

APoLUS User guide version 2.0 (April 1st 2020)

if necessary, please cite as:

Hewitt, R.J. (2020) APoLUS User guide version 2.0 (April 1st 2020). Unpublished user documentation

1. How to obtain APOLUS latest version:

1. Download from the link at <https://simlander.wordpress.com/apolus/>

A zip file called "APOLUS.zip" will be downloaded from a google drive account.

2. Before continuing the rest of this user manual, read "APOLUS_model_full_system_documentation.pdf" which is in the folder you just downloaded. The present document just tells you what to do, without much explanation, while APOLUS_model_full_system_documentation.pdf tells you what the model was designed for and how it works.

2. Most important advances from previous versions

1. all updates are now recorded in file "readme.txt" in the main download directory of the APOLUS model, and appended at the end of the file allocator.R in the ALLOCATOR folder.

3. How to open and run a calibrated model

1. Open R. Check you have installed the gWidgetstcltk, raster and rgdal, I think the others (tcltk, sp, digest) come in automatically. Should run fine on all modern versions of R (3.0+). Tested on linux, Mac, windows. I run it in a simple command window, but it should also work on Rstudio. But if you struggle on Rstudio, I suggest to return to simple command interface.

2. copy the model root folder and all of its subdirectories to your chosen location and set the working directory to the model root. e.g. At the R command line, type something like:

```
> setwd("/home/richard/R/APoLUS9_5")
```

if you are on Windows don't forget to use "\\\" or "/" because "\\\" isn't recognised.

3. load the calibrated model. "sim01.RData".

```
> load("sim01.Rdata")
```

4. Take 2 minutes to read "sim01_explanation.doc" (it's one page, most of which is a table)
5. Scroll down to line 432 of the allocator script (ALLOCATOR/allocator.R). I refer to the lines: (eval_anim and eval(parse(text=eval_anim)). If you're on linux, install ImageMagick (great little program) comment this line back in and the animations should be generated ok in the OUTPUTS/animations directory. For windows users, just do nothing, in the OUTPUTS file I have included a windows program called Animate gif that you can use to animate the .png files manually.

6. Type:

```
> source("SWITCH/switchb.R")
```

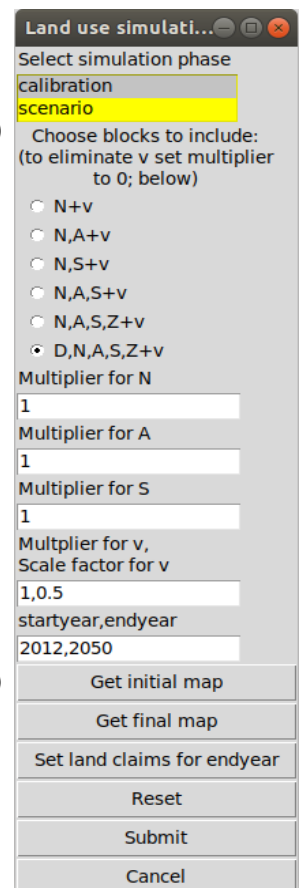
This will load the main switchboard (top right). Hit "8-Run Model", and then "Submit" in the simulation control dialog (below right). You will be given the startmap information, and you will be asked if you want to use the tendencies file (hit "y" for "yes").

```
> [1] "no new startmap chosen, will use existing map F_lu12 for start date 2012"
```

```
[1] "No new demand tables!"
```

```
Tendencies file exists, compute demands from tendencies file? (y/n)
```

The model will set land use demands according to the file in "SWITCH/tendencies.csv" and check available space to see if it's possible to simulate this amount of change up to the chosen final year (2050). It will inform us that for the moment actors are not influencing land claims (this setting is currently disactivated in the current version of APOLUS)



```
[1] "running checks on demands and available space according to tendency table"
[1] "ACTORS are not allowed to influence land claims.."
Run simulation? (y/n) n
```

(hit “n” for “no”). Then hit enter again, and, if you are on windows, hit “Cancel” at the annoying dialog box that comes up. Linux and Mac should just return the cursor.

7. At the cursor type “NewLUDTendencyTable”, and ignore the stupid warning messages.

```
> NewLUDTendencyTable
LU_cat demand_2012 landuse status      X1      X2      X3      X4      X5
3      3      7456    urb      a 7534.288 7612.576 7690.864 7769.152 7847.44
4      4      233    solar     a  279.600  326.200  372.800  419.400  466.00
5      5      116    wind      a  127.600  139.200  150.800  162.400  174.00
      X6      X7      X8      X9      X10     X11     X12     X13
3 7925.728 8004.016 8082.304 8160.592 8238.88  8317.168 8395.456 8473.744
4  512.600  559.200  605.800  652.400  699.00  745.600  792.200  838.800
5  185.600  197.200  208.800  220.400  232.00  243.600  255.200  266.800
      X14     X15     X16     X17     X18     X19     X20     X21
3 8552.032 8630.32 8708.608 8786.896 8865.184 8943.472 9021.76 9100.048
4  885.400  932.00  978.600 1025.200 1071.800 1118.400 1165.00 1211.600
5  278.400  290.00  301.600  313.200  324.800  336.400  348.00  359.600
      X22     X23     X24     X25     X26     X27     X28     X29     X30
3 9178.336 9256.624 9334.912 9413.2  9491.488 9569.776 9648.064 9726.352 9804.64
4 1258.200 1304.800 1351.400 1398.0 1444.600 1491.200 1537.800 1584.400 1631.00
5  371.200  382.800  394.400  406.0  417.600  429.200  440.800  452.400  464.00
      X31     X32     X33     X34     X35     X36     X37     X38
3 9882.928 9961.216 10039.5 10117.79 10196.08 10274.37 10352.66 10430.94
4 1677.600 1724.200 1770.8  1817.40  1864.00  1910.60  1957.20  2003.80
5  475.600  487.200  498.8   510.40  522.00  533.60  545.20  556.80
```

The numbers show the results of the calculation done with the information fed into the model from the “SWITCH/tendencies.csv”.

	A	B	C	D	E	F	G	H	I	J
1	LU_cat	demand_2012	landuse	status	growthrate	multiplier	tendency			
2	3	7456	urb	a	1.05	1				
3	4	233	solar	a	20	1				
4	5	116	wind	a	10	1				
5										

At the moment, the model is set up to grow the three active land uses each year by using the number in the “growthrate” field to calculate a percentage of the demand at the starting time (2012 here) and then add that number to the total for each year of the simulation. So, $(1.05/100 * 7456) = 7534.288$ which is the value you will see in column X1 of NewLUDTendencyTable. So you now understand NewLUDTendencyTable.

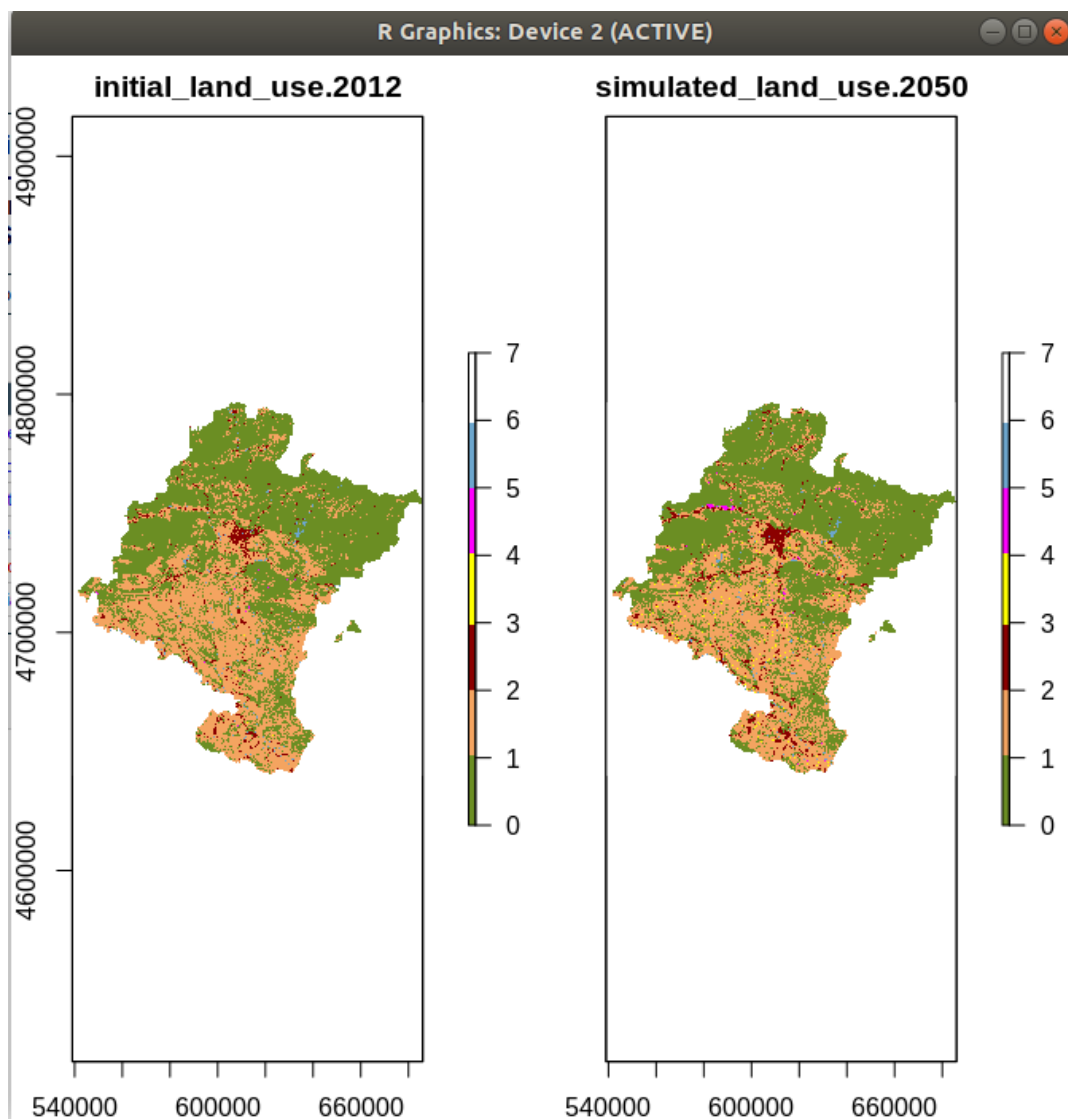
If you want to see the code where this is calculated, open “SWITCH/demandtendency.R” and look at lines 48 to 68. You can see what will happen if you change the “l” in the tendency column in SWITCH/tendencies.csv” to “e”. Of course, you will need to change the values of “growthrate”, too, or you won’t be able to the run the model (an exponential with these values for growthrate will try to allocate way too many cells that there is space for in the model).

8. Now go back to step 6 and hit “y” for “yes” at the Run simulation request.

```
> "Run simulation? (y/n)y
```

The model will run (see example terminal output below) and plot the two maps (see example plot below), unless you are on a headless machine with an X server, in which case the plots may not work. It isn't important anyway, it's mainly a way of showing that the simulation has run correctly.

```
Terminal
File Edit View Search Terminal Help
[1] "warning - zero values, map 'unallocated' generated, unassigned cells added to category 1"
[1] "this is usually because some function classes have changed location"
[1] "simulation finished, now creating new individual land use maps from Tn for next run"
[1] "running checks on demands and available space"
[1] "Total available space in the model is 245001"
[1] "FINISHED: landuse simulation for 2049"
===== | 97%[1] "entered simulation loop.."
[1] "created demand matrix 'rdemand' from tendency table"
[1] "created and stacked TP for urb"
[1] "created demand matrix 'rdemand' from tendency table"
[1] "created and stacked TP for solar"
[1] "created demand matrix 'rdemand' from tendency table"
[1] "created and stacked TP for wind"
[1] "finished random maps .."
[1] "created TTP for urb"
[1] "created TTP for solar"
[1] "created TTP for wind"
[1] "transition potentials computed and stacked..."
[1] "line 254 28"
[1] "running Linux 64bit R.."
[1] "line 305 28"
[1] "warning - zero values, map 'unallocated' generated, unassigned cells added to category 1"
[1] "this is usually because some function classes have changed location"
[1] "simulation finished, now creating new individual land use maps from Tn for next run"
[1] "running checks on demands and available space"
[1] "Total available space in the model is 244864"
[1] "FINISHED: landuse simulation for 2050"
===== | 100%
NULL
> |
```



9. Check that the active land uses (red highlighted) have been correctly allocated in the final land use map and compare with the last column of NewLUDTendencyTable. The final land use map is called "Tn".

```
> freq(Tn)
      value count
[1,]      1 149088
[2,]      2  95776
[3,]      3  10431
[4,]      4   2004
[5,]      5    557
[6,]      6   1958
[7,]      7 319366
```

(copied from above)

```
X38
3    10430.94
4     2003.80
5      556.80
```

If the last column of NewLUDTendencyTable and the values given by freq(Tn) agree for the 3 active land uses, the model has run correctly. This doesn't mean it's a good model, and indeed it may seem quite silly. To improve it, you will need to play with calibration settings, neighbourhood, accessibility, suitability, actor dynamics or zoning (see "APOLUS_model_full_system_documentation.pdf").

Note that if you run the model again you get the same basic pattern, but very slightly different results because of the random factor. This is intentional – you can run 10 or more simulations with the same settings, and arrive at a probability estimate of where land use change will most likely go according to your calibrated model. This is much more reliable than a single snapshot of 2050.

4. How to build and calibrate a new model from scratch.

1. Open R. Check you have installed the gWidgetstcltk, raster and rgdal, I think the others (tcltk, sp, digest) come in automatically. Should run fine on all modern versions of R (3.0+). Tested on linux, Mac, windows. I run it in a simple command window, but it should also work on Rstudio. But if you struggle on Rstudio, I suggest to return to simple command interface.

2. copy the model root folder and all of its subdirectories to your chosen location and set the working directory to the model root. e.g. At the R command line, type something like:

```
> setwd("/home/richard/R/APoLUS9_5")
```

if you are on Windows don't forget to use "\\\" or "/" because "\\" isn't recognised.

3. Type:

```
> source("SWITCH/switchb.R")
```

This will load the main switchboard (top of this document).

Inputting data and calibrating the model

The switchboard is designed to help you through the calibration process. You need to carry out each numbered task in the order it appears. The model directories already contain example data in the folder "APOLUS[version]/ASCII". It is suggested that the relevant data for each part of the model should be stored inside ASCII in the appropriate subfolders. Take the time to replace all the test data in these folders with your own data, ensuring that x and y corners, ncols, nrows and resolution are all the same for all maps. The model was originally designed to work with ESRI ascii raster data, with file extension .asc, but now uses Geotiff files instead, for which you need to have the rgdal library installed on your system. If you prefer to work with .asc files (or any other type of GIS format file accepted by rgdal), you will need to edit the pattern search commands, and the "raster" and WriteRaster" commands in the various scripts, e.g. ACCESSIBILITY/access.R, ALLOCATOR/allocator.R. .asc format is handy just in case you can't install the gdal libraries on your system – typically if you are working on a remote server and the sysadmin won't install things. If you are happy to use only .asc files, then you don't need rgdal, and can work with just raster. Of course, some scripts may "require" rgdal so you'll also need to comment this out.

Import region maps

This dialog allows the user to import region maps (e.g. from the ASCII folder, see above). For a regional study, the national and regional maps will be the same (the regional map – if no other regions are studied, national influence will be at the level of the region). The names are given for convenience only, and relate simply to three levels of hierarchical administrative boundaries – national could in fact be a map of the world, regional could be a continent, e.g. Europe, and local could be a country.

Once loaded the maps are stored in R under the names “national”, “regional” and “localm”

Import land use maps

Before running this dialog, check that the land use maps have the same number of categories and that there are no “NA” values. This saves substantial headache. Before running the model, NA values should be assigned a category like the other uses, e.g., in a map with 7 land uses, the area outside the region could be number 8. Also before running this dialog, set the desired color palette and correct number of breaks for the number of land use categories. The color scheme you choose here will be used to display the results of the model. You can set this in the script START/startlu.R, on lines 61 and 62. You need to set one more breakpoint and color than there are categories in your model, e.g. if you have 7 categories, you need 8 colors and 8 breakpoints. Breakpoints need to start at 0, and if your last category is outside the study area, it is recommended that your last colour should be very innocuous, e.g. white. Type “colors()” at the command prompt for a list of colors supported in R.

When you run this dialogue you will be asked to enter the start year and the end year for the simulation, and then you will enter the map for each date. For example, if you want to calibrate from 2000 to 2010, start year will be 2000, and for land use map 1 you will select your map “lu2000.asc” or something similar from the ASCII directories. endyear will be 2010 and you will select the corresponding land use map for land use map 2. The start map and the end map will be plotted. If the colours and breakpoints you set didn't come out right you don't need to run the import tool again, just paste lines 61-66 of the START/startlu.R script into the command window and modify them until the maps look right.

The first time you run the model, its a good idea to choose the interactive option at the prompt “Land use categories and status: type ‘t’ to import ‘landuse.csv’ or ‘i’ for interactive mode ...” You will be prompted to name the uses, say whether they are active or static classes, and whether they respond to actor dynamics. Follow the command line instructions and input these data carefully, as going back can be a headache. You will be prompted for all the categories, and the dialog will close once all the categories have been named and assigned a status. At this point OUTPUTS/landuse.csv will be written (or overwritten – there is an example already there) and your new project will be saved as “project.Rdata”. If you crash out after this point you can restore your work without having to go through the arduous land use dialogue just by setting the directory to the model root and doing >load(“project.Rdata”). Likewise, if you need to modify your land use settings, do it directly in OUTPUTS/landuse.csv and then start the land use import dialog again, this time choosing “t” to import the modified landuse.csv file.

Set actors' characteristics

If this is the first time you have calibrated a cellular automata land use model, and you don't have much interest in replicating the behaviour of different land use actors on land use allocation, **you can skip this bit completely, just remember to select the N,A,S,Z+v radio button in the simcontrol dialog (page 1 of this guide) and not D,N,A,S,Z+v.**

But if what you really care about are the way different actors may compete to facilitate or block the allocation of certain land uses, read on....

When you hit the button for “3. Set actors' characteristics”, your will be asked:

List of actors: type ‘t’ to import ‘actlist.csv’ or ‘i’ for interactive mode:
you can create your own actlist.csv (following the OUTPUTS/EXAMPLE_actlist.csv template) inside the OUTPUTS folder and import it, or you can enter the actors individually at the command line. Since actors must be associated with a region map, the script will check if step 1 “import region maps” has been undertaken successfully. If not, you won't be able to continue. Just go back and do step 1 first, and then try again. You will then be asked to rate the actors characteristics respect to the implementation of *the land uses you have associated with the actors* in the import land use dialog. These values will be used to create actor suitability maps that will affect the spatial location of the associated uses. This is very useful if you are working with several municipalities or regions, since the regions whose actors are most strongly in favour will

get the new land uses in preference to the others¹

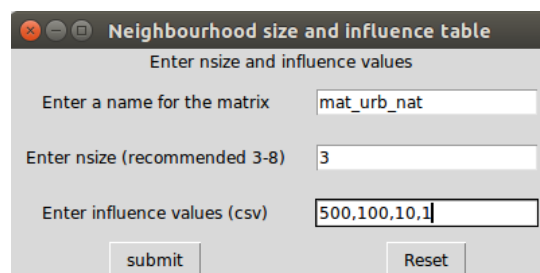
Once the actor characteristics have been set and the transition potential map generated (actormapsTTPs), the values selected (stored as a table called "actorchars" in the workspace) will be written to OUTPUTS/actorchars.csv. The values assigned to these characteristics assigned (stored as a table called "actorcharsvals" in the workspace) will be written to OUTPUTS/actorcharvals.csv.

```
print(actorcharvals)
write.csv(actorcharvals, file = "OUTPUTS/actorcharvals.csv")
```

How to calibrate the neighbourhood effect

To modify the cell neighbourhood attraction and repulsion effects, click on "4.Calibrate Neighbourhood". This script works fine but doesn't exit cleanly, so when you finish, just hit enter in the R window and "cancel" if you are given an exit dialog.

Choose the land use_to_land use matrix you want to modify from the list, and hit "e" to change the matrix. In the neighbourhood dialogue (below), choose the neighbourhood size you want (2 or 3 is a good starting point), and enter the values separated by commas for the influence at each cell distance. There must be one more influence value than nsize, since the first value is for distance 0 (on the cell itself). At left is an example rule for the attractiveness of urban land to natural land.



The first land use listed after the word "mat" is the active use; i.e. the one that will occupy the other. In the example model, only urb, solar and wind are active uses, so don't modify any rules that don't begin "mat_urb", "mat_solar" or "mat_wind", or whatever your active uses are called. At some point the selectmatlist script needs to be changed so the matrices for these other uses don't appear in the list.

Once you've finished modifying a neighbourhood rule, hit "q" in the on-screen dialog and the model will be saved in its current state, incorporating the new rule, in the file "project_nrules.Rdata". Also, a graph displaying the new rules will be written to the NHOOD directory as a .png file.

To see the result of the rule change, run the simulation. At present, solar is attracted to urban land, as can be seen in the simulation results. All model results are output to the OUPUTS file. The images of each simulation are stored in the OUTPUTS/animation directory.

To experiment with allocating lots of land use for fun, change the annual demands² (e.g. andem3 = annual demand for land use 3). Only change land active land use demands - 3,4, and 5 (urban, solar, wind). e.g.

```
> andem3 <- 350 #will allocate 350 extra cells of land every year for land use 3 (urban).
```

If you want to know the current demand settings, just call up the land use table

```
> tablu1
```

at each step of the model, the number of cells corresponding to each land use in the first map (2007 in this case) plus the annual increment will be allocated. e.g. demand3 + andem3 for land use 3. The annual demand is simply the difference between the number of cells in map1 and map 2 divided by the number of years. To simulate future change, "demand" for each use is set to the number of cells for that use last map (2012), and annual demand for each successive year is computed according to values chose by the user.

Calibrate accessibility

1. To calibrate accessibility, you need to create a map in .asc raster file format of euclidean (straight line) distances from features for each accessibility type, e.g. roads, rail, electric lines. This is easy to do in GIS. There are examples in the relevant subfolders of the ASCII directory. All the ascii distance maps should be in the same directory. When the calibrate accessibility button is clicked, you will be prompted for the location containing these maps, which will be imported to the workspace. You will be prompted to set the accessibility for each accessibility type to each land use. If accessibility is not required for a land use, just hit "n" for "no".

¹ Of course, if you are working with only one region, even if all actors are against implementation of these uses, the transition potentials will be very low, but those uses will still be implemented as long as there is existing demand in the model. This is not very realistic of course, so we need to use these results to modify the demand too. [This wouldn't work for the calibration phase of course...]

² This is relevant for manually allocating land claims in a future simulation, i.e. for calibration. If you are using the file SWITCH/tendencies.csv to plot final land use demand according to a tendency curve (see Section 3, step 6, above) you can ignore andem1, andem2, andem3 etc and demand1, demand2, demand3 etc.

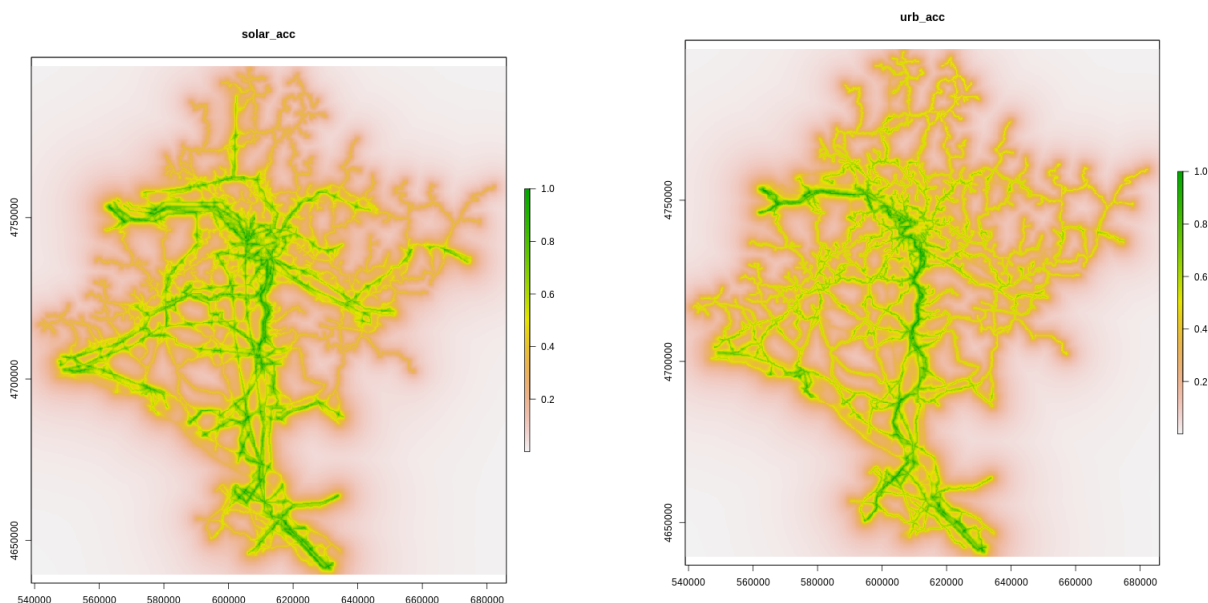
If you select “n” for no, this land use map will be skipped, and accessibility will be assigned to value 1 everywhere.

```
> [1] "3 layers imported: "
set accessibility for land use urb y/n?
```

2. You will be prompted to set the weight of each weight of each accessibility type to that particular land use class.

```
>input weight (any number) for dist_elec relative to other maps (999 to quit): 1
[1] "success"
```

For example, for wind turbines, rail networks might be significant (e.g. for transport of parts), but roads are probably more important for servicing and general access, while the electric grid is of fundamental importance since it is unlikely that new turbines will be installed far from an existing supply line. So rail might be given weight 1, road given weight 5, and electric given weight 10. (see below – highest accessibility values in green, lower in brown, lowest in white). Roads are yellow in the right-hand figure (urban), as opposed to brown in the left-hand figure (solar) indicating greater relative importance of accessibility by road for urban development than for solar development, where the electric network is considered more important (green on left hand figure) for solar development than for for urban development. The area that is green in both figures is the rail network, which has been given a high weight for both urban and solar development.



3. Then you will be asked to input the **coefficient**.

```
> input coefficient (any number) for dist_elec for land use urb (999 to quit):
```

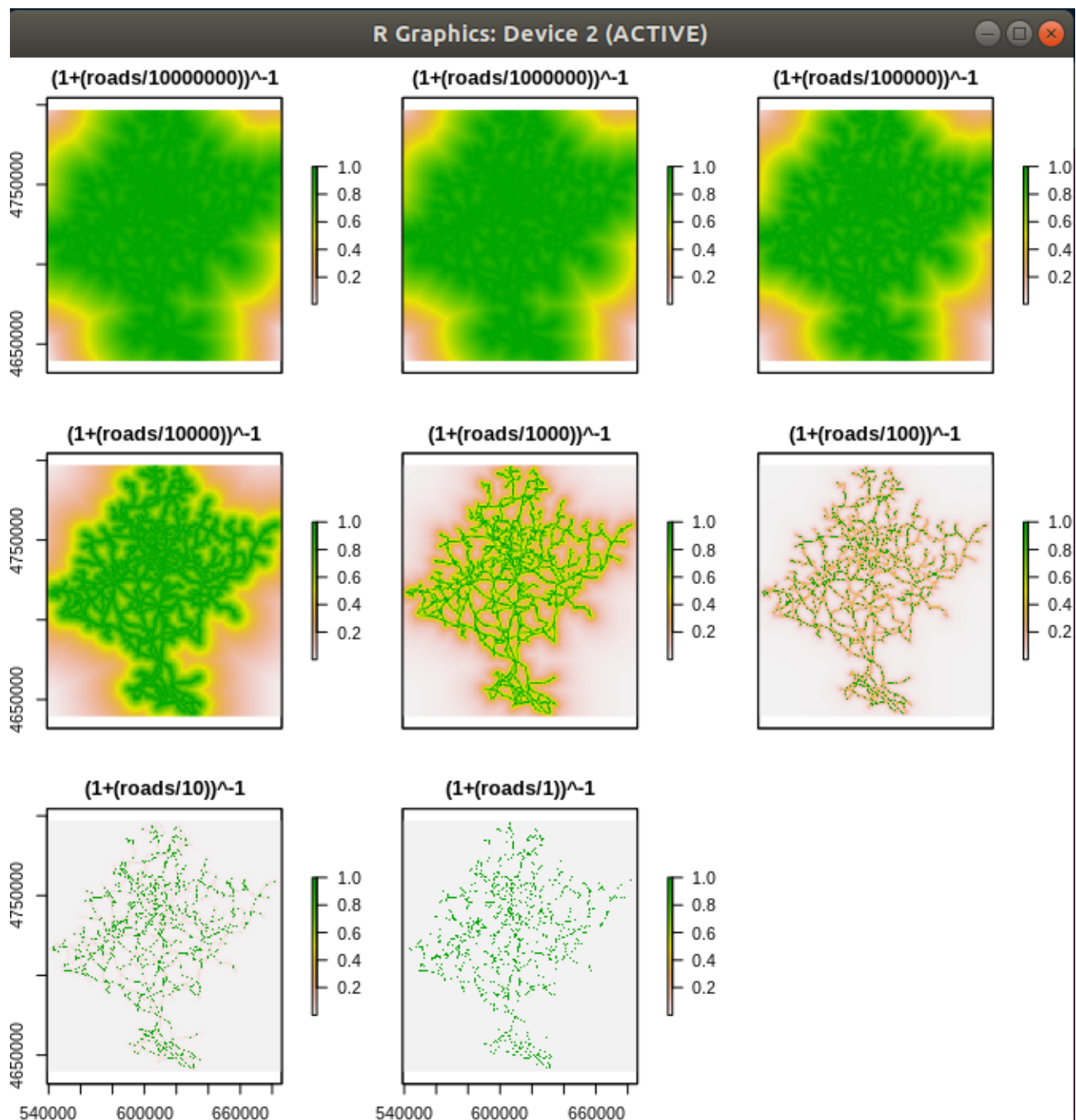
This relates to White et al's (1997) local accessibility equation, which determines the shape of the curve:

$$a_j \left(1 + \frac{D}{\delta_j} \right)^{-1} \quad (\text{Eq. 2})$$

(from White et al 1997)

where D is the euclidean distance from the cell to the nearest cell in the network, and δ_j is the coefficient expressing the importance of accessibility for the desirability of the cell for land activity j; a_j is then inserted as a coefficient in equation 2 (see above).

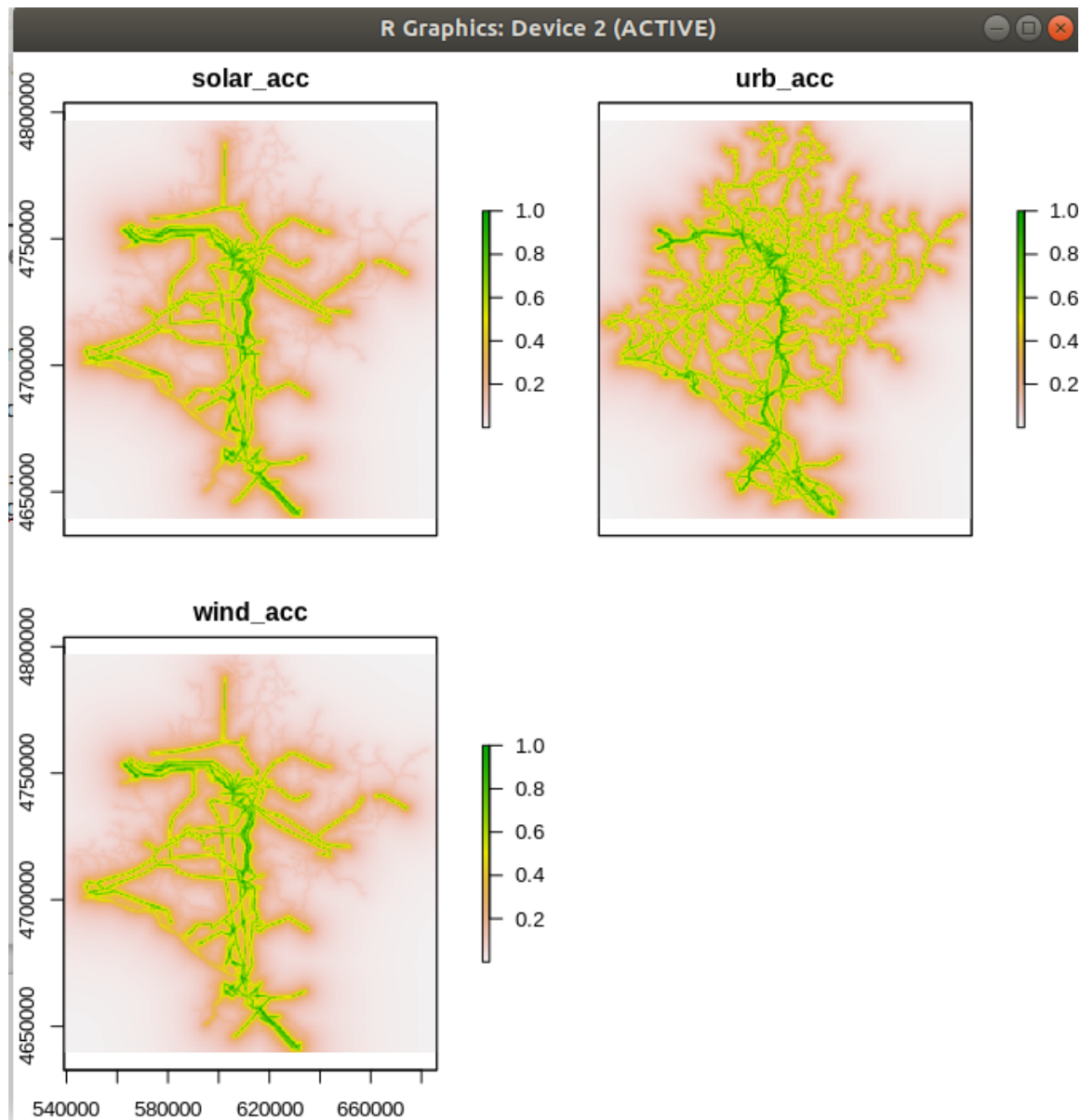
The higher the value of the coefficient that you input here, the weaker the accessibility effect (below).



4. Once you have set weights and coefficients for all maps, total accessibility will be calculated for each land use and stored as a map named `[land use]_acc`.

You can find them and plot them in the workspace using a pattern search command:

```
> nmaps <- ls(pattern="acc$")
> nmapss <- stack(mget(as.character(nmaps)))
> plot(nmapss)
```

5. Finally, check out the information tables, which show the accessibility values you have assigned for each land use. If no accessibility has been assigned for a given land use, the table will exist, but have value “1”

```
> ls(pattern="accinftab$")
[1] "solar_accinftab" "urb_accinftab"   "wind_accinftab"
> solar_accinftab
> solar_accinftab
```

	Land_use_map	distance_map	coefficient	Min	Max	weight
1	solar	Aimap_dist_elec	100000	0	64051.54	3
2	solar	Aimap_dist_rail	100000	0	93048.38	1
3	solar	Aimap_dist_road	100000	0	54011.85	2

```
Accessibility_map
1      solar_acc
2      solar_acc
3      solar_acc
```

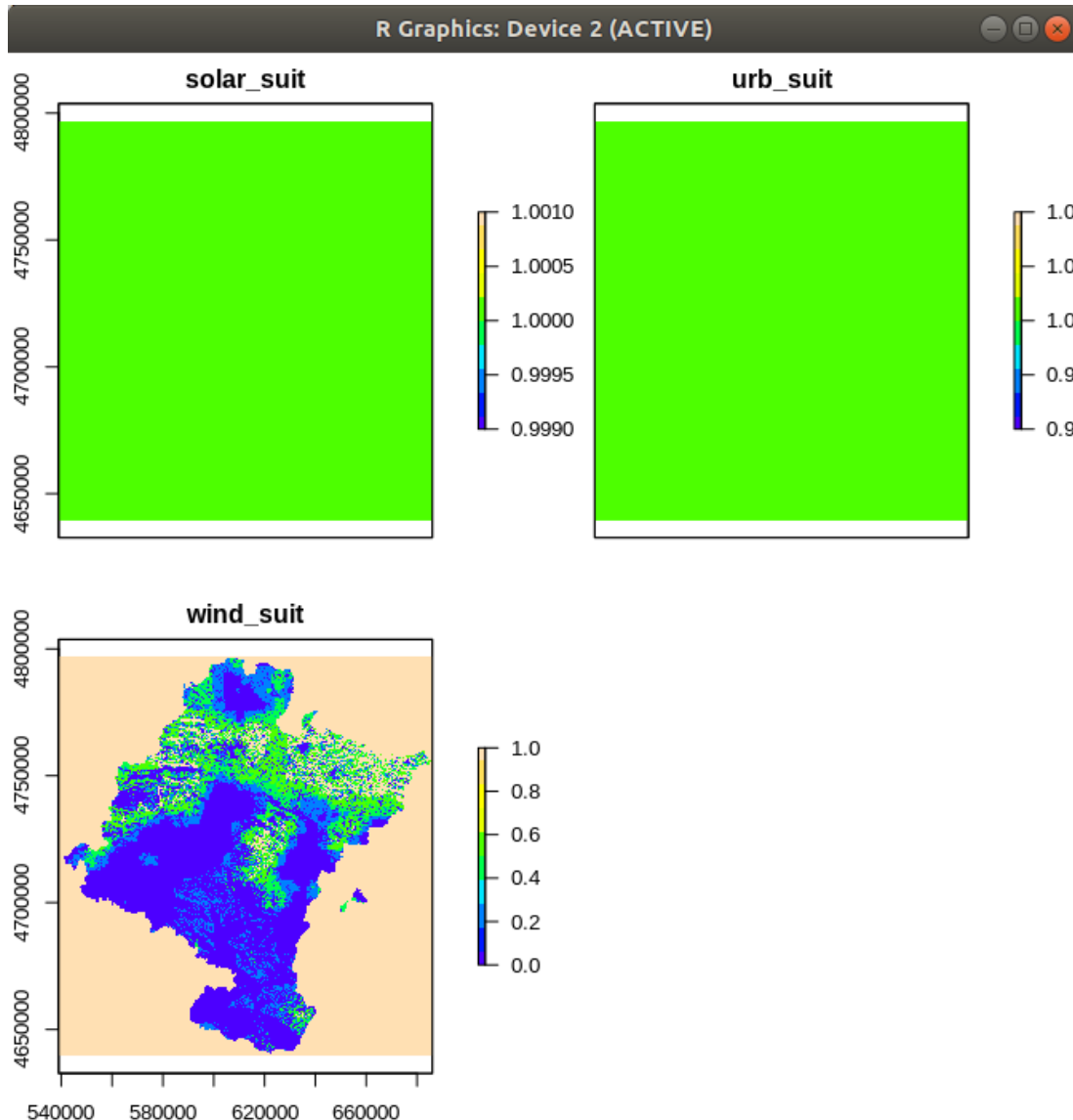
Calibrate suitability

1. When the calibrate suitability button is clicked, all raster .tif format files in the ASCII/SUITABILITY folder

will be automatically imported, just as for the calibrate accessibility tool (above). This folder should contain only suitability base maps in tif format – i.e. rasters containing raw, unclassified data values for the desired suitability variable e.g. elevation, slope. Note that, if desired, preclassified maps can also be used. E.g for slope, either the actual slope map in degrees or % (e.g. min 0, max 34.5 degrees) or the slope map reclassified into categories (e.g. min 1, max 5, where 1 is least sloping and 5 is most sloping) are both acceptable – in either case the user will be asked to assign suitability values to the relevant categories in the map.

```
> [1] "4 layers imported: "
set suitability for land use urb y/n?
```

2. If you select “n” for no, this suitability map will be skipped, and suitability will be assigned to value 1 everywhere. See below, where only land use wind has been assigned suitability values.



If “y” for “yes” is selected, you will then be prompted to input suitability for each category in the input map:

```
[1] "input suitability influence scores for wind to suitability basemap slope200
or type 999 to quit"
[1] "slope200: min;max = 1;6"
input format is low,high,value,low,high,value etc: -
Inf,2,0,2,3,0.1,3,4,0.2,4,5,0.3,5,6,0.5,6,7,0.7,7,8,0.8,8,9,0.9,9,Inf,1
```

3. At this prompt, enter the breakpoints and influence values for the land use class chosen for each suitability map in the form startval,endval, influence value. There is no need to use values higher than one, but no

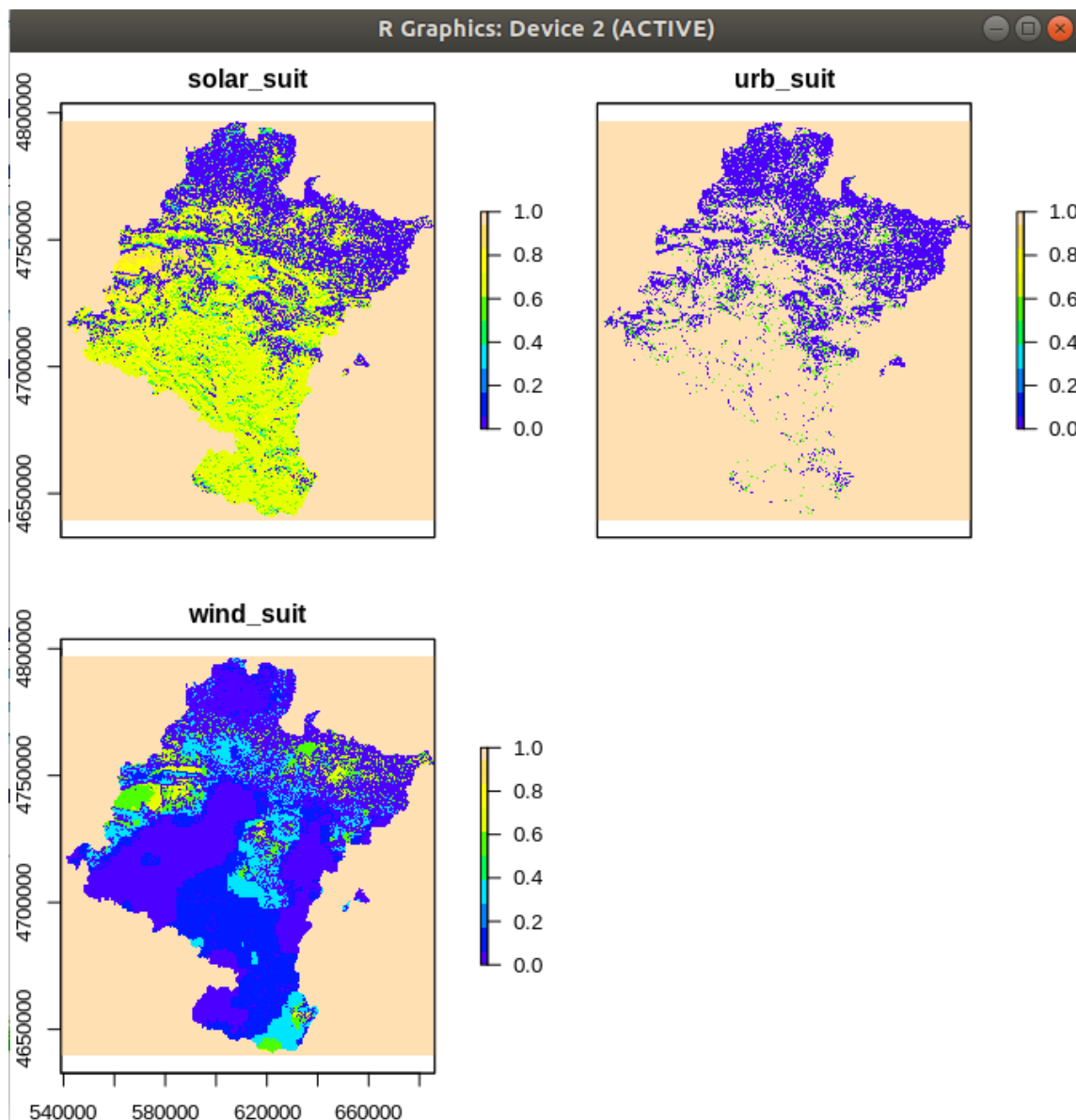
harm in doing so either, since all values are unity normalized later.

remember to use `-Inf` and `Inf` to ensure that you always include the lower and upper bounds.

If a suitability map is not important or not significant for a particular land use class, just reclassify to `-Inf,Inf,1` which will ensure that it is not taken into account in the final suitability computation for that land use class.

4. To view the final maps and check that nothing went wrong, it's a good idea to search for the maps and plot them together as a stack (or individually - not all graphical display configurations correctly plot stacks). See below.

```
> nmaps <- ls(pattern="suit$")
> nmapss <- stack(mget(as.character(nmaps)))
> plot(nmapss, col = topo.colors(10))
```



5. Finally, as for accessibility, check out the suitability information tables:

```
> ls(pattern="suitinftab$")
[1] "solar_suitinftab" "urb_suitinftab"  "wind_suitinftab"
> solar_suitinftab
  Land_use_map      base_map
```

```

1      solar Simap_elev500
2      solar Simap_slope200
3      solar Simap_solar200
4      solar Simap_wind200

reclass_vals
1      -Inf,Inf,1
2      -Inf,4,1,4,5,0.5,5,Inf,0
3 -Inf,2,0,2,3,0.1,3,4,0.2,4,5,0.3,5,6,0.5,6,7,0.7,7,8,0.8,8,9,0.9,9,Inf,1
4      -Inf,Inf,1

      Min      Max suitability_map
1 500.0011 2310.292      solar_suit
2  1.0000    6.000      solar_suit
3  1.0000   10.000      solar_suit
4  1.0000    8.000      solar_suit

```

Calibrate zoning

1. The zoning calibration procedure is virtually identical to that described for accessibility and suitability, except that the procedure is simpler. It is only necessary to specify the level of restriction appropriate to each map and whether to include the map or not (1 or 0) for the given land use. Strictly restricted areas will not be changed, Weakly restricted areas have lower probability of change (reduced by a factor of 10), and Permitted areas will be no different than areas with no zoning assigned.

```

> [1] "2 layers imported: "
set zoning for land use urb y/n? y
[1] "input zoning scores for urb to zoning basemap eepp_200 or type 999 to quit"
[1] "eepp_200: min;max = 1;78"
[1] "Use 1 for permitted, 0.1 for weakly restricted, 0 for strictly restricted:

```

2. This requires the user to input a simple reclassification series, as above. There are only 3 options.

-Inf,Inf,1 - permitted

-Inf,Inf,0.1 - weakly restricted

-Inf,Inf,0 - strictly restricted

e.g.

```

[1] "input zoning scores for urb to zoning basemap eepp_200 or type 999 to quit"
[1] "eepp_200: min;max = 1;78"
[1] "Use 1 for permitted, 0.1 for weakly restricted, 0 for strictly restricted:
"
input format is low,high,value,low,high,value etc: -Inf,Inf,0
[1] "success"
[1] "input zoning scores for urb to zoning basemap north_navarre or type 999 to quit"
[1] "north_navarre: min;max = 1;1"
[1] "Use 1 for permitted, 0.1 for weakly restricted, 0 for strictly restricted:
"
input format is low,high,value,low,high,value etc: -Inf,Inf,1
[1] "success"

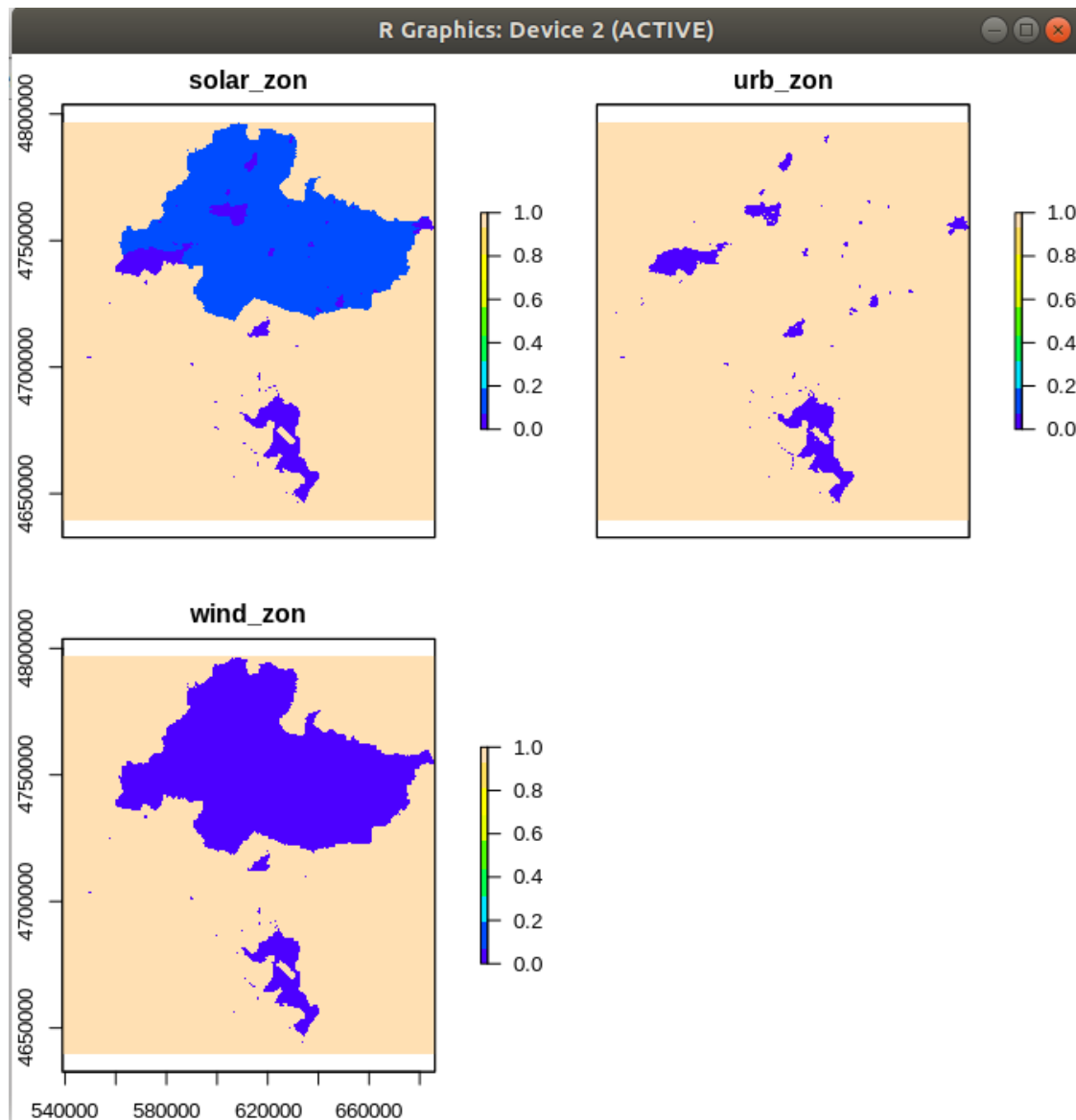
```

3. As with accessibility and suitability maps, it's a good idea to stack and plot the maps:

```

> nmaps <- ls(pattern="zon$")
> nmaps
[1] "solar_zon" "urb_zon" "wind_zon"
> nmapss <- stack(mget(as.character(nmaps)))
> plot(nmapss, col = topo.colors(9))

```



4. And examine the zoning information tables:

```
> ls(pattern="zoninftab$")
[1] "solar_zoninftab" "urb_zoninftab" "wind_zoninftab"
> solar_zoninftab
  Land_use_map      base_map reclass_vals Min Max zoning_map
1      solar      Zimap_eepp_200  -Inf,Inf,0  1  78 solar_zon
2      solar Zimap_north_navarre -Inf,Inf,0.1  1   1 solar_zon
>
```

Preparing to run the model

Once all parameters have been set the model can be run. First, however, it is strongly recommended to run the cleaner script³ to remove temporary files created during the calibration phase, e.g.

```
> source("ALLOCATOR/cleaner.R")
```

Once the cleaner script has been run issue command `ls()` to check that all necessary system files are available for the model to be run. The system files will be the names of each land use (e.g., urb, nat, agr), and their extensions for the various blocks (mat_, _acc, _suit). To run the model correctly startyear, endyear, startlist, lastcat, ext, lu1, lu2 (land use maps), national, regional, localm (region maps) and all the objects

3. As this line has been incorporated into the allocator.R script this is no longer necessary. But it may be helpful for visualisation purposes.

prefixed “demand” and “andem” (one for each land use) are needed as well as the dataframes tablu1 and OriginalDemandTable. The dataframes actorchars, actorcharvals, suitinftab and accinftab are not needed to run the model but may be useful for your info. Just type the name of the variable to view them in the workspace (e.g. actorchars). Finally breakpoints and colors are needed for plotting. Remember that if you have modified lines 63-65 of the START/startlu.R script AFTER running the import land use maps tool in step 2, you need to paste these lines into the command prompt to execute them and overwrite the old values for colors and breakpoints.

Running the model

When you are ready to run the model (i.e. you have been through steps 1-6 successfully, you have cleaned up the workspace with the cleaner script and you are happy with the land use map color legend, hit “7. Run model”. If the model runs successfully, the end year and the simulation of the same map will be plotted side by side. The OUTPUTS directory will contain a .png file of each simulated year and an ascii raster map of each simulated year.

Troubleshooting

If the allocation is not what you are expecting, your first point of call should be the total transition potential (TTP) maps for each land use. Note that you will only be able to see the final TTP maps (i.e. for the last iteration). These are available in the workspace as a raster stack called TTPs, and can be plotted together, by typing simply “plot(TTPs)”. They are also available individually, named TTP_<landuse>. Usually if something has gone wrong at the calibration stage, one or more of these maps may be blank. A typical problem is entering an invalid character somewhere in the set accessibility or set suitability dialogue, which may not throw an error, but generate a null result. For this reason it's recommended to plot the suitability and accessibility maps on screen e.g. plot(urb_suit) after running these tools (see above) to ensure that the maps have been properly generated.

5. How to run a future simulation

Assuming you have a calibrated model, either by loading the example or by calibrating the model from scratch in Section 4, above, you will want to go to the next step and see how the land change tendencies you have established in the calibration phase (which hopefully are a good fit to your second land use map) play out at a hypothetical future date. Section 3 (above) shows how to do this using the simcontrol dialog and a land use tendencies file. However, it is often useful to do this manually without passing through the simcontrol dialog.

To do this you need to do 4 things:

Set a new startyear and endyear:

```
> startyear
[1] 2007
> endyear
[1] 2012
>

> startyear <- 2012
> endyear <- 2050
> startyear
[1] 2012
> endyear
[1] 2050
>
```

2. Import new start map (unless you want to run your simulation from your calibration startdate right through to a later date). Looking at the existing maps:

```
> lu1
class      : RasterLayer
dimensions : 788, 735, 579180 (nrow, ncol, ncell)
resolution : 200, 200 (x, y)
extent     : 539000, 686000, 4639400, 4797000 (xmin, xmax, ymin, ymax)
```



```

coord. ref. : NA
data source : in memory
names      : F_lu07
values     : 1, 7 (min, max)

```

```

> lu2
class      : RasterLayer
dimensions : 788, 735, 579180 (nrow, ncol, ncell)
resolution : 200, 200 (x, y)
extent     : 539000, 686000, 4639400, 4797000 (xmin, xmax, ymin, ymax)
coord. ref. : NA
data source : in memory
names      : F_lu12
values     : 1, 7 (min, max)

```

We see that we can use the existing lu2 map (2012)

```

> lu1 <- oldlu1 #keep a copy of it just in case...
> lu1 <- lu2 #VERY IMPORTANT that the new map you want to use as a start date is called lu1.

```

3. Change the starting demand ("demand<somenunder>, where <somenunder> is the the value in column LU_cat in the land use table "tablu1") to match the starting map. In this case, since we will used land use in 2012 as the start map, we need to set the demand to the amount of cells of that land use in 2012. We can do this with reference to our original land use table, "tablu1"

```

> tablu1
  LU_cat demand_2007 demand_2012 landuse status actor_influence
1    1    149089    149237   nat    p            0
2    2    101092    100814   agr    p            0
3    3     7416     7456   urb    a            0
4    4      84     233   solar  a            1
5    5     117     117   wind    a            1
6    6     2015     1958  water    s            0
7    7    319367    319366 outside  s            0

```

First, check that the demand already set corresponds to the value for our original map in tablu1

```

> demand1
[1] 149089
> demand2
[1] 101092
> demand3
[1] 7416
> demand4
[1] 84
> demand5
[1] 117
> demand6
[1] 2015
> demand7
[1] 319367

```

then modify the demand to match the value for 2012, e.g.

```

> demand1 <- 149237
> demand1
[1] 149237
and so on...

```

4. Finally, check the annual demand ("andem<somenunder>, where <somenunder> is the the value in column LU_cat in the land use table "tablu1").

```

> andem1

```

```
[1] 29.6
> andem2
[1] -55.6
> andem3
[1] 8
> andem4
[1] 29.8
> andem5
[1] 0
> andem6
[1] -11.4
> andem7
[1] -0.2
```

When the model runs, the total number of cells to be allocated for each land use at a particular timestep is created by:

```
round(demand",j,"+(andem",j,"*ii),0)
```

To see how this works, consider the following example. The variable demand3 gives the initial demand for urban land use, and andem3 gives the annual demand increment for urban land use. That means that in timestep 2, for example, the model will allocate 7472 cells of urban land ($7456 + (8 \times 2)$) and in timestep 4, for example, the model will allocate 7488 ($7456 + (8 \times 4)$) cells of urban land.

Since the model must always allocate this demand somewhere, as long as there is space in the model, the active uses, (for which the user has set neighbourhood rules that occupy other land uses), will be allocated to their full required demand. The active uses with the highest transition potentials will take over other active land uses with lower transition potentials according to the defined neighbourhood rules.

For the initial future simulations, it is recommended not to change the values for “andem”. This allows us to see how the land change tendencies observed in the calibration period play out in future. Once you are happy, you can then modify these values to obtain a particular future target, e.g. 1000 cells of solar energy land use in 2040. If you don’t know the final amount of land use you want to allocate, you can use a linear or exponential tendency (see Section 3, Step 7).

NB: If you are going to use a tendencies file (see Section 3, Step 7), you can ignore the andem and demand variables, since these will be set automatically from the tendencies file by the function SWITCH/demandtendency.R

Finally, to run the simulation, make sure the preparation work is complete (see Section 3) and then hit “Run Model” on the switchboard, or enter

```
>source("ALLOCATOR/allocator.R")
```

The simulation image files (png) and raster maps will be created in the OUTPUTS directory. The final simulation map is always called:

```
sim_<endyear>
```

which is the same as Tn once the simulation is complete.

so, in our case, we can plot the map to the display with “plot(sim_2050)” and examine the cell frequency for the simulation with “freq(sim_2050)”.

Have fun!

Richard Hewitt, April 1st 2020