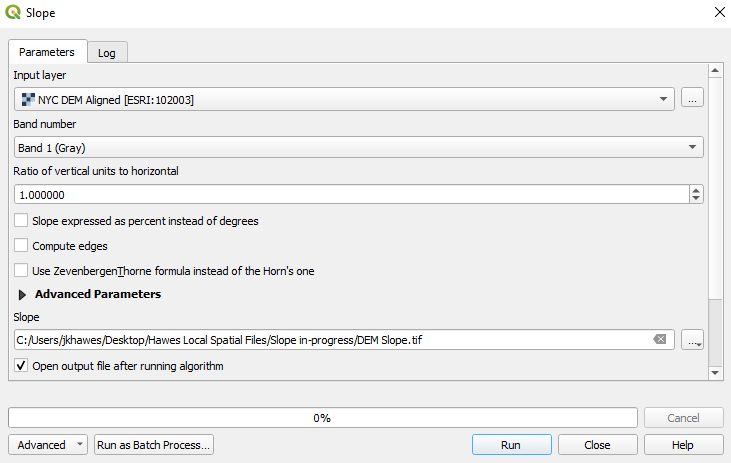


## Calculate Slope from the DTM

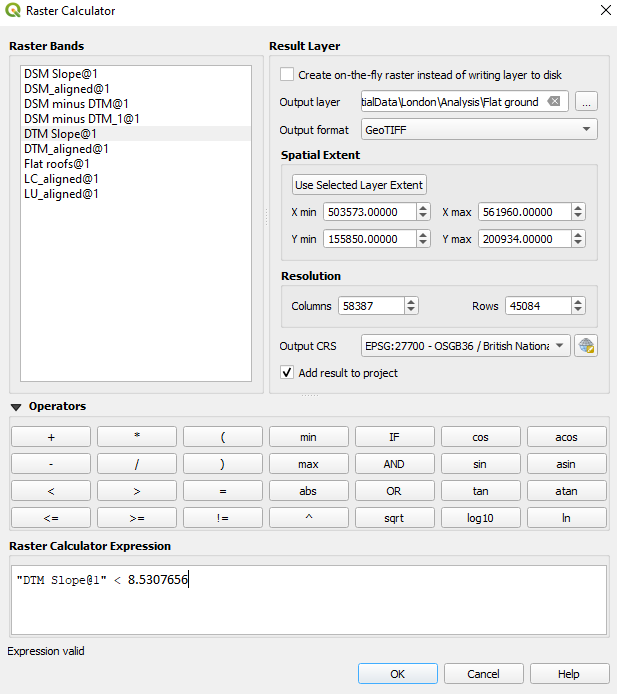
Now that we have clean layers to work with, we can derive slope on the ground. We can simply use the ***Slope*** function under Raster > Analysis on the DTM we just aligned. This will yield a slope layer for all ground cells in the city. We’re interested in places where the slope is less than 15%. Unfortunately, the “Slope expressed as percent instead of degrees” function seems to return absolutely outrageous values, so I don’t recommend using that. Instead, it seems better just to convert the 15% to degrees and use that in the raster calculator in the next step.

If you want to save yourself time later, run this as a batch process with the DSM as well.

\*\*Ignore the occasional screenshots from NYC, the commands are the same - just different file names

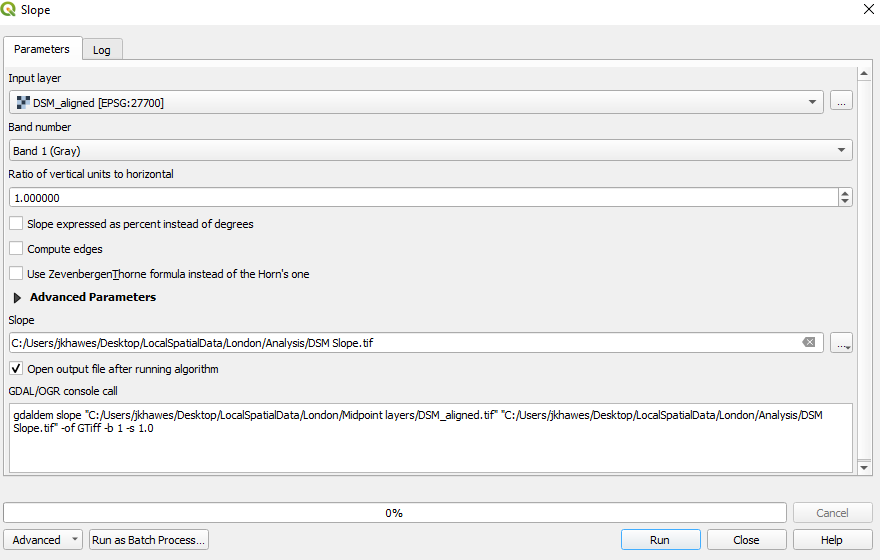


Once we have this Slope file, we can do some simple raster algebra to determine where this is greater than and less than 15%. Expressed as degrees, a 15% slope is arc-tangent of 0.15, which is 8.5307656. So we want to find places where the slope layer is less than that.

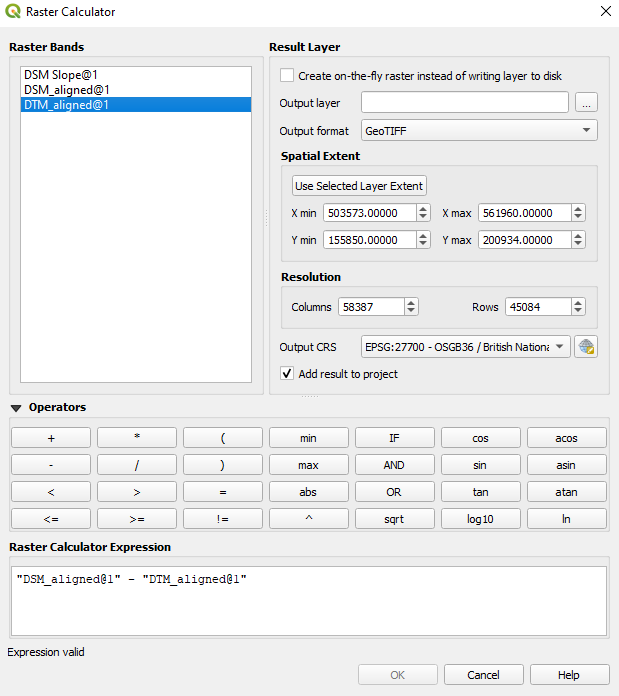


## Identify flat roofs from the DSM

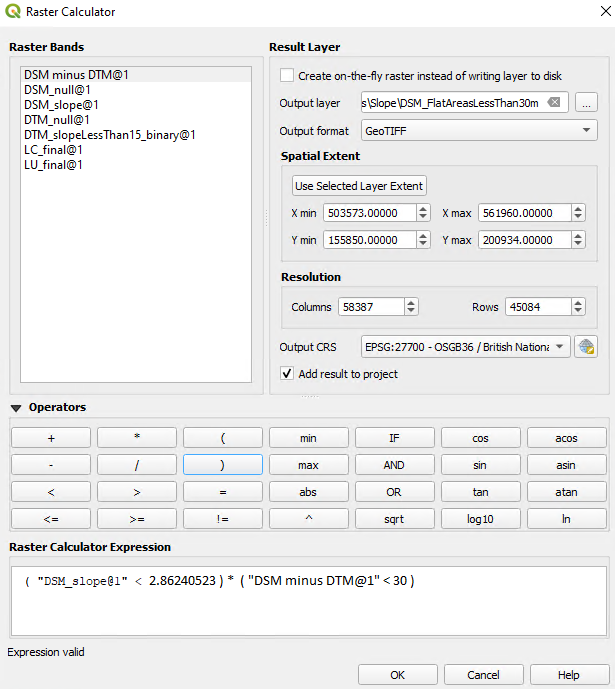
Next, we need to calculate the roof slopes - this is a bit trickier and requires several steps. First, we can run the Slope function under Raster > Analysis on the DSM.



Next, when we need to subtract the DEM from the DSM to make sure the ground level is zero all over the map.

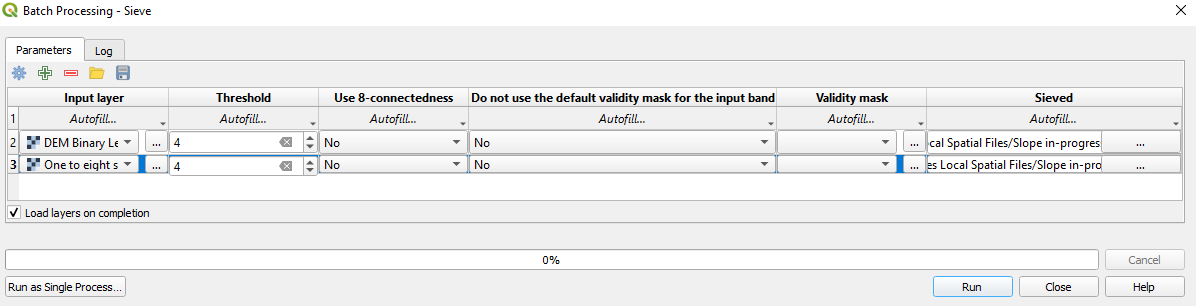


We can then run some simple raster calculations, finding the places where the Slope layer is less than 5% (less than arctangent of 0.05 = 2.86240523) and the Height of the DSM-DEM layer is less than 30m (8 stories or less) - note that this height is why we need to do the subtraction. If not, we can’t use 30 or something as a roof height, because ground level differs, so some roofs are below ground level in other places in the city. In some older versions of this, I also used 3.5 m as a minimum, but this really isn’t necessary since we use the building land cover classification below. Just need to limit the max so we’re not working on top of the Empire State building.



## Clean up layers with the sieve tool

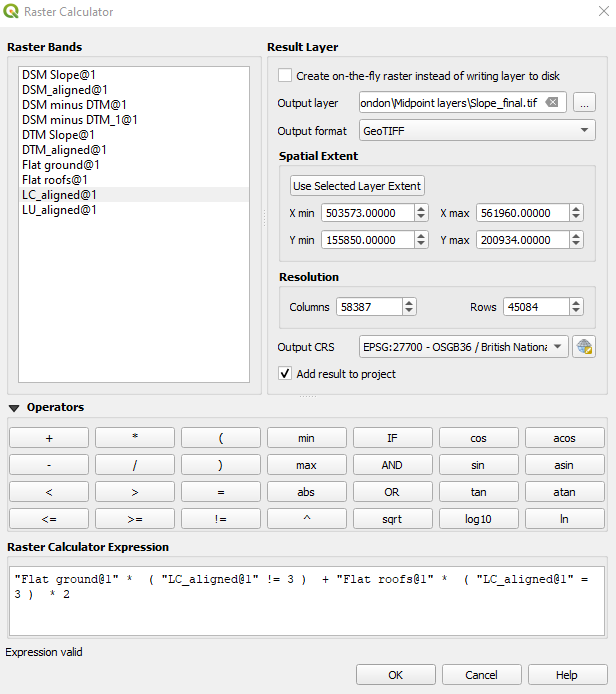
Once we have useful results for both round and rooftops, we can use the sieve tool to clean it up a bit. I used 4 as a threshold because it tends to catch edge cases most effectively. Make sure to fix the symbology to 0-1 after running the function.



## Final raster calculations

Lastly, we need to combine these layers into something intelligible - we will keep flat roofs and flat ground separate for now just for the sake of preserving information. We can always reclassify later. So our goal is: 0 = > 15% slope, 1 = flat ground, 2 = flat roof. How do we get there?

First, we’ll need to make all the building footprints zero in the ground slope layer and convert everything outside of buildings to zero in the roof slope layer. To keep the information discrete, we can do one more raster calculation - RoofsLayer \* 2 + GroundLayer. In the end, we can do this all in one step. For the building layer conversion, we can simply use the LC layer we derived earlier. Since everything has been aligned, we should be able to use it directly. The raster calculation is exactly the same for the two layers, but inverted. See below.



After this calculation is complete, don’t move all the Slope files off the hard drive just yet - make sure to keep the aligned DSM - we will use it as our first input in the sunlight layer.

# Sunlight Availability

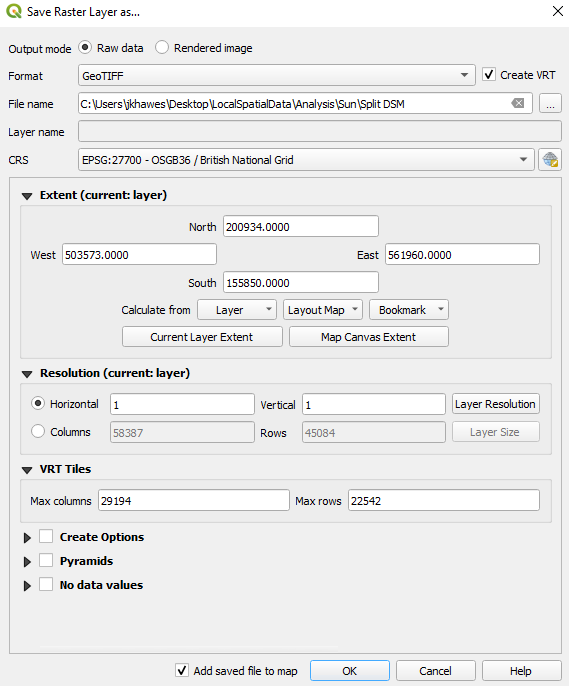
The final layer we will derive is a sunlight availability layer. This will take advantage of the r.sun package in GRASS, which takes the DSM and converts it to solar irradiance. This one is fairly complex, so I derive my process from an [example done in Canada](https://www.sciencedirect.com/science/article/pii/S0038092X10000812?via%3Dihub#fn15). Based on that paper, we will need a few inputs to make this work, including:

| Canadian layer/example | London layer/notes | Source link |
| --- | --- | --- |
| Digital Elevation Model (DEM) | DSM - include shading from buildings and trees |  |
| Slope/inclination | Derived from DSM |  |
| Aspect/orientation | Derived from DSM |  |
| Latitude | Not necessary if we use a proper projection |  |
| Albedo: the ratio of diffusely reflected radiation on a surface to its incident radiation. | Albedo can probably be calculated for each city with this function, or we can use urban averages. For i.albedo, just need landsat imagery: https://grass.osgeo.org/grass78/manuals/i.albedo.html |  |
| Mean days and corresponding angular position of the sun. | Can use the same mean days if we do want to do the calculation for every month. “ Table 1.6.1 in Duffie and Beckman (1991) readily provides the day of month, day of year and δ (sun declination) values to input into the simulation -- J.A. Duffie, W.A. Beckman, Solar Engineering of Thermal processes (second ed.), John Wiley & Sons (1991)” |  |
| Linke turbidity: a convenient approximation to model the atmospheric absorption and scattering of the solar radiation under clear skies. | If all we want is very high level stuff, we can get that from the same place the example paper did. Resolution is about the scale of NYC. Have three different raster cells for whole city, all the same value. Able to make a raster with the resolution of our DEMs and DSMs by downsampling. | <http://www.soda-pro.com/help/general-knowledge/linke-turbidity-factor> |
| Ground-measured values of global horizontal irradiation (GHI). | Available from NASA SSE POWER program - GHI is the first value (ALLSKY\_SFC\_SW\_DWN CERES SYN1deg All Sky Surface Shortwave Downward Irradiance (kW-hr/m^2/day)) while GHI under Clear-Sky conditions is the second value (CLRSKY\_SFC\_SW\_DWN CERES SYN1deg Clear Sky Surface Shortwave Downward Irradiance (kW-hr/m^2/day)) | Available at a 1x1 degree resolution. It claims to be ½ by ½ but doesn’t seem to output that for 2019 at least. – <https://power.larc.nasa.gov/data-access-viewer/> |
| Clear sky index Kc: “Ratio of the global horizontal irradiance to the global horizontal irradiance under clear sky conditions. It is important not to confuse and hence misuse this definition with those for insolation clearness index and clear sky insolation clearness index.” | Available in the POWER suite of indicators as ALLSKY\_KT. We do not use the normalized value - this transforms the Kc with the latitude - Kc = shortwave direct horizontal (GHI) / shortwave direct top-of atmos -- I haven’t figured out the use for the normalized parameter yet. Maybe comparing different locations? See [here](https://www.star.nesdis.noaa.gov/smcd/emb/radiation/solar_resource_definitions.php) for simple definitions. See [here](https://www.star.nesdis.noaa.gov/smcd/emb/radiation/documents/SRDB_1.0_Parameter_Definitions.pdf) for other details.  Definitions (rather unhelpful, except it explicitly mentions GHI): <https://power.larc.nasa.gov/#resources> | The regional data access panel at the website above allows “NetCDF” export, which can be imported as a raster in Q: <https://ereefs.aims.gov.au/ereefs-aims/help/how-to-open-a-NetCDF-file-with-ArcMap-and-QGIS>  So basically we turn this into a raster of the appropriate resolution and multiply to get a final value from r.sun. |

With this set of inputs, we should be able to calculate the shading effects in essentially any city around the world. The DSM is the hardest thing to find, and we should be able to simulate this with building height data, which is more often available. Let’s test it out in NYC.

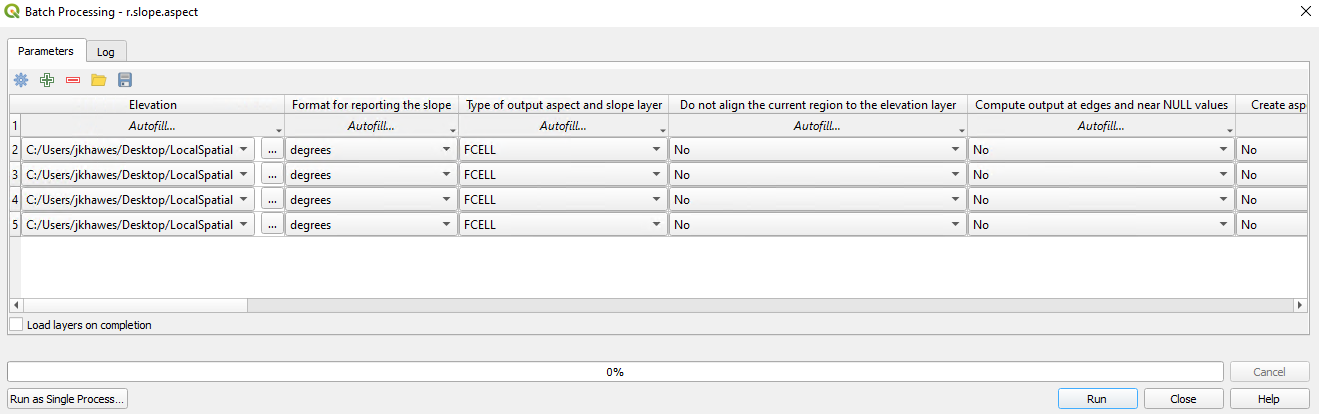
## Clip DSM to work better with GRASS

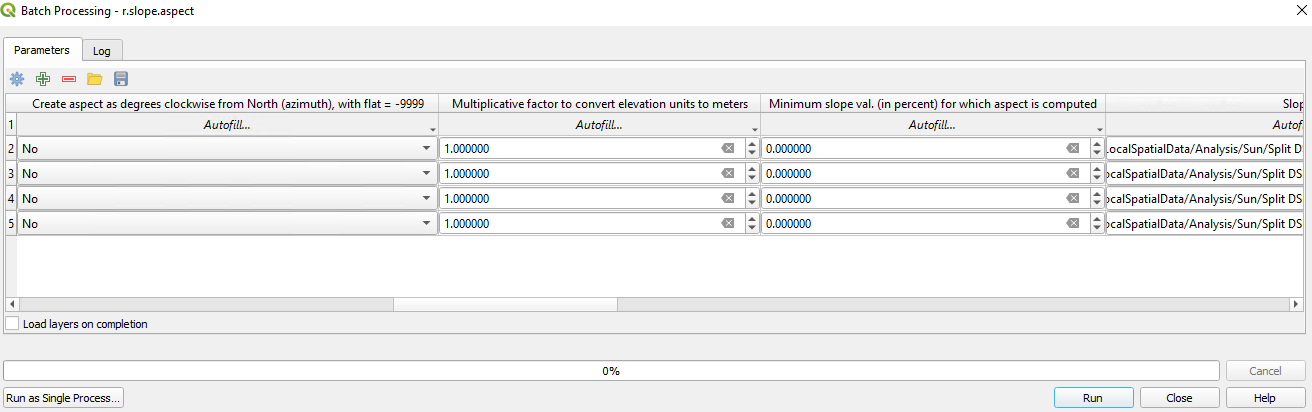
For some reason, of all the commands we use, the GDAL commands embedded in r.sun are the only ones that use the default maximum raster size, and they won’t save anything too big. The max size is about half the size of the London rasters. So before moving forward, we have to clip the DSM to work with this limitation. This is best achieved by saving the raster in several chunks. We can do this in a whole variety of ways, but the simplest is by just Exporting the DSM layer and saving it as four chunks. To do this, we right click on the layer in QGIS, then we go to Export > Save As. At the top of the box, check Create CRT, which should open up the option to create multiple tiles. The reason this is called a VRT is that it saves a VRT file along with the tiles that will load the whole thing - but you can treat the individual tiles as rasters like anything else. You can set the VRT tiles options to half the size of the raster, which should turn it into a 2x2 grid that will run nicely with the r.sun options.

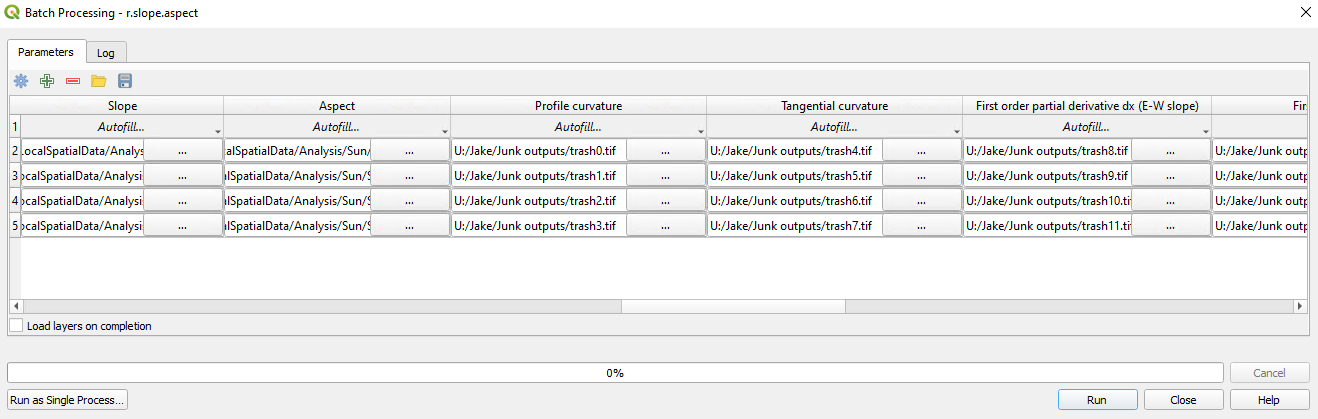


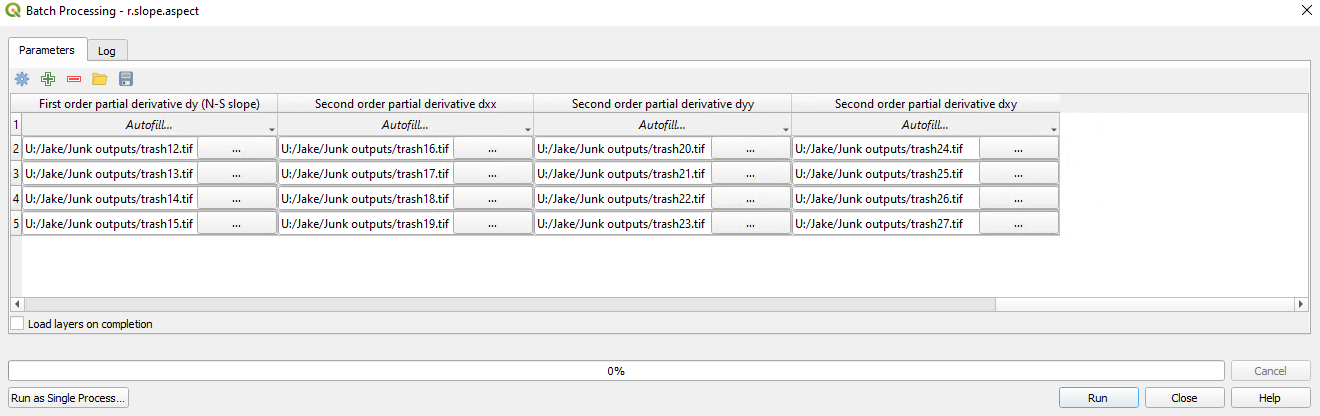
## r.slope.aspect

Once we have a useable DSM, we need to produce maps of the slope and the aspect based on our DSM. We could use the DSM slope map we already have, but it’s just as easy to just run it all within GRASS to make sure everything is formatted the way r.sun wants it to be. This command is fairly straightforward, only a couple things need to be customized. First, we need to uncheck the box that asks about aligning with the elevation region. We do want to align all of our calculations with that region. Second, if you are willing to run these things individually, you want to suppress the outputs other than slope and aspect. If, like me, you want to run them overnight and not worry about it, then you’ll want to set the files to save on a network drive of some sort so you don’t fill your hard drive with junk. Remember that these are very large files and producing 28 extra 10GB temp files would do a number to even the hardiest computer. Saving them on the network might take longer, but it should allow the whole thing to run smoothly without filling up.



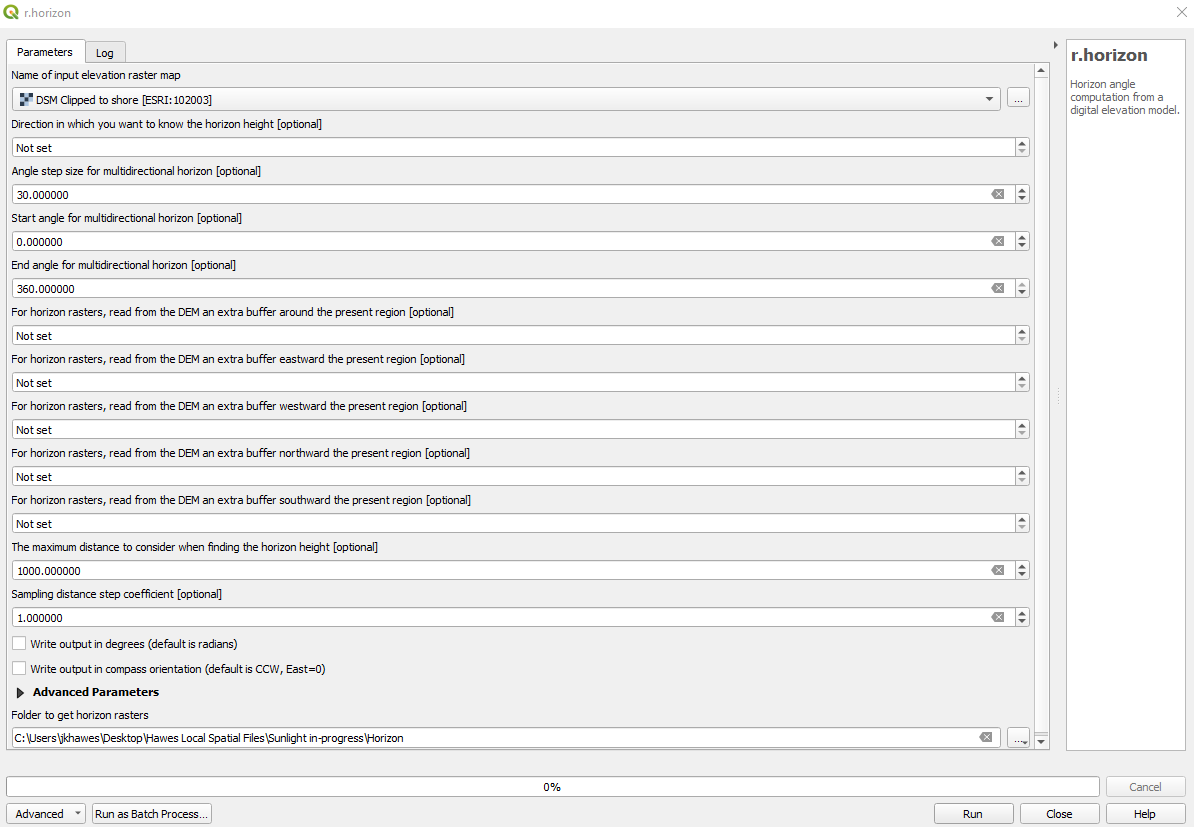




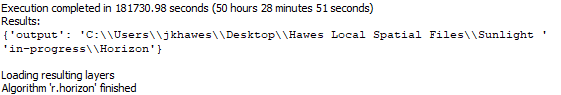


## Optional: r.horizon

If you are planning to use GRASS GIS more directly via commands, it can be very helpful to run r.horizon to determine the horizon height at all locations in the city. Unfortunately, if you plan to run the r.sun suite via QGIS, the interface does not play nice with loading an entire directory, which is the required format of the r.horizon output and the r.sun input.

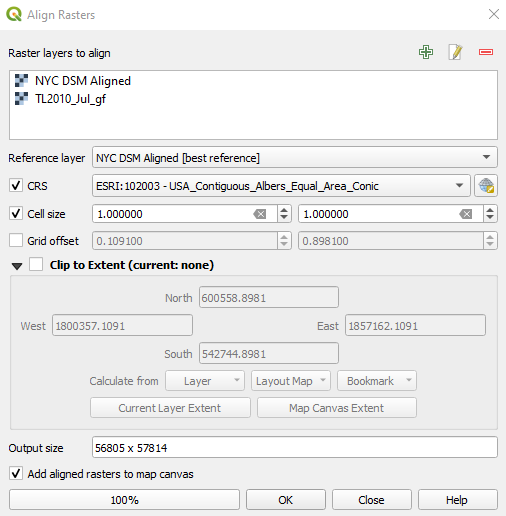


This will take a very long time to run - on the order of 50 hours on the Beast at 1m resolution. In fact, the last time I ran this, it ran for 50 hours and only saved 120-360, so you may have to run it twice to convince it to save everything correctly.



## Aligning Linke

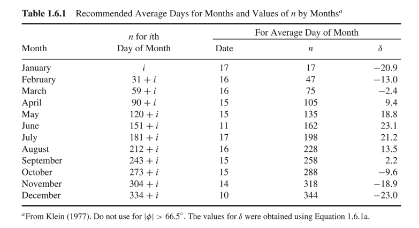
The last step before running the final command is to align the Linke values with the files we’ll be using. This will also clip the raster and will probably take 10-15 minutes.



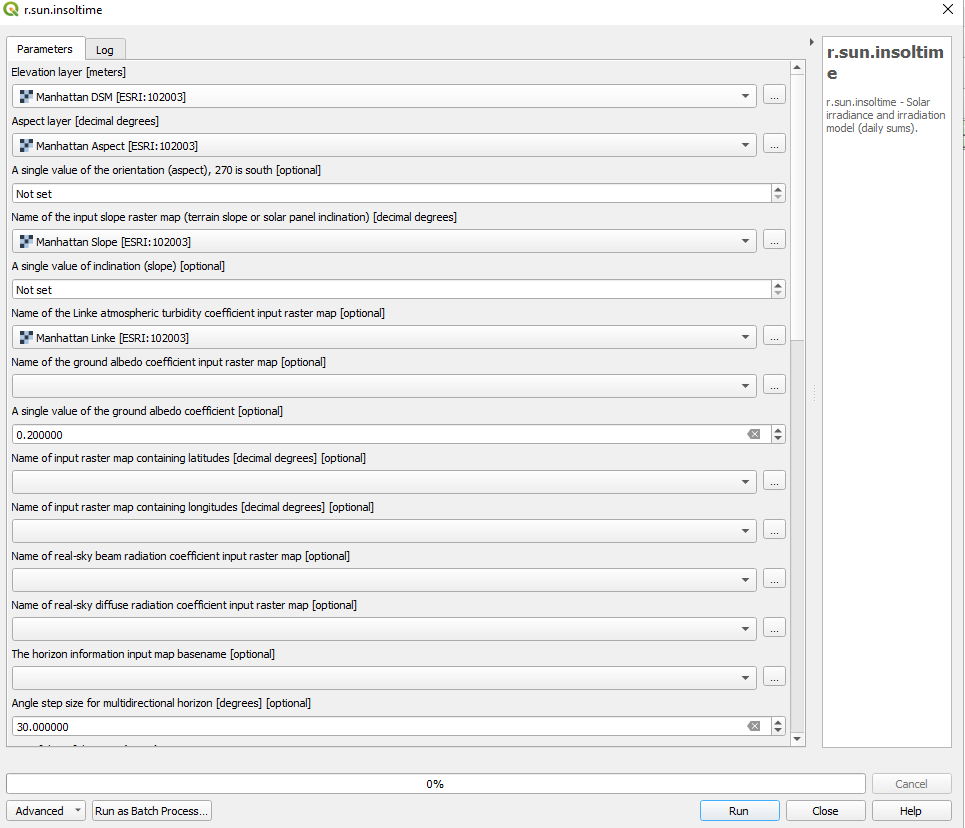
Once the Linke values have been converted to a 1x1m raster, we should be able to clip this to the borough boundaries, as we’ve done with the others.

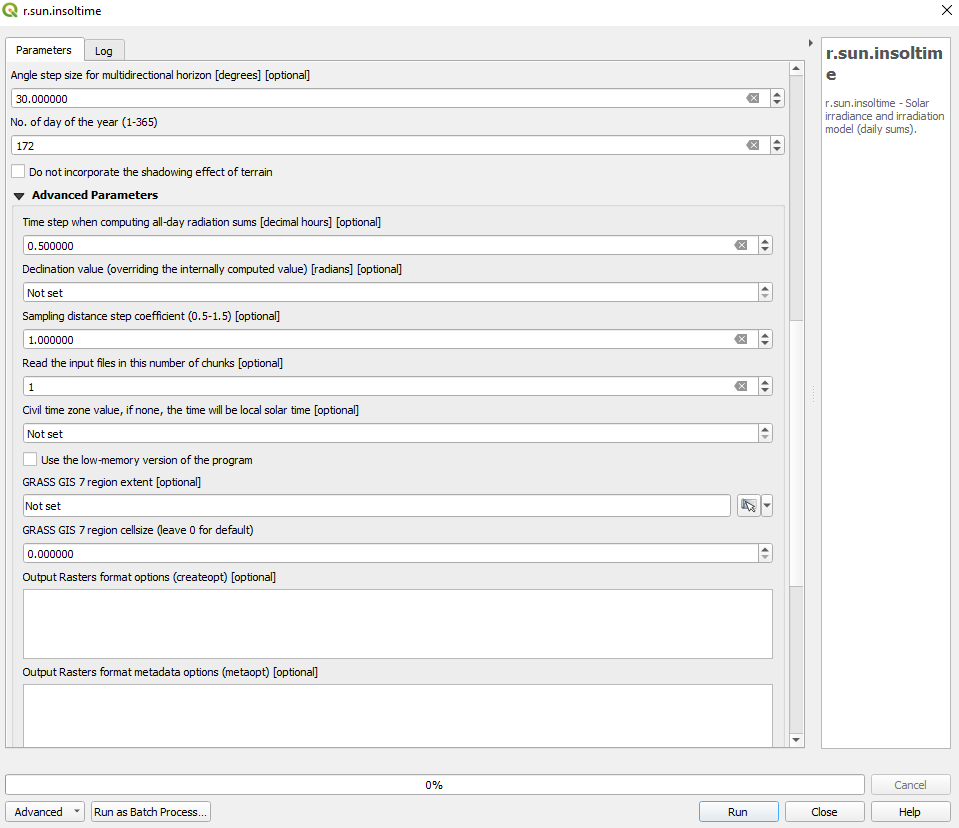
## r.sun.insoltime

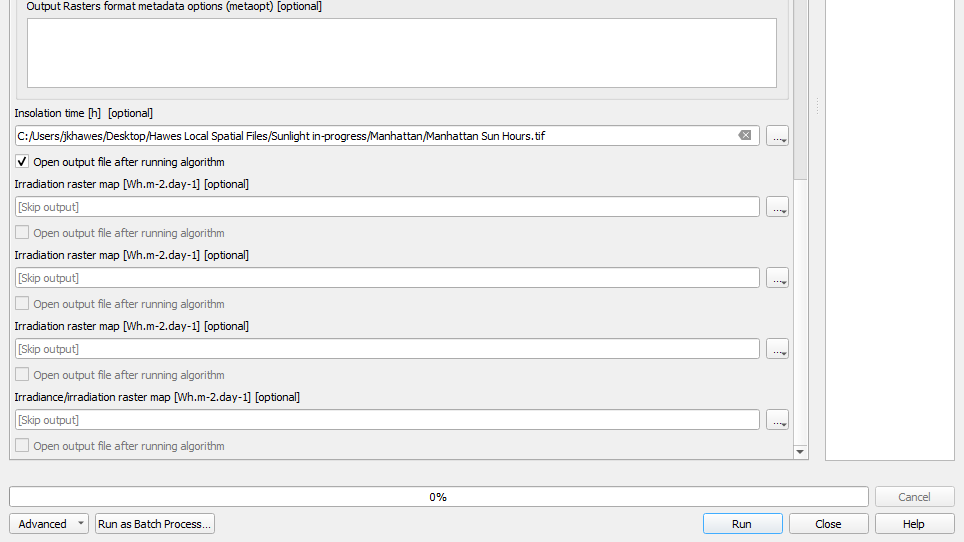
If we are strictly interested in the number of hours of sunlight, then we can simply ignore the more complex aspects like levels of radiation from the NASA data. We can retrieve average days from the book cited in the Canadian paper:



This means that the inputs look like this:





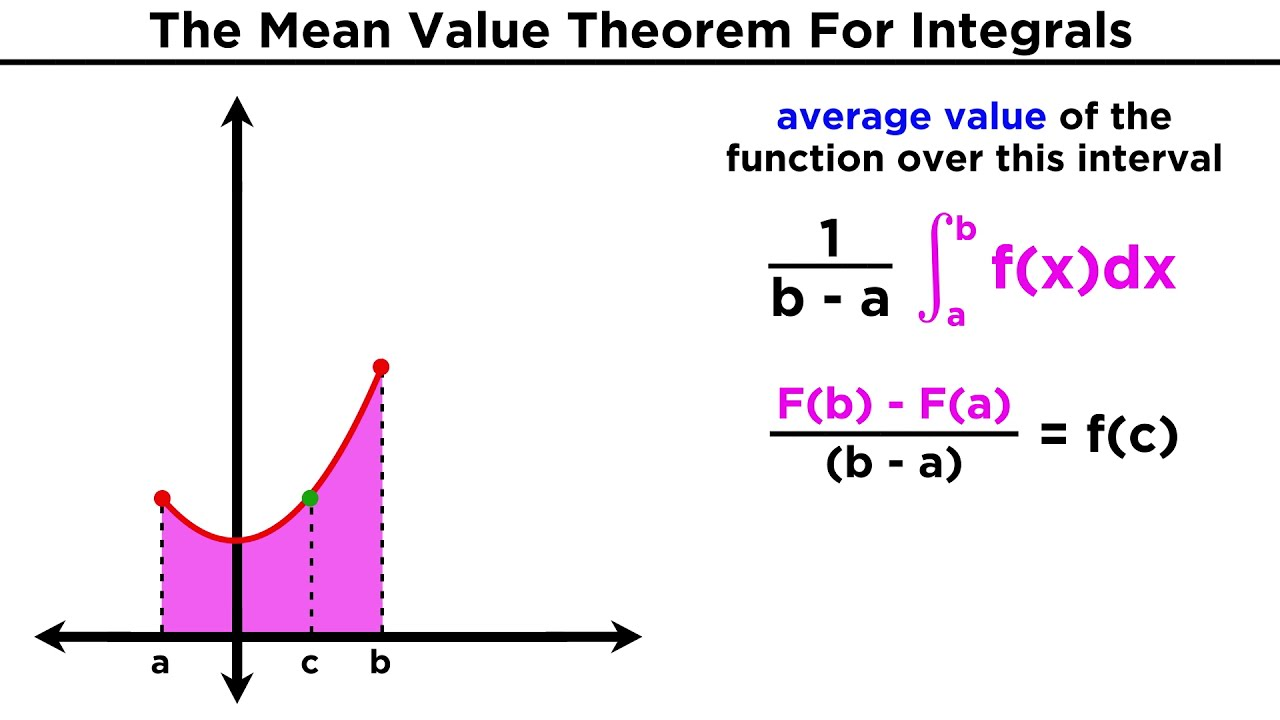


These commands run slowly, but stubbornly. If you tile it like I did, then set 0 is the largest and takes about 2 days to run. 1 and 2 both take between 1 and 1.5 days to run, and 3 is slightly less than a day. For some reason, things run faster as the process proceeds, so by the time you get to October all of these numbers are smaller. At the end of it all, it took about two weeks to run for me on a high-powered modeling machine.

## Merge results and calculate average

First, we obviously need to recombine these tiles. This can be run as a batch process and should take several hours to run.

Once we have full sunhours layers for each month, we can take the average value of the function over the desired interval (the time between April average and October average). We can either try to fit and upside down parabola or we can assume two piecewise linear functions. The second produces more reproducible math between locations and on each cell, so we live with that simplification. The rest of the calculus in this whole process has been hidden in algorithms in QGIS, but you can actually follow along as we derive this particular equation. The mean value theorem says that we can calculate the average value of a function over any particular interval on which it is continuous, which our piecewise function is. For piecewise functions, we calculate the mean value theorem of the constituent pieces and then take a weighted mean.



We derive our equation with the standard y = mx+b, assuming that slope is linear:

For April to July, this is:

For July to October, this is:

So once we conduct the integration, the y goes away and the x gets filled in, but that still leaves us with b1 and b2. So before we can jump to the mean value theorem, we have to calculate the value of b1 and b2. We can do this by simply plugging in values we have already:

If we plug July into April to July, we have:

Solving, we end up with:

If we plug July into July to October, we have:

Solving, we end up with:

We can run both of these as raster calculations and end up with b1 and b2 as rasters. Now we can go ahead and integrate.

This is obviously a more complicated bit of math. For April to July, we end up with:

Following this same math for July to October, we end up with:

At the end, this turns out to be five raster calculations. We calculate and first with separate raster calculations, then we can directly calculate and .

The final calculation is just one final raster calculation - .

For b1, the raster calculation looks like this: "July Sunhours@1" - ( 198\* ( ( "July Sunhours@1" - "April Sunhours@1" ) / 93 ) )

For b2, the calculation looks like this: "July Sunhours@1" - ( 198 \* ( ( "October Sunhours@1" - "July Sunhours@1" ) / 90 ) )

For MeanValue1, the calculation looks like this: ( 28179 / 17298 ) \* ( "July Sunhours@1" - "April Sunhours@1" ) + "b1@1"

For MeanValue2, the calculation looks like this: ( 43740 / 16200 ) \* ( "October Sunhours@1" - "July Sunhours@1" ) + "b2@1"

For MeanValueOverall, the calculation looks like this: ( ( 93 \* "MV1@1" ) + ( 90 \* "MV2@1" ) ) / 183