Paris 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. An absolute treasure trove of datasets is available at the National Institute of Geographic and Forest Information (IGN) [website](https://www.ign.fr/institut/identity-card). The website has a built-in English translation for the basic pages - not sure how well it works for the [catalog pages](https://geoservices.ign.fr/catalogue). Among useful files we don’t use here are the nationwide [Ortho dataset](https://geoservices.ign.fr/documentation/donnees/ortho/bdortho), [LiDAR](https://geoservices.ign.fr/lidarhd), etc. Instead, we use:

* BD Topo - [download](https://geoservices.ign.fr/bdtopo#telechargementshpdept) and [metadata](https://geoservices.ign.fr/sites/default/files/2023-01/DC_BDTOPO_3-3.pdf)
* DSM (MNS, modèle numérique de surface) - [download and metadata](https://geoservices.ign.fr/modeles-numeriques-de-surfaces-correles)
* DTM (MNT, modèle numérique de terrain) - [download and metadata](https://geoservices.ign.fr/rgealti#telechargement1m)
* Vegetation height - [Hauteur Vegetation](https://opendata.apur.org/datasets/hauteur-vegetation-2021)
* Infrastructure information layer - [Equipement emprise infrastructure](https://opendata.apur.org/maps/equipement-emprise-infrastructure-transport)
* Parcel Express - [download](https://geoservices.ign.fr/ressource/218607) and [metadata](https://geoservices.ign.fr/documentation/donnees/parcellaire/parcellaire-express-pci)
* Cadastral parcels - from a [WMS server](https://wxs.ign.fr/parcellaire/geoportail/r/wms?SERVICE=WMS&VERSION=1.3.0&REQUEST=GetCapabilities)

At the end of this analysis, we will have transformed these inputs into four aligned rasters at 1m resolution. All analysis will be conducted in EPSG 2154, RGF93 v1 which is based on a Lambert Conformal Conic reference system. This system is used by the French government and records distance in meters, so it is consistent with our other cities. Analysis is conducted at the administrative boundary of cities, which varies substantially by region. In France, Paris municipality is the same as Paris Department - department 75. So this is our boundary for analysis. Based on the findings of this analysis, we will be able to develop a variety of scenarios that capture the possible areas of expansion for urban agriculture in Paris.

Notes on spatial extent of analysis, since some of these data will likely change by the time you’re reading this, and it may be helpful to consider how I thought through the various scales of analysis: Paris as a city is relatively small, so it may be interesting to conduct analysis at a higher spatial scale - one interesting option is at the level of the Ile de France Region, which is effectively the state of which Paris is the capital (as well as being the national capital). For more information on administration in Paris, see [here](https://en.wikipedia.org/wiki/Administration_of_Paris). As far as I can tell, most data we are using are available at the level of the Ile de France if we choose to do this. However, this is a huge region that far exceeds the size of the Paris urban agglomeration. A third option for consideration, then, is the Paris urban agglomeration, sometimes called the Paris Metropole or Grand Paris. This is a recently created distinction (2016) that functionally rests some substantial planning power at the level of overall urban area - see [here](https://metropolegrandparis.fr/fr/la-metropole-du-grand-paris). Unfortunately, it does not seem to be a level that IGN or other data groups recognize - and there are small sections of Grand Paris that extend across rational administrative boundaries. In this case, we would need all the data at the Ile de France level, which we would then clip to the administrative boundary with the file at [this WMS server](https://ogc.geo-ide.developpement-durable.gouv.fr/wxs?map=/opt/data/carto/geoide-catalogue/1.4/org_38134/efc7b741-039a-4c6d-adb4-385bf6eda43a.internet.map&service%20=WMS&request=GetCapabilities) - link from [here](https://www.geocatalogue.fr/Detail.do?id=2096453). This is probably the most precise option, but it may require some data at the state level that don’t exist. This, then, brings us to the final option, which is another way of dicing a “scaled up” option. We could conduct this analysis on the city of Paris and the Petite Couronne - or Paris and the three departments bordering it. All but seven communes in the Paris Metropole are included in Paris and the Petite Couronne, meaning that it is a pretty good approximation of the new urban agglomeration planning district. If we cannot locate all the state level data, we could do this instead. I was able to piece together data at the level of Ile de France, the various Regions, and the Paris Metropole and do the analysis for the entire Paris Metropole. I cannot guarantee that these data will always be available at this level, given the administrative complexity.

# 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. DSM (MNS, modèle numérique de surface) - [download and metadata](https://geoservices.ign.fr/modeles-numeriques-de-surfaces-correles)
2. DTM (MNT, modèle numérique de terrain) - [download and metadata](https://geoservices.ign.fr/rgealti#telechargement1m)

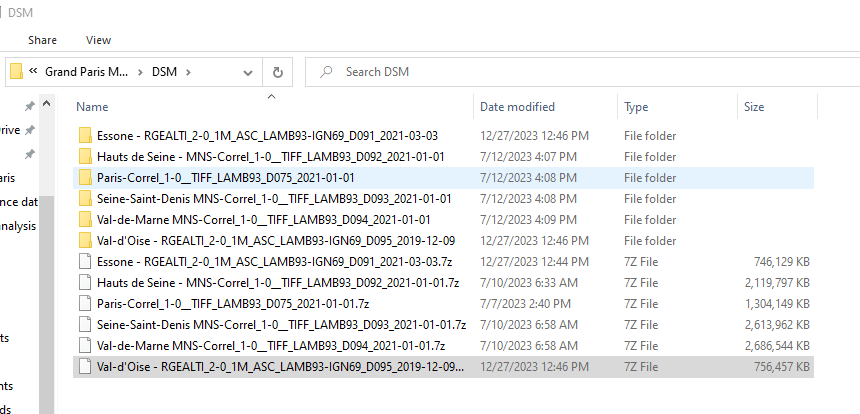
We will begin by preparing city-wide DTM and DSM layers, then we will proceed with the flat ground analysis. We will then identify flat roofs. Finally, we will mask buildings from the ground layer and add in the buildings results. This takes a while because of the format of download and size of the files.

## Download IGN data products

IGN produces a suite of products useful for this analysis, including a number of LiDAR- and orthophotography-derived products. For the slope derivation, we can use the DTMs (RGE ALTI) and DSMs (MODÈLES NUMÉRIQUES DE SURFACES CORRELÉS) made available through the online data download. Unfortunately, it’s not perfectly straightforward to run this analysis at the Metro level, because the files are only available at the scale of the Departments, not the Region or the Metropolitan area. So we have to download the data of all the constituent departments:

* Department 75: Paris
* Department 92: Hauts de Seine
* Department 93: Seine-Saint-Denis
* Department 94: Val-de-Marne

You will also need a few tiles from Departments 91 and 95 (Essonne and Val-d’Oise, respectively). You’ll have to download the whole thing for both of these, but you’ll only use a small subset of the tiles. Unzip (using 7-zip) each of these departments into their own folders so you can keep track of inputs. For the two larger departments (91 and 95), when you do the DSM you have to download all parts of the zip file (.001 thru .004) and highlight them all, then ask 7-zip to extract. It will look something like this at the end:

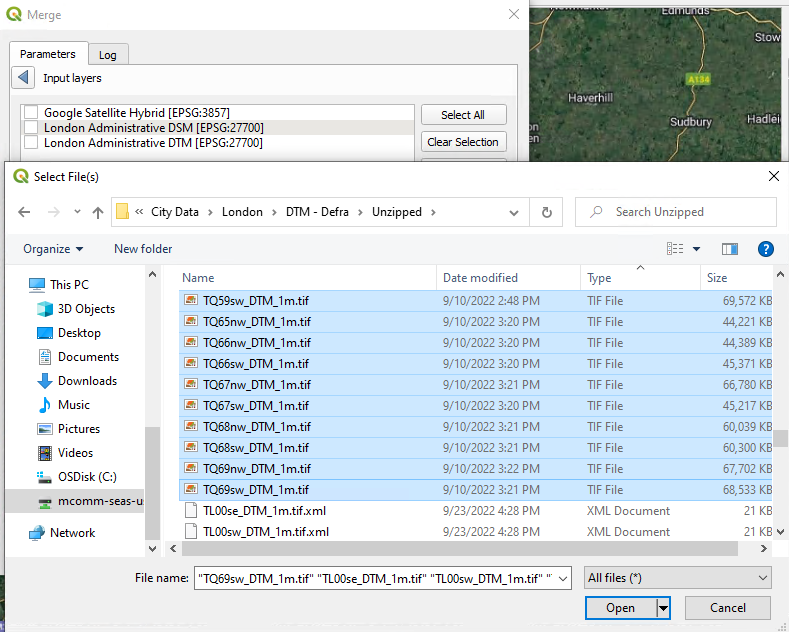


## Merge DSM and DTM files and clip to administrative boundary

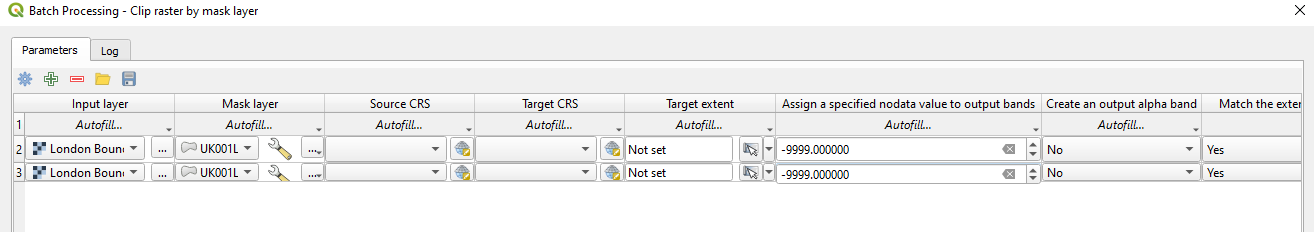
Since these files are downloaded as individual tiles, they will need to be merged to be useful in city-wide analysis. Under Raster >> Miscellaneous, the Merge tool will take a bunch of input layers and create one merged raster layer. You do this with the base DSM files shared in the Modèles Numériques De Surfaces Correlés download, and you do this with the MNT files in the RGE Alti files. DST and SRC provide metadata. You do not have to load the files into QGIS to merge them. So, to make this work, you will merge all the tiles from Departments 75, 92, 93, and 94 and add in the following tiles for Department 91 and 95:

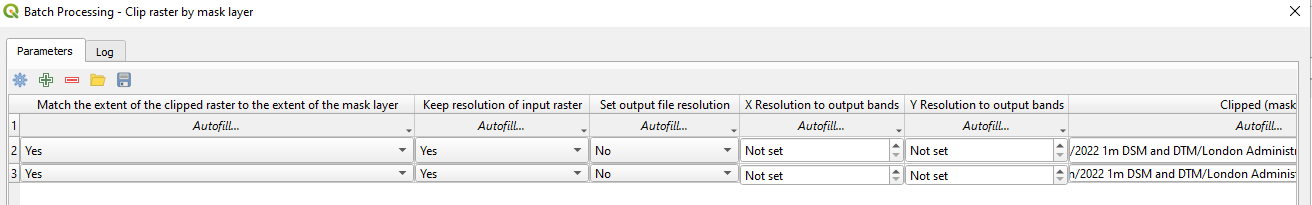
| Dept 91:   * 0649-6844 and 6845 * 0650-6843 thru 6847 * 0651-6842 thru 6849 * 0652-6839 thru 6848 * 0653-6839 thru 6846 * 0654-6840 thru 6846 * 0655-6842 thru 6846 * 0656-6846 |  |
| --- | --- |
| Dept 95:   * 0641-6872 thru 6874 * 0642-6871 thru 6875 * 0643-6872 thru 6876 * 0644-6873 thru 6875 * 0645-6873 thru 6875 * 0646-6874 and 6875 |  |

Functionally, this is the same as what we did in London - see below for an example screenshot in addition to the instructions. Since Departments 75, 92, 93, and 94 are all in one folder, you can go in to the Merge tool, click Add Layer, then click Add Files. Go to the directories where you unzipped everything to and add all the tif files. Next, go to the Essonne and Val-d’Oise folders and add the select few tiles we need. AFter everything is added, click “OK” otherwise the list of inputs does not always save. You should end up with somewhere around 1000 files being merged (I had 928 for my DTM and 1038 for my DSM - because of duplicates across departments). Save the file as something like “Metropole du Grand Paris DTM Unclipped” 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 do two things to save some computational time. First, upscale this to 1m resolution - it defaults to 0.5m. This is as easy as right-click -> Export -> Save As. Set the directory and the resolution to 1m x 1m, then save the new file. You should also clip the files to the administrative boundary - batch processing works for this. Using the Metropolitan Boundary shapefile layer, you can use the Clip Raster by Mask Layer tool and make everything outside the administrative boundary null. The order of these actions doesn’t really matter. You save a little time doing it in this order, but it’s pretty minor all things considered.





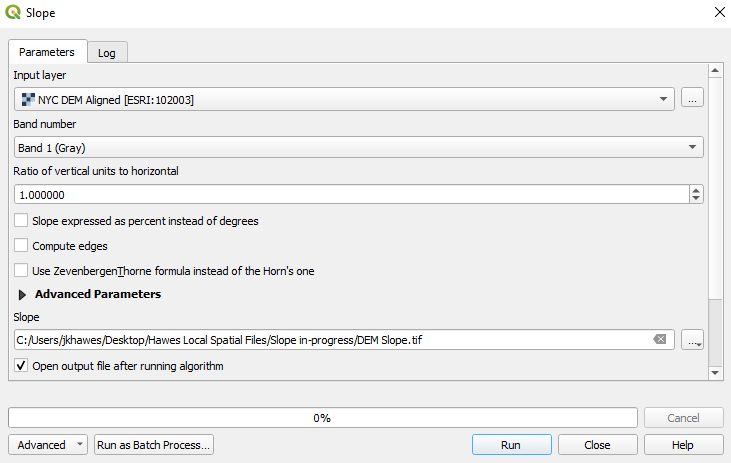
The output should look something like this (but in the shape of Paris, this is an image of London just as an example):



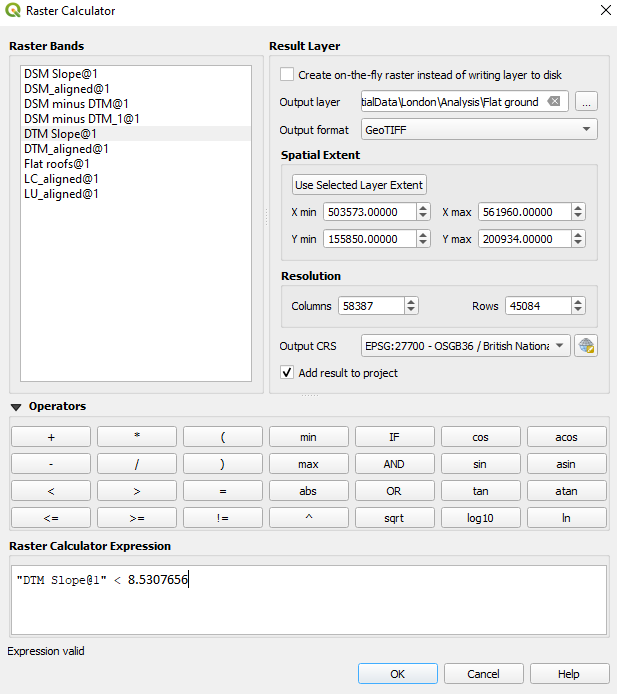
## Calculate Slope from the DTM

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. For the record, the first step on the next section is also going to be running a slope command, so you can run slope on the DSM now if you see fit. Just name it DSM Slope or some such thing and put it in a folder for safe keeping.

\*\*Don’t worry about the occasional screenshots from NYC, all the commands are the same - just different file names

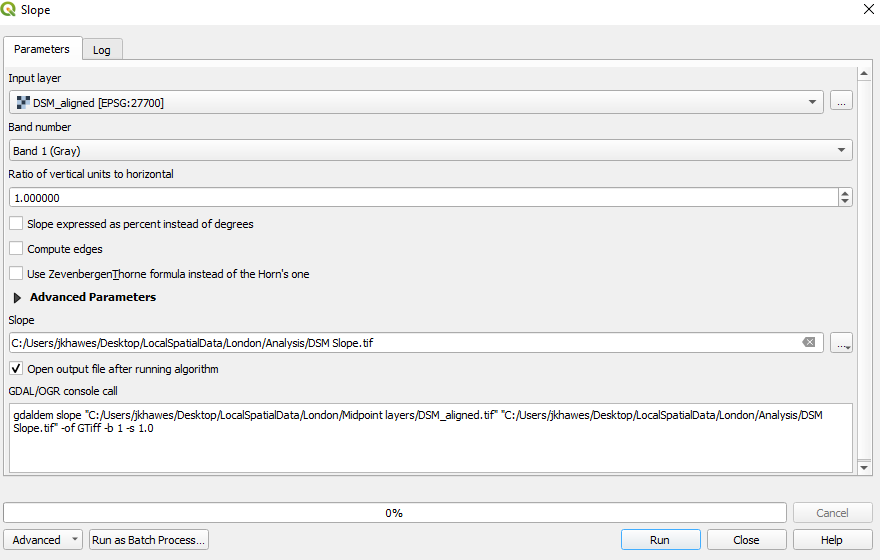


Once we have this Slope file, we can do some simple raster algebra to determine where this is greater than and less than 15%. Expressed as degrees, a 15% slope is arc-tangent of 0.15, which is 8.5307656. So we want to find places where the slope layer is less than that.

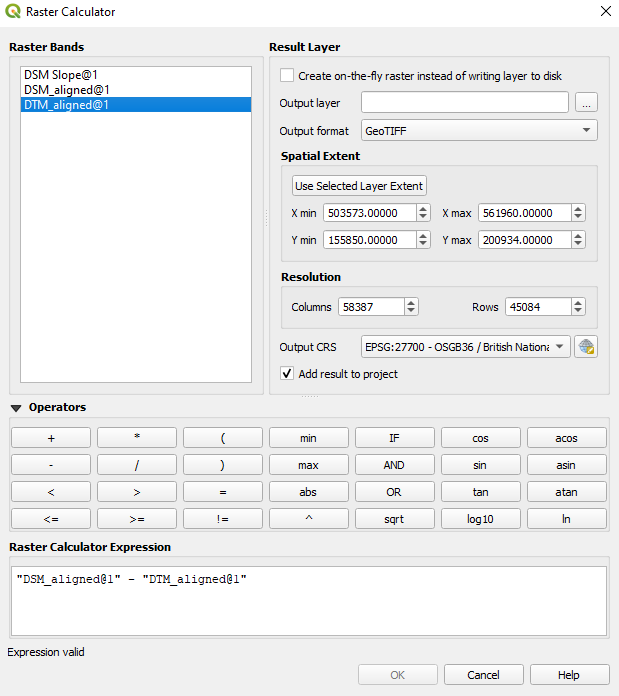


## Identify flat roofs from the DSM

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.

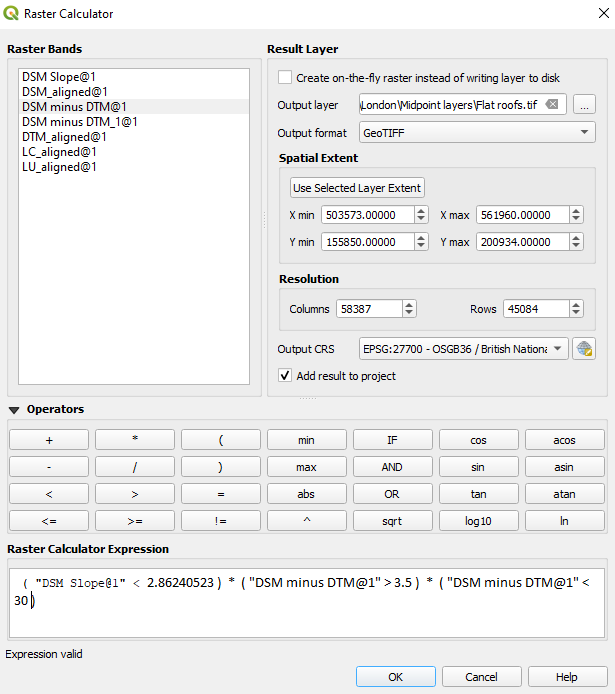


Next, when we need to subtract the DEM from the DSM to make sure the ground level is zero all over the map.



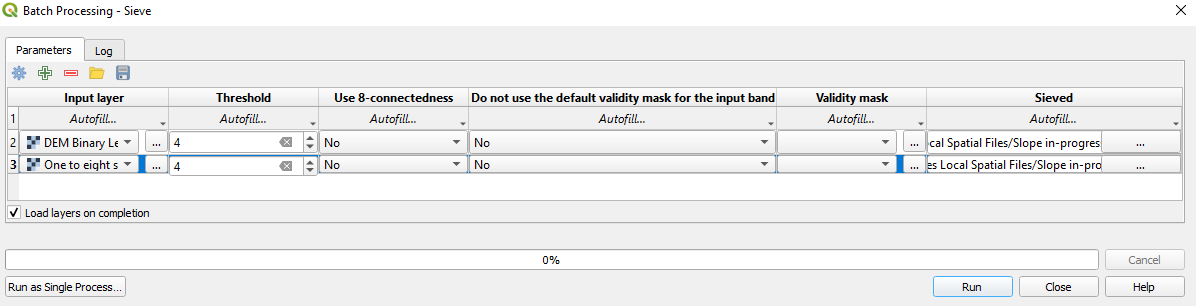
We can then run some simple raster calculations, finding the places where the Slope layer is less than arctangent of 0.05 (2.86240523) and the Height of the DSM-DEM layer is more than 2.5m and less than 30m (between 1 and 8 stories) - note that this height is why we need to do the subtraction. If not, 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.

Code: ( "DSM Slope@1" < 2.86240523 ) \* ( "DSM minus DTM@1" > 3.5 ) \* ( "DSM minus DTM@1" < 30 )



## Clean up layers with the sieve tool

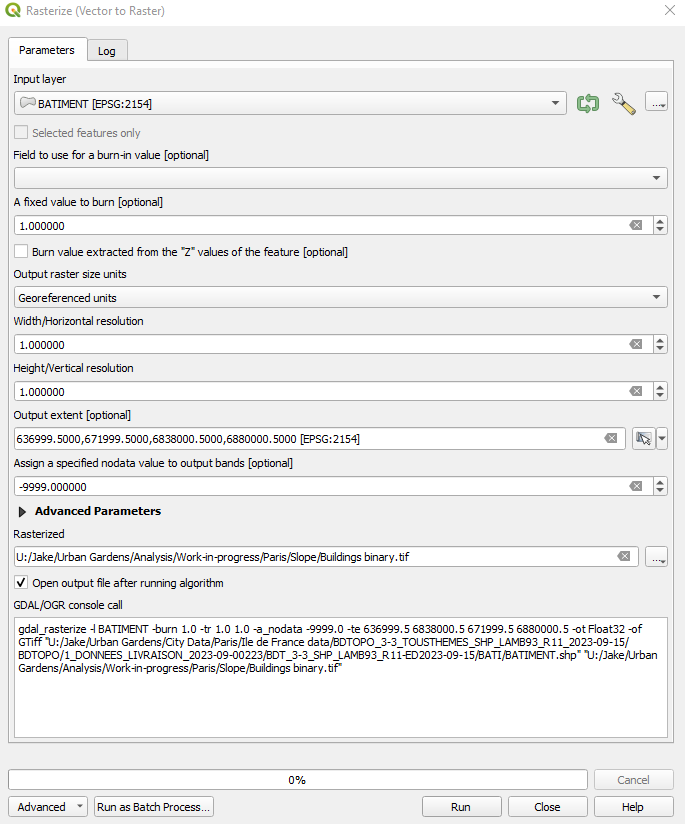
Once we have useful results for both round and rooftops, we can use the sieve tool to clean it up a bit. I used 4 as a threshold because it tends to catch edge cases most effectively. Make sure to fix the symbology to 0-1 after running the function.



## Final raster calculations

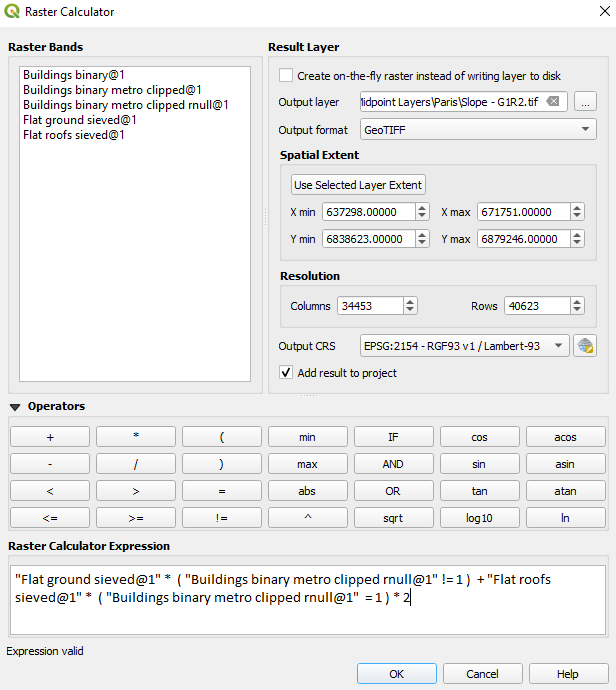
Lastly, we need to combine these layers into something intelligible - we will keep flat roofs and flat ground separate for now just for the sake of preserving information. We can always reclassify later. So our goal is: 0 = > 15% slope, 1 = flat ground, 2 = flat roof. How do we get there?

First, we need a building layer to compare to - we can simply convert the buildings shapefile into a raster, aligning it with the slope rasters as we go. So, step 1 is to load Batiment from the Ile-de-France BD Topo, then rasterize it, setting the extent to match the clipped DSM (which will partially clip and align the building layer - see below for settings). Second, clip this again to the bounds of the metro area, Finally, run r.null to set all null values to zero.



Once we have a building layer, we can use it 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. The raster calculation is exactly the same for the two layers, but inverted. See below.

Code: "Flat ground sieved@1" \* ( "Buildings binary metro clipped rnull@1" != 1 ) + "Flat roofs sieved@1" \* ( "Buildings binary metro clipped rnull@1" = 1 ) \* 2



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.

# 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
* Low vegetation (typically grass or dirt) - 2
* Roof - 3
* Trees (medium or high vegetation) - 4
* Otherwise occupied - 0 (e.g., monument, water, railroad, road)

We derive land cover information from the following layers:

* BD Topo, one of the core parts of this overall analysis - [download](https://geoservices.ign.fr/bdtopo#telechargementshpdept) and [metadata](https://geoservices.ign.fr/sites/default/files/2023-01/DC_BDTOPO_3-3.pdf)
* Vegetation height ([Hauteur Vegetation](https://opendata.apur.org/datasets/hauteur-vegetation-2021)) shows us the high, medium, and low vegetation throughout the city at 1m resolution.
* Open Street Map data (for water) - can download in a number of ways, including direct downloads with QGIS plugin.
* Infrastructure data layer from APUR: [Equipement emprise infrastructure](https://opendata.apur.org/maps/equipement-emprise-infrastructure-transport)

## Reclassify vegetation height to use as base layer

The first thing we will do is use the vegetation height layer to develop a basic “vegetation or not” layer. Specifically, we need to declassify the results of this layer so that all values appear as strictly height, rather than classified by land use, then we can move on to classifying the different heights as high or low vegetation. The online guide has this to say about the declassification:

Private vegetation: vegetation height = pixel value

Public vegetation: vegetation height = pixel value - 100

Vegetation outside MGP: vegetation height = pixel value - 1000

This translates to the following code:

if ( "HAUTEUR\_VEGETATION\_2021@1" < 100, "HAUTEUR\_VEGETATION\_2021@1", if ("HAUTEUR\_VEGETATION\_2021@1" < 1000, "HAUTEUR\_VEGETATION\_2021@1" - 100, "HAUTEUR\_VEGETATION\_2021@1" - 1000 ))

There will inevitably be some artifacts in a file this large, so we can just set anything remaining over 75m to 0. There are no trees taller than 75m in Europe (and the one closest is in Portugal). Unfortunately, this takes two steps. First, we make anything greater than 100 equal to zero and make the other values equal to 1. You can use r.reclass to do this with the code:

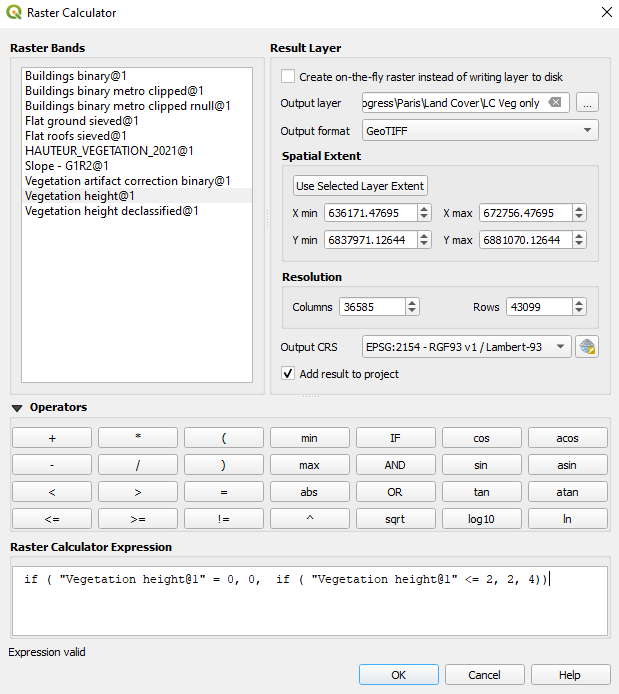
0 thru 75 = 1

75 thru 10000 = 0

Then we multiply the new raster by the old one to get a declassified height file that has nothing over 75.

"Vegetation height declassified@1" \* "Vegetation artifact correction binary@1"

Once we have a layer with no impossible artifacts, we can reclassify it to represent high and low vegetation. The ASPRS uses 2m as a maximum height cutoff for low vegetation, so we will use the same (see Khosravipour, 2014 - citation below). In this case, we should be able to do this with a pretty simple raster calculation: if ( "Vegetation height@1" = 0, 0, if ( "Vegetation height@1" <= 2, 2, 4))



Khosravipour, Anahita, Andrew K. Skidmore, Martin Isenburg, Tiejun Wang, and Yousif A. Hussin. “Generating Pit-Free Canopy Height Models from Airborne Lidar.” Photogrammetric Engineering & Remote Sensing 80, no. 9 (September 1, 2014): 863–72. https://doi.org/10.14358/PERS.80.9.863.

## Reclassify remaining area as pavement, then overlay “other” features

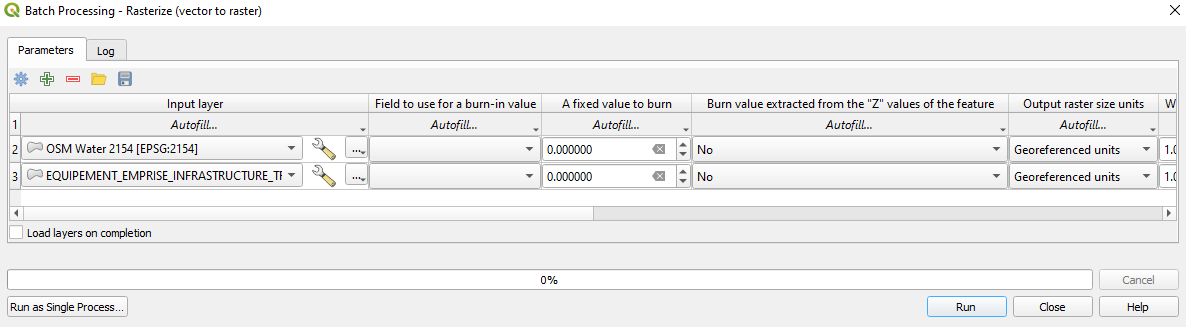
Most of the remaining space is either a building or pavement. We will introduce buildings last, since that is one of the layers we have with the most certainty. So we can reclassify all zeroes as ones, then reintroduce zeroes with overlays. First, r.reclass the LC Veg and Building file with

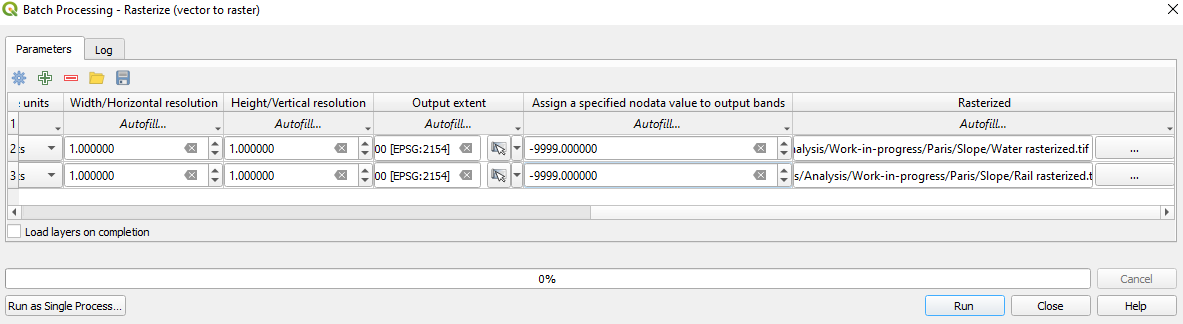
0 = 1

\* = \*

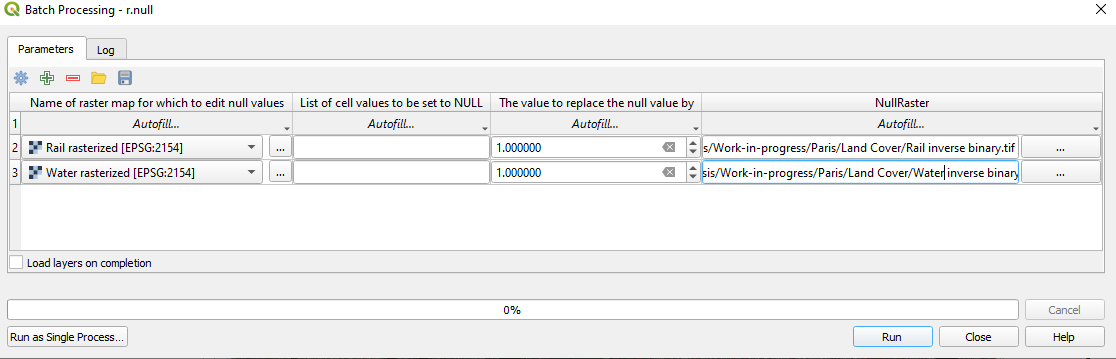
The \* = \* is a special case that tells the algorithm to leave anything not explicitly mentioned as it is.

From here, we add in the river and the rail infrastructure. To do this, we can create reverse binaries - everything except water or rail is 1, and those two are zero. Then we run multiplication with the raster calculator.

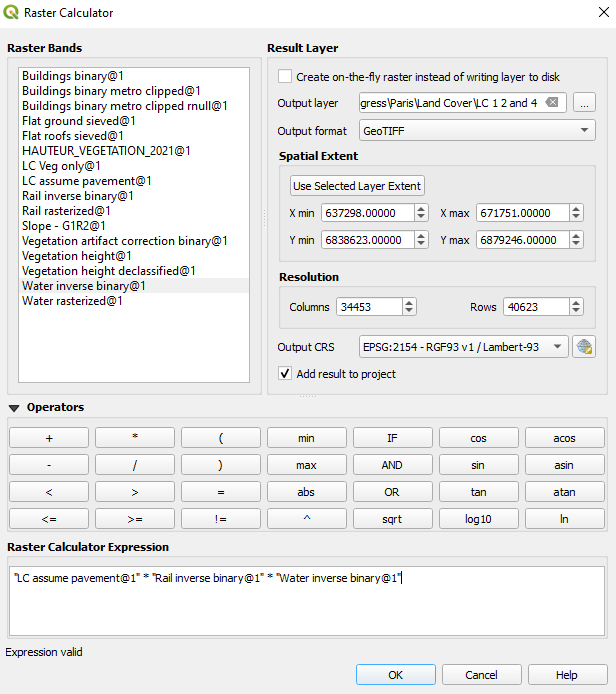
The first step here is to rasterize our water (gis\_osm\_water\_a\_free) and rail layers (EQUIPEMENT\_EMPRISE\_INFRASTRUCTURE\_TRANSPORT) to make them easy to distinguish from pavement. Before we can do that, we need the OSM water file in EPSG 2154 - we use reproject for this. We’ll set the burn-in value to 0 and the null value to -9999 in both cases. Use the buildings layer to set the extent (once again, this is not just useful for speeding things up, it also aligns the rasters. ****

****

Once this is run, we’ll use r.null to reset everything that was null to 1.



Then, we use this multiplication:



## Add buildings layer to LC

Since our building layer (from BD Topo, linked above) is generally high reliability, we’ll consider this as a simple overlay. Lots of ways to do this, but my personal preference is to run it as scaled addition, then use r.reclass. The current building layer is a 1-0 binary. So we multiply this by 10, add it to the existing LC layer, and then reclassify anything above 10 as 3 (buildings):

Raster calculation: ( "Buildings binary metro clipped rnull@1" \* 10 ) + "LC 1 2 and 4@1"

r.reclass:

10 thru 20 = 3

\* = \*

This new file can just be named LC, because it captures the vast majority of variation in land cover in the greater Paris area. This is our final land cover product.

# Land Use 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).

The regional planning commission in Paris recently (fall 2023) upgraded their land use analysis to better support new initiatives related to impervious surfaces. This effort yielded an 81-class land use layer, which is not perfect, but is better than we could derive from a hodge-podge of other layers. The

## New method: Using MOS+ 81-class layer

This is a fairly simple reclassification, since we’re using their layer and just orienting it to our FEW-meter classification scheme. The reclassification guide can be found [here](https://docs.google.com/spreadsheets/d/176HUxNceFgc3QG77QbuSUSxrKWQ1HY6M/edit?usp=sharing&ouid=102554365591872343075&rtpof=true&sd=true).

1. *Various information on this updated land use analysis process can be found online - https://artificialisation.developpement-durable.gouv.fr/bases-donnees/mos-ile-de-france & https://www.club-plui.logement.gouv.fr/IMG/pdf/20220414\_club\_plui\_mos.pdf. Some information about the derivation is available here - https://www.institutparisregion.fr/fileadmin/DataStorage/IauEtVous/CartesEtDonnees/Mos/MOS\_plus\_Presentation.pdf*

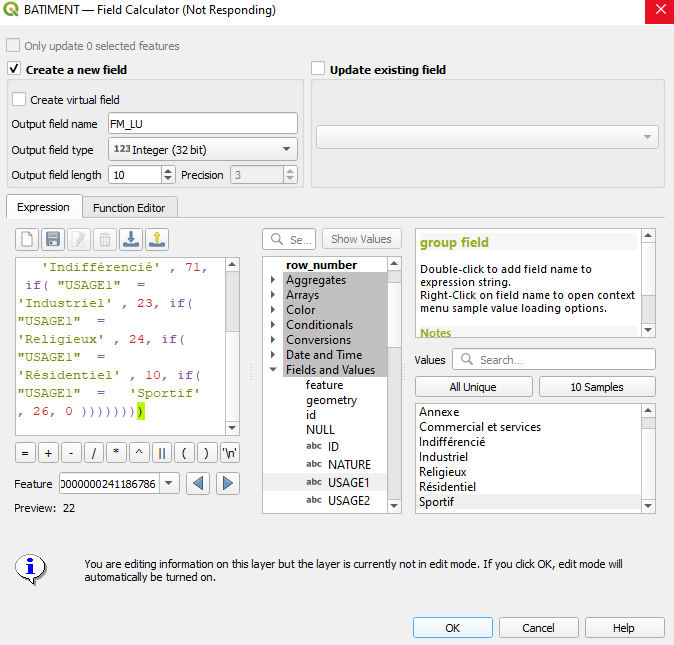
## Old method - overlays to generate LU:

This old version was based on the description of the MOS+ 81-class methods. We used very similar layers and overall generated a fairly reliable map. However, we did not have access to the full layer set or the local knowledge supporting the reclassification, so we default the layer generated locally when it’s available. If for some reason you can’t get the 81-class layer, you may still want to use the old method. For this, we used:

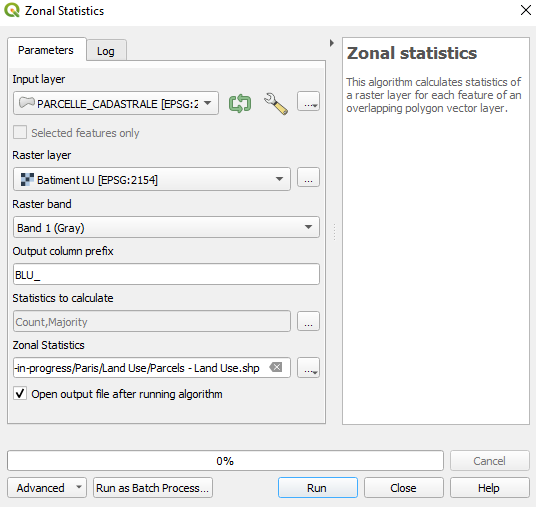
* BDTopo - Batiment
* Parcellaire Express
* Occupation du Sol
* Espaces Publics
* Espaces Verts

### Merge Occupation du Sol data and Recode Land Use with Field Calculator

1. Take Batiment Usage\_1 and send it to parcels (Parcelle\_Cadastrale)
   1. Convert USAGE1 to numeric, then rasterize and take the mode.

****

Rasterize with all the usual settings, burn in the LU value, and set -9999 to the NULL. Make sure to use an existing layer as the extent so it aligns with our other layers - this doesn’t matter much, but it allows us to use this layer in the future if it ever becomes helpful. Once we have a raster, we can run zonal statistics. We’re looking for the ‘mode,’ which QGIS calls ‘majority.’



#### Merge Occupation du Sol files and calculate a generalized land use value from US and CS values.

First, merge the Occupation du Sol files from Departments 75, 91, 92, 93, 94, and 95. As a final effort to speed this up, you could clip this to the boundary of the metropole, but I had a lot of trouble with this and it turned out to be more effort and time than it saved. So I recommend doing the below with the full merged file.

Once you have a merged file, convert CS and US values to our FEW-meter land use values. This is a pretty complex field calculation, so for the sake of posterity:

if( "CODE\_US" = 'US1.1' , 90, if( "CODE\_US" = 'US1.2' , 32, if( "CODE\_US" = 'US1.3' , 25, if( "CODE\_US" = 'US1.4' , 83, if( "CODE\_US" = 'US2' , 25, if( "CODE\_US" = 'US235' , 21, if( "CODE\_US" = 'US4.1.1' , 80, if( "CODE\_US" = 'US4.1.2' , 82, if( "CODE\_US" = 'US4.1.3' , 42, if( "CODE\_US" = 'US4.1.4' , 83, if( "CODE\_US" = 'US4.1.5', 42, if( "CODE\_US" = 'US5' , 10, if( "CODE\_US" = 'US6.1' , 82, if( "CODE\_US" = 'US6.2' , 70, if( "CODE\_US" = 'US6.3' , 70, if( "CODE\_US" = 'US9' , 71, if( "CODE\_US" = 'US3' , if( "CODE\_CS" = 'CS1.1.1.1' , 22, 33), 0)))))))))))))))))

Sometimes long if statements freeze the field calculator - if this happens to you, this Case When code should be lighter.

CASE  
WHEN "CODE\_US" = 'US1.1' THEN 90   
WHEN "CODE\_US" = 'US1.2' THEN 32  
WHEN "CODE\_US" = 'US1.3' THEN 25  
WHEN "CODE\_US" = 'US1.4' THEN 83   
WHEN "CODE\_US" = 'US2' THEN 25   
WHEN "CODE\_US" = 'US235' THEN 21   
WHEN "CODE\_US" = 'US4.1.1' THEN 80   
WHEN "CODE\_US" = 'US4.1.2' THEN 82   
WHEN "CODE\_US" = 'US4.1.3' THEN 42   
WHEN "CODE\_US" = 'US4.1.4' THEN 83   
WHEN "CODE\_US" = 'US4.1.5' THEN 42   
WHEN "CODE\_US" = 'US5' THEN 10  
WHEN "CODE\_US" = 'US6.1' THEN 82   
WHEN "CODE\_US" = 'US6.2' THEN 70   
WHEN "CODE\_US" = 'US6.3' THEN 70   
WHEN "CODE\_US" = 'US9' THEN 71  
WHEN "CODE\_US" = 'US3' THEN  
 if( "CODE\_CS" = 'CS1.1.1.1' , 22, 33)))  
ELSE 0

Once we have this numeric field, we can do exactly the same thing as above and get a parcel file with both land ue derivations. As a reminder, this means to rasterize and then use zonal statistics - with the parcel layer that already has the batiment value.

***Notes:***

1. *Formerly was filtering the CS more stringently in the US3 category - decided it was filtering out a lot of open spaces by labeling them as protected woodlands. Here’s the code that used to do that:   
   if( "CODE\_CS" = 'CS1.1.1.2' OR "CODE\_CS" = 'CS1.1.2.1' OR "CODE\_CS" = 'CS1.1.2.2' OR “CODE\_CS" = 'CS1.2.1' OR "CODE\_CS" = 'CS2.2.1' OR "CODE\_CS" = 'CS2.2.2', 33, if( "CODE\_CS" = 'CS2.1.1.1' OR "CODE\_CS" = 'CS2.1.1.2' OR "CODE\_CS" = 'CS2.1.1.3' OR "CODE\_CS" = 'CS2.1.2' OR "CODE\_CS" = 'CS2.1.3' , 32 , 0)))*

#### Compare the values derived from Occupation du Sol and Batiment

Once these two land use derivations are in the same parcel layer, we can compare them directly. This means converting a series of decision points into a Case When algorithm for Field Calculator.

1. If they’re the same, we call that good.
2. If the building layer was null, we get the value from the OS layer. If one of the two layers was 71 (unknown), use the other. Same with 0.
3. Road, rail, or ag overwrites anything else (80, 82 and 90).
4. If they’re 10 (residential) and 22 (commercial), we can just set it to mixed use (21) - this is pretty common. Same with 21 and 10 or 21 and 22.
5. 10 (residential) can overwrite 32 (silviculture), since that’s a pretty imprecise land use by all indications.
6. There are some cases where there is no clear reason to believe one over the other, so we’ll defer to the buildings layer, since it’s newer (9/23 vs. 2021) and higher resolution in most cases.

*Resulting Case When statement:*

CASE

WHEN "BLU\_majori" = "OdSLU\_majo" THEN "BLU\_majori"

WHEN "BLU\_majori" IS NULL THEN "OdSLU\_majo"

WHEN "BLU\_majori" = 0 THEN "OdSLU\_majo"

WHEN "OdSLU\_majo" = 0 THEN "BLU\_majori"

WHEN ("BLU\_majori" = 10 AND "OdSLU\_majo" = 22) THEN 21

WHEN ("BLU\_majori" = 22 AND "OdSLU\_majo" = 10) THEN 21

WHEN ("BLU\_majori" = 10 AND "OdSLU\_majo" = 21) THEN 21

WHEN ("BLU\_majori" = 21 AND "OdSLU\_majo" = 10) THEN 21

WHEN ("BLU\_majori" = 22 AND "OdSLU\_majo" = 21) THEN 21

WHEN ("BLU\_majori" = 21 AND "OdSLU\_majo" = 22) THEN 21

WHEN ("BLU\_majori" = 10 AND "OdSLU\_majo" = 33) THEN 10

WHEN ("BLU\_majori" = 22 AND "OdSLU\_majo" = 33) THEN 22

WHEN ("BLU\_majori" = 26 AND "OdSLU\_majo" = 33) THEN 26

WHEN "BLU\_majori" = 71 THEN "OdSLU\_majo"

WHEN "OdSLU\_majo" = 71 THEN "BLU\_majori"

WHEN "OdSLU\_majo" = 32 THEN "BLU\_majori"

WHEN "OdSLU\_majo" = 80 THEN "OdSLU\_majo"

WHEN "OdSLU\_majo" = 82 THEN "OdSLU\_majo"

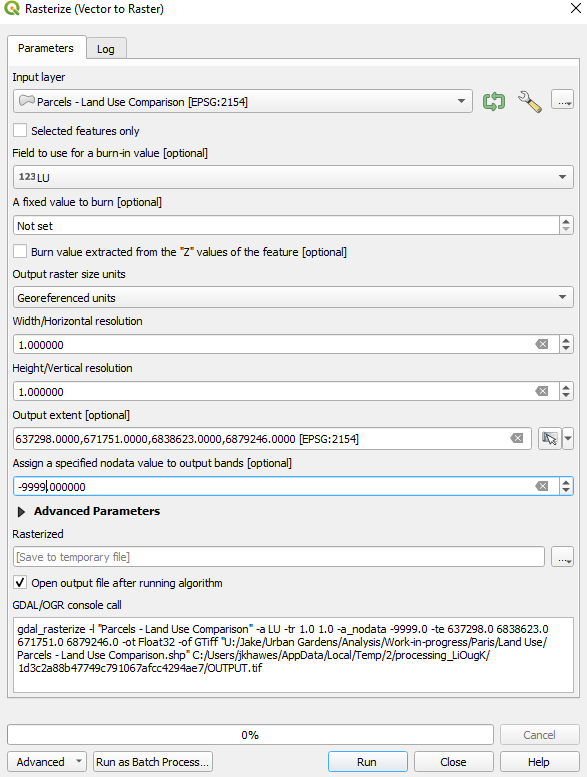
WHEN "OdSLU\_majo" = 90 THEN "OdSLU\_majo"

ELSE "BLU\_majori"

END

### Rasterize all layers and combine

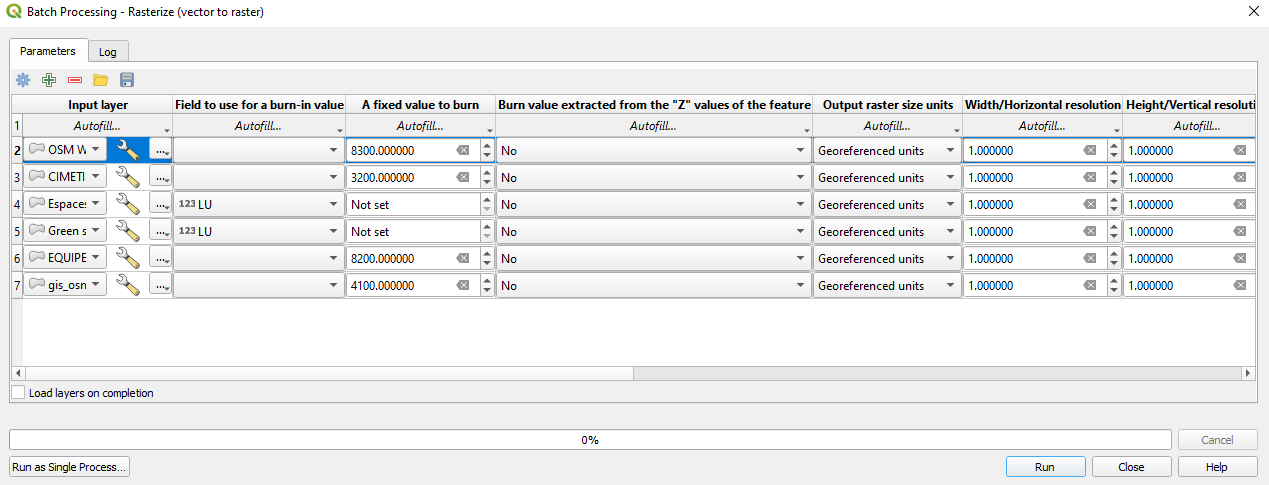
First, we can rasterize the new parcel-based LU we just derived. Burn in the new value we derived, and we align it with one of the existing layers - land cover, for example. As usual, set the no data value to -9999.

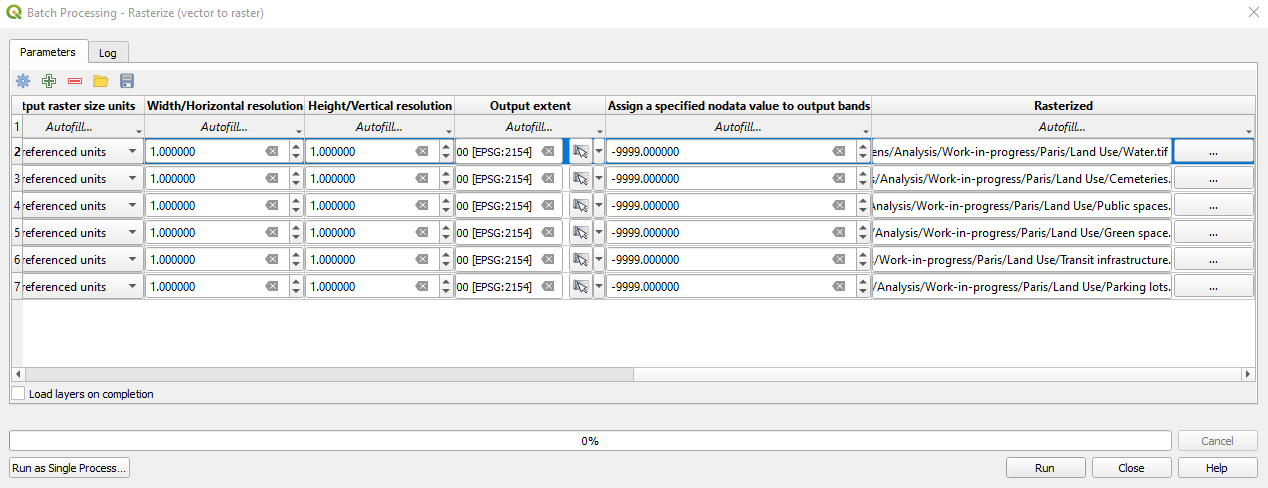


Next, you can rasterize everything else you’re interested in. The strategy here is going to be to: 1. Rasterize the overlay files and set them to a unique burn-in value; 2. Run r.null on all the new rasters to set everything outside the known areas to 0; and 3. Add the new rasters together.

The overlay layers are as follows, from highest to lowest precedence. Several of these require reclassification (and two need to be reprojected before rasterization), and the reclassification schemes are also recorded in the list and [here](https://docs.google.com/spreadsheets/d/176HUxNceFgc3QG77QbuSUSxrKWQ1HY6M/edit?usp=sharing&ouid=102554365591872343075&rtpof=true&sd=true):

1. OSM Water
   1. All water - burn-in value = 8300
2. Cimetiere - Cemeteries
   1. All cemeteries - burn-in value = 3200
3. Espaces Publics - Includes roads and plazas, among other things.
   1. Autoroutes et quasi-autoroutes - Highways and quasi-highways = 8000
   2. Dépendances des chaussées - Roadway dependencies = 8000
   3. Esplanades et places - Esplanades and squares = 3300
   4. Parkings - Parkings = 4100
   5. Piste cyclable - Cycle Path = 8100
   6. Routes et rues - Roads and streets = 8000
   7. Tramway - Tram = 8200
4. Espaces verts - Green space - *Needs to be reprojected to 2154.* 
   1. Bois - Wood = 3300
   2. Cimetières - Cemeteries = 3200
   3. Décorations sur la voie publique - Decorations on public roads = 3200
   4. Ephémères, partagés, pédagogiques - Ephemeral, shared, educational = 3100
   5. Etablissements sportifs - Sporting facilities = 2600
   6. Jardinets décoratifs - Decorative gardens = 3200
   7. Jardins privatifs - Private gardens = 2100
   8. Murs végétalisés - Green walls = 2900
   9. Promenades ouverte - Open walks = 3100
   10. Périphérique - Peripheral = 3200
   11. if( "type\_ev" = 'Bois' , 3300, if( "type\_ev" = 'Cimetières' , 3200, if( "type\_ev" = 'Décorations sur la voie publique' , 3200, if( "type\_ev" = 'Ephémères, partagés, pédagogiques' , 3100, if( "type\_ev" = 'Etablissements sportifs', 2600, if( "type\_ev" = 'Jardinets décoratifs' , 3200, if( "type\_ev" = 'Jardins privatifs' , 2100, if( "type\_ev" = 'Murs végétalisés' , 2900, if( "type\_ev" = 'Promenades ouvertes' , 3100, if( "type\_ev" = 'Périphérique' , 3200, 0))))))))))
5. EQUIPEMENT\_EMPRISE\_INFRASTRUCTURE\_TRANSPORT - Mostly rail, some misc. Buildings of interest.
   1. All rail and public transport - Burn-in value = 8200
6. OSM Traffic\_a\_free\_1 - Parking lots. - *Needs to be reprojected to 2154.* 
   1. All parking spaces - Burn-in value = 4100





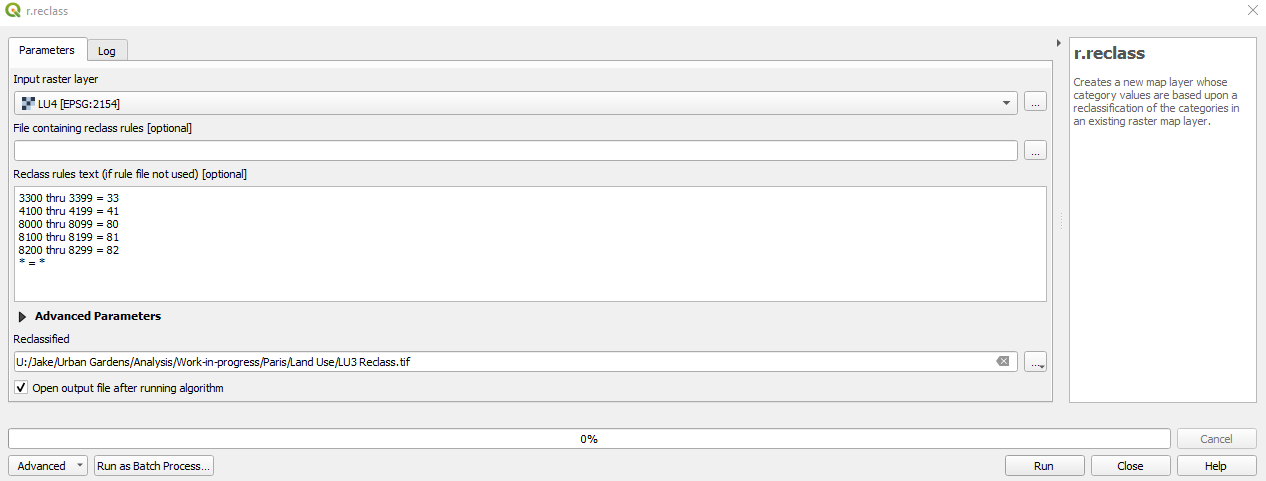
Before we do any raster calculations, we run all the new layers through r.null and set the null spaces to 0 (if you haven’t done this for the parcel LU before, do it now as well). This prevents errors in the addition.

#### Combine layers

If we wanted to develop a really complicated series of classifications, we could add these all together at once, but this is frankly just not worth it. Instead, I use the land use code we want and add two zeroes to make it distinct from the original land use code. We’ll start from the bottom, add it to the Parcel LU, and reclassify before adding the next layer up. This looks like:

1. Parcel LU + Parking spaces -> LU1
   1. 4100 thru 4199 = 41
   2. \* = \*
2. LU1 + Transit Infrastructure -> LU2
   1. 8200 thru 8299 = 82
   2. \* = \*
3. LU2 + Green Space -> LU3
   1. 2100 thru 2199 = 21
   2. 2600 thru 2699 = 26
   3. 2900 thru 2999 = 29
   4. 3100 thru 3199 = 31
   5. 3200 thru 3299 = 32
   6. 3300 thru 3399 = 33
   7. \* = \*
4. LU3 + Public Space -> LU4
   1. 3300 thru 3399 = 33
   2. 4100 thru 4199 = 41
   3. 8000 thru 8099 = 80
   4. 8100 thru 8199 = 81
   5. 8200 thru 8299 = 82
   6. \* = \*
5. LU4 + Cemeteries -> LU5
   1. 3200 thru 3299 = 32
   2. \*=\*
6. LU5 + Water -> LU\_final
   1. 8300 thru 8399 = 83
   2. \*=\*

An example of the reclass code for reference (ignore the LU3 typo in the file name):



# Sunlight Availability

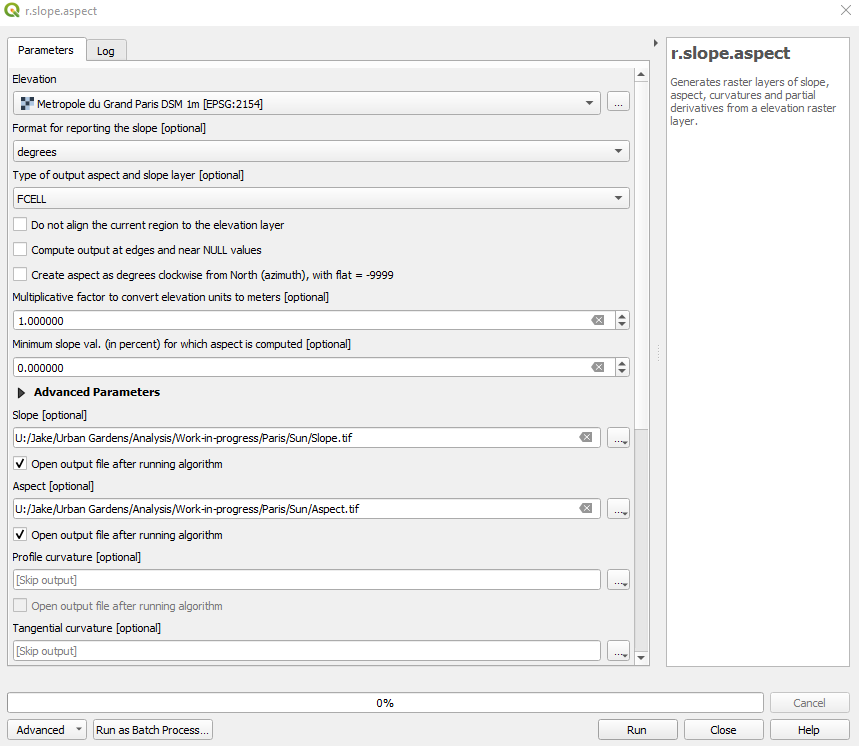
The final layer we will derive is a sunlight availability layer. This will take advantage of the r.sun package in GRASS, which takes the DSM and converts it to solar irradiance. This one is fairly complex, so I derive my process from an [example done in Canada](https://www.sciencedirect.com/science/article/pii/S0038092X10000812?via%3Dihub#fn15).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

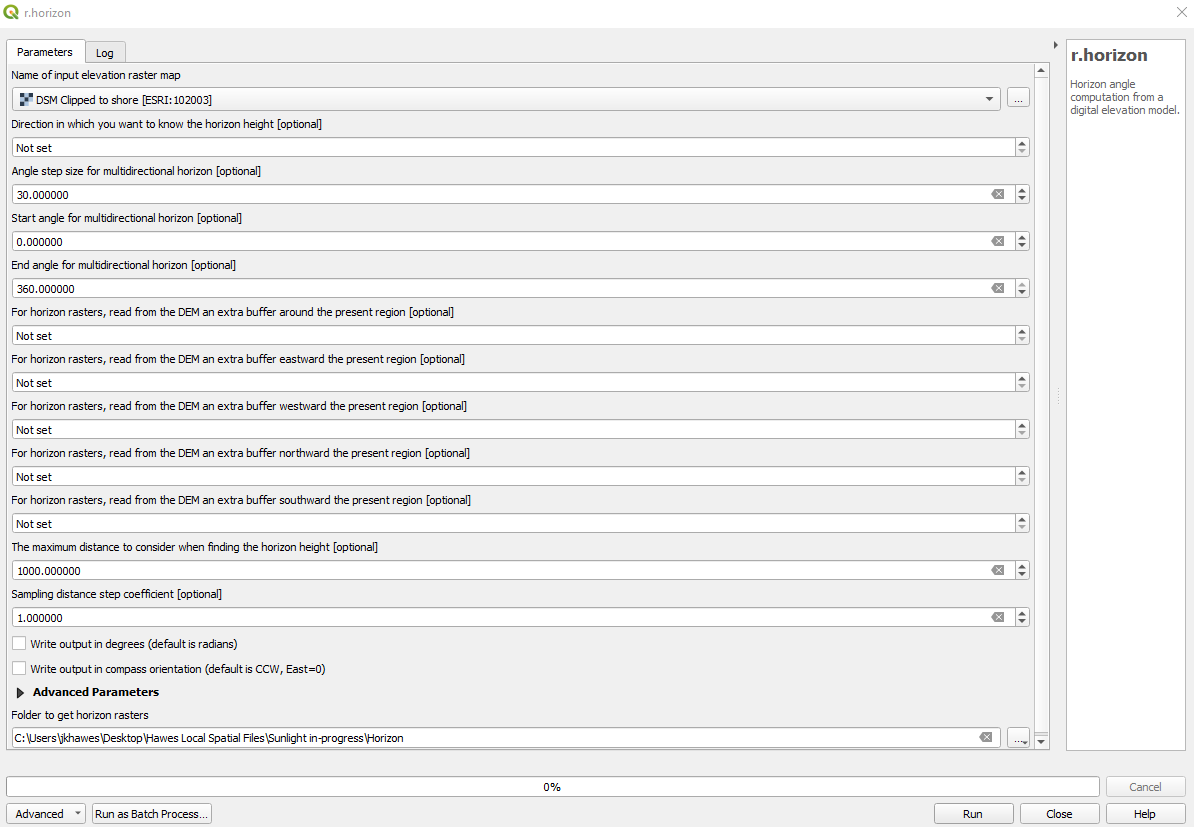
## r.slope.aspect

Assuming we already have a useable DSM, we can get started by producing 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, you want to suppress the outputs other than slope and aspect. See below for settings.

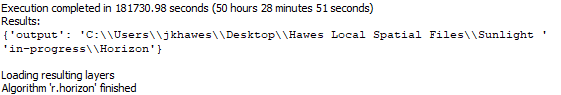


## Optional: r.horizon

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.

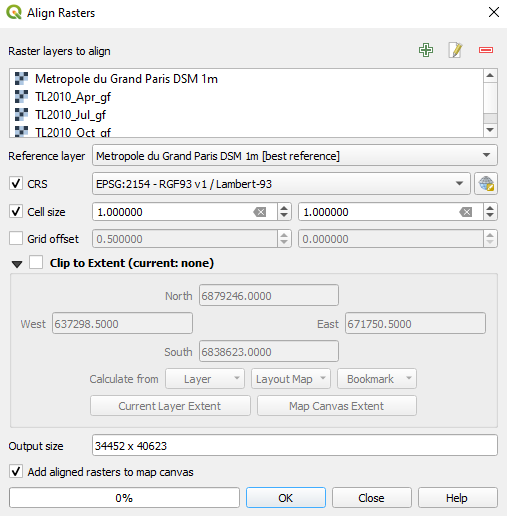


This will take a very long time to run - on the order of 50 hours on the Beast at 1m resolution. In fact, the last time I ran this, it ran for 50 hours and only saved 120-360, so you may have to run it twice to convince it to save everything correctly.



## Aligning Linke

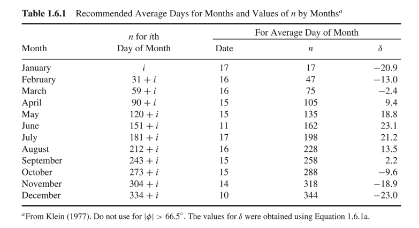
The last step before running the final command is to align the Linke values with the files we’ll be using. This will also clip the raster and will probably take 10-15 minutes.



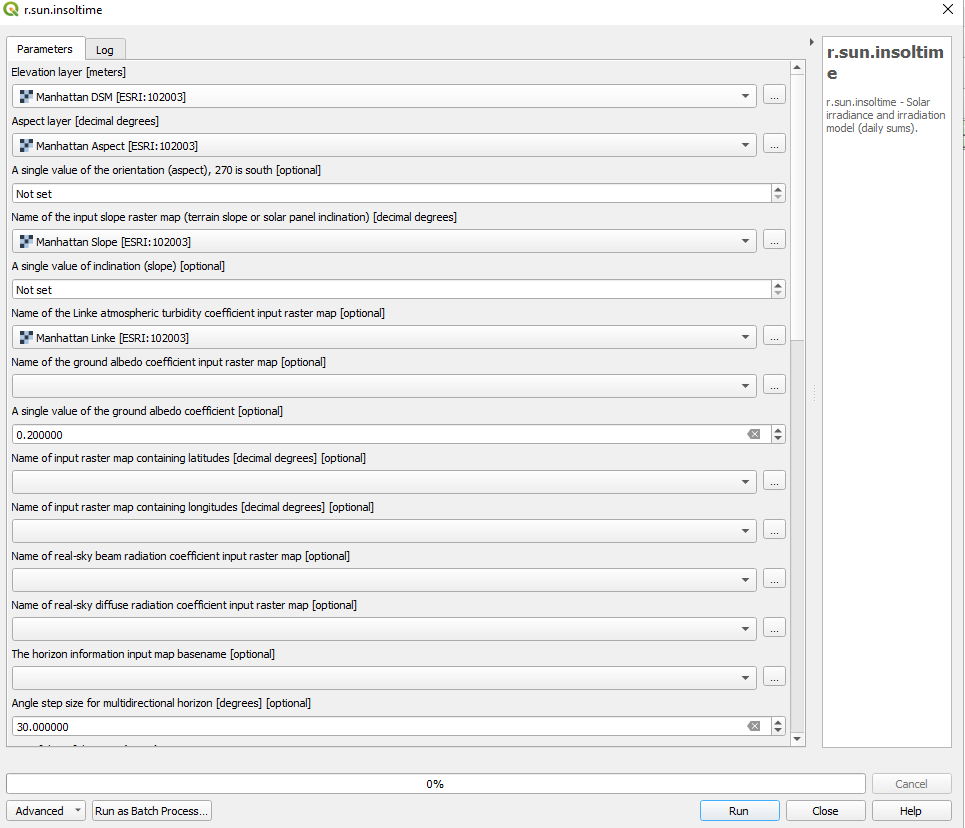
Once the Linke values have been converted to a 1x1m raster, we should be able to clip this to the metropole boundaries, as we’ve done with the others.

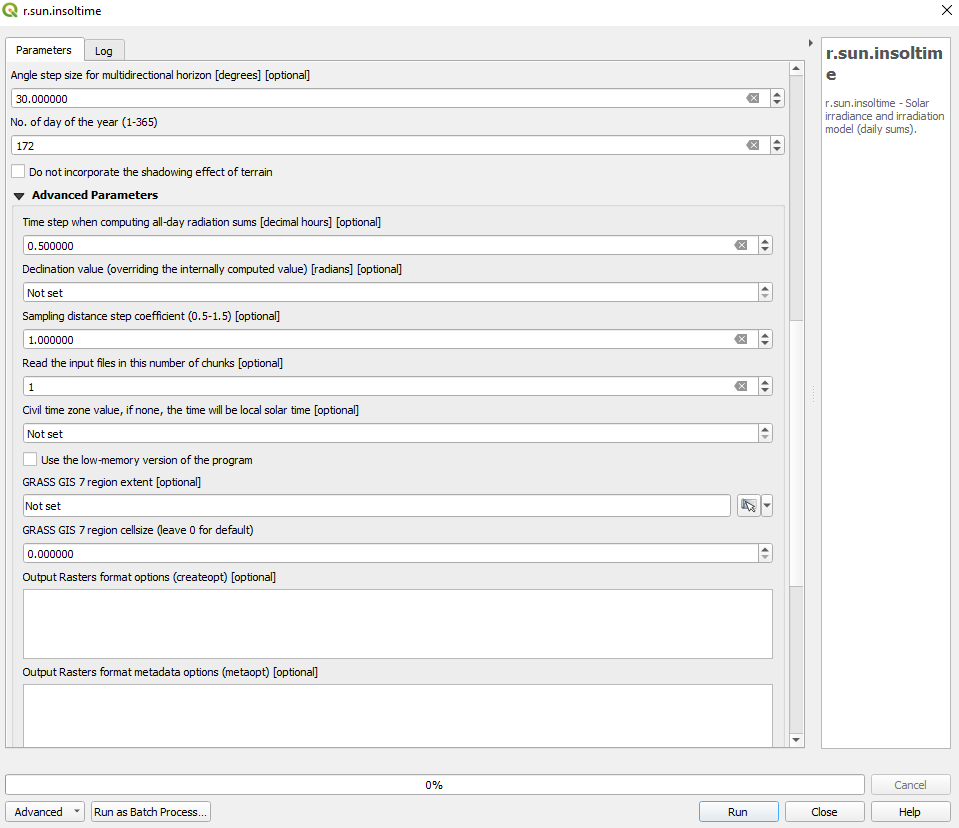
## r.sun.insoltime

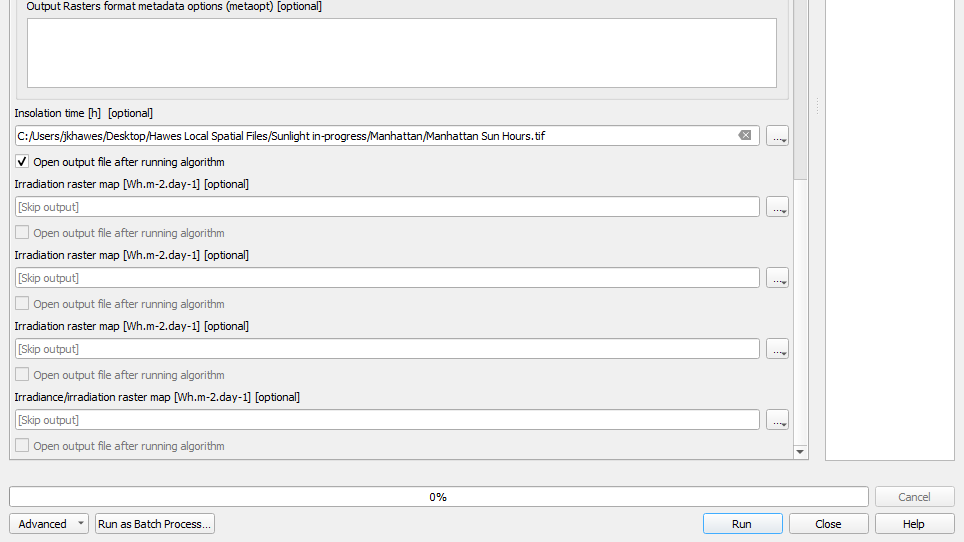
If we are strictly interested in the number of hours of sunlight, then we can simply ignore the more complex aspects like levels of radiation from the NASA data. We can retrieve average days from the book cited in the Canadian paper:



This means that the inputs look like this:







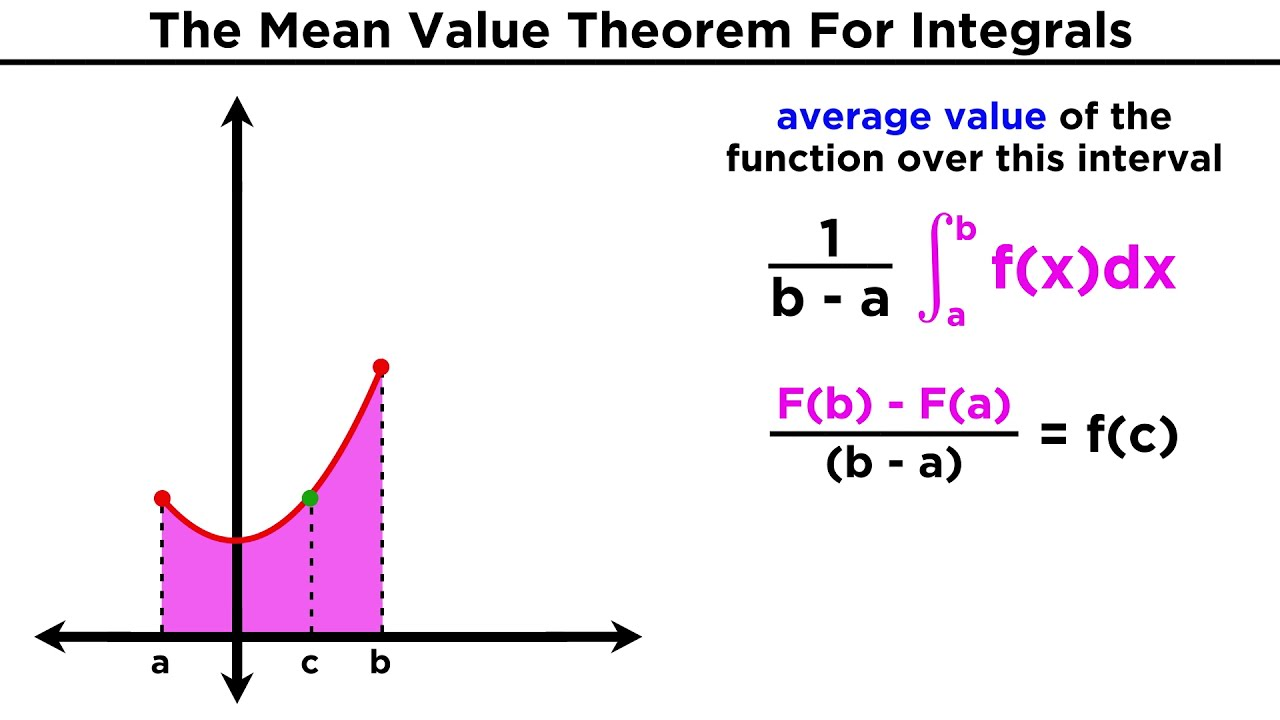
These commands run slowly, but stubbornly. In Paris, it takes about 56 hours to run from the network drive. TBD if desktop speeds this up.



## 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 Sun Hours@1" - ( 198\* ( ( "July Sun Hours@1" - "April Sun Hours@1" ) / 93 ) )

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

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

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

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