Malaria Simulation (MaSim)
Version: 4.1.7.1

Robert Zupko

Compilation date 2024-02-13 for version 4.1.7.1

This document was produced in part through funding from the National Institutes of Health (NIAID R01AI153355), and the Bill and Melinda Gates Foundation (OPP159934, INV-005517).

Contents

Introduction

Malaria is a vector-borne infectious disease that is transmitted by a bite from an infected *Anopheles* mosquito, and is caused by unicellular eukaryote members of the *Plasmodium* genus, which are obligate parasites, requiring a suitable host as part of their life cycle. In humans, *P. falciparum* is the leading cause of malaria deaths, while *P. vivax*, *P. ovale*, and *P. malariae* cause milder infections; less frequent are infections by *P. knowlesi*. Due to the potential severity of cases, emphasis is placed on falciparum malaria and it's eradication; however, eradication is complicated by the tendency of the parasite to develop drug resistance with artemisinin partial resistance now having being identified in several parts of Africa (World Health Organization 2022).

1 Model Description

The Malaria Simulation, or MaSim, is a stochastic, individually-based model (or individually-based micro simulation) designed to investigate the evolution of antimalarial resistance by the *Plasmodium falciparum* in the presence of an antimalarial therapy given to a symptomatic individual, with transmission driven by a force of infection model. The simulation may be run as either a regionally-based model or a geographically-based model. While both approaches include the same levels of population heterogeneity, when run as a regionally-based model spatial heterogeneity (i.e., varying prevalence and transmission intensity) is limited whereas the geographically-based model allows for heterogeneity to be modeled at the configuration resolution of the simulation (e.g., 25 km^2).

1.1 Individuals

All individuals within the simulation represent humans that may become infected by the *P. falciparum* parasite, which leads to either a symptomatic or asymptotic infection. In the event of a symptomatic infection, individuals will seek treatment at the configured rate and will receive a therapy based upon the configuration provided to the simulation. Upon taking a therapy, a simplified pharmacokinetic and pharmacodynamic (PK/PD) is used to determine the blood concentration of the drug(s) in blood. The blood concentration, combined with the drug resistance profile for the *P. falciparum* clone(s) determines the killing rate of the parasite and reduction of blood parasitemia.

1.2 Infection

While *P. falciparum* is transmitted by female *Anopheles* mosquitoes, as a modeling simplification a force of infection (FOI) model is used that presumes that transmission is driven in part by the total parasite load of all individuals within a given area. The likelihood that an individual is bitten is then determined by their individual biting attractiveness and a seasonal adjustment to account for the variable number of mosquitoes present throughout the year. All bites are presumed to be infectious and the probability of infection is determined by the individual's underlying immune response to *P. falciparum*.

1.2.1 Infection Life Cycle

The infection life cycle for an individual begins when they are selected by the simulation to be "bitten" by an infectious mosquito. This triggers a sporozite challenge in the individual in which the infection proceeds to based upon the following equation (Zupko et al. 2022):

$$Pr_{inf} = \begin{cases} Pr & \text{if } \theta < 0.2 \\ 0.1 & \text{if } \theta > 0.8 \\ Pr\left(1 - \frac{\theta - 0.2}{0.6} + 0.1\frac{\theta - 0.2}{0.6}\right) & \text{otherwise} \end{cases}$$

Where Pr is the probability of infection for an immunological naïve individual, θ is the individual's current immunity, and Pr_{inf} is the final probability of infection. Upon failing the challenge the individual's state is set to `exposed`, the infection then proceeds to the liver stage, and seven days later, the blood stage. Upon entering the blood stage the individual's state is set to `asymptomatic`, the parasitemia is set to a random value within the asymptomatic bounds. Next, the individual is checked to see if they have an effective drug in their blood[1], if they do then the infection will remain asymptomatic, otherwise it may progress to a clinical infection.

The probability that an asymptomatic infection will progress to clinical is dependent upon the individual immune response (Nguyen et al. 2015):

$$Pr_{clin} = \frac{0.999}{1 + \left(\frac{\theta}{\theta_{mid}}\right)^{z}}$$

Where θ_{mid} is the configured midpoint infliction of the immunity curve, and Pr_{clin} is the final probability of a clinical infection. In the event that the individual proceeds to a clinical infection, then after the appropriate number of days for their age, the clinical infection manifests. At this point the individual's state is set to `clinical` and the parasitemia is increased to be within the clinical bounds. While the simulation assumes that all clinical infections manifest with recognizable symptoms, treatment seeking may not always occur. As a result, following progressing to clinical, the individual then decides to seek treatment with a probability that is appropriate for their current location and their age group.[2]

When an individual seeks treatment, the simulation first checks to ensure that it will prevent death. If the treatment is insufficient, then the individual state is set to `dead` and the death is processed by the simulation. Otherwise, a series of events are scheduled wherein the individual may take the prescribed therapy each day, and parasites are cleared from the blood based upon the current immune response and level of drugs in the blood. During this time, the individual's state may progress from `clinical` to `asymptomatic` based upon the parasitemia. Once the individual has completed the course of treatment, if there are still parasites present in the blood, then the individual is checked to see if they will relapse based upon the configured value. Otherwise, they continue to clear the parasite based upon their immunity clearance rate. Once the parasite is cleared, and if there are no other clones present, the individual state will be restored to `susceptible`.

In the event that the individual does not seek treatment, the checks to see if the infection will result in death. If so then the individual state is set to `dead` and the death is processed by the simulation. Otherwise, the individual slowly clears the infection via their immunity clearance rate which will progress them from `clinical` to `asymptomatic` to `susceptible`.

In areas with a high prevalence, multiple infections with different clones is possible and the progression is the same as described in this section for each clone. In the event that multiple clones are present in the blood, then the killing rate is calculated for each of the clones based upon the drugs efficacy upon that clone.

1.3 *P. falciparum* Genotypes

Upon model initialization, the circulating genotypes of *P. falciparum* is determined by the configuration; however, when mutations are enabled, there is a small probability of a mutation occurring. While this mutation may confer an evolutionary advantage (e.g., resistance to an antimalarial), if accompanied by a fitness cost, the mutation may be lost or out competed by wild-type parasites without the mutation. Also included in the model is recombination due to interrupted feeding by a mosquito resulting in multiple clones being present in the gut. The probability for this occurring is based upon the configuration provided and the FOI of the circulating genotypes.

[1] This can occur if they are treating a recent infection, participated in a mass drug administration campaign, or are taking seasonal malaria chemoprophylaxis.

[2] In other words, the probability of seeking treatment is determined by the location that the individual is currently in, not their original location or intended destination.

1.4 Policy Interventions

Within the simulation, drug policy interventions (e.g., a change in first-line therapies) are supported with limited support for non-therapeutic interventions (e.g., bed net distribution). Due to the lack of an underlying vector model, interventions targeting the *Anopheles* mosquitoes are not supported.

Geospatial Data

2 Introduction

A key part of the simulation is the spatial component that allows for increased realism due to the spatial heterogeneity that is introduced. However, ensuring that the spatial data is prepared correctly for the simulation can be a challenge since it needs to be done for each country that is simulated. This chapter will walk the reader through the process of preparing the spatial data for a country, using Kenya as an example for the process.

The first step in preparing the geospatial data is to gather all of the relevant sources, this typically includes:

1. Political boundaries - national and sub-national
2. Population
3. Malaria prevalence
4. Travel or friction surface

Possible sources of this data is included in the appendix while Figure 1 offers a summary workflow of raster production.

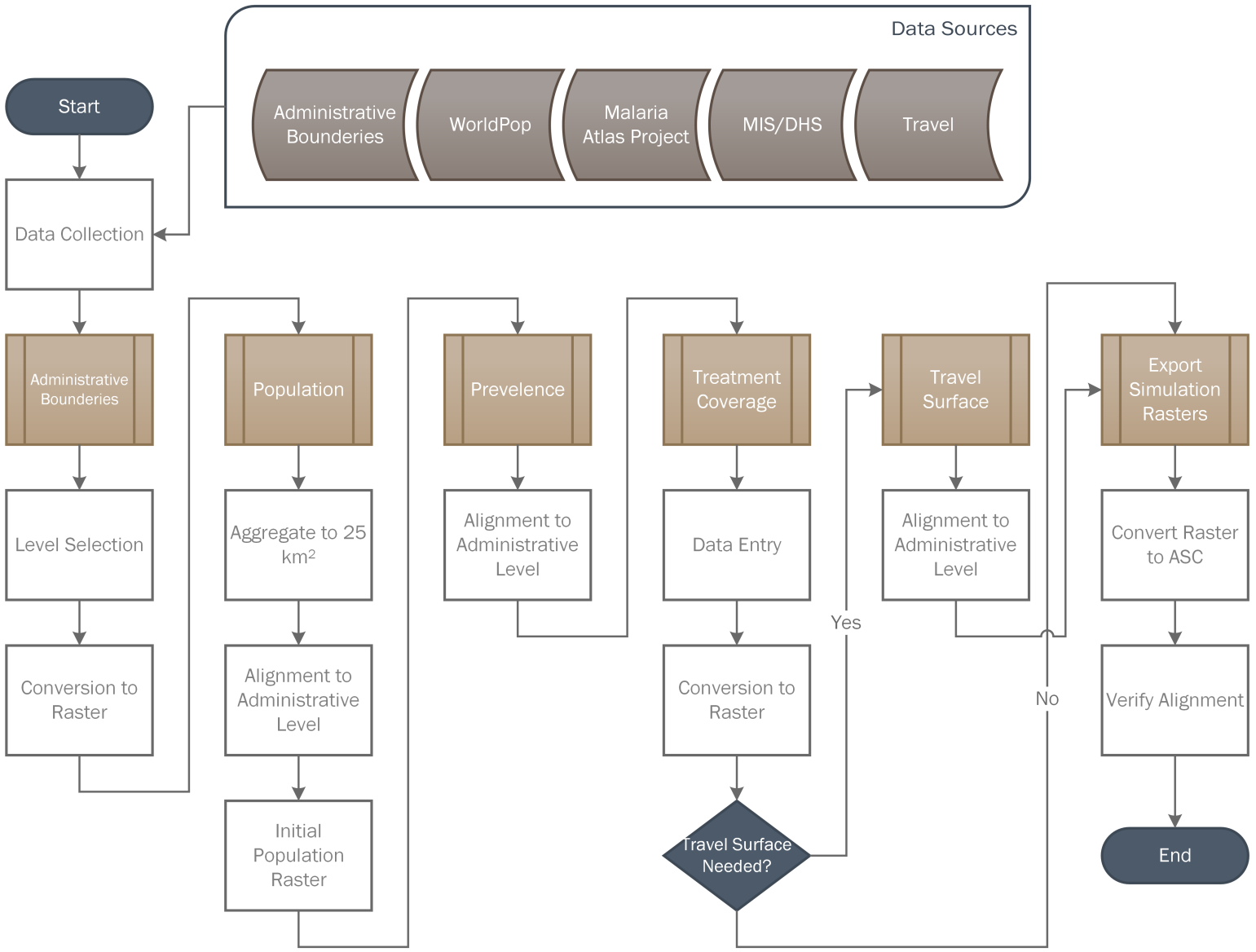


Figure 1: The typical workflow used when preparing the geospaital data for a country

3 Raster Preperation

3.1 Determining the Projection

Once all of the relevant geospatial data has been collected, the first step is determining the *datum* (the frame of reference for measuring locations on the Earth) and *projection* (the method used to portray the spherical coordinates on a flat surface). A useful starting point for this is epsg.io, which maintains a database of over 6,000 coordinate systems.

A typical starting point is to enter the name of the country, which will return a list of all of the relevant records (see Figure 2). Typically, for the purposes of the simulation, using the appropriate Universal Transverse Mercator (UTM) zone projection on the World Geodetic System 1984 (WGS84) datum will provide a reasonable level of accuracy for the simulation. Although large countries may overlap multiple zones, which case the bounding of the projection should be checked to ensure that it covers the majority the country (see Figure 3). Once an appropriate datum and projection has been identified, make a note of it along with the Well-Known ID (WKID) as part of the project documentation since this will be used in the preparation of all of the geospatial data for the simulation.

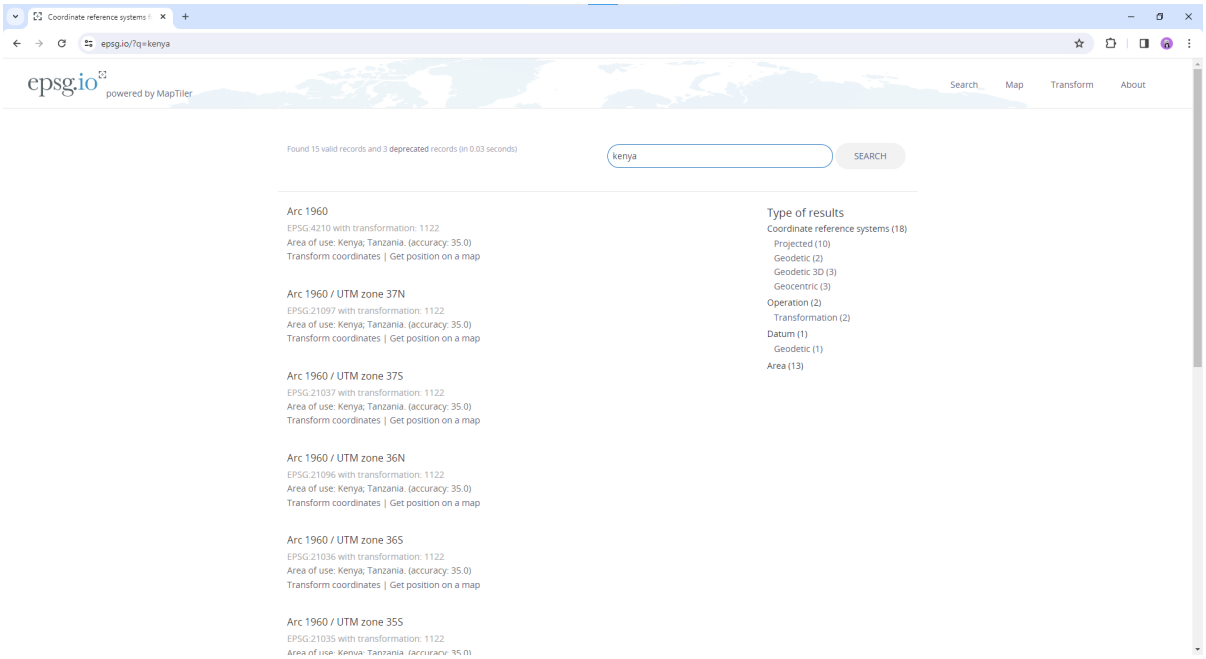


Figure 2: The results of a search for "kenya" on epsg.io

Once the projection has been determined and data gathered, start by creating a new project in ArcGIS Pro via `New Project > Map` for this example call the project `Kenya`, set the location to a folder such as `C:\GIS\`, and make sure `Create a new folder for this project` is checked. This will create a new project with the default map of the United States (see Figure 4). Start by removing the `World Topographic Map` and `World Hillshade` base maps by right clicking on them under `Drawing Order` and selecting `Remove`. You should now have an empty map with no contents listed.

We will now set the coordinate system for the map. Start by right clicking on `Map` under `Drawing Order` and selecting `Properties`, this will load a new window titled `Map Properties: Map`. Click on `Coordinate Systems` on the left hand side of the window, this will display the current coordinate system for the map. By default `WGS 1984 Web Mercator (auxiliary sphere)` is used, but we want to set the map to the coordinate system system selected for the country we are working with. Locate the `Search` box under the `Current Z` box and enter the WKID value identified on epsg.io - using Kenya as our example this would be `21097`. A

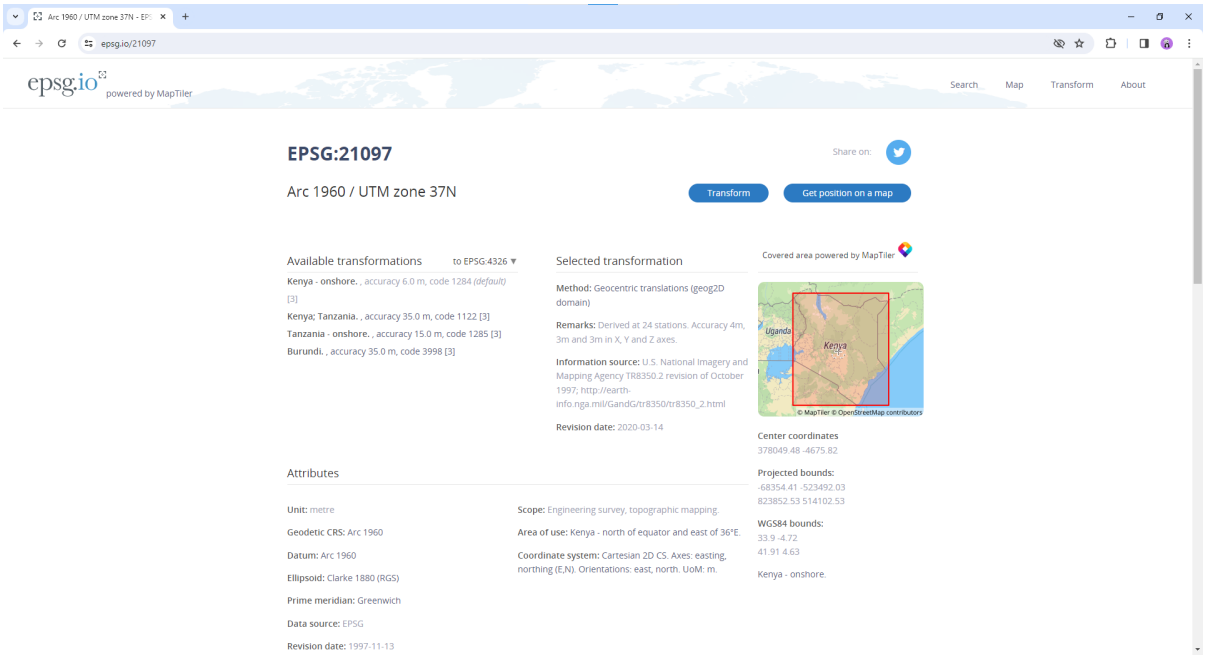


Figure 3: The details page for Arc 1960 / UTM zone 37N on epsg.io. Note the Well-Known ID (WKID) that appears at the top of the page (EPSG:21097) as well as the bounding box that appears over the map of Kenya.

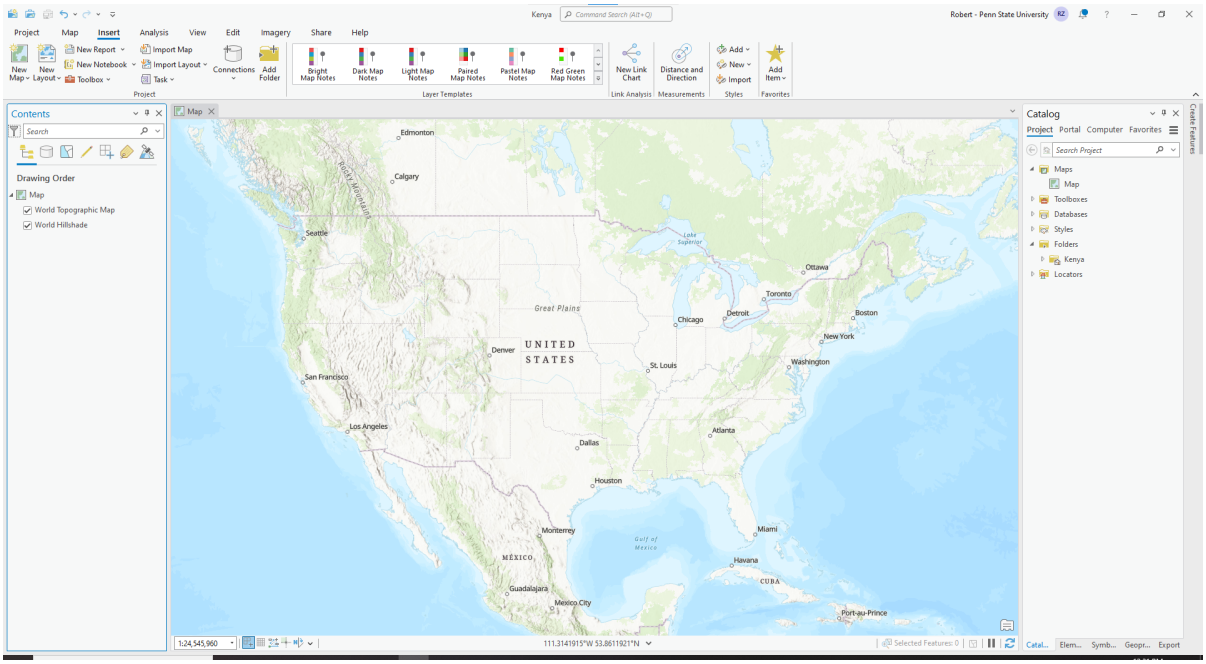


Figure 4: ArcGIS Pro with the default map of the United States

valid WKID should only return a single entry, in this case `Arc 1960 UTM Zone 37N` (see 5). Make sure the new coordinate system is listed under the `Current XY` and click `OK`.[3] The default coordinate system has now been set for the map.

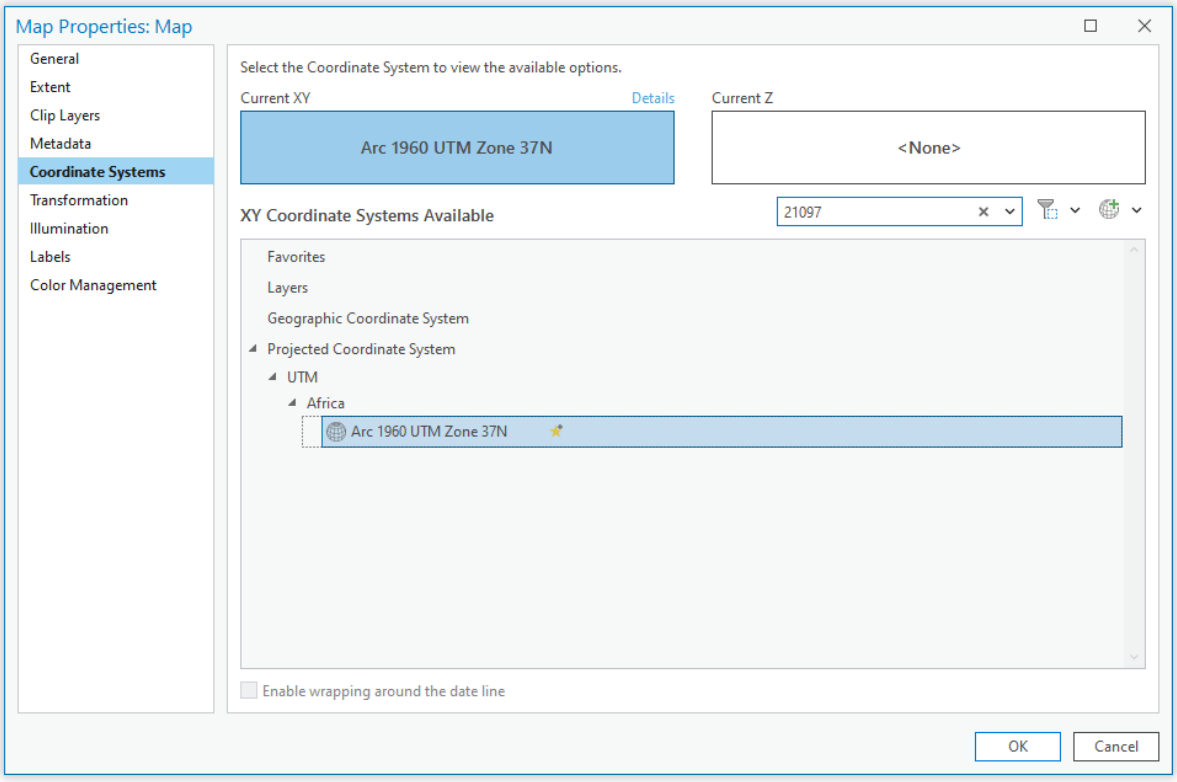


Figure 5: Setting the new coordinate system in ArcGIS Pro

3.2 Preparing the District Raster

Next, we need to load the administrative boundaries for the country and prepare 1) raster data that can be used by the simulation, and 2) a comma separated value (CSV) file that can be used to map the district identification value (i.e., `district id`) assigned when creating the raster data, to the name of the district.

Start by finding the project directory created by ArcGIS, in this case it is `C:\GIS\Kenya` and create a new folder called `data` and one called `simulation`. The `data` folder will be used for storing any data files that are downloaded. Next, locate the administrative boundaries for the country.[4] Typically shapefiles are distributed as a compressed file, decompress the file into the `data` directory and have ArcGIS refresh the folder listing by right clicking on home folder and selecting `Refresh` or pressing the `[F5]` key. You should see something similar to Figure 6.

Next, right click on `Map` under the `Drawing Order` panel and select `New Group Layer`, an entry called "New Group Layer" will appear. Double check on the `New Group Layer` and enter "Administrative Boundaries" as the `Name` in the `Layer Properties: New Group Layer` dialog that appears. Click `OK` and under the `Drawing`

[3] Since the simulation models everything as a flat plane, it is unlikely that `Current Z` will ever be set to anything.

[4] For Kenya, these were found on the Humanitarian Data Exchange (HBX) by searching for "kenya" at https://data.humdata.org/dataset/cod-ab-ken. Note that an advantage of using data from HBX is that the data is open access under the Creative Commons Attribution for Intergovernmental Organisations, this is relevant when reproducing shapefile data in maps used in publications.

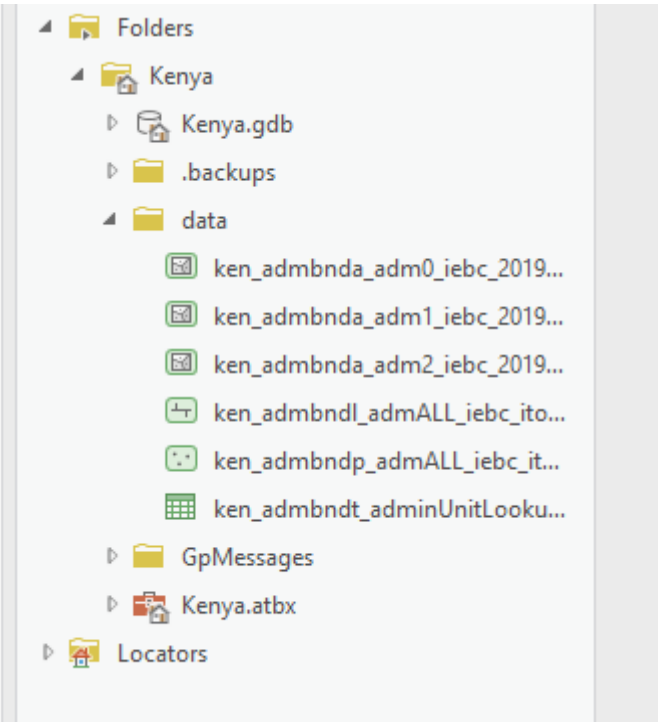


Figure 6: Refreshed listing of all of the shapefiles in the data directory in ArcGIS Pro

`Order` panel you should have an entry called `Map` with an entry called `Administrative Boundaries` below it.

Now, locate the administrative levels zero (or national boundary), one, and two in the `data` folder of the `Catalog` panel and drag them under the `Administrative Boundaries`. Right click on `Administrative Boundaries` and select `Zoom To Layer` you should now have a map similar to Figure 7.

At this point we could start creating the raster files, but first we should adjust the appearance of the shapefiles to make them easier to work with. Start by renaming them by either double clicking on the layer and changing the name in the `Layer Properties` dialog that appears, or single clicking on the layer name and editing it directly in the `Drawing Order` panel.[5]

Next, we will adjust the symbology that is being used to render the layers so that we can see them all. Start by right clicking on the administrative level zero layer selecting `Symbology`, the `Symbology` panel should appear on the right. Click on the symbol and in the gallery that appears, select `Black Outline (2pts)`. The map should now refresh with the national border in bold and the administrative level one appearing (see Figure 8). Repeat this process for for level one, selecting `Black Outline (1pt)`. Finally, for the level two layer, start by selecting `Black Outline (1pt)` but then click `Properties`, this update the dialog to show the full list of appearance settings. Click the color box next to `Outline color`, select `Gray 50%`, and click `Apply`. The map should now appear similar to Figure 9.

Now, we will cover the administrative layer one shapefile to a raster file, and this process is the same regardless of the administrative level of a shapefile. To reduce the visual clutter of the map you may also wish to hide the administrative level two shapefile by clicking the box next to the layer name under the `Drawing Order`.

First, we need to examine the data associated with the administrative level one shapefile. This can be done by right clicking on the layer and selecting `Attribute Table`, a spreadsheet should appear below the map

[5]One suggested format is the country name followed by "Admin NUMBER", or "Kenya - Admin Zero" in the case of our example.

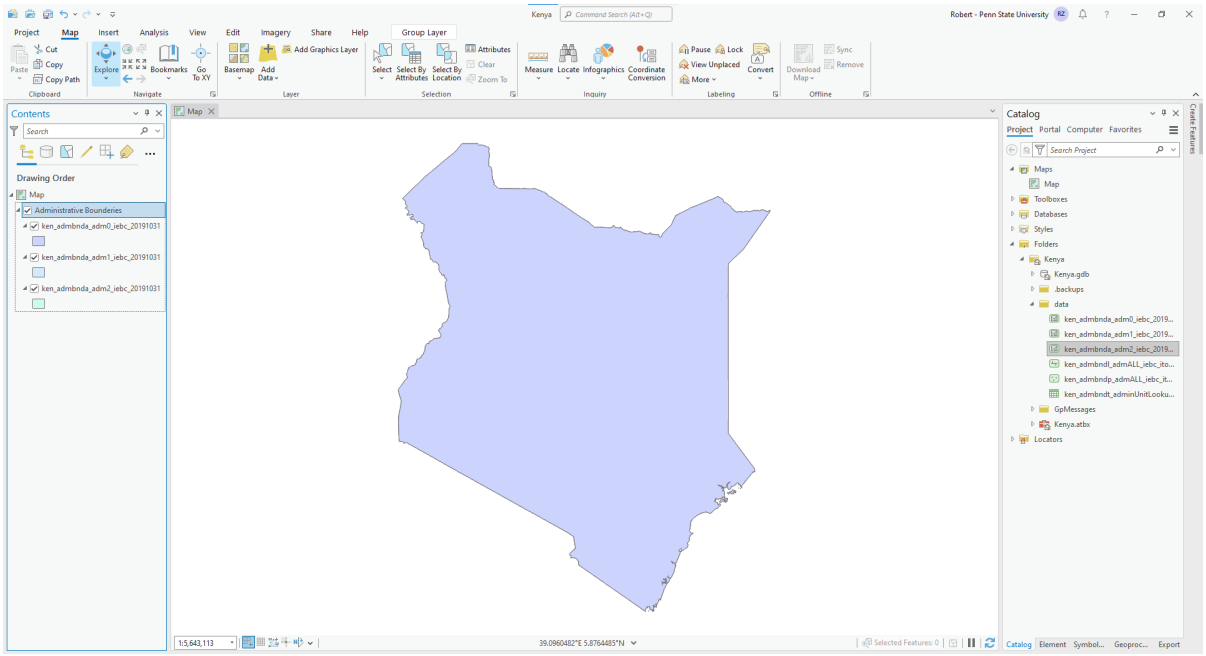


Figure 7: The apperance of the map after all shapefiles have been added. Note that the top most layer – the national boundry – is covering all of the other layers

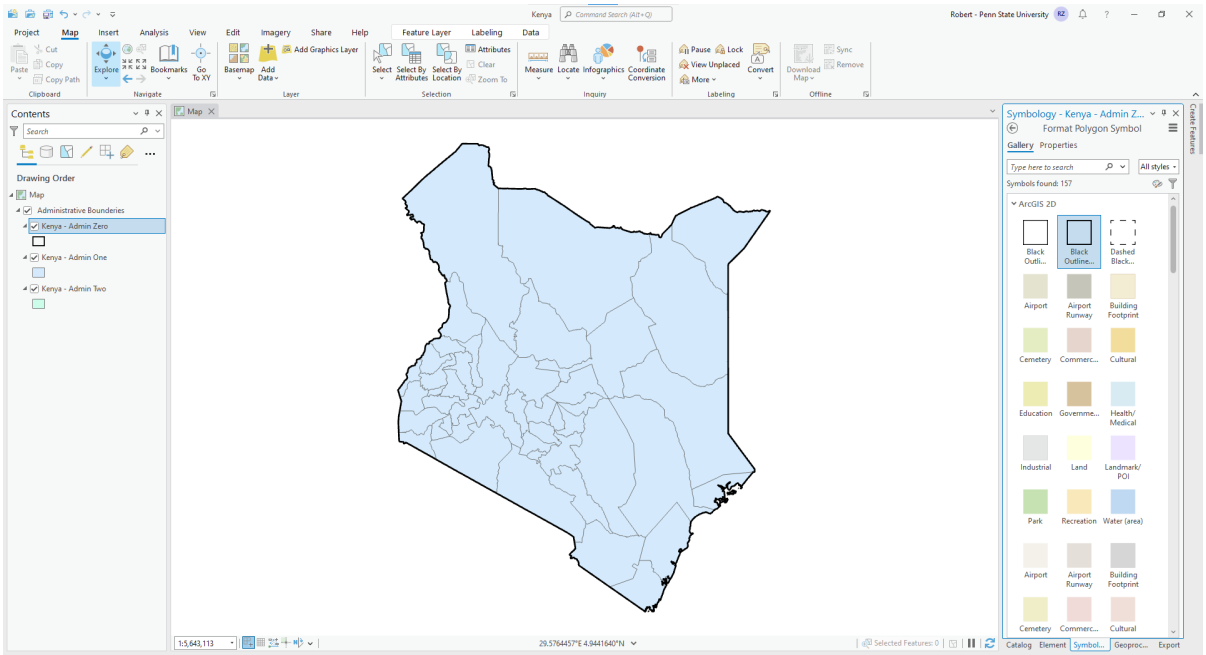


Figure 8: The appearnace of the map after the symbology of the administrative level zero has been changed

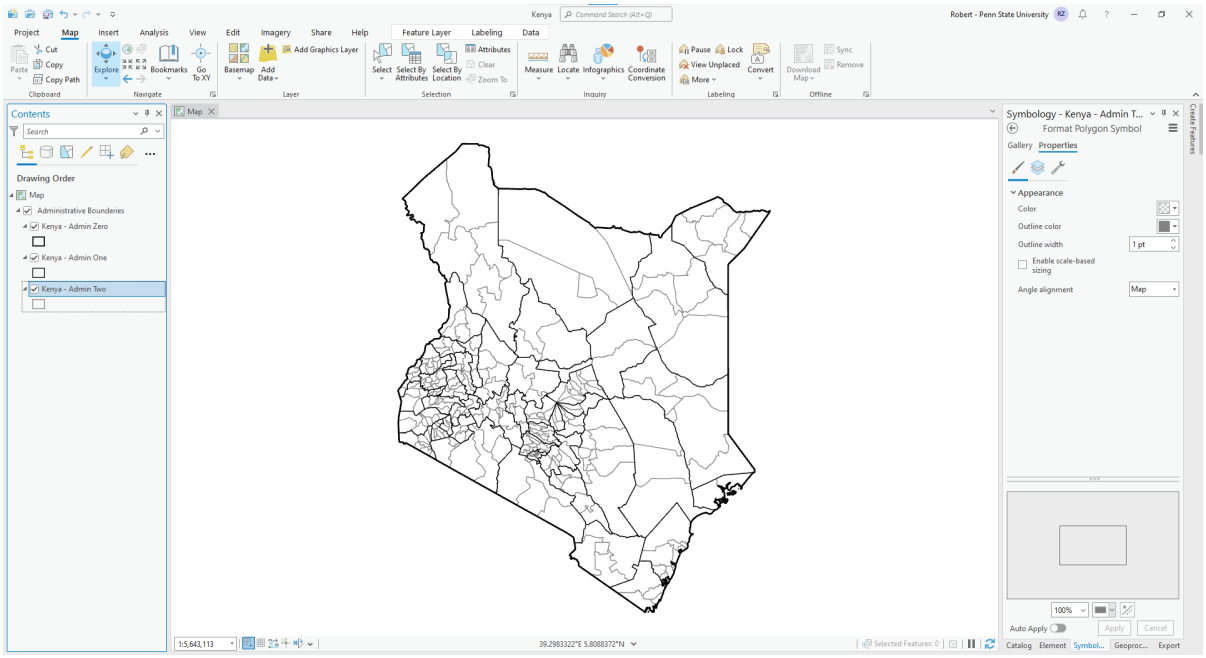


Figure 9: The final apperance of the map after all of the administrative layers have had their symbology updated

(see Figure 10).

In examining the attribute table we want to first check to see if there is a numerically indexed field that we can use as the district id assignment, and what the indexing of the field is. In the case of the data from Kenya, the `FID` field could serve this role, but since it is a reserved field by ArcGIS Pro, it cannot be exported, so we are going to add our own ID field. Start by clicking the `Add` button next to the `Field:` label, this will case the dialog to shift to the fields list for the layer. At the bottom of the list of fields, type "ID" under the `Field Name` and make sure `Short` is selected as the `Data Type`, click save and the updated list should look similar to Figure 11.

Close the `Fields` view and scroll to the right of the attribute table, the new `ID` field should be present with all rows set to zero. Since the `FID` provides a unique value for each row, we can use that as the basis for the district id.

Start by right clicking on `ID` and select `Calculate Field`, a new dialog should appear. In the text box under `ID =` enter "`!FID! + 1`" which will set the id to be equal to the `FID` value, plus one (if the `FID` is one indexed, then you only need to enter "`!FID!`"). Click `OK` and the `ID` column will be updated to have a unique value in for each row.

With the `ID` column now created, we can proceed to creating the raster. Close the attribute table and then under the `View` menu, select the `Geoprocessing` option, this will load the `Geoprocessing` panel on the right side of the screen. Type "polygon to raster" and select `Polygon to Raster (Conversion Tools)` when it appears in the list, the dialog will update with the setting prompts for the tool. Next, enter the following settings:

1. Under `Input Features` select the administrative level one shapefile
2. Under `Value field` select `ID`
3. Click the folder icon next to `Output Raster Datasest`, select the databases created by ArcGIS Pro for the project (ex., `Kenya.gdb`), enter the name for the new raster (ex., `ken_admin_one`), and click `Save`

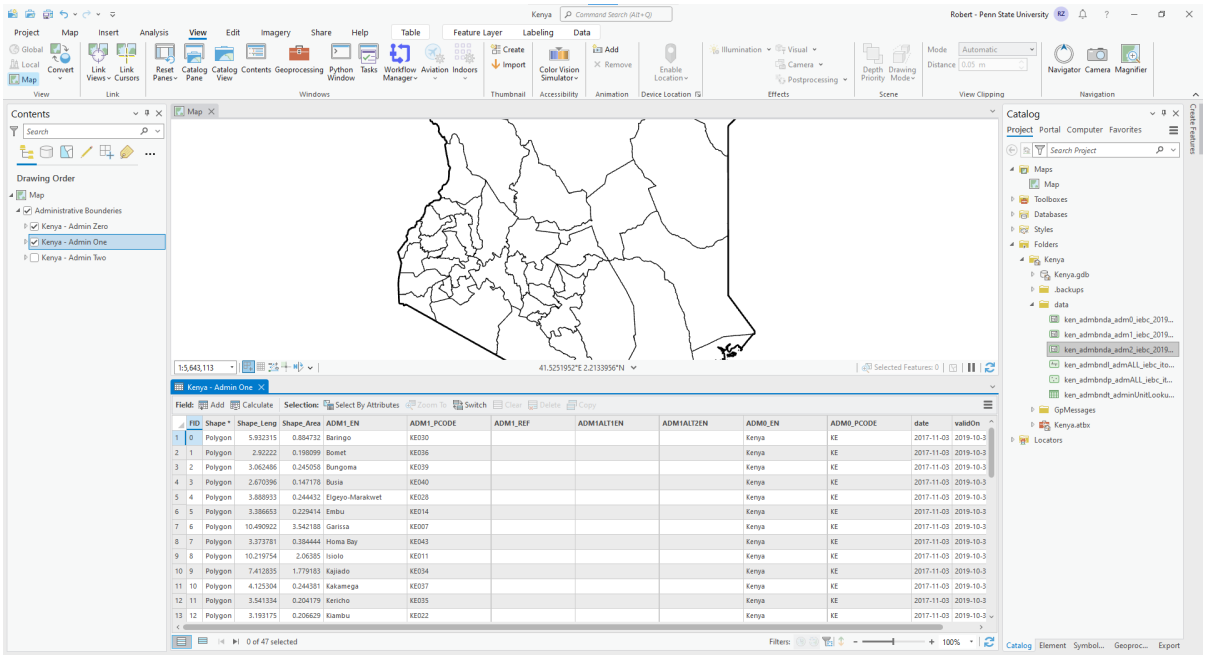


Figure 10: The attribute table for the adminsitriave level one shapefile for Kenya. Note that the FID column is zero indexed with the provience Baringo being associated with the value zero

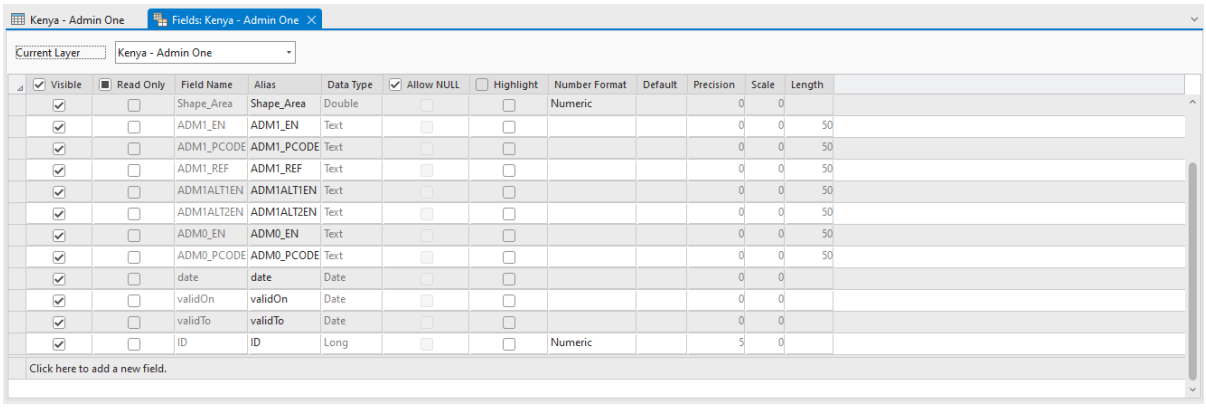


Figure 11: The updated fields list after the ID field has been added

4. Click the `Environments` tab next to `Parameters` and under `Output Coordinate System` select `Current Map [Map]` this will cause the selected projection to be filled in (`Arc_1960_UTM_Zone_37N` as previously entered for Kenya)
5. Click the `Parameters` tab to return to the previous view, you will note that the `Cellsize` has changed
6. For `Cellsize` enter 5000, this corresponds to 5000 meters or 5 kilometers across the x or y axis of a cell
7. Make sure `Build raster attribute table` is checked and click `Run`

Once the calculation is complete you should see a map similar to Figure 12. Note that the scale should go from one to the number of administrative regions (47 provinces in the case of Kenya) and the symbology selected by ArcGIS Pro may be different. At this point you may wish to create another Group Layer for simulation rasters via the process previously outlined for creating the `Administrative Boundaries` grouping, move the new raster into this grouping. The rendering order of the shapefiles and rasters can be adjusted by changing their order under the `Drawing Order`.

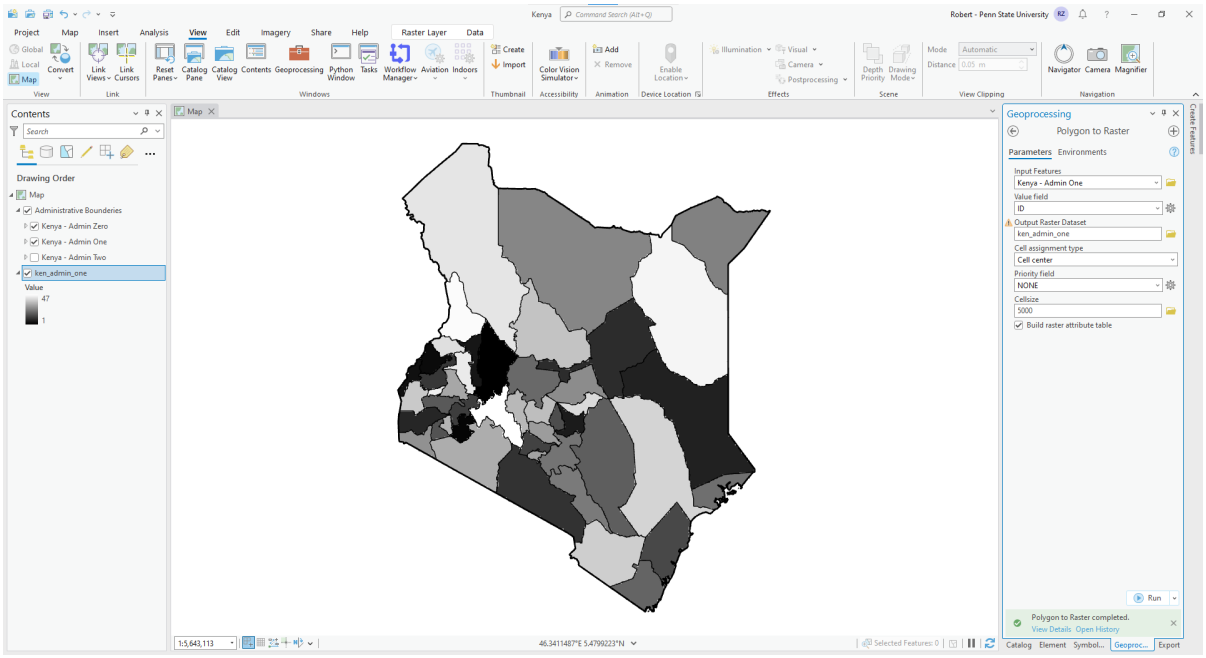


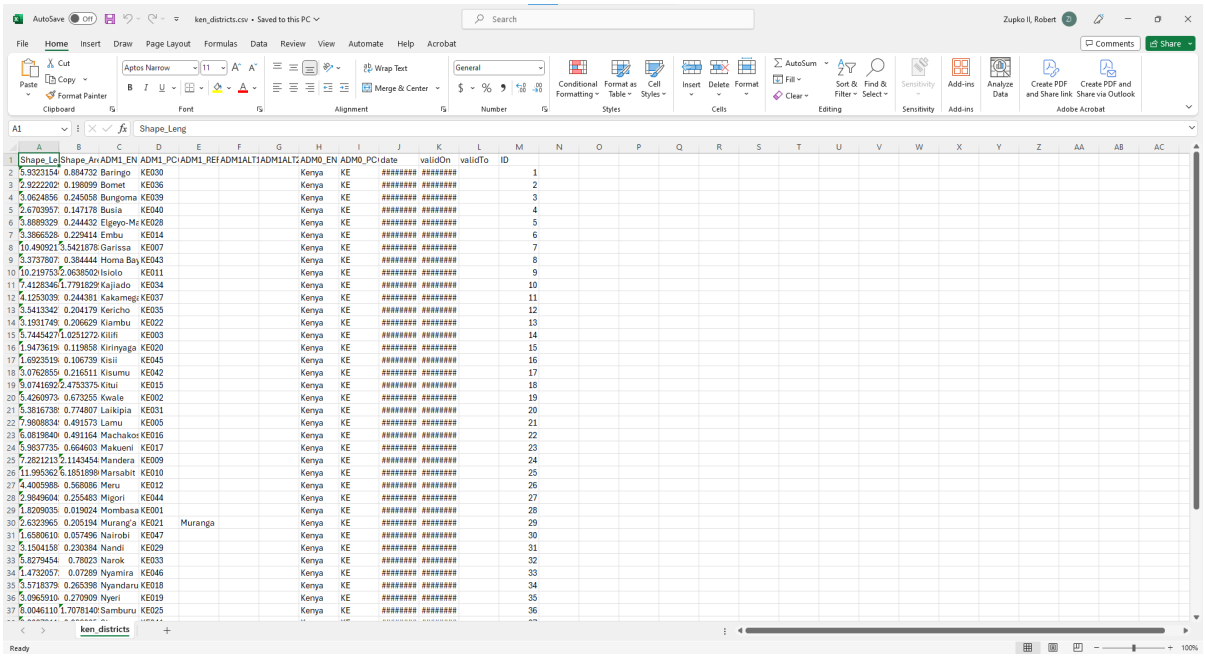
Figure 12: The updated fields list after the ID field has been added

3.2.1 Exporting the District Mapping File

Now is a good time to export the mapping CSV file by right clicking on the layer under `Drawing Order`, and selecting `Data > Export Table`. This causes a new dialog to appear, click the folder icon next to the `Output Table` and the select the location to save the CSV file (e.g., the `simulation` folder created for the project) and enter the file name *inclusive of the .csv extension* and click `Save`, click `OK` to export the CSV file and close the dialog.

Locate the CSV file on disk and open in spreadsheet software such as Microsoft Excel. If you are using Excel do not allow it to convert any of the data since we will be deleting most of it and changing the order. Upon loading the CSV the contents should appear similar to Figure 13. Delete all columns except for the administrative region name in English and the ID column. Once complete you should have two columns left. Next, change the order of the columns so that the ID column is first, and the English name column is second. Save the CSV file. At this point the district mapping CSV file is ready and the first row will be

used as the field headings.[6]



Shape_Le	Shape_Ar	ADM1_EN	ADM1_PCI	ADM1_REF	ADM1ALT1	ADM1ALT2	ADM0_EN	ADM0_PCI	date	validOn	validTo	ID
5.9323154	0.884732	Baringo	KE030				Kenya	KE	########	########		1
2.9222202	0.198099	Bomet	KE036				Kenya	KE	########	########		2
3.0624856	0.245058	Bungoma	KE039				Kenya	KE	########	########		3
2.6703957	0.147178	Busia	KE040				Kenya	KE	########	########		4
3.8889329	0.244432	Elgeyo-Ma	KE028				Kenya	KE	########	########		5
3.3866528	0.229414	Embu	KE014				Kenya	KE	########	########		6
10.490921	3.5421878	Garissa	KE007				Kenya	KE	########	########		7
3.3737807	0.384444	Homa Bay	KE043				Kenya	KE	########	########		8
10.219753	2.0638502	Isiolo	KE011				Kenya	KE	########	########		9
7.4128346	1.7791829	Kajiado	KE034				Kenya	KE	########	########		10
4.1253039	0.244381	Kakamega	KE037				Kenya	KE	########	########		11
3.5413342	0.204179	Kericho	KE035				Kenya	KE	########	########		12
3.1931749	0.206629	Kiambu	KE022				Kenya	KE	########	########		13
5.7445427	1.0251272	Kilifi	KE003				Kenya	KE	########	########		14
1.9473619	0.119858	Kirinyaga	KE020				Kenya	KE	########	########		15
1.6923519	0.106739	Kisii	KE045				Kenya	KE	########	########		16
3.0762855	0.216511	Kisumu	KE042				Kenya	KE	########	########		17
9.0741692	2.4753375	Kitui	KE015				Kenya	KE	########	########		18
5.4260973	0.673255	Kwale	KE002				Kenya	KE	########	########		19
5.3816738	0.774807	Laikipia	KE031				Kenya	KE	########	########		20
7.9608834	0.491573	Lamu	KE005				Kenya	KE	########	########		21
6.0819840	0.491164	Machakos	KE016				Kenya	KE	########	########		22
5.9837735	0.664603	Makueni	KE017				Kenya	KE	########	########		23
7.2821213	2.1143454	Mandera	KE009				Kenya	KE	########	########		24
11.995362	6.1851898	Marsabit	KE010				Kenya	KE	########	########		25
4.4005988	0.568086	Meru	KE012				Kenya	KE	########	########		26
2.9849604	0.255483	Migori	KE044				Kenya	KE	########	########		27
1.8209035	0.019024	Mombasa	KE001				Kenya	KE	########	########		28
2.6323965	0.205194	Murang'a	KE021	Muranga			Kenya	KE	########	########		29
1.6580610	0.057496	Nairobi	KE047				Kenya	KE	########	########		30
3.1504158	0.230384	Nandi	KE029				Kenya	KE	########	########		31
5.8279454	0.78023	Narok	KE033				Kenya	KE	########	########		32
1.4732057	0.07289	Nyamira	KE046				Kenya	KE	########	########		33
3.5718379	0.265398	Nyandarua	KE018				Kenya	KE	########	########		34
3.0965910	0.270909	Nyeri	KE019				Kenya	KE	########	########		35
3.0046110	1.7078140	Samburu	KE025				Kenya	KE	########	########		36

Figure 13: The data exported by ArcGIS Pro to CSV via the Export Table function

3.3 Preparing the Population Raster

After preparing the district raster, the next step is to prepare the population raster. In doing so, there are three approaches that can be used:

1. Using an unconstrained global mosaic of populations at 30 arc seconds[7]
2. Using an unconstrained national mosaic of population at 30 arc seconds[8]
3. Using a unconstrained national mosaic of population at 3 arc seconds[9]

These data sets are *unconstrained* meaning the processing does not attempt to restrict the population based upon building footprints or other restrictions on residency such as national parks.[10] For the purposes of malaria modeling, unconstrained data sets are preferable since we want allow people to move around the landscape and occupy locations that they may not typically (or legally) domicile at.

Note that each of these data sets have a resolution of arc seconds, which are 1/60th of a nautical mile or 30.87 meters are the equator. However, it is important to remember that arc seconds of longitude decrease mathematically as one moves toward the Earth's poles. As such, it is important to remember that distortion **will** occur when projecting data that is originally supplied in degrees.

Here we are going to use second option of processing an unconstrained national mosaic of the population at 30 arc seconds (about 1 km at the equator). Start by dragging the raster on to the map, when the pop-up dialog to build pyramids (used to improve the performance of displaying the raster) and statistics appears,

[6]Most of the tools that use the district mapping are agnostic as to the column names, although some may expect them to be called `DISTRICT` and `NAME`.

[7]Such as WorldPop spatial distrubiton of poulation in 2020

[8]Such as WorldPop spatial distribution of popuation in 2020 for Kenya at 30 arc

[9]Such as WorldPop spaital distrubiton of population for 2020 for Kenya at 3 arc

[10]See https://www.worldpop.org/methods/top_down_constrained_vs_unconstrained/ for more details.

make sure both `Build` and `Calculate` are checked and click `OK`. After the calculations are complete, the raster should appear on the map (see Figure 14).

At this point it is recommended to create a new Group Layer for the raster data (e.g., "Raster Data") and to adjust the symbology of the raster to make it easier to see. Here the raster has been updated so the color scheme is inverted and the "Stretch Type" is set to "Standard Deviation" (see Figure 15).

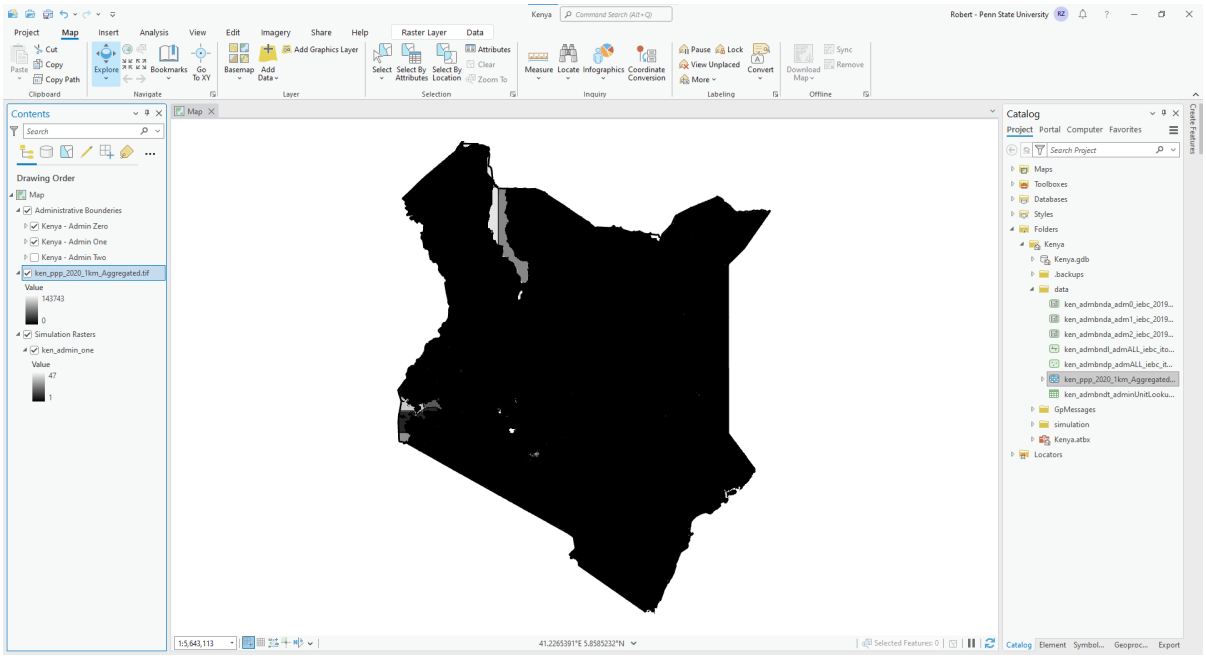


Figure 14: The population raster after it has been initially loaded by ArcGIS. Note that much of the map appears to be black, suggesting a population of zero.

Assuming you are working with data from WorldPop, the next step is to create a new raster whose projection is in meters as opposed to degrees. Under the `Geoprocessing` tab, type "project raster" and click the `Project Raster (Data Management Tools)` command that appears. This will load the `Project Raster` tool to the right side of the screen. Enter the following:

1.`Input Raster` is the population raster that you loaded (e.g., "ken_ppp_2020_1km_Aggregated.tif" in the example, where "ppp" is population per pixel)
2. Select `Current Map [Map]` for the `Output Coordinate System`, this will cause the map's coordinate system to be filled in, along with the correct `Geographic Transformation`
3. The `X` and `Y` values are the cell (i.e., pixel) size, in the case of Kenya the values are both `926.456581851104`, for both of these fields enter "1000"
4. Click `Run`

After processing this will create a new raster and add it to the map (see Figure 16). By default ArcGIS Pro will postfix the names of rasters or shapefiles with the tool used to process them (e.g., "ken_ppp_2020_1_ProjectRaster") and store them in the geodatabase created for the project ("Kenya.gdb" in this case). In most cases these are effectively temporary files and can be deleted once the workflow is complete.

Next the resolution of the raster needs to be reduced from 1000 m (1 km) to 5000 m (5 km). In the `Geoprocessing` tab, type "aggregate" and click the `Aggregate (Spatial Analyst Tools)` command that appears. This will load the `Aggregate` tool to the right side of the screen. Enter the following:

1. `Input Raster` is the projected raster we just created ("ken_ppp_2020_1_ProjectRaster" in this case)

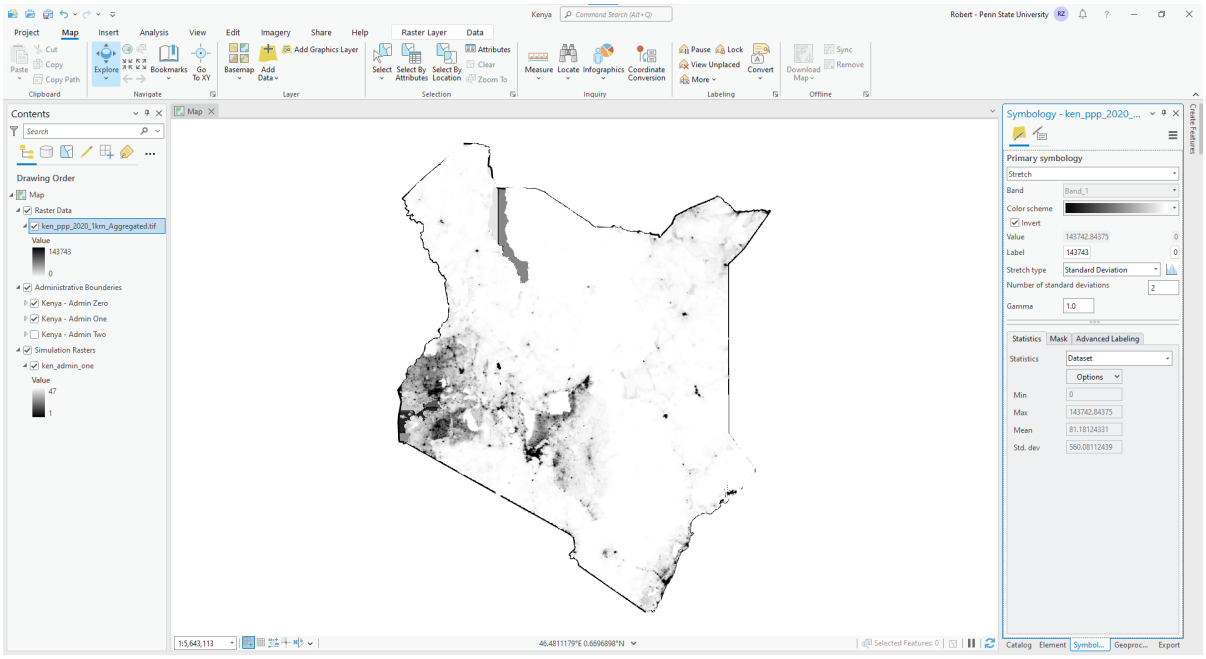


Figure 15: The population raster after it has been moved to a new Group Layer and the symbology is adjusted.

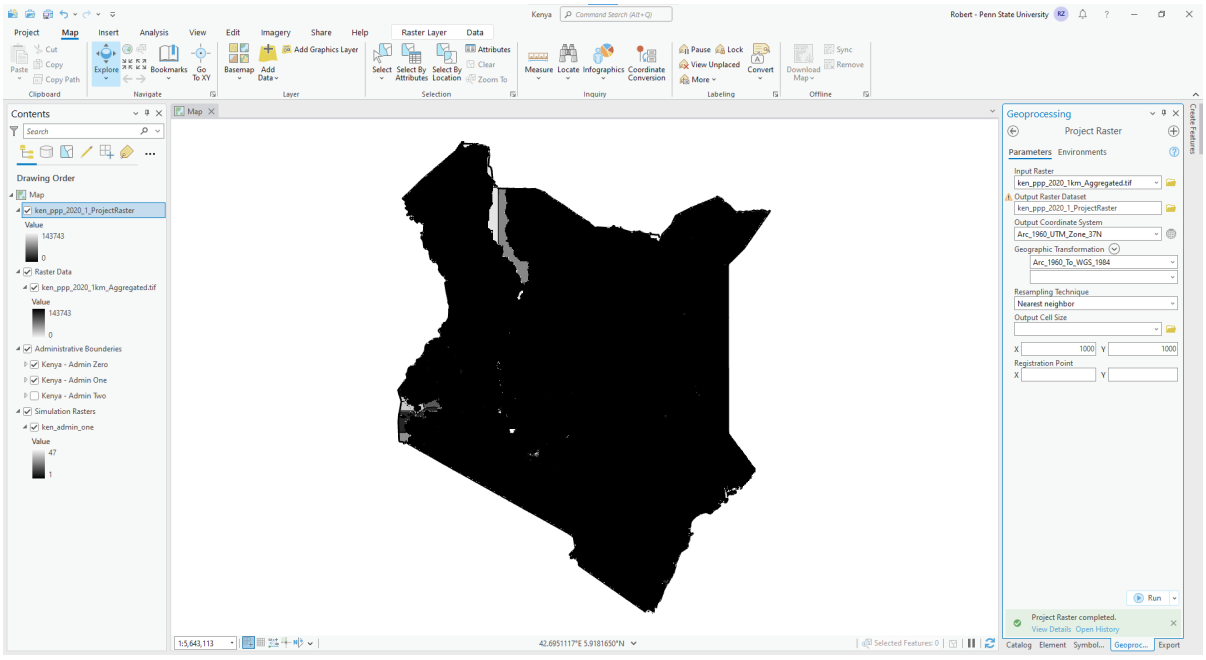


Figure 16: The projected population raster, now in 1000 m resolution. Note the default symbology has again been applied by ArcGIS.

2. `Cell Factor` is the down sampling factor, since we are going from 1 km to 5 km we want to enter 5
3. `Aggregation Technique` is how the aggregation will occur, since we are working with population we want to make sure `Sum` is selected
4. Click the `Enviornments` tab at the top of the tool window so that additional options appear
5. `Snap Raster` will adjust the extent of rasters so that they align with each other, select the previously created district raster (e.g., "ken_admin_one")
6. Click `Run`

Once complete the raster should appear similar to the one in Figure 17. However, perpetration of the population raster is still not complete since the underlying WorldPop data may contain fractional data. For the purposes of the simulation we want to ensure that all pixels contain either `NODATA` or whole integers.

On the `Geoprocessing` tab, type "raster calculator" and click the `Raster Calculator (Image Analyst Tools)` that appears. This will cause the `Raster Calcuator` to load on the right side of the screen. In the text box enter `RoundUp("RASTERNAME")` where `RASTERNAME` is the name of the raster created by the `Aggregation` tool.[11] Under `Output raster` enter the final name for the raster (e.g., "ken_ppp_2020_5km") and since it will write to the geodatabase by default, you may wish to change the output location. Click `Run` and the final raster will appear (see Figure 18).

At this point temporary rasters created during the projection and aggregation can be deleted. You may also wish to reorganize the location of the raster to be with the previously created district raster in the group layers (see Figure 19).

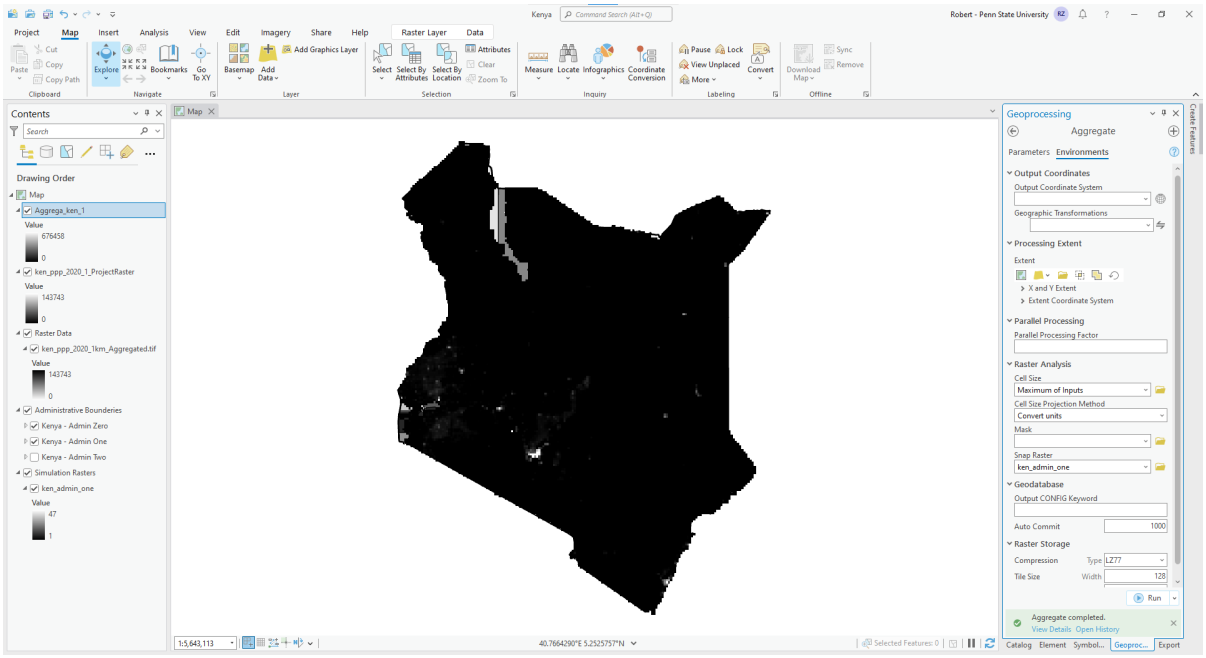


Figure 17: The population raster aggregated up to 5 km. Note that the data now appears slightly more pixelated

[11] The reason that we round up is that we assume that any decimal value represents a whole person.

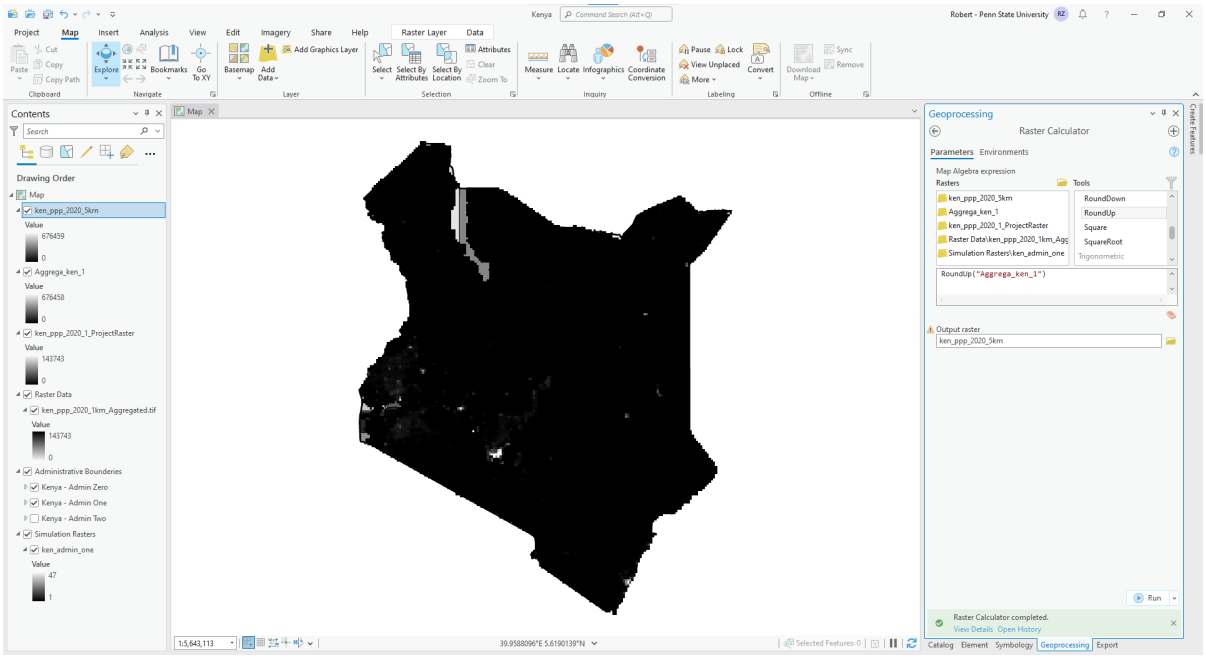


Figure 18: The output of the Raster Calcluator, note that the new raster has the final name applied to it but temporary rasters are still in place.

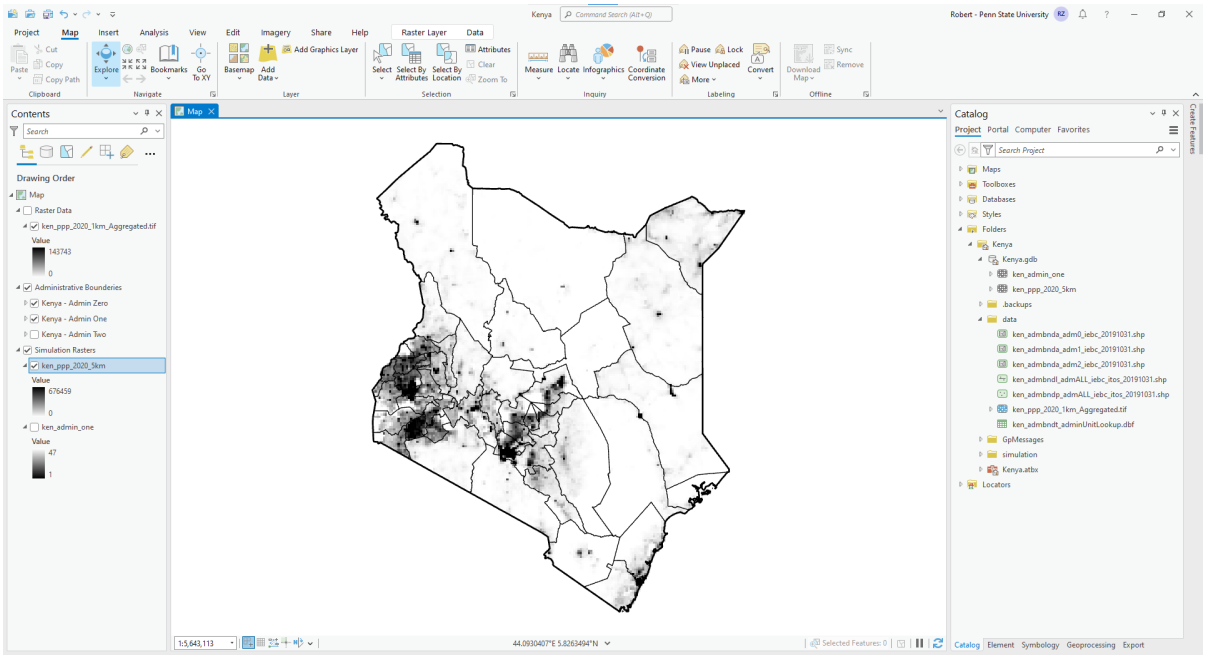


Figure 19: The final popuation raster, with population aggregated up to 5 km. Note the organization of the raster under the "Simulation Rasters" Group Layer and the storage of the raster in the geodatabase.

3.3.1 Preparing the Initial Population Raster

Once the population raster has been prepared for the target calibration year, a derived raster can be created that will be used as the initial population raster in the simulation. Since the simulation assumes a fixed birth rate, the initial population raster is used to instantiate the simulation as opposed to the historical population in the equivalent year of the simulation.

Start by first determining the crude birth rate in the calibration year (i.e., the same year as the previous population data).[12] Next we calculate p', representing the adjusted population in the initial year of the simulation as such:

$$p' = \frac{1000}{\left(1 + \frac{cbr}{1000}\right)^n}$$

Where *cbr* is the crude birth rate and n is the number of years from model initialization to to the calibration year.[13] The value of 1,000 is used as a proxy population for the calculation. Using Excel for the calculation gives us a rounded value of 738.03, which we then use in the following:

$$multiplier = 1 - \left(\frac{1000 - p'}{1000}\right)$$

The resulting *multiplier* is the value that will be used in ArcGIS, with four digits of precision typically being sufficient. An Excel workbook is also included in the simulation repository under `manual/tools/Population-Multiplier.xlsx`, see Figure 20.

Once the *multiplier* has been calculated, the `Raster Calculator` can be used to produce a new raster using the formula:

```
RoundUp("Simulation Rasters\ken_ppp_2020_5km"*multiplier)
```

Where the path is adjusted to the 25 km^2 population raster that was created and *multiplier* is replaced with the calculated value, see Figure 21. Note that `RoundUp` is used to ensure that all decimals are rounded up to the nearest whole number.

3.4 Preparing the Prevelence Raster

Now that the population raster has been prepared, the next step is to prepare the *Plasmodium falciparum* prevalence raster. Typically this is done using the prevalence in individuals aged two to ten years, thus the raster is refereed to as the $Pf\text{PR}_{2\text{-}10}$ raster.

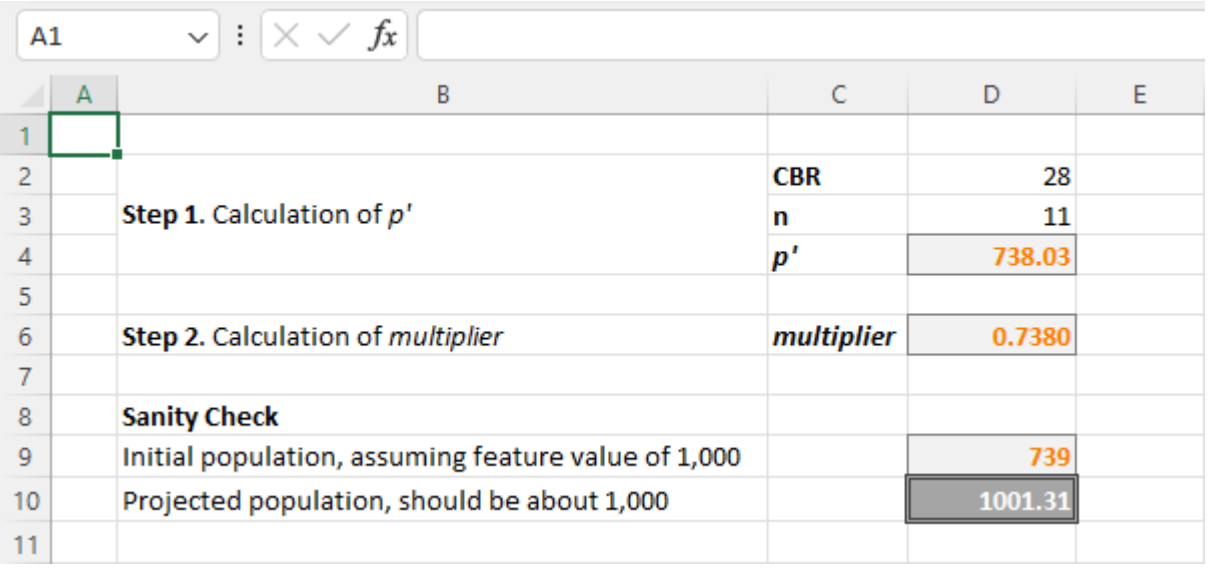
When preparing the $Pf\text{PR}_{2\text{-}10}$ raster, the data from the Malaria Atlas Project (MAP) is considered to be the canonical source; however, there are two things to be aware of when working with data from MAP. First, is that the prevalence raster is based upon a Bayesian space-time geostatistical model that predicts the $Pf\text{PR}_{2\text{-}10}$ for each pixel in the raster (Weiss et al. 2019). As such, while MAP is considered the canonical source for prevalence rasters, it should not be treated as the *definitive* source of data as there may be a lag between new data publication and regeneration of the prevalence rasters. When working with MAP data be sure to note the version number associated with the raster since the underlying data likely will change over time along with the algorithm.

The second thing to keep in mind is that the national boundaries for clipping used by MAP may not be in agreement with other data sources (see Figures 22 and 23). The underlying cause of this is the treatment of disputed boundaries and this can be a politically sensitive issue. In the case of the Kenyan data we are using as an illustrative example for this chapter, since the northern border contains the Ilemi Triangle disputed area with the territory claimed by both Kenya and South Sudan (Collins 2004).[14]

[12] For Kenya we are assuming a crude birth rate of 28 per 1,000 inhabited based upon data in Statista

[13] Typically this will be able eleven years to account for model burn-in.

[14] At the time of writing Kenya appears to have *de facto* control of the region.



	A	B	C	D	E
1					
2			**CBR**	28	
3		**Step 1.** Calculation of *p'*	**n**	11	
4			***p'***	738.03	
5					
6		**Step 2.** Calculation of *multiplier*	***multiplier***	0.7380	
7					
8		**Sanity Check**			
9		Initial population, assuming feature value of 1,000		739	
10		Projected population, should be about 1,000		1001.31	
11					

Figure 20: Example of the population calculation in Excel.

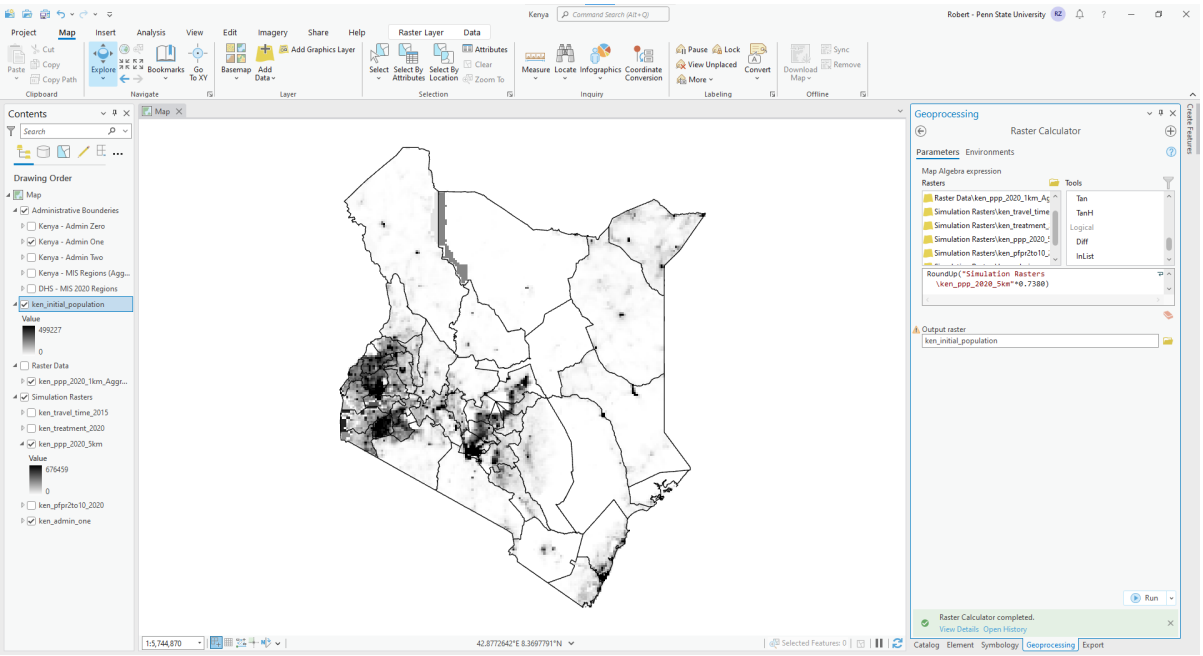


Figure 21: The intial popuation raster produced by the 'Raster Calculator' with updated symbology. Note the highest cell value is lower than in the orginal 2020 population raster..

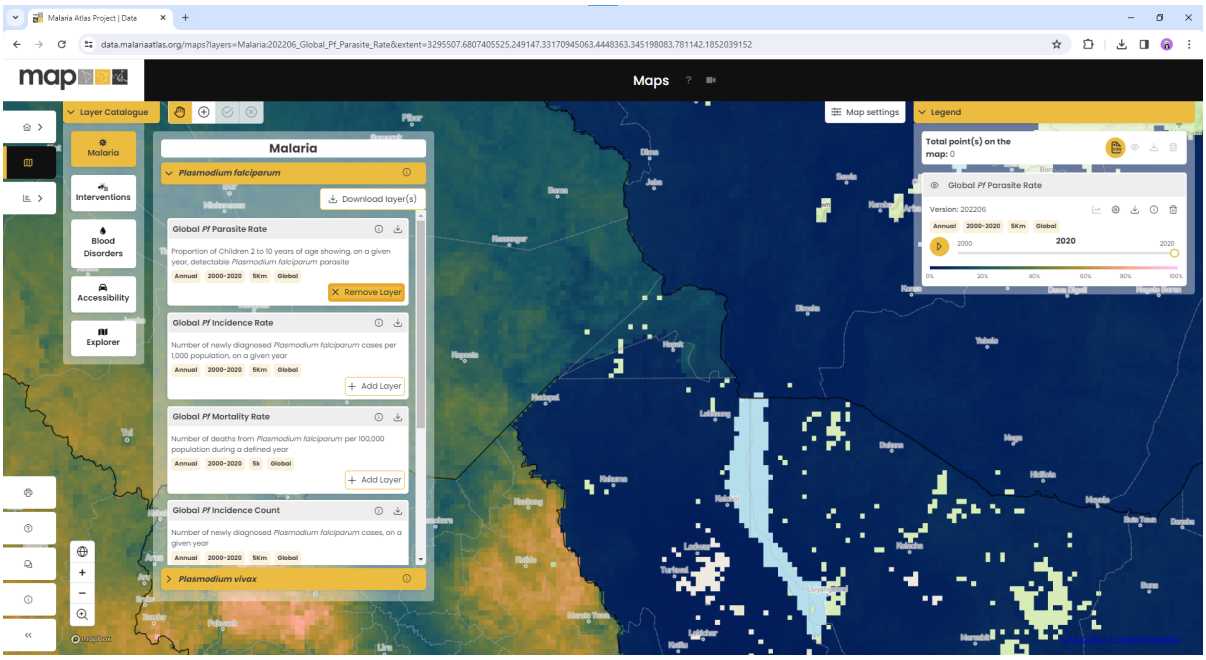


Figure 22: The MAP Global *P. falciparum* Rate zoomed to the area of the disputed border between Kenya and South Sudan. Note the dotted gray lines indicating the disputed boundry and the sudden shift in prevelence.

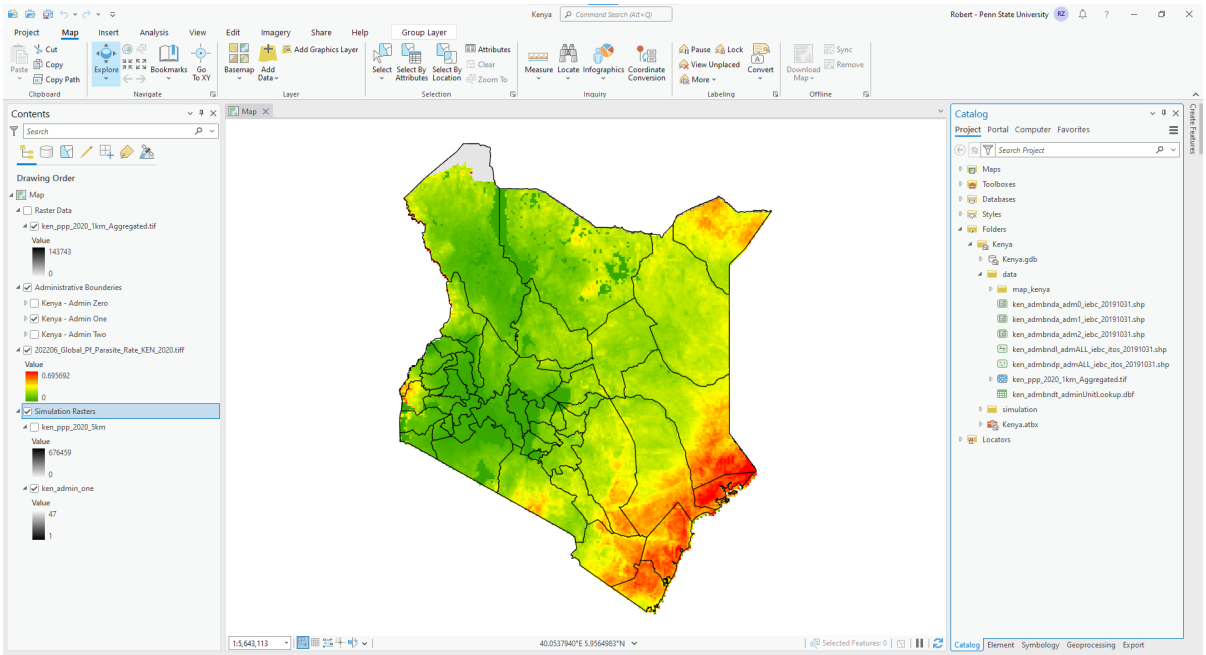


Figure 23: The *Pf*PR 2-10 data from map, downloaded via the "Clip to Country" feature with symbology adjusted in ArcGIS. Note the admin one raster is visable in the north due to the cliping boundry being smaller than boundry defined by the administrative shape boundry.

Due to the discrepancy between the clipped raster from MAP and the admin one raster, we are going to produce our own $Pf\text{PR}_{2\text{-}10}$ raster from MAP's global raster. Start by downloading the data from MAP, and decompress the data into a convent location (ex., a sub-directory of the `data` directory created earlier; see Figure 24).[15] Select the raster that is appropriate for the project – typically the most recent year of production – and drag it to the map. When prompted to calculate statistics, click "Yes". Your map should now appear similar to Figure 25.

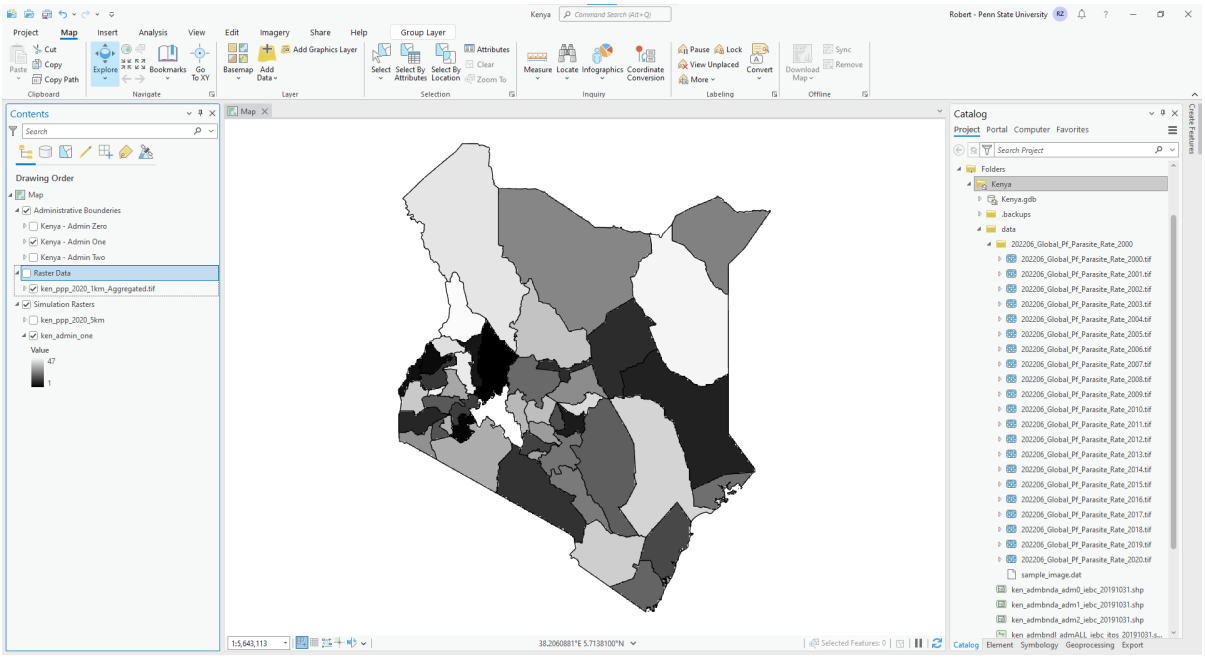


Figure 24: Contents of the MAP global rasters directory in ArcGIS.

Since raster data from MAP is distributed with metadata, ArcGIS should recognize the raster as having a datum of WGS 1984 with an angular unit of degrees. We will start by using `Project Raster` to create a temporary raster using the projection that was selected for the country. Make sure that the `X` and `Y` values for the `Cell Size` are set to "5000" and "5000". While this will result in the loss of some data due to re-sampling; however, the impact will be nominal since spatial correlation will still apply.

Next, in the `Geoprocessing` tab, type "extract by mask" to search for the `Extract by Mask (Spatial Analyst Tools)`, click the result to load the tool's controls. Enter the following:

1. `Input raster` is the projected raster we just created
2. `Input raster or feature mask data` is the raster that will be used to extract data from the map, select the admin one raster
3. Enter the preferred final name and location for the `Output raster`, here `ken_pfpr2to10_2020` is used to indicate this is the $Pf\text{PR}_{2\text{-}10}$ for Kenya in the year 2020
4. Make sure `Inside` is selected for `Extraction Area`
5. Select the `Environments Tab`
6. Under `Output Coordinate System` select `Current Map` to load the map's projection
7. Under `Snap Raster` select the admin one raster
8. Click `Run` to execute the tool

Once complete you should see something similar to Figure 26. At this point you can delete the global raster that was created by projection, and high or remove the original MAP raster. It is also recommended to

[15] Be aware that these will be larger files. At the time of writing the compressed file was 189 MB with the compression acting more as a convenience for bundling the 21 TIFF files additional metadata files.

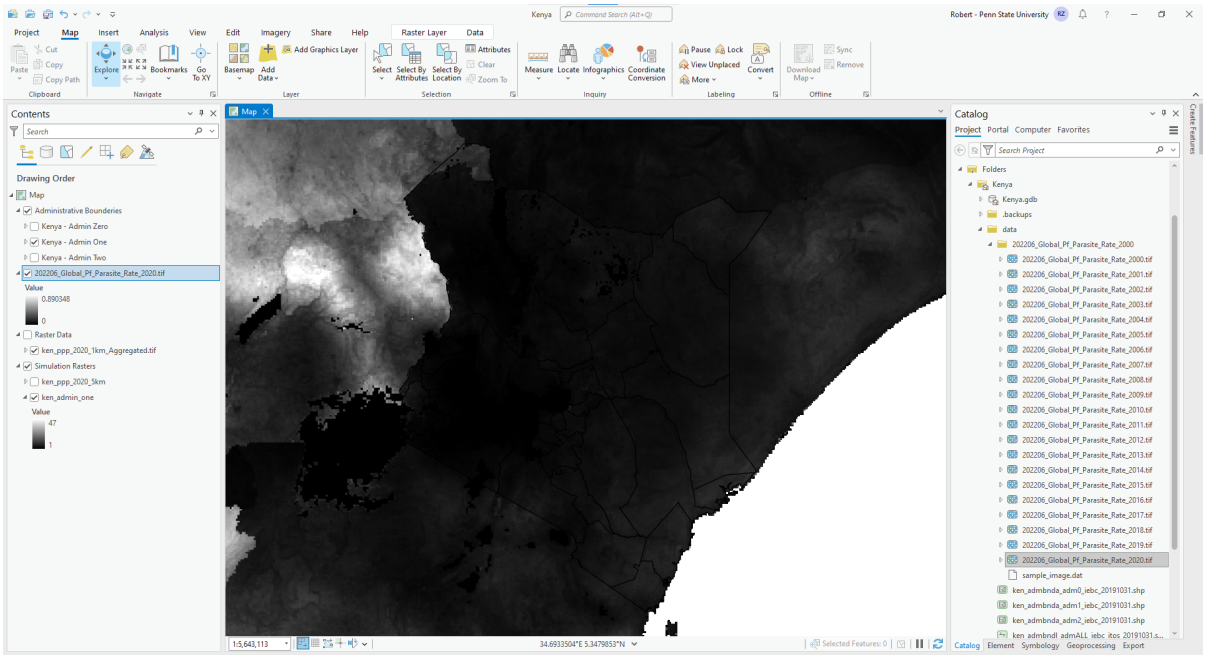


Figure 25: Apperance of ArcGIS after the global raster has been loaded, note the use of the default color scheme.

adjust the symbology of the $Pf\text{PR}_{2\text{-}10}$ raster as well as adding it to an appropriate group layer (see Figure 27)

3.5 Preparing the Treatment Coverage Raster

Now that we have the administrative regions, population, and $Pf\text{PR}_{2\text{-}10}$ rasters prepared, the last required raster for the simulation is the under-5 and over-5 treatment seeking behavior. The typical source for this data is the Malaria Indicator Survey (MIS) from the Demographic and Health Surveys (DHS) Program, run by USAID.[16] In this example we will be using the 2020 MIS data for Kenya (Division of National Malaria Programme (DNMP) [Kenya] and ICF 2021).[17]

Typically, the MIS will follow the administrative boundaries of a country, but it's not also uncommon for boundaries to be re-drawn for a survey, as was the case for the Kenyan 2020 MIS (see Figure 28). As such, the first step will be to acquire the survey boundary data via the DHS Spatial Data Repository.[18] Once the shapefile has been imported into ArcGIS it is important to assess the survey boundaries and note any discrepancies that are present. In the case of the Keyna MIS 2020 boundaries we can see that the Ilemi Triangle disputed area is excluded along Lake Turkana and Lake Victoria.

In this case, visual inspection of the shapefiles suggests that the MIS boundaries are being drawn based upon the admin two boundaries. As such we can use the MIS boundaries to inform the selection of the appropriate admin two boundaries, followed by dissolving the data into a shapefile that can be edited and used for creation of raster containing the treatment seeking values. Let's start by first preparing the work space so that all of the MIS regions are correctly labeled and the layer is grouped with the other administrative boundaries. Once complete you should have something similar to Figure 31.

Visual inspection of the map suggests that the majority of the admin two districts have their centroid

[16] https://dhsprogram.com/

[17] Note that due to the survey process there is typically a lag between when the survey is conducted, when the results are released. At the time of writing, the 2020 MIS data is the most recent for Kenya.

[18] https://spatialdata.dhsprogram.com/home/

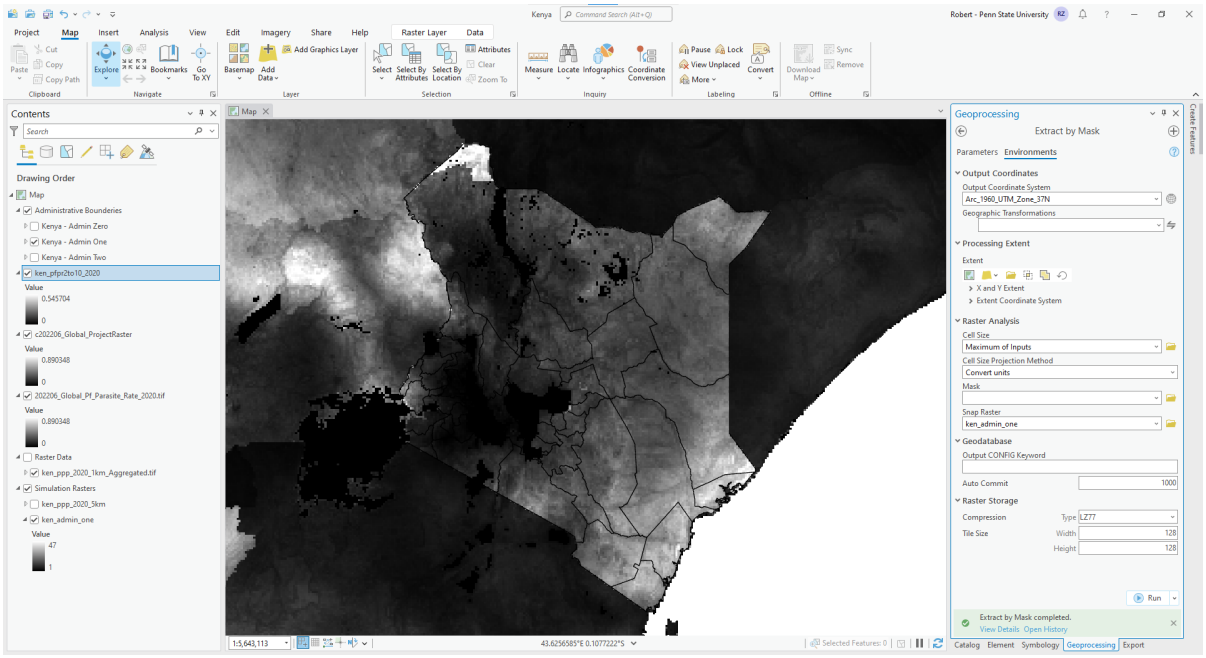


Figure 26: Apperance of ArcGIS after the data has been extracted, note that despite the default color scheme the new raster can be distinguished.

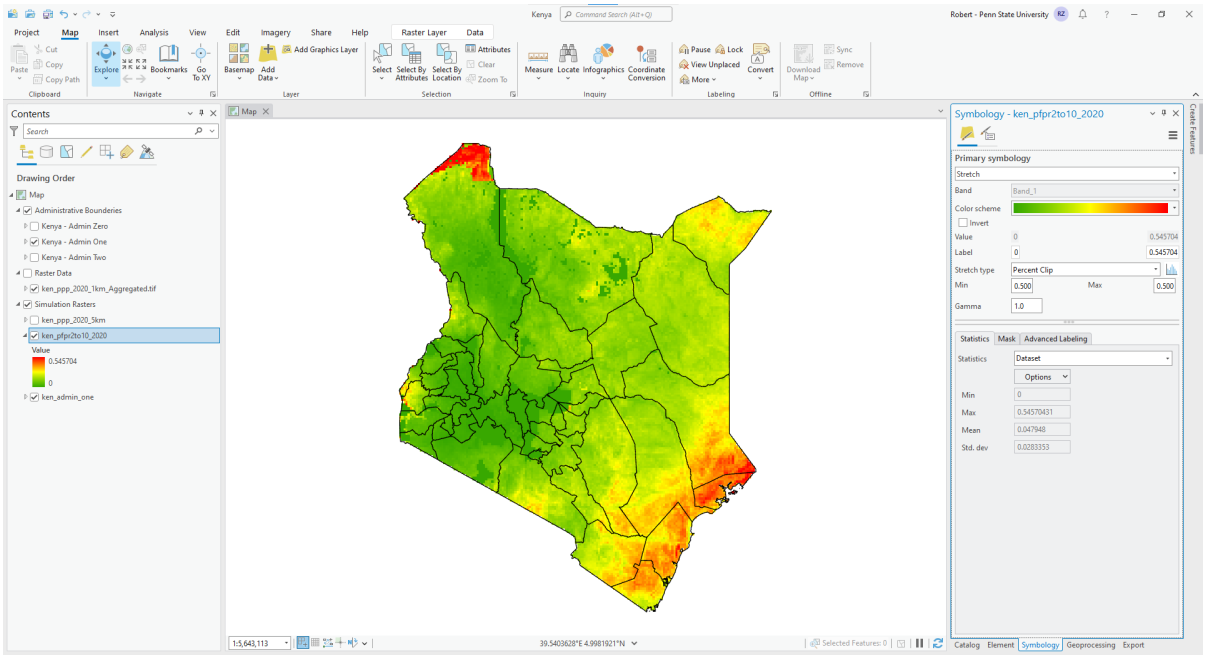


Figure 27: Final *P. falciparum* prevelence raster for Kenya after the symbology was adjusted and the workspace cleaned up. Note the higher projected prevelnce in the disputed Ilemi Triangle region in the north.

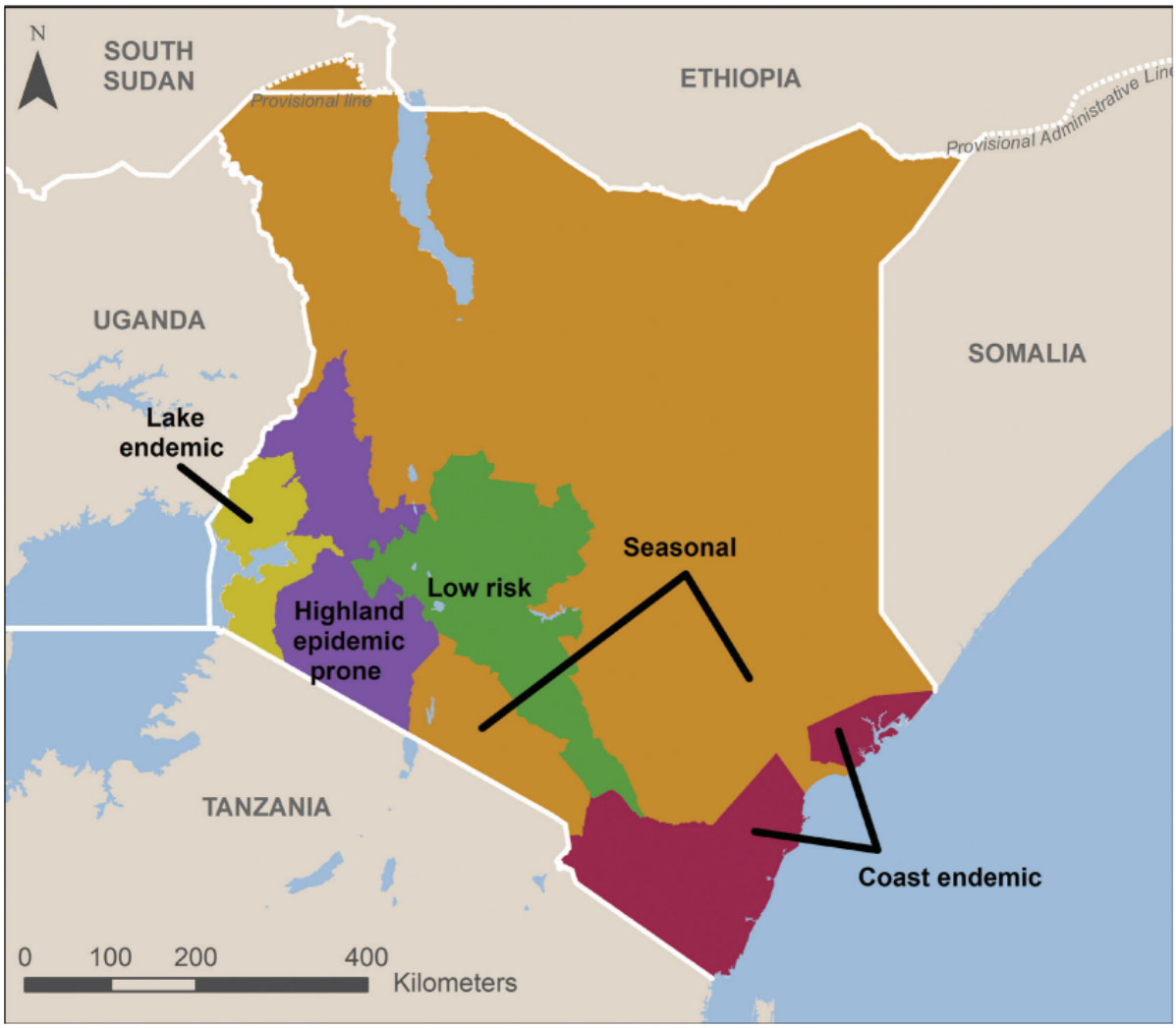


Figure 28: Comparison of the EIR versus the prevalence under different population sizes, note the inclusion of calibration points prepared by Malaria Modeling Consortium members (Reproduced from Division of National Malaria Programme (DNMP) [Kenya] and ICF (2021), p. xxii)

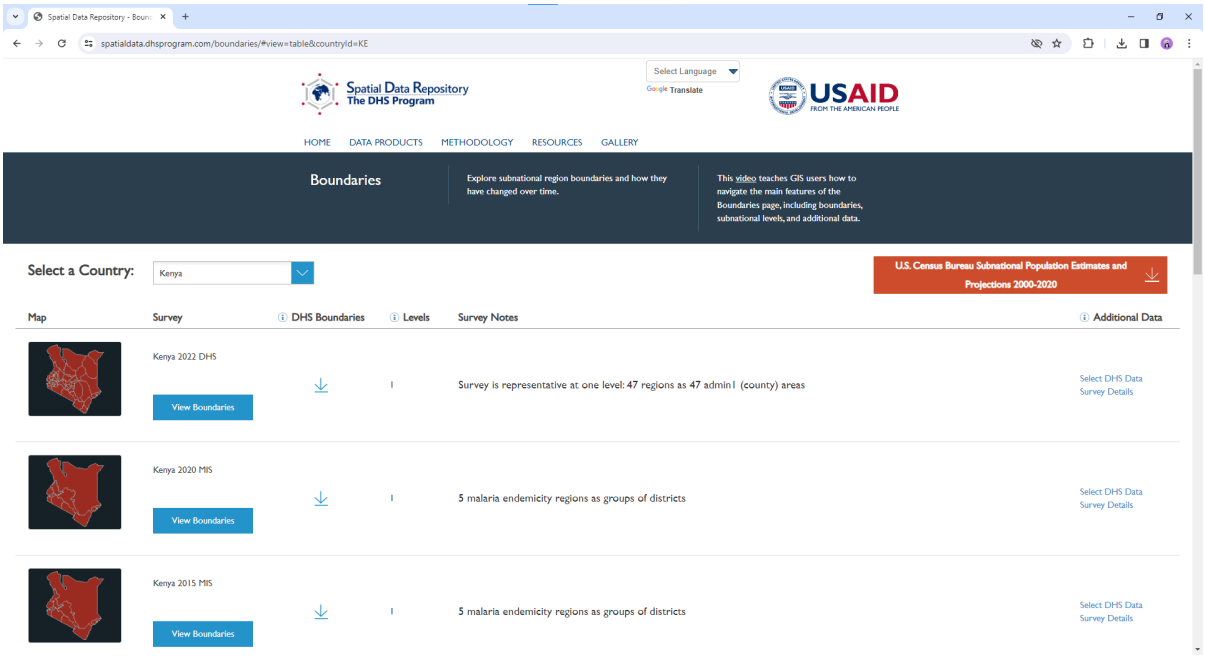


Figure 29: The DHS Program's Spatial Data Repositry showing the results of a search for Kenya

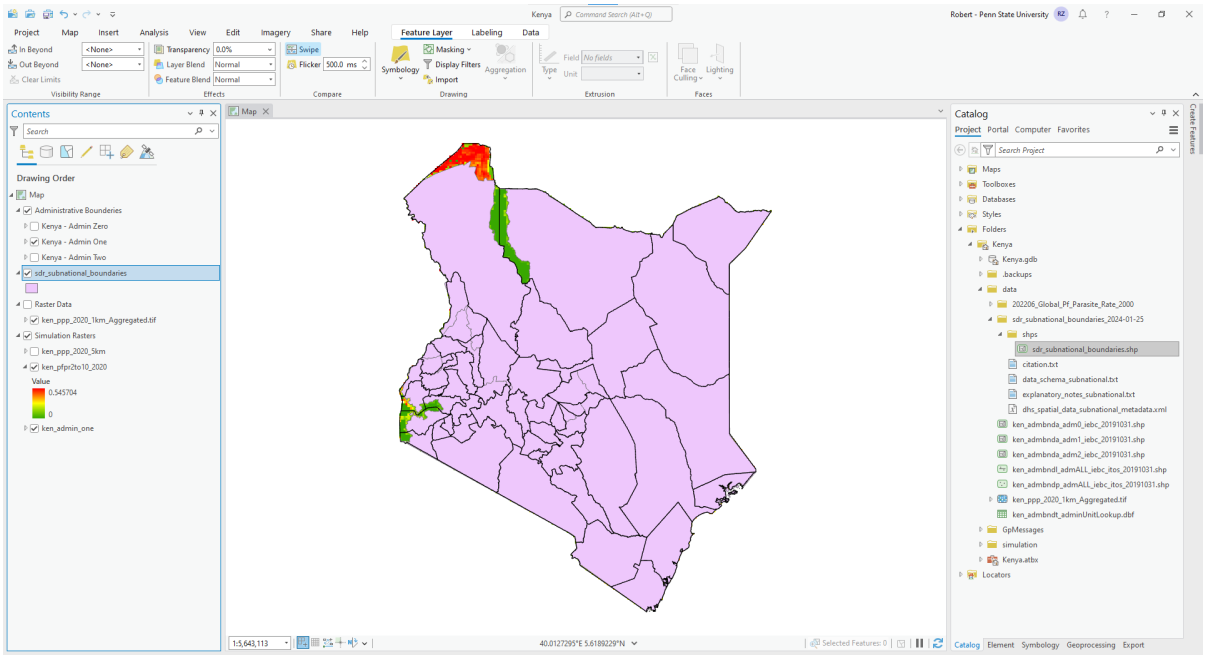


Figure 30: The 2020 MIS boundries imported into ArcGIS with the default symbology. Note the appearnace of the survey bounderies in gray while the admin one bounderies are in black. The overlay suggests that the survey was defined by a lower adminsitraive level. Also note that some reasons were excluded from the survey.

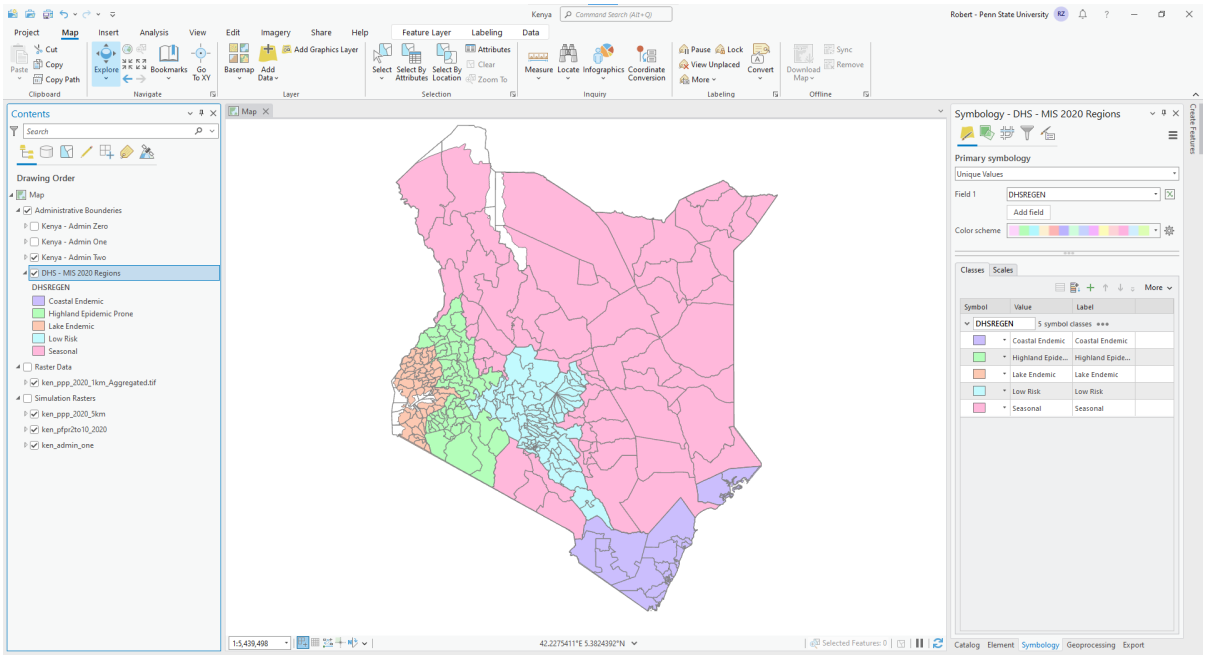


Figure 31: The workspace prepared for processing. Note the updated symbology and layer organization.

contained within the MIS regions; however, for the Lake Endemic region in the west, this may not be the case, so we will start there. However, since this workflow will modify the original admin two file, you may want to back-up the shapefile, or duplicate it with a new name.

Start by selecting the `Select By Attributes` tool under the `Map` on the ribbon menu, the `Select By Attributes` dialog will appear. Make sure the MIS regions shapefile is selected for `Input Rows`, select `DHSREGEN` for the start of the `Where` clause and then select `Lake Endemic` when the values appear.[19] Click `Apply` and the dialog will not close, but the regions will be selected (see Figure 32). This is so you can verify the query results, if the Lake Endemic region is highlighted, click `OK` to close the dialog.

Next, right click on the MIS region layer and select `Selection` then `Make Layer From Selected Features` from the sub-menu that appears. This will create a new *temporary* layer as shown in Figure 33. Here *temporary* means that the layer has not been saved as a new shapefile or in a geodatabase, but as part of the map itself. So if you close ArcGIS the layer will still be present. This is potentially a source of errors (or confusion) since any changes to the temporary layer will be applied to the original layer as well. Additionally, if you create a raster from a temporary layer it will preserve the extents of the original layer as `NODATA` as opposed to clipping the raster to the extents of the polygons. This will result in more rows or columns than expected.

Note that features are selected by ArcGIS, click the `Clear` button on the ribbon bar under `Map`, this will clear any selected features. Since some of the ArcGIS tools operate on selected features, it is important to make sure selections are cleared as soon as work is done with them.[20]

Now we are going to select the shapes from the admin two shapefile, start by toggling off the viability of the MIS regions. You should see the outlines of the admin two regions as well as the filled shape that contains the Lake Endemic region. Click the `Select By Location` under the `Map` tools in the top ribbon bar, a new dialog should appear. Under `Input Features` select the admin two layer, and under `Selecting Features` select the temporary layer that was just created.[21] For the `Relationship` select `Have their center in`

[19] As you might suspect, functionally what ArcGIS is doing is preparing a SQL query against the data set.

[20] As you get more comfortable with ArcGIS this is actually a very useful feature since you technically do not need to create a temporary layer to update the labels. However, since different symbology can be applied to temporary layers, they are easier to work with when visually assessing data.

[21] Typically ArcGIS will place it at the top of the other layers and append `selection` to the name.

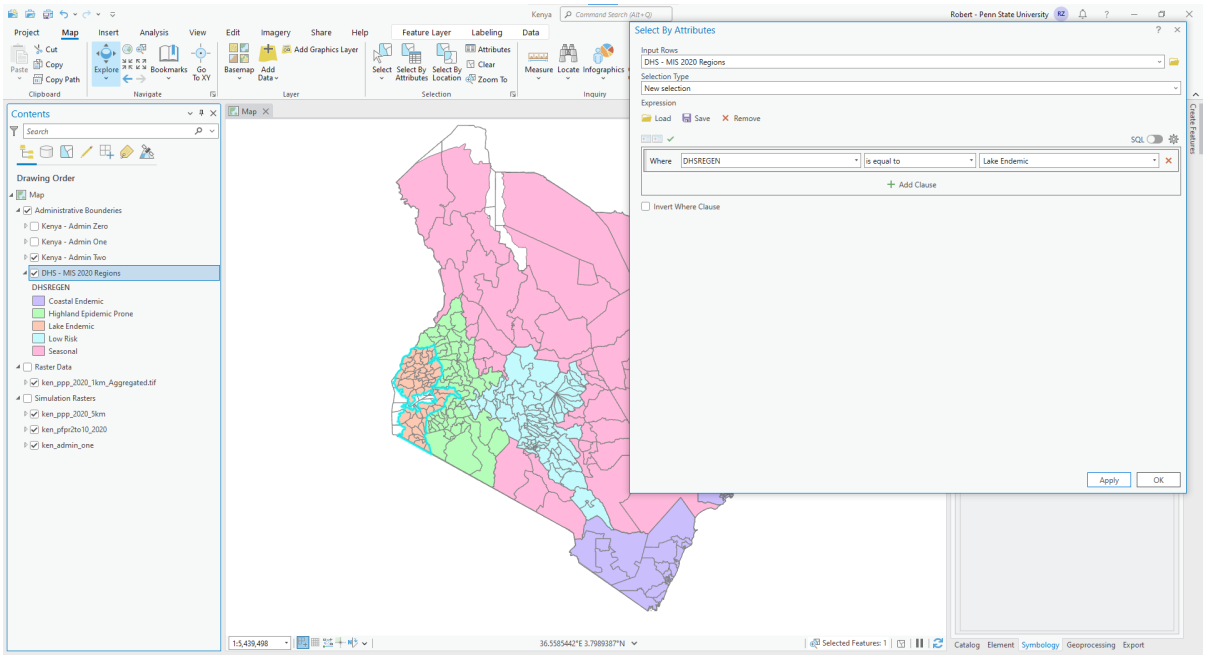


Figure 32: Previewing the selection of the Lake Endemic features before closing the Select By Attributes dialog.

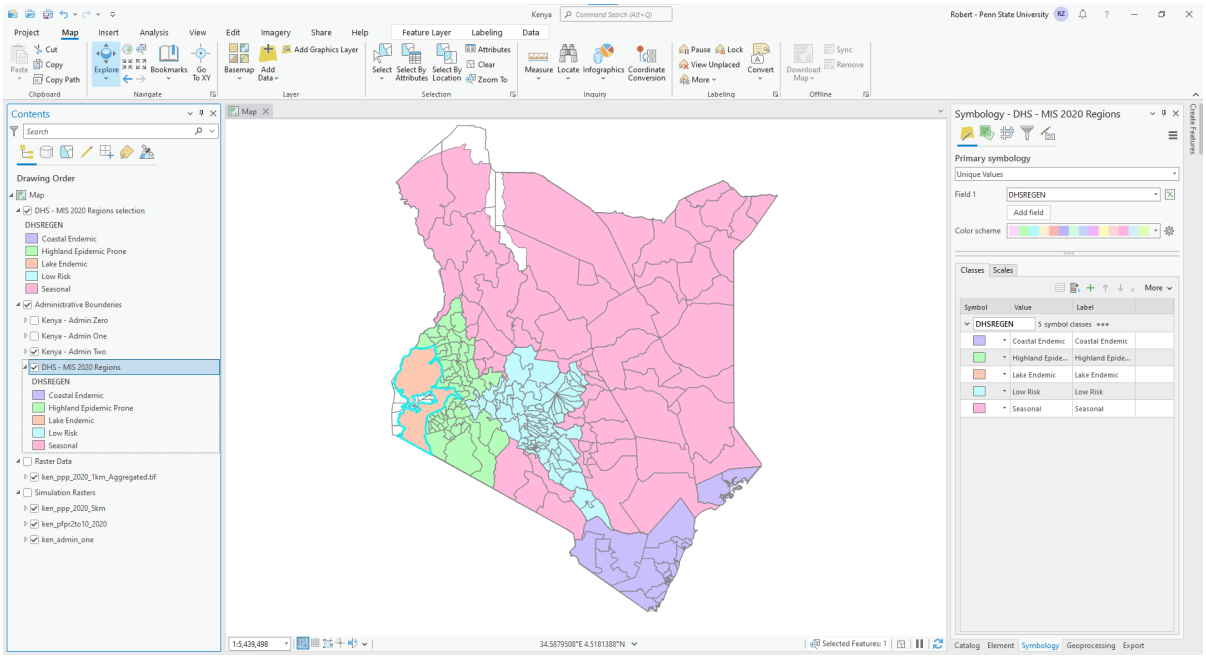


Figure 33: The temporary layer created after creating a layer from the selected features.

and click `Apply` to preview the selection (see Figure 34).[22] If the selection was successful, click `OK` to close the dialog.

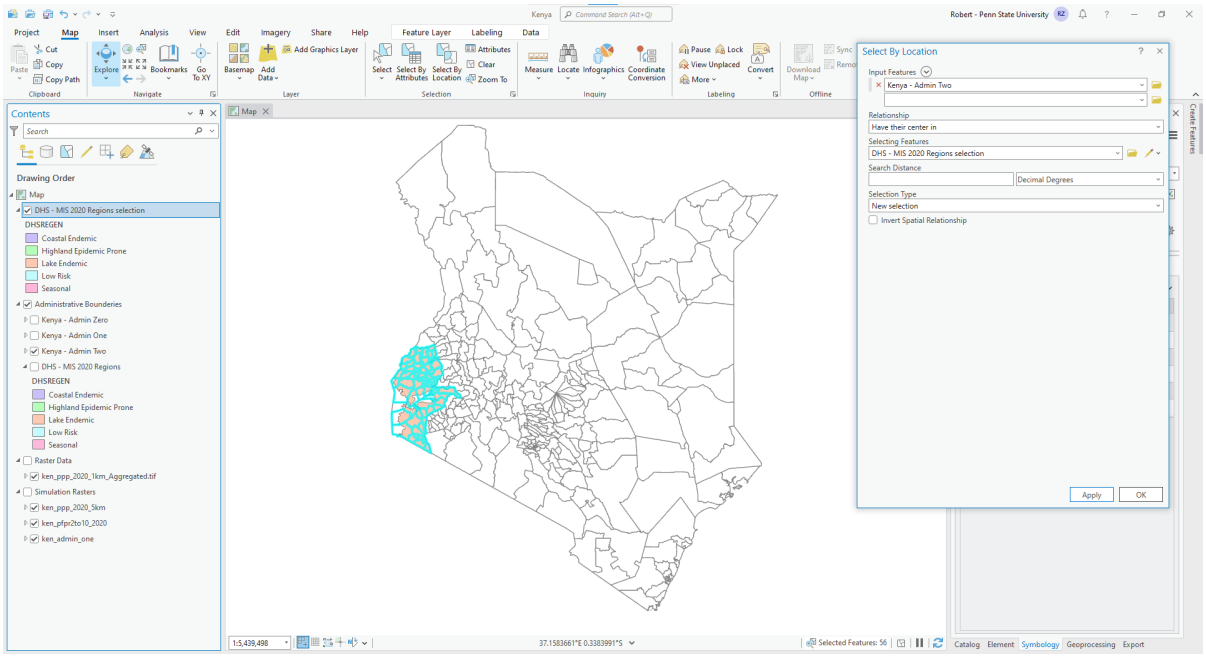


Figure 34: Previewing the selection fo the admin two regions based upon the Lake Endemic region.

Zooming in on the selected region find that one admin two region is select unselected. Select the region by first clicking the `Select` button in the ribbon menu under `Map`, next hold `[SHIFT]` and click in the two unselected regions.[23] If successful the region should look like Figure 35. Right click the admin two layer and select `Selection` then `Make Layer From Selected Features` to create a new temporary layer from the selected features. Clear any selected features, update the symbology, and rename the temporary layer to help in keeping track of it (e.g., "Admin Two - Lake Endemic").

We now have a temporary layer that draws from the admin two regions, but contains all of the MIS Lake Endemic regions. However, since we will ultimately want a single shapefile, now is a good time to add a new column to the shapefile which will be used to distinguish the regions. Open the attribute table, and select `Add` to create a new field. In the resulting dialog enter the new field name (e.g., "DHSREGEN" for DHS region") and set the `Data Type` to `Text` (see Figure 36). Click `Save` to save the new field and close the `Fields Editor`. Scroll the Attribute Table to the left until you find the new column. Right click the field name and select `Calculate Field`, this will open the `Calculate Field` dialog which will allow us to apply a label to all regions at once. Make sure the `Input Table` and `Field Name` are correct and in the text box below `DHSREGEN =` – assuming you entered "DHSREGEN", otherwise it will be the field name you used – enter `"Lake Endemic"` (with quotes). Click `Apply` to apply the change without closing the dialog, you should see `Lake Endemic` for all rows in the column (see Figure 37), click `OK` to close the dialog. Remember that the temporary layer is a subset of the original layer, so we just added the new column and updated the rows in that layer as well.

Barring adding a new column, this workflow can be applied to create and label temporary layers for the remaining regions. Once you are complete you should have a map that is similar to Figure 38. The next step is to aggregate the labeled admin two regions into single shapes, which can be done by dissolving the smaller shapes. Under the `Geoprocessing` tab search for "dissolve" and select the `Dissolve` tool. In the

[22] While there are several other selections available, the `Have their center in` bases the selection on if the centroid is in the selecting feature or not. The default `Intersect` option will select not only the shapes in the the selecting feature, but ones that share a border with the selecting feature.

[23] The second region is small and can be hard to spot, you may want to create a temporary layer and apply a fill to spot it.

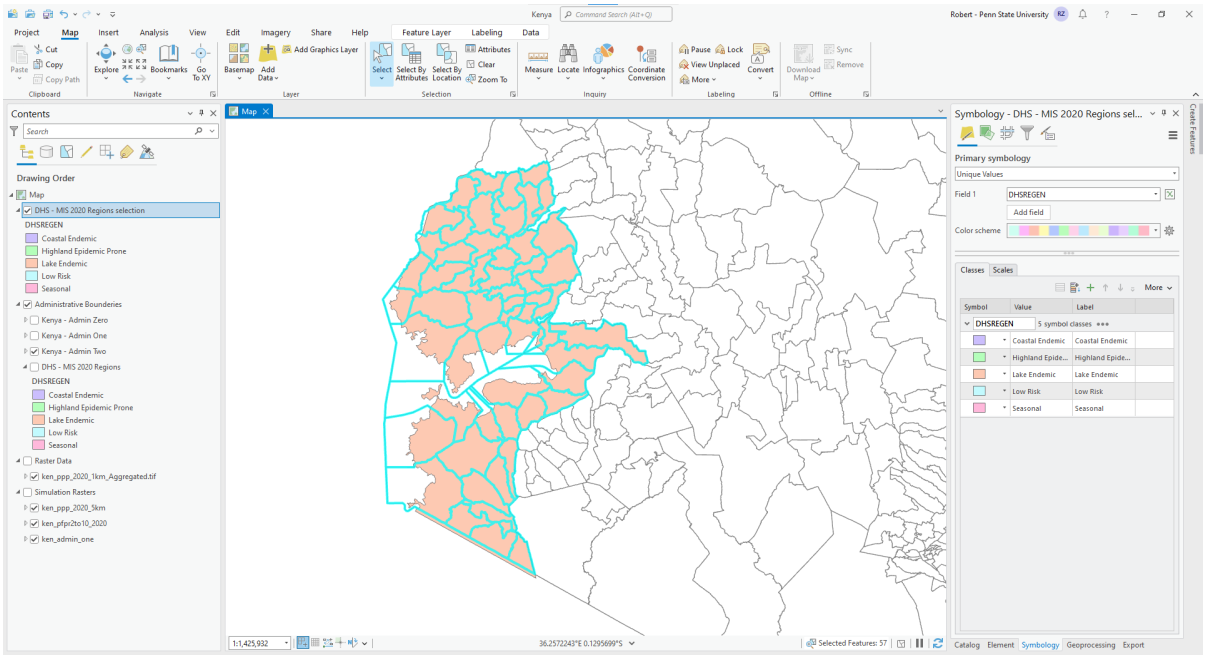


Figure 35: Selection of all admin two regions in the MIS Lake Endemic region.

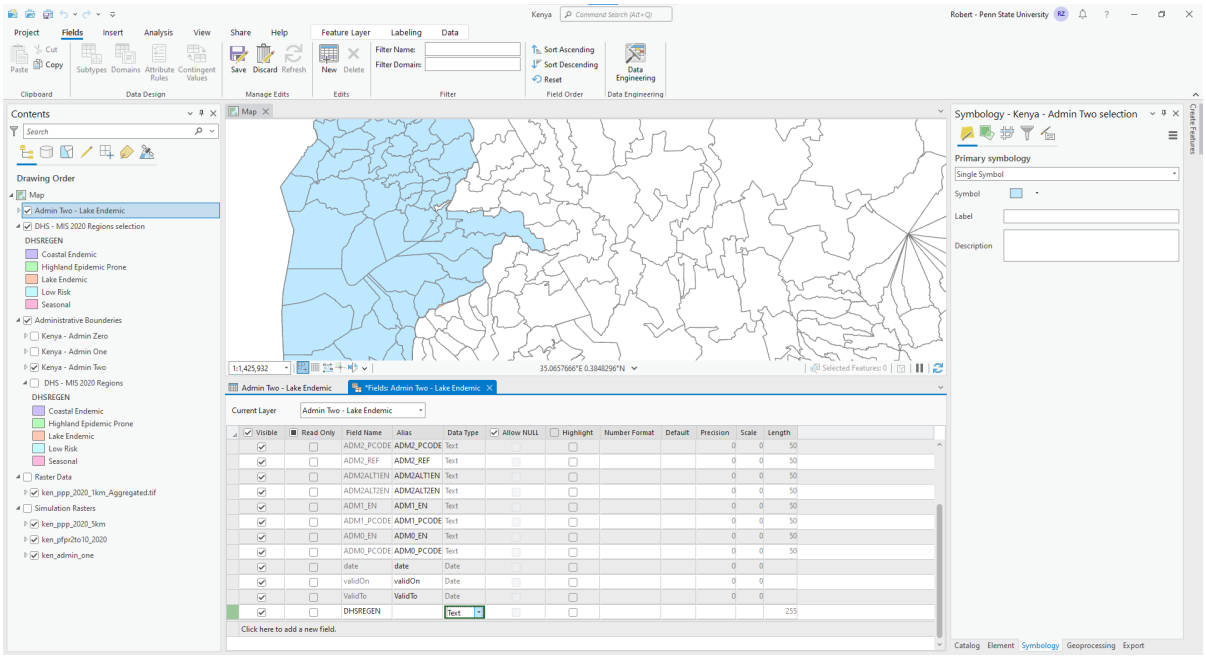


Figure 36: Adding the new column for to store the DHS region.

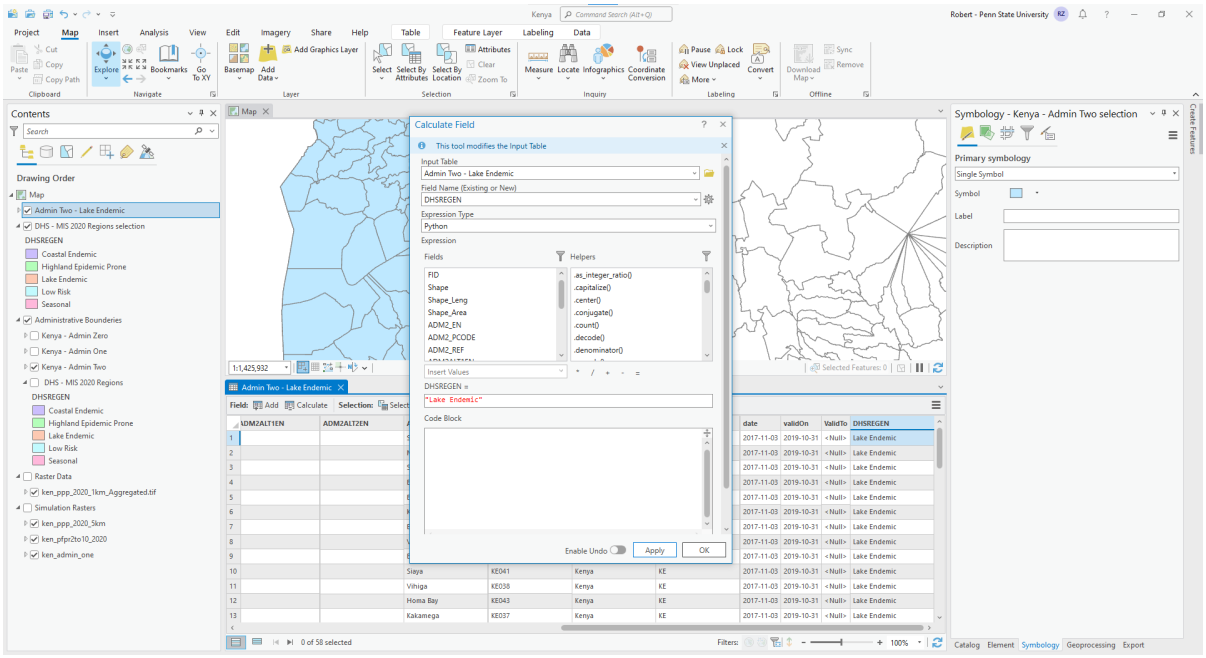


Figure 37: Result of using Calculate Field to apply the "Lake Endemic" label

controls that appear select layer that contains all of the labeled admin two regions (ex., “Kenya - Admin Two - MIS Regions”), for `Output Feature Class` select the location you want to save the data at and enter an appropriate name (ex., “ken_mis_regions_aggregated”). Under `Dissolve Fields` select the name of the field that contains the labels (ex., “DHSREGEN”) and on the `Enviornments` tab select `Current Map` for the `Output Coordinate System` and click `Run`.

At this point you should have a map that is similar to Figure 39. Take a moment to clean-up and reorganize your work space. Open the Attribute Table for the new layer, and you will note that there is now one row per region label despite some regions having multiple disconnected parts. This is due to the Dissolve tool creating multipart features, which is useful for the purposes of preparing data for the simulation since it reduces data entry steps. Create a new field called “Treatment” and make sure `Double` is selected as the `Data Type`, a new column will appear in the Attribute Table called “Treatment” and at this point you can directly edit the values like a spreadsheet. Once the values have been entered you should have a table similar to Figure 40, be sure to save via `Save` under `Edit` on the ribbon bar.

At this point the raster can be generated using the `Polygon to Raster` tool using the following settings:

1. `Input Features` the aggregated MIS region shapefile that was created.
2. `Value Field` is “Treatments”.
3. `Output Raster Dataset` is the preferred file name and location.
4. `Cellsize` is 5000 (in meters).
5. On the `Enviroments` tab the `Current Map` is selected for the `Output Coordinate System`.
6. The admin one raster is selected as the `Snap Raster`

Once complete the final raster should looks similar to Figure 41. In the event that a second treatment raster is needed for the over-5 group, it is recommended that a second value field be created (ex., “Over5Treatments”) and the original value field be renamed (ex., “Under5Treatments” to reflect the differences).

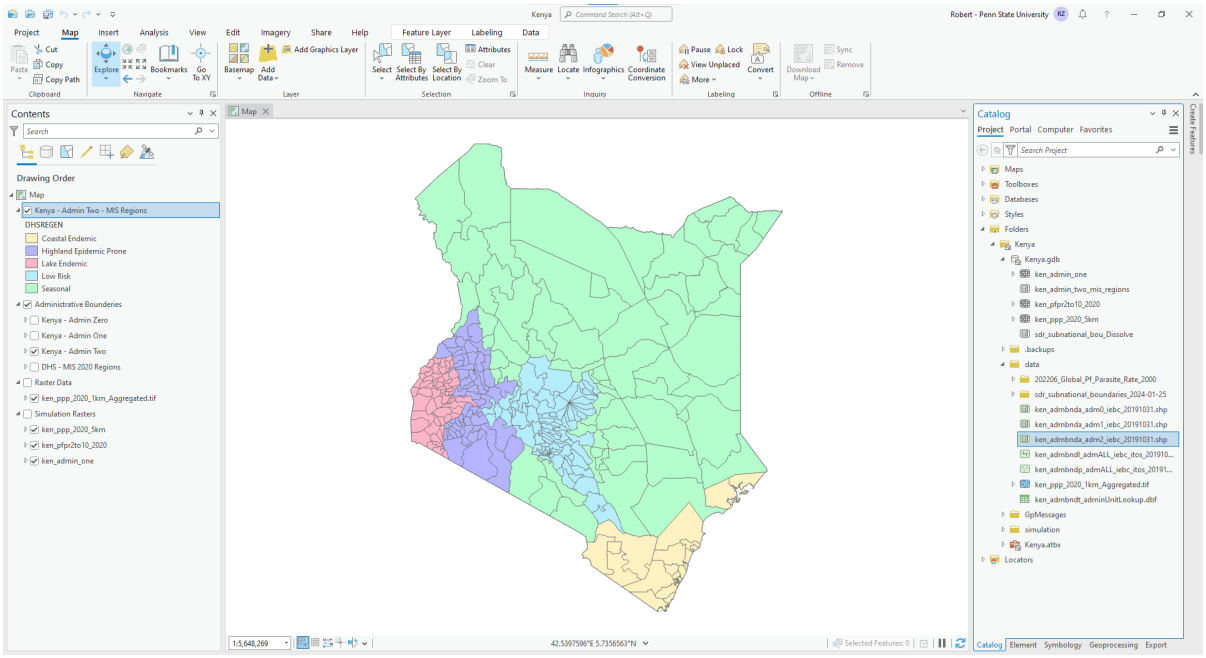


Figure 38: The result of the workflow to label admin two regions with the MIS survey region. Note the "ken_admin_two_mis_regions" shape data in the geodatabase, which is the duplicated copy of original admin two shapefile.

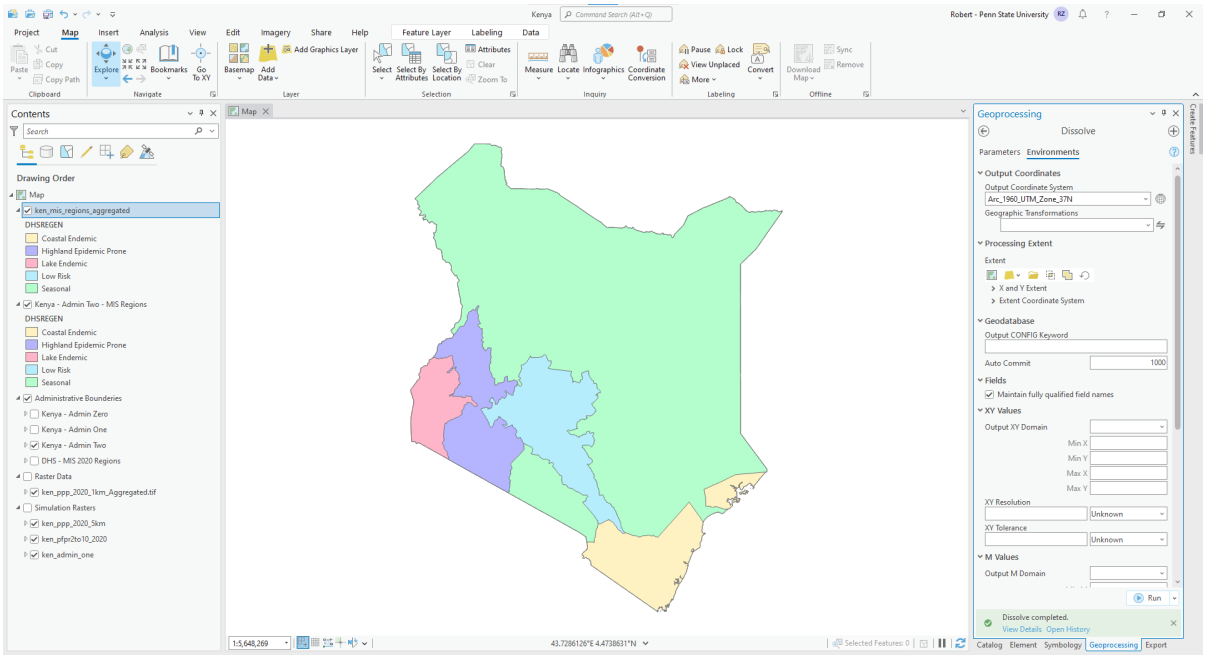


Figure 39: Apperance of the map following the process of labeling the admin two regions and disolving them into aggregated MIS regions.

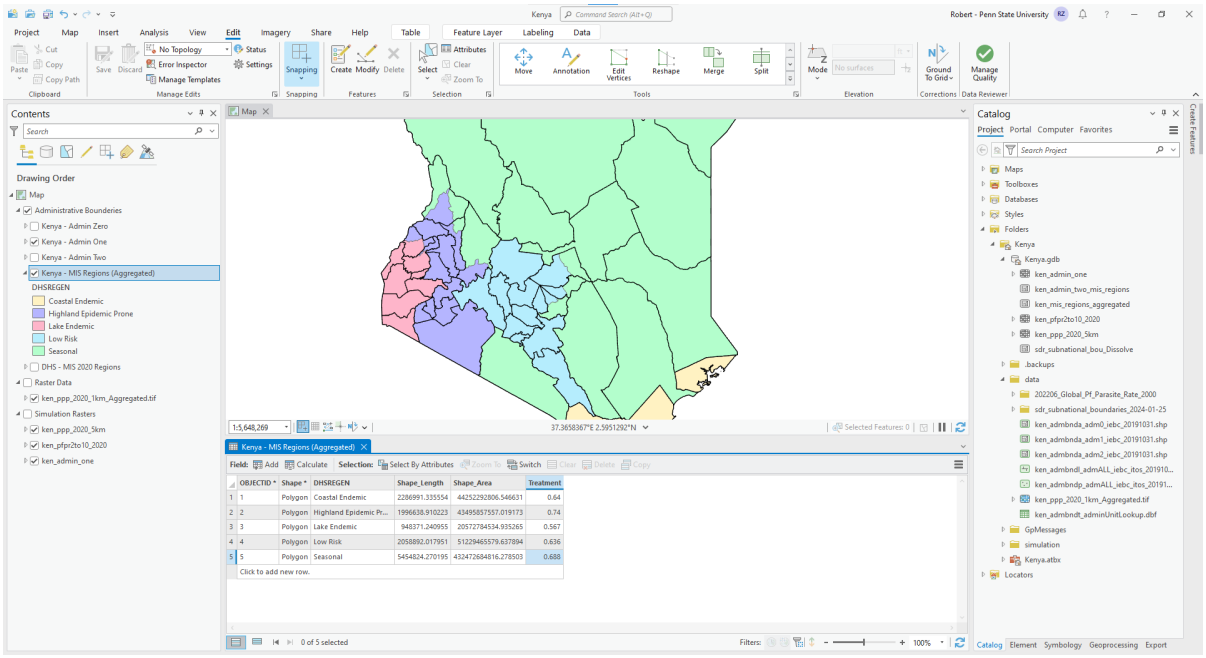


Figure 40: Adding the treatment seeking rate via the Attribute Table, note the percentages are entered as values from zero to one. (Under-5 treatment seeking from Division of National Malaria Programme (DNMP) [Kenya] and ICF (2021), p. 62)

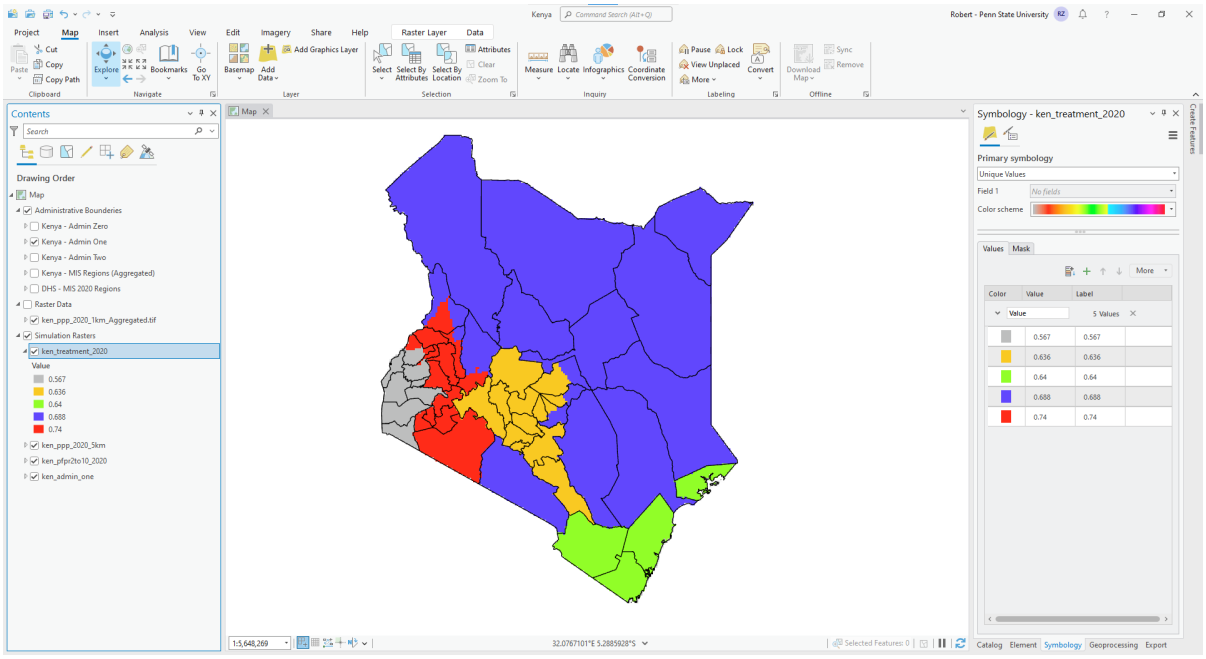


Figure 41: The final treatment seeking raster with the admin one border overlaid. Note that the symbology has been updated to unqiue values, with only the five entered under-5 treatment seeking rates present.

3.6 Preparing the Travel Surface

Production of a travel surface – also referred to as a friction surface – is optional and dependent upon the movement model selected for a country in the simulation. However, in the case of our example of Kenya, the size of the country is large enough that accounting for travel time is likely to be a necessary part of the country's calibration.

In addition to *Pf*PR$_{2\text{-}10}$ rasters, the Malaria Atlas Project (MAP) also produces three travel time surfaces that can be used with the simulation:

1. Global Motorized Travel Time to Healthcare
2. Global Walking Only Travel Time To Healthcare
3. Global Travel Time to Cities

Each of these travel time rasters serves a different purpose, and presently the generalized movement model in the simulation that supports a travel surface (i.e., `WesolowskiSurface`) works best with the travel time to cities raster.

Since the travel time raster is produced by MAP, using their option to "Clip to Country" is going to produce the same raster extent as the *Pf*PR$_{2\text{-}10}$ raster. In the case of Kenya this means that the Ilemi Triangle will not be included due to being disputed territory.

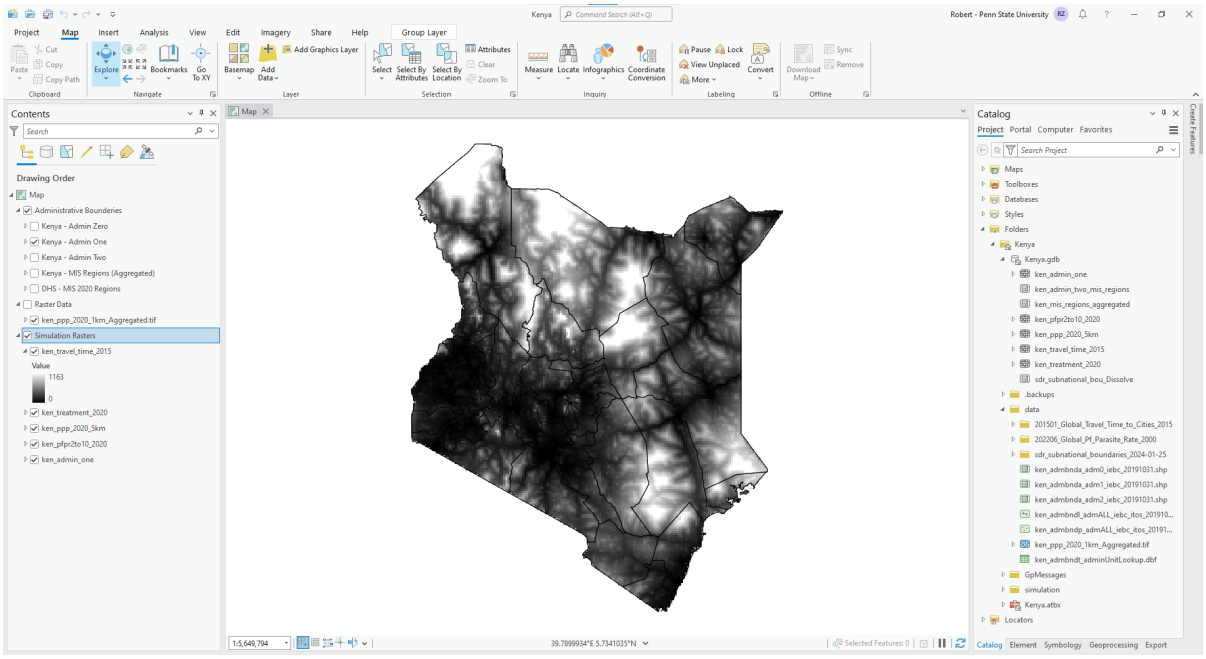


Figure 42: The final treatment seeking raster with the admin one border overlaid. Note that the symbology has been updated to unqiue values, with only the five entered under-5 treatment seeking rates present.

The workflow to prepare the raster is the same as the $Pf\mathrm{PR}_{2\text{-}10}$ raster:

1. Start by downloading the global travel time data, unzip it to a convenient location for the project.
2. Add the global travel time raster to the map, be sure to build pyramids and calculate statistics.
3. Use `Project Raster` to project the raster to the same projection as the map with the X, Y `Cell Size` both set to 5000 using `Majority resampling` for the `Resampling Technique`.
4. Use `Extract by Mask` tool, using the admin one raster as the mask and snap raster, the output coordinate system should be the same as the map.

After cleaning up the work space you should have travel time raster that appears similar to Figure 42.

While this is a streamlined process, the underlying resolution of the travel time raster is similar to the population raster from WorldPop. As such, this will result in considerable data loss in the process of going from 30 arc seconds to 5,000 meters. If data accuracy is necessary then the workflow can be adjusted to following one similar to creating the population raster – although the mean or median should be taken as part of the aggregation as opposed to sum – however, this should be contextualized against the 25 km^2 that each cell in the raster will represent. While a more elaborate workflow based upon aggregation as opposed to resampling may be more accurate in a technical sense, the difference will not have a significant impact upon the model fidelity due to the distance involved in crossing a single cell.[24]

3.7 Exporting the ASC Files

By this point, all of the rasters necessary for model calibration should have been completed; however, the rasters are currently in a format that is not supported by the simulation (i.e., GeoTIFF). The next step is going to export the rasters to a ASCII text coded raster (i.e., an ASC file, Esri ASCII rater, or Esri grid) that can be loaded into the simulation.[25] At a minimum the following rasters be exported from ArcGIS for the simulation:

1. The *initial population raster*, which will be used to create the initial population when the simulation starts.
2. The *administrative districts raster*, which will be used to aggregate cell level data to a district.
3. The *under-5 treatment seeking raster*, which will be used to determine the likelihood that an individual under-5 will seek/receive treatment.

The other mandatory raster for the simulation – the beta's surface – is created by the workflow outlined in the chapter on Model Calibration. However, the following rasters may need to be exported from ArcGIS depending on the circumstances:

4. The *over-5 treatment seeking raster*, which will be used to set a different likelihood of seeking/receiving treatment for individuals over-5.
5. The *travel surface raster*, which will be used with the movement model to control how individuals move around the simulated landscape.
6. The *climate zone raster*, which is used to determine which climate zone a cell is within, the production of which is tied to the production of climatic data for the simulation.

[24] The average walking speed of adults is about 5 km/h, while a motorized vehicle on roads (or good terrain) will be quicker, the key point here is that in practical terms the cells are actually quite large!

[25] Documentation on the formation can be found in the ArcGIS Manual or in Wikipedia. As to why the simulation uses the Esri grid format as opposed to a binary raster, the reasons are two fold. First, the Esri grid files are human readable which allows for manual edits to be performed if necessary without the need for additional software, so simple simulations can be manually created in a purely text-based environment which is useful for scientific computing where being connected via SSH to a remote system is still likely. Second, during development of the simulation's geospatial codebase, a review of use cases was conducted and it was deemed unlikely that simulated area would be so big as to render the Esri grid files unmanageable.

In this case, all of the rasters that have been created thus far will be exported. It is recommended by first ensuring that there is a folder in the relevant country repository (e.g., `/GIS/` that will contain the files that are created.[26] Next, the `Raster to ASCII` tool is run in ArcGIS for each raster file, with the results written to the folder that was created, using the the `*.ASC` output format. Once complete the output will look similar to Figure 43

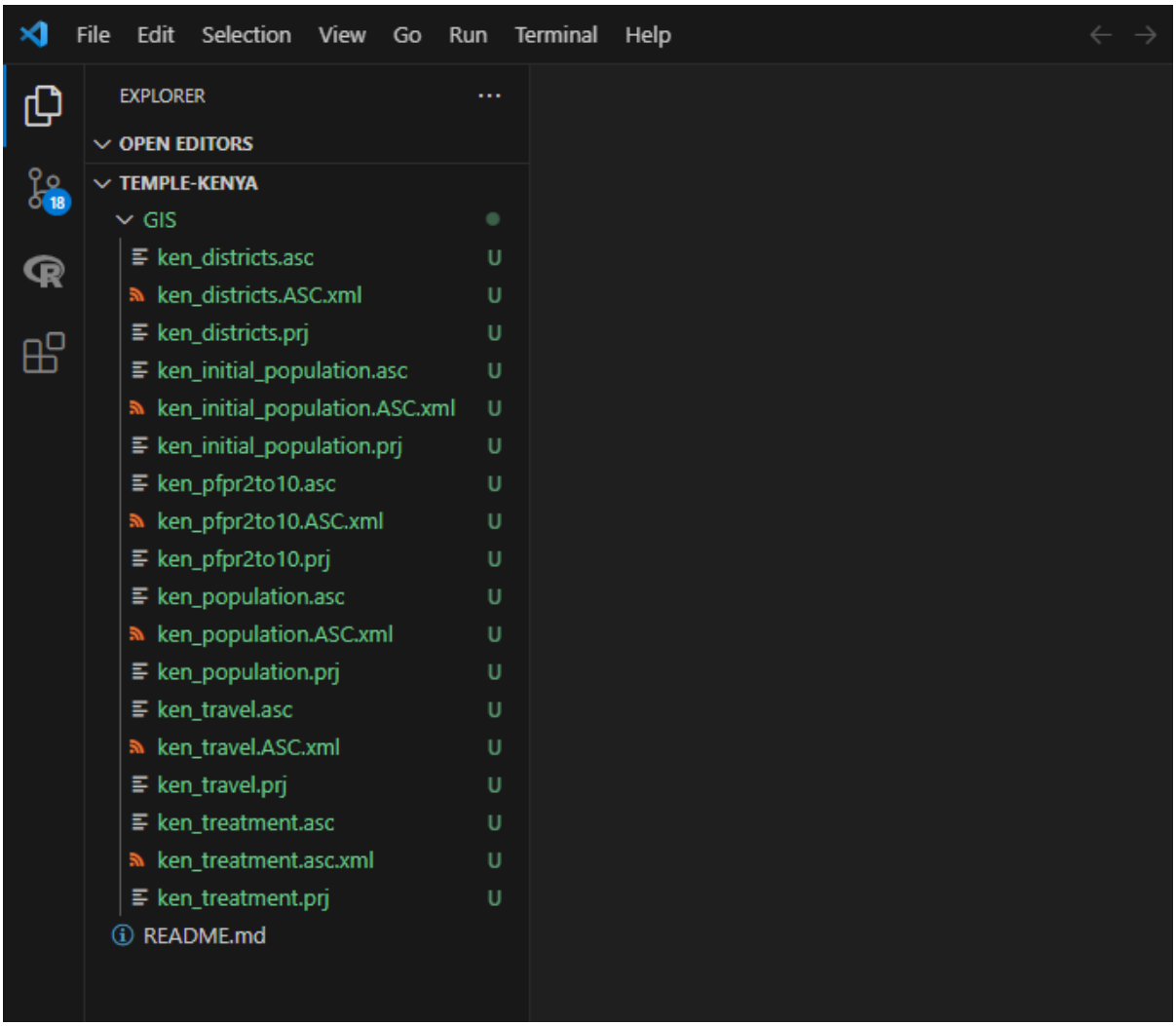


Figure 43: The contents of the GIS directory in VS Code following exporting in ArcGIS. Note the presence of projection files (*.prj) and ArcGIS data files (*.xml).

Exportation via the `Raster to ASCII` tool results in the creation of projection files (*.prj) as well as ArcGIS data files (*.xml) in the directory. It is recommended that the projection files be retained for reference, while the ArcGIS data files be deleted since they will not be used. Once all ASC files have been exported, run the `validateraster` tool from `PSU-CIDD-MaSim-Support` to check that all of the rasters have been correctly aligned. In this case, we find that there there is some misalignment in the files created for Kenya:

[26] While best practices for reproducible science would call for *all* files to be stored in a repository, in practice the workflow has typically been for geospatial files to get added to the repository if they are necessary to run the simulation or produce data used in analysis. Effectively, if the geospatial is produced by a different organization it's included as a reference, whereas files produced by members of the lab are included as part of the data analysis.

```
Using ken_districts.asc as reference
178 != 179 ncols
224 != 228 nrows
-67203.471654228 != -72203.471654228 xllcorner
-518695.09422505 != -528695.09422505 yllcorner
Error with alignment between ./ken_districts.asc and ./ken_initial_population.asc
178 != 179 ncols
224 != 228 nrows
-67203.471654228 != -72203.471654228 xllcorner
-518695.09422505 != -528695.09422505 yllcorner
Error with alignment between ./ken_districts.asc and ./ken_population.asc
5 files crosschecked, 6 total files
```

This tells us that the something has gone wrong with the creation of the population and initial population rasters. Specifically, the two population rasters have more rows and columns than the other rasters. The next step is to isolate the likely cause of this problem, which can typically done by adjusting the symbology to show Nodata values, as shown in Figure 44.

Based upon the results reported by `validateraster` and visual examination of the rasters in ArcGIS, the suggestion is that at some point some extra data was produced for the population rasters. This is quite common and can be addressed by using `Extract by Mask` to create an updated raster based upon the smaller raster. In this case we will be using the population rasters as the `Input Raster` and the admin one raster as the `Input raster or feature mask data`. Pay close attention to ensure that the `Analysis Extent` is the same as the smaller raster (this can be updated by clicking the down arrow next to the `Extent of a Layer` icon and selecting the appropriate raster). If this is done correctly you should have a raster similar to Figure 45. Be sure to double check the number of rows and columns using the `Properties` dialog for the raster since color bleeding can result in it looking like the process has failed. The ultimate check some be the the number of rows and columns matching the smaller raster.

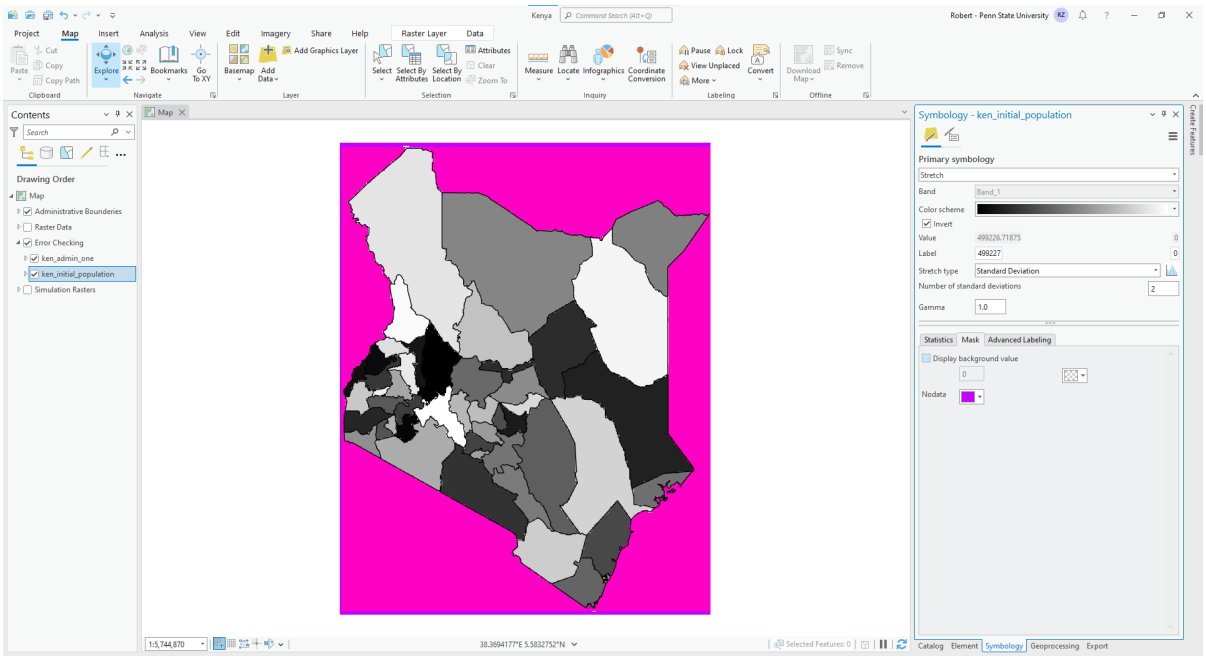


Figure 44: By overlaying the smaller districts raster on the larger initial popuation raster, with Nodata being displayed, the source of the extra rows at the top, bottom, and left side can be seen. Note that the underlying initial population raster also suggests the presence of data outside the bounds suggested by the district raster.

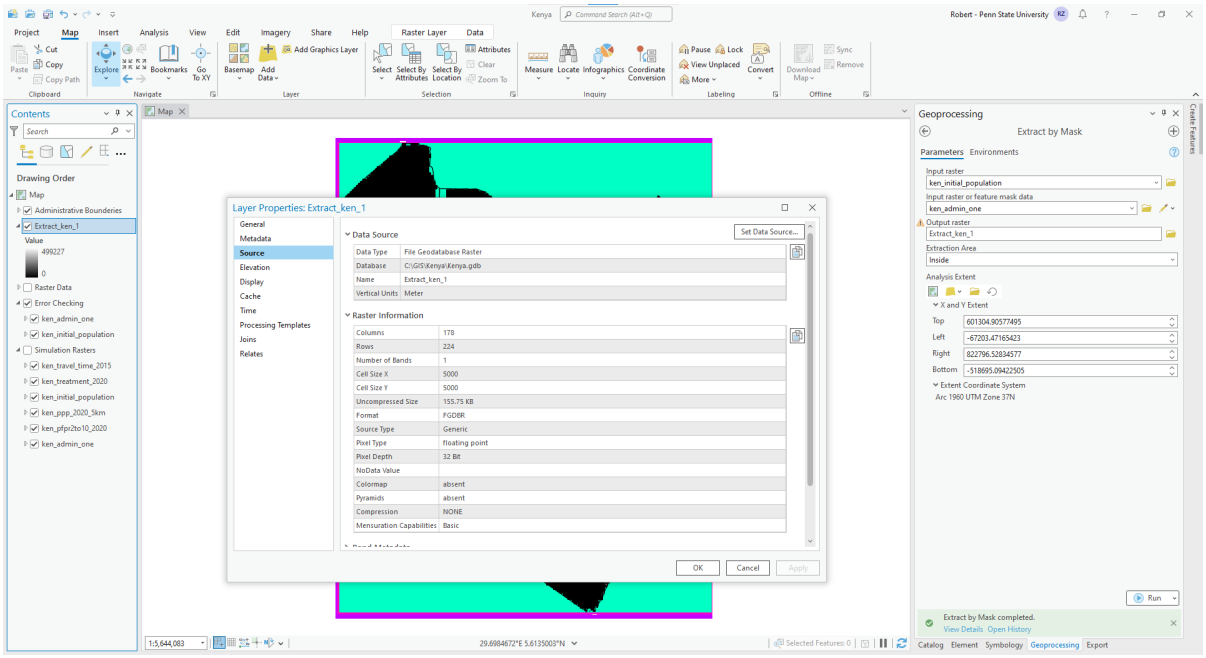


Figure 45: Upated rows and columns following the use of the Extract by Maks tool. Note that some color bleeding is present due to how ArcGIS is rendering the layered rasters.

At this point running `validateraster` suggests that while the alignment issues have been addressed, the data is still misaligned:

```
Using ken_districts.asc as reference
Mismatched nodata at 0, 32
One: 42.0, Two -9999.0
Mismatched nodata at 0, 33
One: 42.0, Two -9999.0
Mismatched nodata at 11, 43
One: 42.0, Two -9999.0
Mismatched nodata at 12, 43
One: 42.0, Two -9999.0
Mismatched nodata at 12, 44
One: 42.0, Two -9999.0
Mismatched nodata at 13, 43
One: 42.0, Two -9999.0
Mismatched nodata at 13, 44
One: 42.0, Two -9999.0
Mismatched nodata at 14, 44
One: 42.0, Two -9999.0
Mismatched nodata at 21, 47
One: 42.0, Two -9999.0
Plus 311 additional errors
```

In this case the `validaterasters` tool is reporting that one raster – `ken_districts.asc` – has a value present that that another raster is reporting as Nodata (i.e., -9999). If these rasters are used to run the simulation it will result in an error during model initialization since there will be a mismatch between the number of cells with data and between the various layers.

Here we need to know what the root cause is before we can adjust the data. As before start by using a contrasting color for the Nodata values, followed by using the `Swipe` tool under `Raster Layer` in the ribbon bar to examine the two layers. In doing so we find parts of the map that look like Figure 46, given that these areas contain major bodies of water (Lake Victoria to the west, Lake Turkana to the north) it appears that the WorldPop data is reporting bodies of water as Nodata as opposed to a population of zero. This hypnosis can be confirmed by adjusting the Nodata setting for the source raster as in Figure 47.

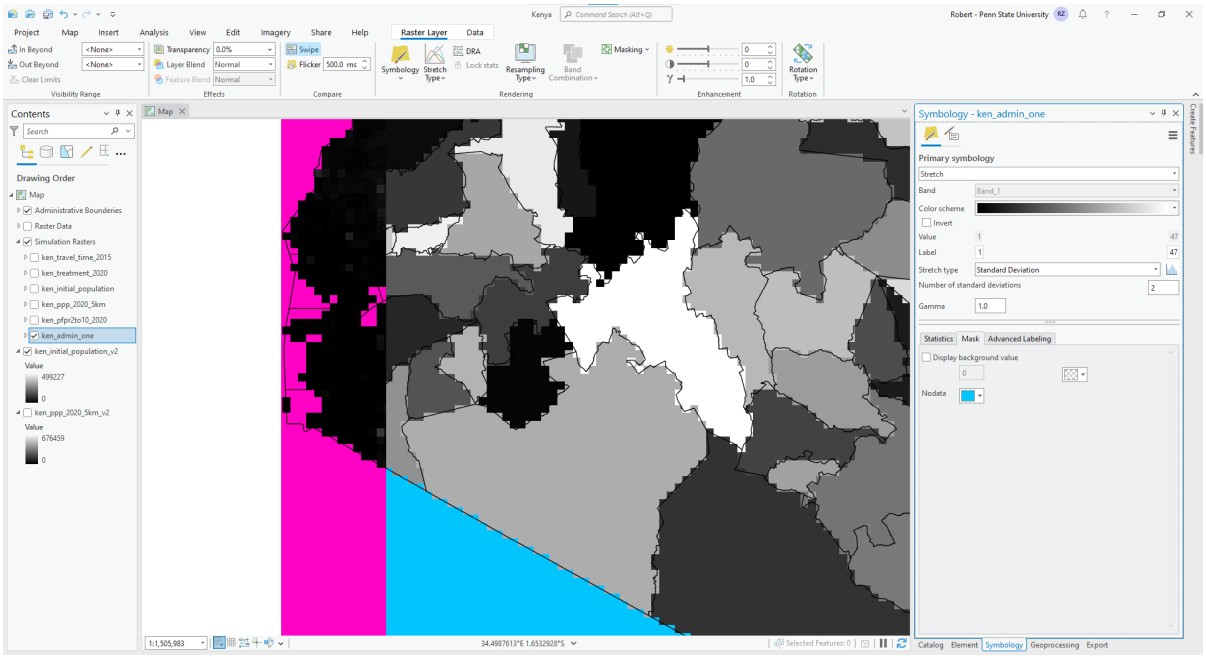


Figure 46: Using the Swipe tool we find that the newly extracted inital popuation raster is missing data where the admin one raster is reporting data.

This suggests that the Nodata values that appear within the bounds of Kenya should be adjusted to zero for the population. There are two approaches that can be used, the first is to edit the pixels to the correct value, while the other is to reclassify the Nodata value and extract an updated raster. Typically the second approach is quicker, and will be used here.

In the Geoprocessing menu search for and load `Reclassify (Spatial Analysis Tools)`, select the appropriate raster and under the `Value` column enter "NODATA" and under `New` column enter "0", as in Figure 48. This will produce a new raster which should then be used as the `Input Raster` for the `Extract by Mask` tool, as previously described. Once complete, the new raster should have zeros present instead of Nodata within the extent of the admin one districts.[27]

Once all issues have been addressed, `validateraster` will report no errors in a manner similar to the following:

```
Using ken_districts.asc as reference
5 files crosschecked, 6 total files
No errors detected
```

At this point, model calibration can begin if malaria does not follow a seasonal pattern. Otherwise, a seasonal adjustment will need to be prepared based upon climatic data, as described in the next section.

[27]The challenging part of this workflow is typically keeping track of the rasters. If you have multiple rasters with issues it's recommended to address them one at a time, and clean-up the temporary and invalid rasters before moving on to the next.

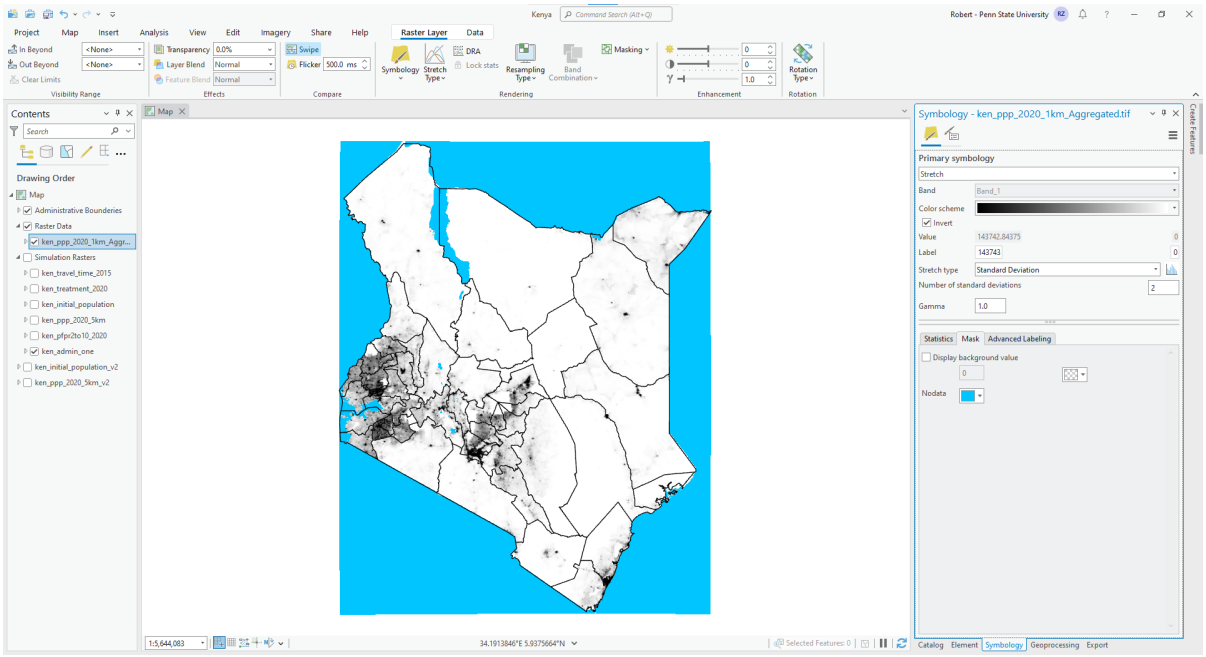


Figure 47: Rediplaying the WorldPop data with the Nodata symbology adjusted, note the presence of missing data.

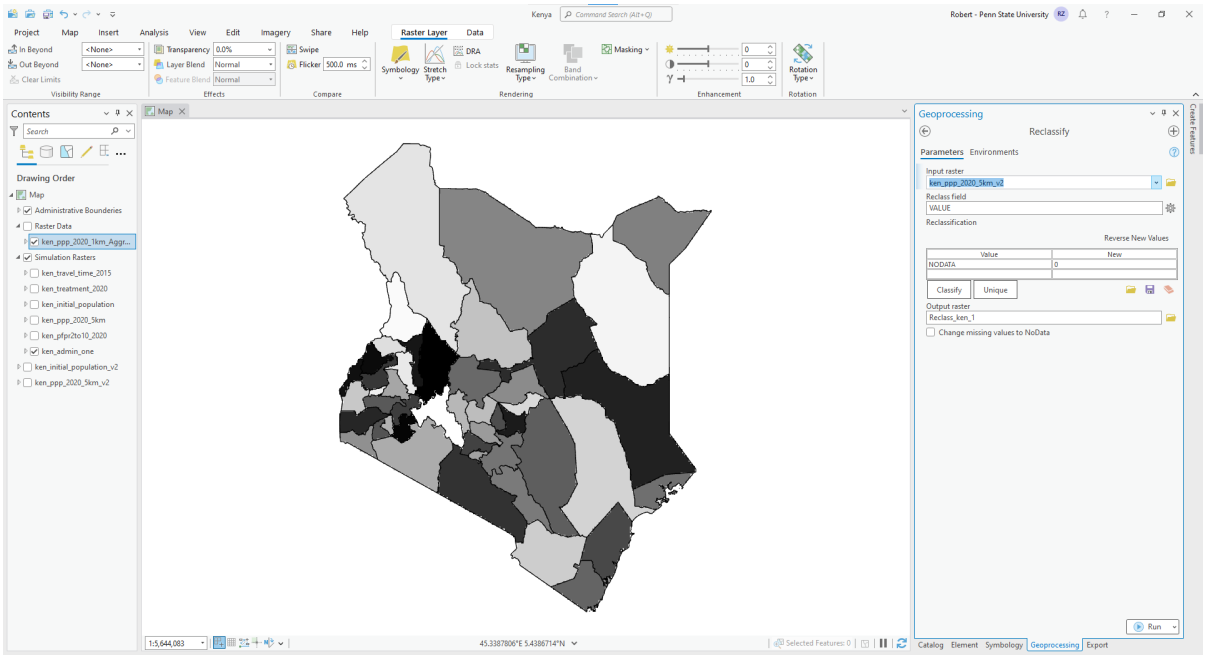


Figure 48: Reclassifing a population raster such that Nodata will be zero.

4 Climatic Data Production

Typically, malaria incidence increases and declines with the local rainfall variability due to the underlying habitat requirements for *Anopheles* mosquitoes (Pascual et al. 2008; Bomblies 2012). In the context of the simulation – and model calibration – this is typically managed by adjusting the daily beta (i.e., transmission intensity) based upon a seasonality model. Two approaches have been used to develop this seasonality mode: a equation-based model for each of the three climatic zones in Burkina Faso (Zupko et al. 2022), and a rainfall-based model with the daily beta adjusted based upon historical rainfall data for Rwanda (Zupko et al. 2023). Selection of the approach to use will typically depend upon the country being modeled; however, the preference is for the rainfall-based model since it has a closer coupling to real data.

When it is possible to use a single climatic model for the entire country, the script `era5_rainfall_doy.js` in the `PSU-CIDD-MaSim-Support` repository can be used to get the daily average rainfall, based upon ten years of climatic data from the Copernicus Climate Change Service (C3S).[28] The script will highlight the country being processed (see Figure 49) and also allow a CSV file to be exported to Google Drive which contains the daily rainfall data (see Figure 50). After which it will be necessary to smooth the data, off-set it to account for the *Anopheles* lifecycle, and scale it such that it is bounded between one and an appropriate floor (ex., 0.4 and 1.0 in the case of Rwanda, (Zupko et al. 2023)).[29] When this approach is used, an additional raster for the simulation is not needed.

The alternative approach is an equation-based model that can be developed on the basis of the malaria incidence or seasonal rainfall patterns. In addition to allowing for the alternative data source, this approach also allows a country to be segmented into multiple climate zones (see (Zupko et al. 2022)). The workflow for producing the climate zones varies, although in the case of Burkina Faso it was based upon the underlying precipitation data from WorldClim (Fick and Hijmans 2017), which was then converted into an aligned raster using the processes previously described in this chapter.

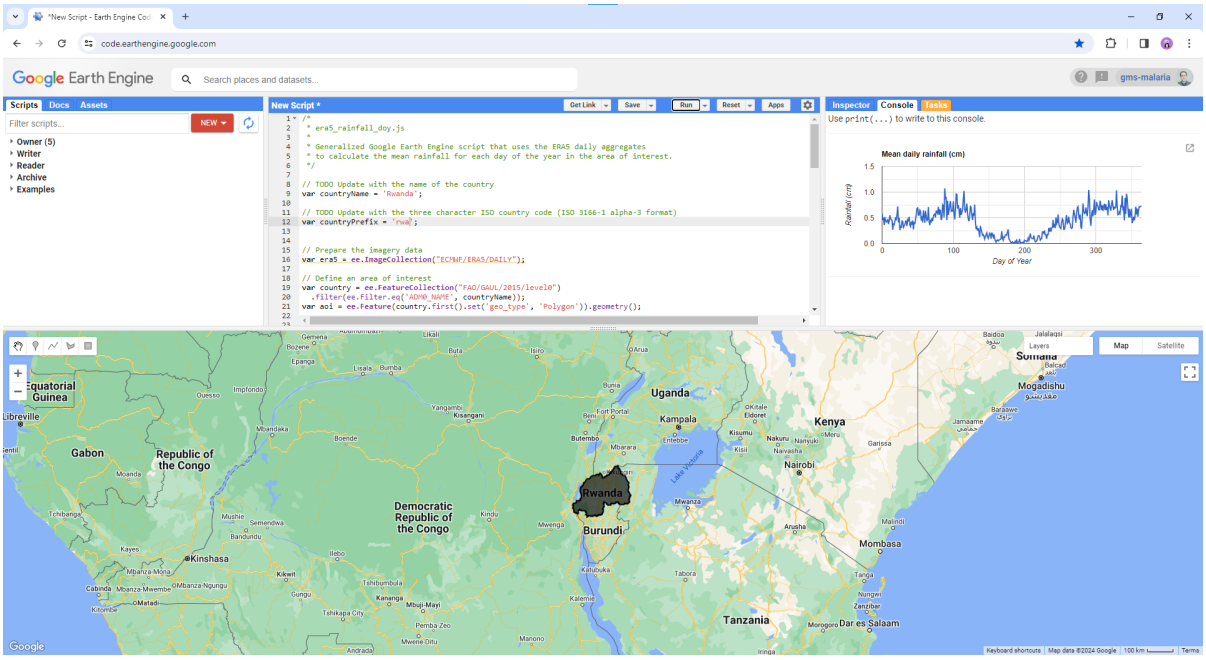


Figure 49: Example output of the daily median rainfall script being run in Google Earth Engine.

[28] see https://cds.climate.copernicus.eu/cdsapp#!/dataset/reanalysis-era5-single-levels?tab=overview

[29] Precise details for this process are stored in the Rwanda repositories: `PSU-CIDD-Rwanda` and `malariaibm-spatial-Rwanda-561H`.

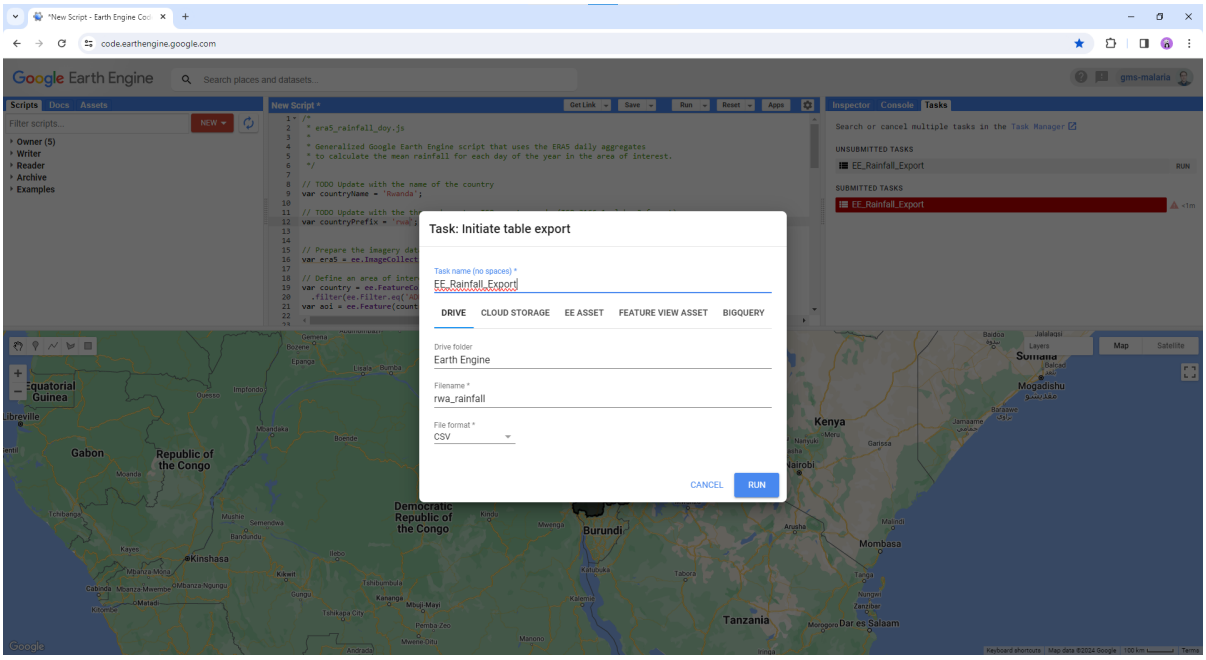


Figure 50: The CSV data export task dialog in Google Earth Engine, note that a task history is retained (as indicated by the failed task) and the Drive Folder must be present in Google Drive for the user running the script.

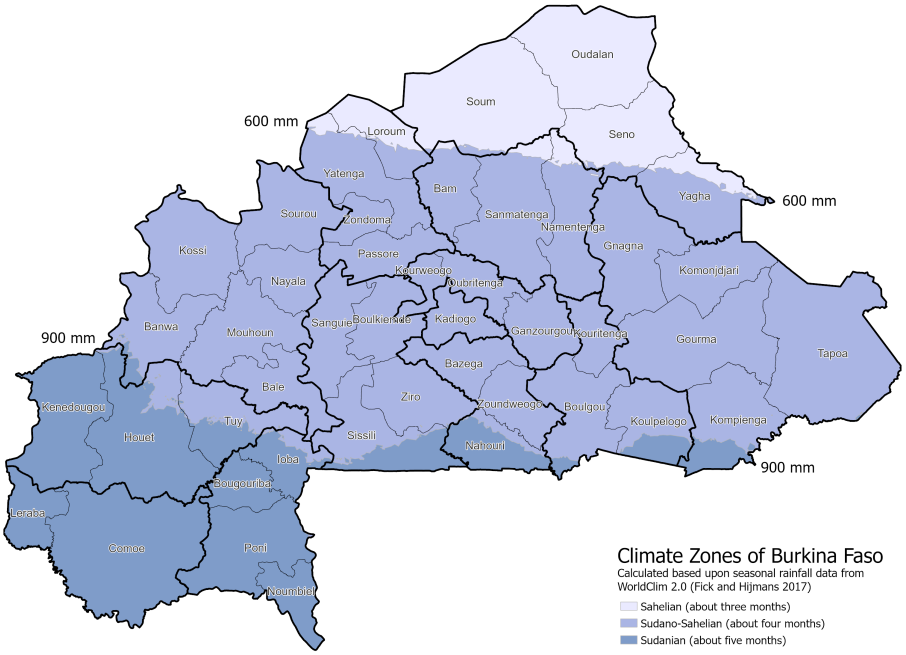


Figure 51: Example of raster segmentation based upon climate zones using precipitation data from Fick and Hijmans (2017), from the supplemental to Zupko et al. (2022).

Simulation

5 Model Calibration

This chapter will introduce the reader to calibration of the simulation when spatial modeling is involved. The directions assume that work is being using a Linux workstation, or Windows workstation with Windows Subsystem for Linux (WSL) installed. Scripts referenced can typically be found in the PSU-CIDD-MaSim-Support repository and prior to running the scripts it will be necessary to add them to the `PATH`.

5.1 Geographic Data

By design the configuration of the simulation is flexible in terms what must be supplied as raster data and what can be supplied as a single value in the configuration file. In general, raster files containing the following are required to run the model spatially:

- *Beta*: The β parameter determines, in part, how many additional infections occur as a result of a single infection and ultimately influences the *Pf*PR. These values must be calculated for each country based upon the inputs.
- Population: The model is initialized with the population values supplied in each of the cells in the raster. This file is mandatory when running spatially and can be generated by down sampling the spatial resolution a publicly available source while preserving the population count.

While the following are required to calibrate the model spatially:

- $Pf\text{PR}_{2\text{-}10}$: While not directly used by the IBM, a raster of the *Pf*PR values is needed as part of the beta raster generation. In situations where $Pf\text{PR}_{2\text{-}10}$ is not available, $Pf\text{PR}_{0\text{-}59}$ can also be used. Attempting to calibrate using the entire population is not recommended since the prevalence is not uniformly distributed across the population.

The following raster files a may or may not be needed depending upon the configuration and research goals:

- Access to treatment: In the event that access to treatment is variable depending upon the location within the country, it is necessary to supply a raster file defining the percentage of infected individuals that seek / receive treatment.
- Climate or malaria seasonality: If there are multiple seasonal variations to the malaria season in the country, it is necessary to supply a raster file that links the cell to the seasonal equation to be used.
- Political divisions: Since results are recorded in connection to a political district or region, a file is required to supply these values.
- Travel time or friction surface: Depending upon the movement algorithm selected, a travel time or friction surface is needed to ensure that individuals move in a reasonable fashion.

Once all of the GIS data has been acquired it and aligned, it is necessary to convert it to an Esri ASCII raster format (`asc` extension) to be loaded into the model.

5.1.1 Intitial Population

Typically the simulation is configured to use a constant crude birth rate during execution, which in turn requires that an *initial population raster* be produced. Assuming a reference population of 1,000 individuals, the initial population can be calculated by applying the equation:

$$p' = \frac{1000}{\left(1 + \frac{cbr}{1000}\right)^n}$$

Where *cbr* is the crude birth rate, n is the number of years prior to calibration year, and p' is the initial population. The cell multiplier can then be calculated as:

$$multiplier = 1 - \left(\frac{1000 - p'}{1000}\right)$$

The resulting *multiplier* can then be applied using the Raster Calculator functionality in geographic information system (GIS) software (e.g., ArcGIS Pro, GRASS GIS, or QGIS) where it is recommended that the floating point result be rounded up to the nearest integer value (i.e., Round Up function in ArcGIS Pro) to ensure that cells with a population always maintain a population. Once applied, the results can be validated by applying the equation:

$$projected = population * \left(1 + \frac{CBR}{1000}\right)^n$$

Where *projected* is the projected population n years after the initial *population* size for the given crude birth rate. Depending upon the number of digits used when applying the *multiplier* there may be some difference in the projected population versus the reference population and while ± 1 individual is accepted for large populations, the result can typically be corrected by increasing the number of digits in the *multiplier*. For cells with a small reference population (ex., < 10) in the focus should be on ensuring that a population remains in the cell since low population cells are subject to high variance in malaria dynamics during model execution.

5.2 Database Preparations

It is recommended that a new database be created for each country being studied. One of the limiting factors in the design of the database and IBM is that it presumes that the sequence of genotypes in each configuration file is the same. Once a new database has been created, the genotypes need to be loaded via the following command:

```
./MaSim -l -i [CONFIGURATION]
```

5.3 Beta Calibration

5.3.1 Setting Bounds

Once the GIS files have been prepared, work can proceed to generation of the β parameters.

1. Begin by ensuring that all of the ASC format raster files are aligned correctly using `validateraster`
   1.1. If they are not, it will be necessary to determine why they are not aligned correctly.
   1.2. In general, rasters should be aligned to the population and $Pf\text{PR}_{2\text{-}10}$ rasters, based upon which one has the most complete information.

2. Following alignment, the `generatebins` command should be run to generate the first calibration script to be run on the cluster this will set the bounds for when the betas are likely in relation to the $Pf\text{PR}_{2\text{-}10}$ values for the country.
   2.1. First, run `generatebins` using the primary configuration file (e.g., `Studies/rwa-configuration.yml`).
   2.2. Login to the cluster and create a directory do the calibration from (e.g., `rwa-calibration`).
   2.3. Copy the following files to the cluster via `sftp` into the previously created directory:

- From the country repository (e.g., PSU-CIDD-Rwanda):
  - `Studies/out/calibration.sh`
  - `Studies/Calibration/rwa-calibration.yml`
- From the support repository:
  - `bash/calibrationLib.sh`
  - `bash/manage.sh`
  - `bash/runCalibration.job`
  - `bash/template.job`
  - `bash/population.asc`
  - `bash/zone.asc`

2.4. Being the control script via the following command: `qsub runCalibration.job`
2.5. During calibration be sure to monitor for any errors, in general if replicate files are not deleted by the management script the error log should be checked.

3. Once all of the replicates have been run by `runCalibration.job` run the `createbetamap` command to load the calibration data from the database and create the first beta map.

5.3.2 Refining Alignment

Once the bounds have been found it is necessary to reduce the epsilon values to an acceptable margin of error compared to the reference data. The is done by the iterating on the beta values using the `reduceepsilons` script until the sub-national annual rates approximate reference values.

1. Run the `createbetamap` command and note the maximum epsilon value output.
   1.1. If the value is outside of acceptable bounds, run the `reduceEpsilons` command using the maximum epsilon rounded down (e.g., if the maximum is 0.0314 then start with 0.01) and the same decimal place for the step value (i.e., 0.01 per the previous example).
   1.2. The script will generate `out/reduction.csv` and `out/script.sh` which needs to be uploaded to the cluster.
   1.3. Once the files have been uploaded run the job on the compute cluster.
2. Once all of the replicates have run on the cluster, run the `createbetamap` script.
   2.1. Note the maximum epsilon.
   2.2. Examine the map of epsilon values (`out/epsilons_beta.asc`) and note the distribution of high epsilon values.
   2.3. If they correspond to higher population areas repeat Step 1
   2.4. If they are concentrated largely in low population areas proceed to Step 3.
3. As the epsilon values are reduced, they need to be checked against the reference values by performing a validation run.
   3.1. Using the beta map generated and the status quo parameters for the country, run the model at scale.

5.3.3 Calibration Assessment

The primary metric for evaluating the β calibration is a comparison of the projected prevalence against the population weighted prevalence for the targeted administrative region level. Typically this is done using data from the Malaria Atlas Project (Weiss et al. 2019), although the preference should be to use the most reliable and timely data that can be sourced. As shown in Figure 52 on the left (Zupko et al. 2022), in administrative regions with a high malaria burden, calibrations that are within $\pm 5\%$ of the reference prevalence are very achievable. However, as the malaria burden declines, the target calibration bounds typically increases to the

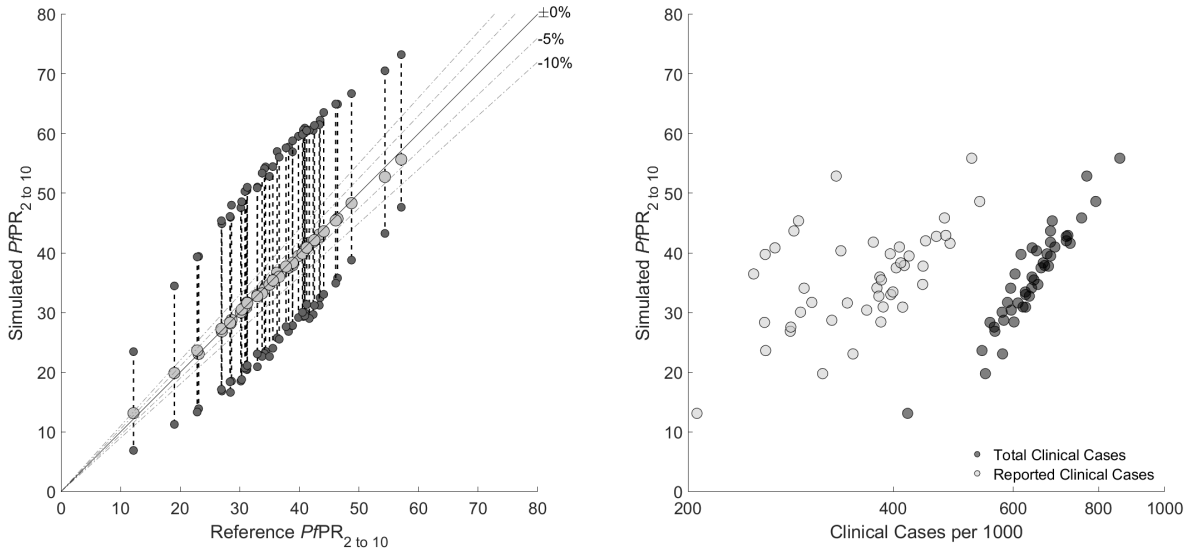


Figure 52: Example of a calibration assessment comparing the projected prevalence versus the reference prevalence. (Reproduced from Zupko et al. 2022, CC BY 4.0)

range of ±10%. This expansion should also be accompanied by a critical evaluation the model calibration to ensure that it is acceptable for the research question being evaluated. Likewise, due to the prefect information that is available in the simulation, in most regions the number of clinical cases is going to exceed the number of reported clinical cases reported (Figure 52, Right). Awareness of what this differential should be is an important part of ensuring the model is properly calibrated, and typically this differential is presented as the total number of clinical cases versus the total number of treated cases (i.e., those clinical cases that are reported to a health system).

One important thing to keep in mind when evaluating the beta calibration is that as the population size decreases, the variance between individual runs increases. This is illustrated by Figure 53 in which the EIR is plotted against the $Pf\text{PR}_{2\text{-}10}$ under different population sizes. Given that the β can act as a proxy for the EIR, increasing the β can be expected to produce increase the EIR, and thus such a figure can be produced by sweeping the β space from zero to n. While the expected behavior is that an increase in β, and thus an increase in the EIR, should produce an increase in the $Pf\text{PR}_{2\text{-}10}$, this is only observed under the larger population sizes. Under small population sizes, while the general trend is for increasing EIR and $Pf\text{PR}_{2\text{-}10}$, the this may not be observed between small changes in the EIR and/or β. This a known limitation of the model.

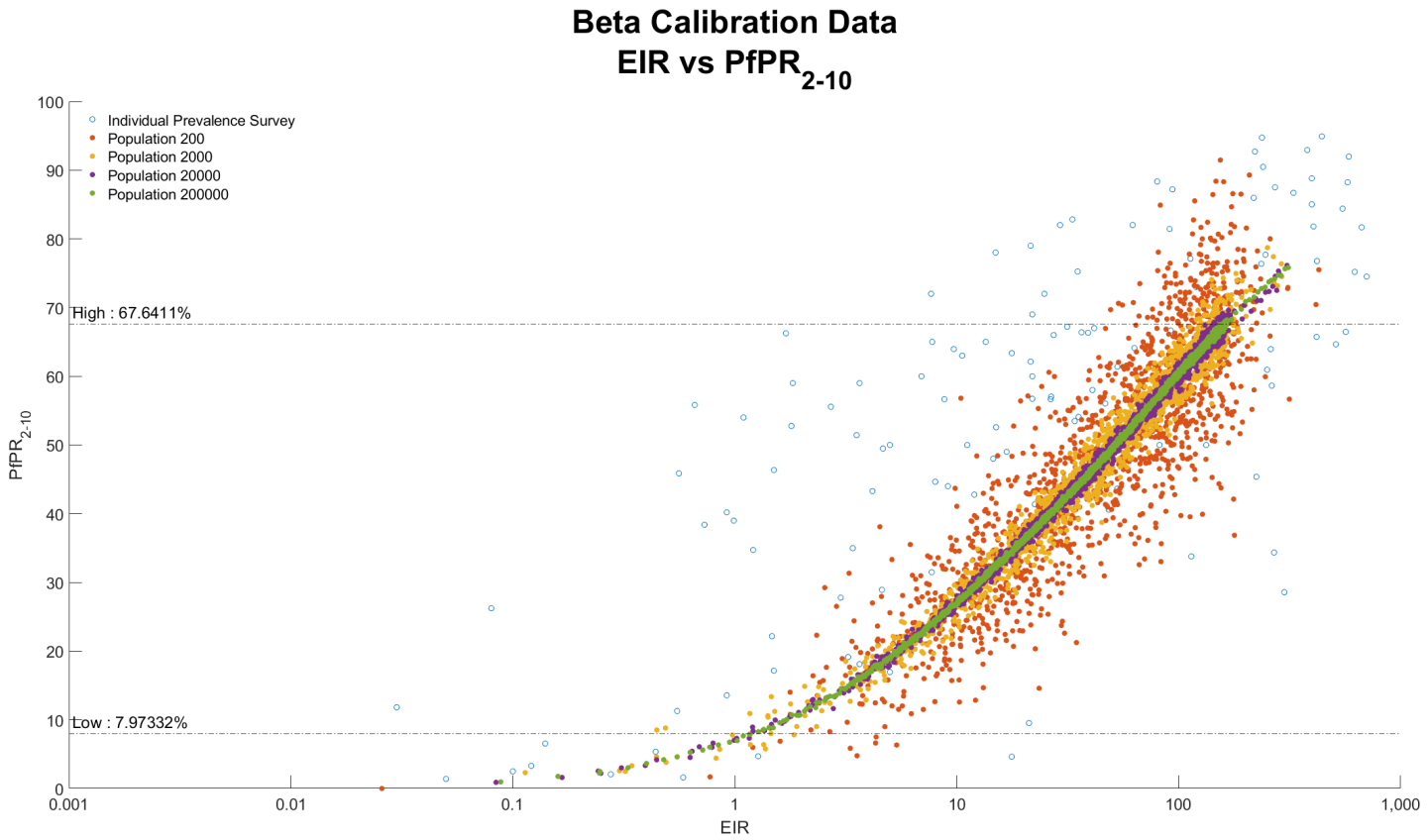


Figure 53: Comparison of the EIR versus the prevalence under different population sizes, note the inclusion of calibration points prepared by Malaria Modeling Consortium members (Hay et al. 2005; Cameron et al. 2015)

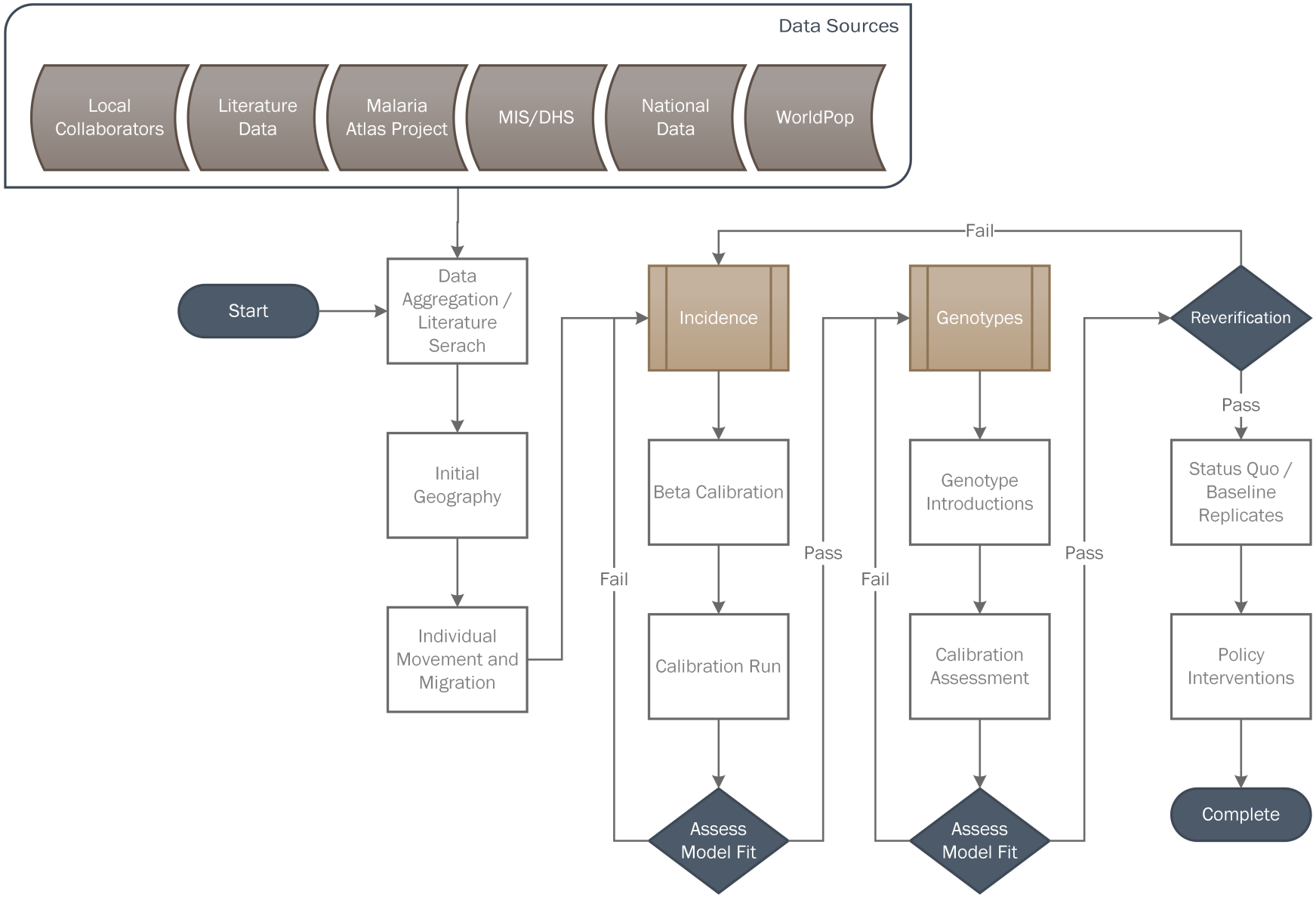


Figure 54: The typical workflow used when preparing the simulation for a new country.

6 Running the Simuation

6.1 Model Exeuction

When the simulation is executed, the operation can be broken into two distinct phases:

1. *Initialization* in which the simulation reads and verifies the YAML configuration file, and instantiates model data collector, population, and other relevant objects (see Figure 55).
2. *Execution* during which the simulation is executed by executing all scheduled operations and events for each time step that the simulation is configured to run (see Figure 56).

6.1.1 Initalization

The main entry point for the simulation is via the `main` function, which first configures last chance error reporting (on Linux platforms), followed by parsing of the command line, configuration of logging, and model initialization (see Figure 55).

Model initialization proceeds by first reading the configuration file, which entails the sequential processing of YAML elements.[30] If the YAML file indicates that the simulation is based upon a raster model of geography, then the geographic data (e.g., population distribution, beta values, etc.) is loaded and used to prepare the internal location database (i.e., `location_db`). If a location model of geography is then present in the YAML file, an error will be produced. Likewise, if any errors are encountered during the processing of YAML notes, an error will be produced.[31]

Once the YAML has been successfully parse, the remaining model initialization proceeds in the following order:

1. The random seed for the simulation is generated and stored.
2. The model reporters are initialized sequentially in the order that they appear on the command line.
3. The event scheduler is initialized.
4. The initial treatment strategy is initialized.
5. The initial treatment coverage model is loaded.
6. The model data collector is initialized.
7. The initial population is initialized based upon the population described in the YAML file.
8. The movement model is initialized.
9. Initial malaria infections are introduced based upon the genotypes indicated in the YAML file.
10. Any environmental or population events that are configured in the YAML file are added.
11. Miscellaneous configuration occurs.

Typically the major operations involved in model initialization are preparing the initial population and infections; however, depending on the movement model and geography involved the one-time calculations involved in model movement may consume a significant amount of time.

6.1.2 Execution

Once initialization is complete, any pre-simulation operations are preformed as part of before run operations, followed by the primary simulation loop which is managed by the `Scheduler::run` function (see Figure 56).

Each time step of the simulation proceeds in the following manner:

[30] While the order of elements in the YAML file does not matter, the simulation is highly sensitive to the order that configuration items are read and changes to the order of entries in the `src/Core/Config/Config.h` file should be carefully tested.

[31] If a superfluous node is encountered in the YAML file, no errors are produced; however, the lack of a required node will produce an error.

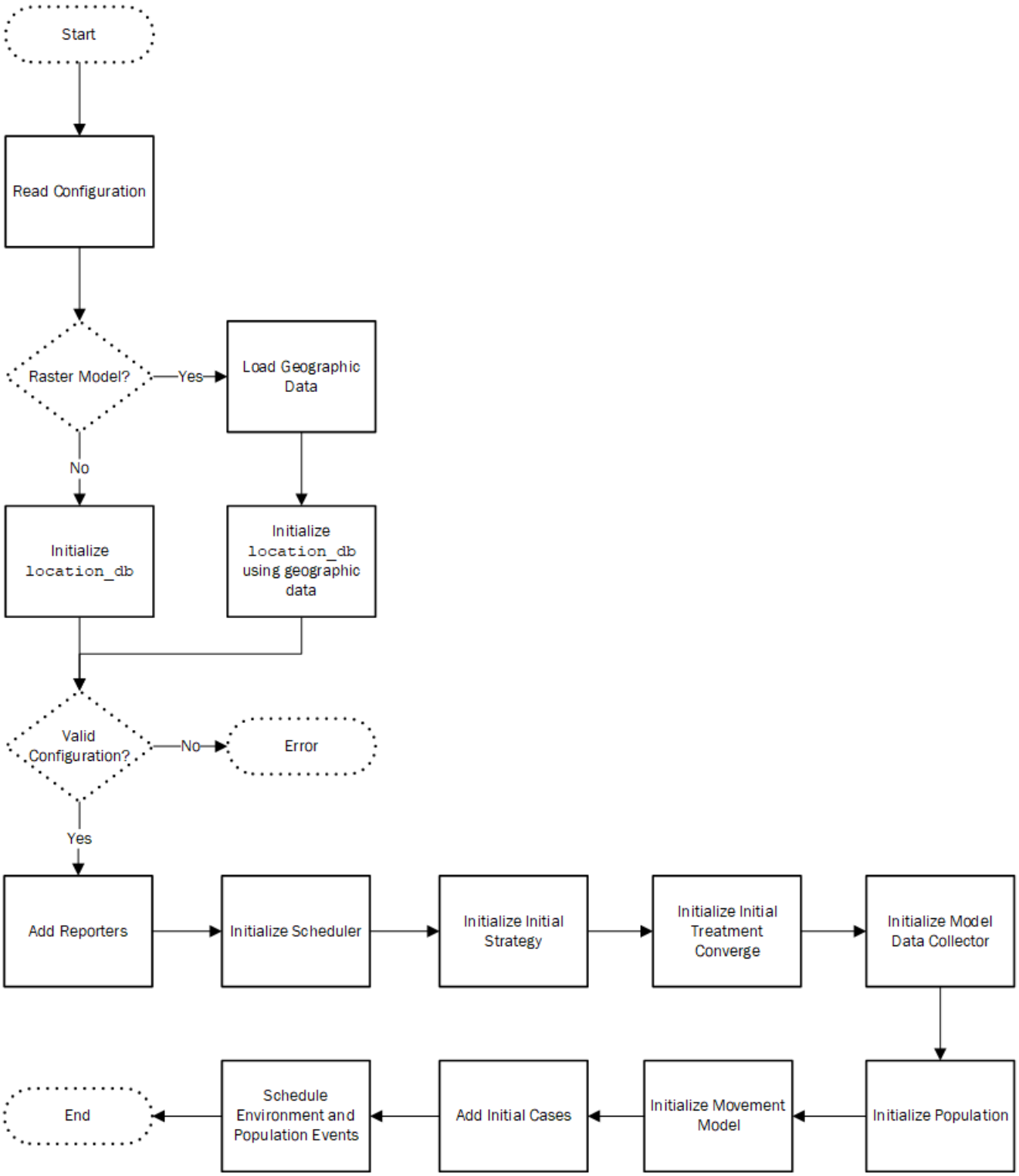


Figure 55: Typical model initialization sequence

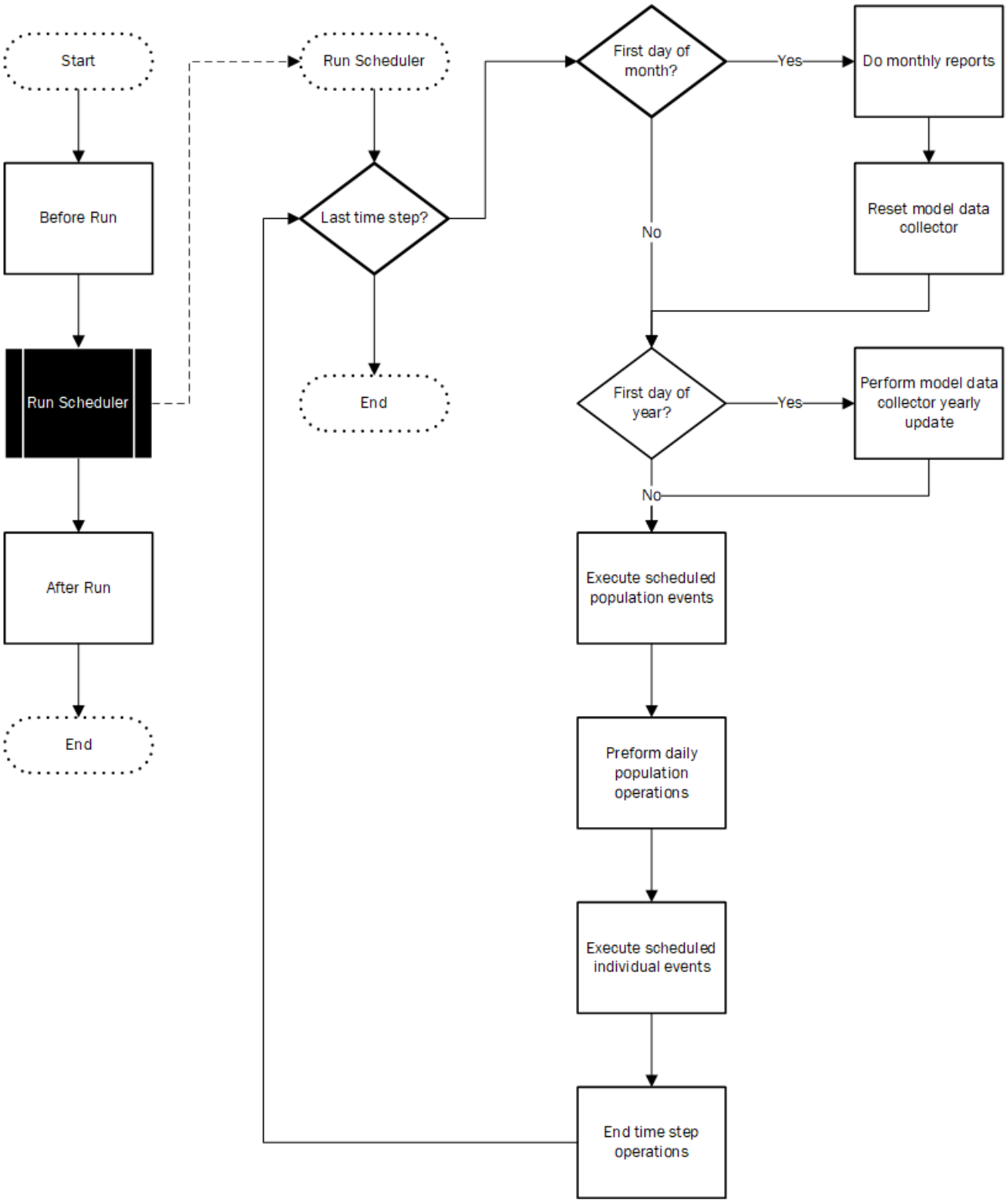


Figure 56: Typical model time step sequence

1. If it is the first day of the month,
    1. The `monthly_report` function of each reporter supplied is executed in the order that they were added to the reporters list.
    2. The Model Data Collector resets any variables that are tracked by the month.
2. If it is the first day of the year,
    1. The Model Data Collector resets any variables that are tracked by the year.
3. Any scheduled population events are executed.
4. All daily population operations are executed,
    1. Infection events are preformed.
    2. The birth event is preformed.
    3. The circulation (i.e., population movement) is preformed.
5. Any scheduled individual events are executed.
6. End of time step operations are preformed.

As a result of the order of operations for the simulation, the model data is incremented before the monthly reporter is executed resulting in the date logged reflecting the start of the next month. However, because the Model Data Collector has not been reset, it will reflect all of the data of the previous month.

6.2 Troubleshooting

6.2.1 EOF encountered while reading data

This error can be caused by several sources, but if the file being read ends with a `.asc` extension, then it is most likely an Esri ASCII raster format file[32] (or ASC file) and the simulation encountered the end of the file before all of the expected data was read. Since the number of rows and number of columns is supplied at the start of an ASC file, the simulation attempts to read that number of cells and will produce this error if it runs out of data. The most likely cause of this error is edits to the ASC file that causes a mismatch between the number of cells and declared rows and columns.

6.2.2 MD5 hash collision

An MD5 hash collision error is produced by either the `DbReporter` or the `DbDistrictReporter` detects that YAML file has the same same file name and produces the same MD5 hash of an existing configuration in the database.[33] Under normal operations this error should be prevented by the unique constraint upon `sim.configuration.md5` and `sim.configuration.filename` although older instances of the database may contain the error. Correction of the error typically requires manual deletion of the duplicated hashes and all replicates associated with them via the `deleteReplicates.py` script in `PSU-CIDD-MaSim-Support/Python` followed by the addition of the constraint to the database:

```
ALTER TABLE IF EXISTS sim.configuration
    ADD CONSTRAINT configuration_md5_unique UNIQUE (md5, filename);
```

[32] https://desktop.arcgis.com/en/arcmap/latest/manage-data/raster-and-images/esri-ascii-raster-format.htm

[33] The `DbDistrictReporter` is an inheriting class of `DbReporter` so the error is logged as being produced by `DbReporter::prepare_configuration` regardless of which reporter is being used.

7 Demonstration

7.1 Introduction

The configuration included in the `demo` directory (`demo.yml`) is configured for a single cell and derives the majority of the settings from the Burkina Faso project, allowing it to emulate a realistic starting point for a model. In addition to the configuration file, there is also a basic raster file (`population.asc`) which informs the model on how to create a single cell with 1,000 individuals in it. The raster files used by the simulation are compliant with the Esri ASCII raster format and can be edited using a text editor, or appropriate geographic information system (GIS) software.

In order to use this demonstration code you first need to build the simulation using the guide outlined in Development, after which is recommended that you copy the binary (`masim`) and contents of the `demo` directory (i.e., `demo/demo.yml` and `demo/population.asc`) into the same location. The simulation can then be executed using the following command:

```
./MaSim -i demo.yml -r CellularReporter
```

Using the `CellularReporter` will generate a comma separated values (CSV) file in the directory containing model output for select parameters such as population, $Pf\mathrm{PR}_{2\text{-}10}$, and infected individuals to name a few. Running the simulation a second time will cause this file to be overwritten unless the job number (`-j`) switch is used. The `CellularReporter` is intended for models with only one location (i.e., cell) and will generate an error if multiple cells are present in the configuration.

7.2 Column Descriptions

The following tables are present in the output from the `CellularReporter`:

Table 1: Column format for the `CellularReporter`

Column	Description
DaysElapsed	Model days elapsed, should correspond to the length of months.
Population	Total population in the simulation.
PfPR2to10	The *Plasmodium falciparum* parasite prevalence for individuals aged 2 to 10 years.
TreatmentCoverage	The current treatment coverage in the simulation as an average for the under-5 and over-5 groups.
InfectedIndividuals	The number of individuals in the simulation who have *any* level of parasitemia.
ClinicalIndividuals	The number of individuals in the simulation who have a parasitemia level of clinical symptoms to manifest.
ClinicalU5	The number of newly clinical individuals under the age of 5.
ClinicalO5	The number of newly clinical individuals over the age of 5.
NewInfections	The number of new infections in the past month.
Treatment	The number of clinical individuals who received treatment.
NonTreatment	The number of clinical individuals who did not seek treatment.

Column	Description
TreatmentFailure	The number of clinical individuals who sought treatment and the treatment did not clear the infection in the configured amount of time.
ParasiteClones	The number of parasite clones present in the simulation, one individual may be infected by multiple clones.
Theta	The mean measure of immunity to the parasite in the simulation, zero means no immunity.
580yWeighted	The weighted number of occurrences of the 580Y genotype, such that the sum of all clones in an individual equals one.
508yUnweighted	The total number of occurrences of the 580Y genotype in the simulation.
Plasmepsin2xCopyWeighted	The weighted number of occurrences of the Plasmepsin double copy mutation, such that the sum of all clones in an individual equals one.
Plasmepsin2xCopyUnweighted	The total number of occurrences of the Plasmepsin double copy mutation in the simulation.

7.3 Example Console Output

```
[INFO] [default] MaSim version 4.0, experimental
[INFO] [default] Model initializing...
[INFO] [default] Read input file: demo.yml
[WARNING] [default] p_infection_from_an_infectious_bite used default value of 0
[INFO] [default] location_db appears to have been set by raster_db
[WARNING] [default] as_iov used default value of 0.2
[INFO] [default] Random initializing with seed: 1625174844985602
[INFO] [default] Starting day is 2007-01-01
[INFO] [default] Location count: 1
[INFO] [default] Population size: 1000
[INFO] [default] Model starting...
[INFO] [default] Perform before run events
[INFO] [default] Simulation is running
[INFO] [default] 17:27:24 - Day: 0
[INFO] [default] 2007-01-01 : turn mutation off
[INFO] [default] 17:27:25 - Day: 30
[INFO] [default] 17:27:25 - Day: 60
[INFO] [default] 17:27:25 - Day: 90

...

[INFO] [default] 17:28:19 - Day: 5400
[INFO] [default] 17:28:19 - Day: 5430
[INFO] [default] 17:28:19 - Day: 5460
[INFO] [default] Perform after run events
[INFO] [default] Model finished!
[INFO] [default] Final population: 1412
[INFO] [default] Elapsed time (s): 54.991
```

8 DxG Generator

The DxG Generator is used to calculate the efficacy of therapies in the simulation based upon the therapy's drug combination and the genotypes that are present in the YAML file for a given region. While this tool uses the same simulation code base, the executable is separate from the primary simulation (i.e., `masim`) and can be invoked as follows:

```
./DxGGenerator -i [input]
```

With the full scope of options supported:

```
-g                  Get the efficacies for a range of genotypes
-h / --help         Display the help menu
-i / --input        The YAML configuration to read
-t                  Get the efficacies for a range of therapies
--ec50              EC50 for AS on C580 only
--iov               AS inter-occasion-variability
--iiv               AS inter-individual-variability
```

Note that the DxG Generator is very sensitive to the YAML configuration since the *full* configuration is still loaded prior to the efficacy calculations. As such it is recommended that a second, simplified (i.e., non-spatial) configuration be prepared for the DxG Generator. The tool is also only comparable with single therapies (i.e., `SFTStrategy`), which can be used in the configuration as follows:

```yaml
strategy_db:
  0:
    name: SFTStrategy
    type: SFT
    therapy_id: 0
initial_strategy_id: 0
```

Additionally, the DxG Generator was written with the assumption that `log_parasite_density_detectable` represents the lower detectable limit for a careful clinical study, typically 10 parasites per microliter of blood. However, if `log_parasite_density_detectable_pfpr` is set, then that value will be used instead. Since the *Pf*PR is driven by the more common detectable range of 50 to 100 parasites per microliter of blood, this will result in the efficacies returned being higher than expected. As such, it is recommended that the `log_parasite_density_detectable_pfpr` setting be deleted from the YAML configuration when it is prepared for the DxG Generator.

Development

9 Development

This chapter outlines the basic steps that are needed to build the simulation across various development environments. The guide is written from the standpoint of a "clean" workstation with no other dependencies in place for users intended to work with the simulation's codebase. Typically the best place to start is by running the `config.sh` script in the root of the repository, which prepares the development environment on Linux and Windows Subsystem for Linux (WSL) systems, followed by creation of a database administrator if needed. However, if the script doesn't work, the full workflow for installing the tool chain dependencies is included here.

9.1 Tool Chain Dependencies

9.1.1 Windows Subsystem for Linux / Linux

Starting with either the Windows Subsystem for Linux of your choice (Ubuntu recommended, or Red Hat) on a a Linux system:

Step 1. Ensure that `apt` is up to date.

```
sudo apt update
sudo apt upgrade
```

Step 2. Install the build dependencies

```
sudo apt install build-essential
sudo apt install cmake
sudo apt install libgsl-dev
sudo apt install libyaml-cpp-dev
sudo apt install libfmt-dev
```

Step 3. Add PostgreSQL to the Apt Repository listing

Note that while this step is necessary when working from a new installation of Ubuntu, it may not be necessary, and you should check to see if the package can be found by first performing `apt search postgresql-10`

```
sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt $(lsb_release \
  -cs)-pgdg main" > /etc/apt/sources.list.d/pgdg.list'
wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add -
sudo apt-get update
```

Step 4. Install the PostgreSQL and libpqxx dependencies

Note that this should be done from a directory that you are comfortable with git repositories being stored in.

```
sudo apt install postgresql-10
sudo apt install libpq-dev
git clone https://github.com/jtv/libpqxx.git
cd libpqxx
```

```
git checkout 7.8.1
./configure --disable-documentation
make
sudo make install
```

Step 5. Create PostgreSQL administrative user

If you are going to be using the PostgreSQL database locally for development purposes, then it is recommended that you also create an administrative user and install pgAdmin (see Installation of pgAdmin).

```
sudo -u postgres createuser --interactive --pwprompt
```

Step 6. Install Git Large File Storage

While not necessary for building the simulation, when working with country data or archiving projects it is recommended that Git Large File Storage (LFS) is installed as follows:

```
sudo apt install git-lfs
git lfs install
```

9.1.2 Upgrading to WSL 2

WSL 2 offers some performance improvements which may be relevant during development so upgrading may be of interest. To do so the following needs to be performed, note that these steps will also enable WSL 2 if WSL 1 has not already been installed. However, one drawback of WSL 2 is that disk access may be slow compared to WSL 1, so this upgrade should only be done if necessary.

Open `PowerShell` with administrator permissions and enter the following:

```
dism.exe /online /enable-feature /featurename:Microsoft-Windows-Subsystem-Linux /all `
  /norestart
dism.exe /online /enable-feature /featurename:VirtualMachinePlatform /all /norestart
```

Once complete, restart the computer, after which the version needs to be set in `PowerShell`:

```
wsl --set-default-version 2
```

If you have already installed a distribution, updating is recommended. Otherwise, you can now install a WSL compliant distribution.

9.2 Building

9.2.1 Local WSL Builds

Before building the first time it is necessary to create the build directory within the local clone of the `PSU-CIDD-Malaria-Simulation` repository:

```
mkdir build
cd build
```

After which the following command can be run to build the simulation under WSL:

```
cmake -DCMAKE_BUILD_TYPE=DEBUG -DBUILD_WSL:BOOL=true ..
make -j 8
```

Generally it is recommended to create a `build.sh` script that runs the build commands.

9.3 Execution

9.3.1 Local Runs

The first step in performing a basic model check is load the genome data in the database, this must also be done whenever a new database is corrected:

```
cd build/bin
./MaSim -i ../../misc/input.yml -l
```

Once the genome data has been loaded the basic input file can be run as follows:

```
cd build/bin
./MaSim -i ../../misc/input.yml
```

Note that while care was taken in places to ensure the code is performant, the amount of RAM needed during execution can be quite high (ex., 32 GB or more). When the model is run in Linux environments where the necessary memory is not available, you may find that the program is killed without notice due to being out of memory.

9.4 Development Tools

9.4.1 Isolating Segmentation Faults in Linux

When developing new functionality in the model it is sometimes necessary to isolate segmentation faults. One of the easier ways to do this is through the use of `gdb` in a Linux environment. First, compile the program with the `debug` flag set, this will ensure that there are debug symbols in the binary. Then use `gdb` to open the gdb console:

```
file bin/MaSim               # Load the specified executable
run -i ../input/sample.yml   # Run the program with the following command line parameters

...                          # Program output omitted

bt                           # Generates the stack trace
```

9.4.2 Profiling

While there are many approaches to code profiling, as a quick way to get up and running, valgrind is recommended along with gprof2dot. Following installation, the simulation can be profiled using:

```
valgrind --tool=callgrind ./bin/MaSim - ../input/sample.yml
gprof2dot -f callgrind callgrind.out.* | dot -Tpng -o output.png
```

This will generate a PNG file containing a node graph of where most of the simulation's time is spent during execution along with the percentage of time spent in a given function.

Note that `valgrind` adds **considerable** overhead to the execution of the simulation, so it is highly recommended that profiling be limited to small simulations.

9.4.3 Valgrind under WSL 2 with CLion integration

When using CLion as an IDE, integration with Valgrind is possible. Presuming that WSL 2 has been enabled and `valgrind` has been installed, CLion simply needs to be informed of the path under WLS 2 via **File > Settings > Build, Execution, Deployment > Valgrind** and the path will typically be `/usr/bin/valgrind`.

10 Database Infrastructure on Servers

Due to the complexity of data storage requirements for simulations involving multiple cells, a database is provided through the `dbreporter` reporter class. The schema is outlined below and was developed against PostgreSQL due to the high storage requirements. Note that the `notes` table presume the usage of the MaSimLIMS and can be removed if not needed with no impact upon the simulation.

10.1 Installation

The following guide is intended provide a walk through of how to get the simulation database up and running. While the document is kept as general as opposed, there may be some differences that you encounter due to your local IT requirements. The following guide assumes that the server is running a clean installation of Ubuntu 18.04 LTS 64-bit with ports 80, 443, and 5432 open.

10.1.1 Hardware Requirements

The specific requirements for the server are dependent in part upon the number of instances that will be connecting to it while a simulation is running. However, as a baseline the following is a reasonable starting point for a virtual machine:

- 4 CPUs
- 8 GB RAM
- 800 GB primary disk
- Ubuntu 18.04 LTS 64-bit

10.1.2 Installing PostgreSQL

1. Connect to the server

```
ssh [User]@[IP address] -p [port]
```

2. Set the default encoding on the server, the setting can be verified by running `locale`

```
export LANG=en_US.UTF-8
```

2. Install the GPG key and repository for PostgreSQL packages

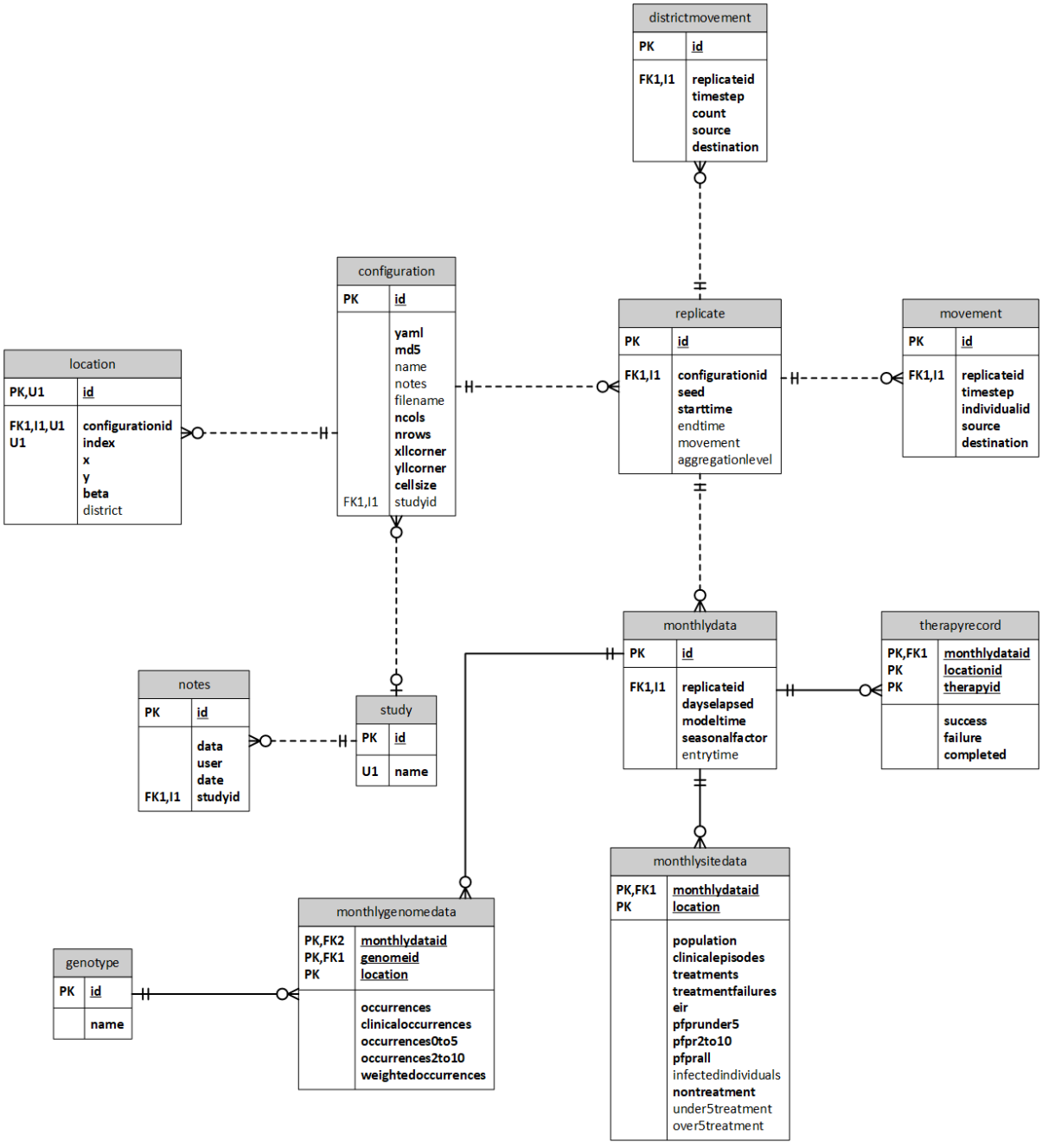


Figure 57: Database schema

```
sudo apt-get install wget ca-certificates
wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add -
sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt/ \
  `lsb_release -cs`-pgdg main" >> /etc/apt/sources.list.d/pgdg.list'
```

3. Install PostgreSQL

```
sudo apt-get update
sudo apt-get install postgresql postgresql-contrib
```

4. Verify connection to PostgreSQL

```
sudo -u postgres psql postgres
postgres-# \conninfo
postgres-# \q
```

5. Create administrative user

Here it is recommended that you create an administrative user (e.g., `dbadmin`) that will be used to connect to the database and to perform administrative task. When prompted be sure to supply the password and select yes when asked if the user should be a super user.

```
sudo -u postgres createuser --interactive --pwprompt
```

6. Configure the server to listen for connections

This is done by first editing the file `/etc/postgresql/11/main/postgresql.conf` and updating the line `listen_addresses='*'`. Next, the file `/etc/postgresql/11/main/pg_hba.conf` needs to have the line `host all all 0.0.0.0/0 md5` added at the end. Note that this means that the server will listen to connections from *any* IP address. If this is not desired behavior, a more restrictive configuration should be used.

7. Enable service

```
sudo update-rc.d postgresql enable
```

8. Optional update for locale

Update default locale for the database template. This is necessary if you get an error that states "Encoding UTF8 does not match locale en_US. The chosen LC_CTYPE setting requires encoding LATIN1" from pgAdmin when creating a database

```
sudo -u postgres psql postgres

update pg_database set datistemplate=false where datname='template1';
drop database Template1;
create database template1 with owner=postgres encoding='UTF-8' lc_collate='en_US.utf8' \
  lc_ctype='en_US.utf8' template template0;
update pg_database set datistemplate=true where datname='template1';
\q
```

10.2 Installation of pgAdmin

The preferred way of installing pgAdmin on a clean Ubuntu installation is though the use of `apt`[34]:

1. Configure the system for the pgAdmin APT repository.

Start by installing `curl` if necessary:

```
sudo apt install curl
```

Next, install the public key for the repository:

```
curl -fsS https://www.pgadmin.org/static/packages_pgadmin_org.pub | \
  sudo gpg --dearmor -o /usr/share/keyrings/packages-pgadmin-org.gpg
```

Finally, create the repository configuration file:

```
sudo sh -c 'echo "deb [signed-by=/usr/share/keyrings/packages-pgadmin-org.gpg] \
  https://ftp.postgresql.org/pub/pgadmin/pgadmin4/apt/$(lsb_release -cs) \
  pgadmin4 main" > /etc/apt/sources.list.d/pgadmin4.list && apt update'
```

2. Install pgAdmin:

```
sudo apt install pgadmin4-web
```

3. Run the pgAdmin configuration script:

```
sudo /usr/pgadmin4/bin/setup-web.sh
```

Once installed and configured you should be able to access pgAdmin at http://127.0.0.1/pgadmin4/ or http://localhost/pgadmin4/ the first time you connect, the local database server will not be listed. This can be added by right clicking on 'Servers', selecting 'Register' > 'Server' which will give you the 'Register - Server' prompt. Enter the name you wish to assign (e.g., 'localhost') before clicking the 'Connection' tab, which will prompt you for connection information. Then enter the following:

Field	Entry
Host name/address	localhost
Port	5432
Maintenance Database	postgres
Username	< Admin Username >
Kerberos authentication	Off
Password	< Admin Password >
Save password?	Off recommended
Role	< Leave Blank >
Service	< Leave Blank >

Where the username and password correspond go the username and password created for the administrative user (e.g., `dbadmin`) when installing PostgreSQL.

[34] https://www.pgadmin.org/download/pgadmin-4-apt/

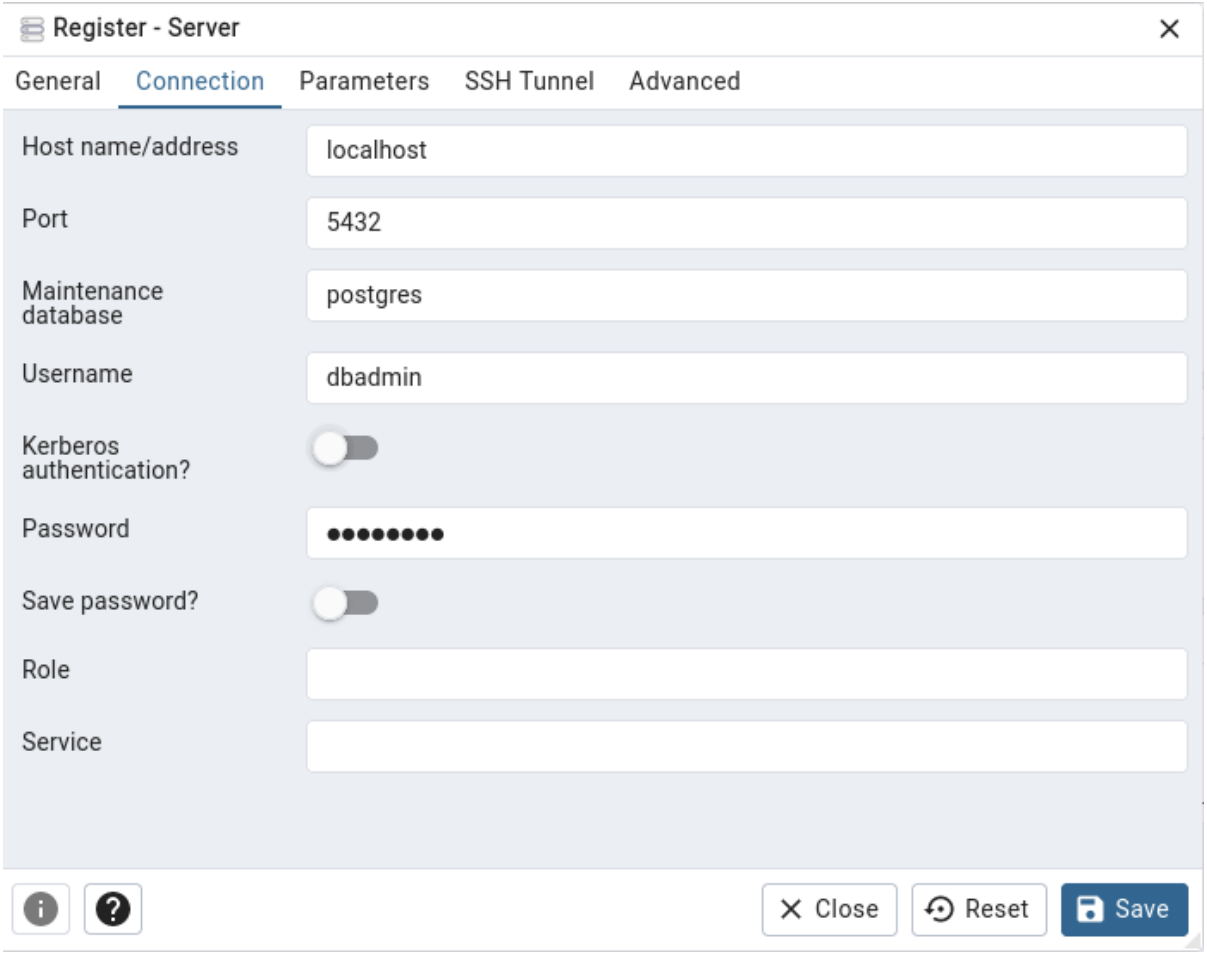


Figure 58: pgAdmin showing the dialog to add a new server, note that here we are adding the localhost

10.2.1 Optional Configuration of Apache for pgAdmin

Since newer version of pgAdmin are designed for web environments, configuration following installation may be limited editing the `pgadmin4.conf` configuration to reside at the server root:

```
WSGIDaemonProcess pgadmin processes=1 threads=25 python-home=/usr/pgadmin4/venv
WSGIScriptAlias / /usr/pgadmin4/web/pgAdmin4.wsgi

<Directory /usr/pgadmin4/web/>
    WSGIProcessGroup pgadmin
    WSGIApplicationGroup %{GLOBAL}
    Require all granted
</Directory>
```

The configuration can then be reloaded as follows:

```
sudo a2dissite 000-default.conf
sudo a2ensite pgadmin4.conf
sudo systemctl restart apache2
```

At this point you should be able to connect to the pgAdmin control panel at `http://[SERVER IP ADDRESS]`. Login to the control panel using the credentials supplied during configuration. One logged in, you should be able to add the localhost via "Add New Server" and proceed with administration of the databases using pgAdmin. Additional deployment information can be found on pgAdmin.org under Server Deployment.

11 Using the Database

11.1 Creation of the Simulation User

11.2 Creation of Simulation Database

(ref:fig-pgadmin-create)pgAdmin showing the dialog to create a new database, note that here we are creating the `template_masim` template database

(ref:fig-pgadmin-script)pgAdmin showing the results of the `database.sql` script being run. Note that tables have been created under the `sim` schema.

After logging into the pgAdmin control panel, start by creating the user `sim` with the password `sim` and ensuring they have permissions to login to the database. This is the user that will be the simulation to write results to the database during model execution. Next, run the script `database.sql` which can be found under the `/database` directory of this repository.

11.3 Cloning databases

For the purposes of development or archiving it may be necessary to clone databases. The following SQL commands can be used from `psql` on the server to do so:

```
UPDATE pg_database SET datallowconn = false WHERE datname = 'masim';
CREATE DATABASE development WITH TEMPLATE masim OWNER sim;
UPDATE pg_database SET datallowconn = false WHERE datname = 'masim';
```

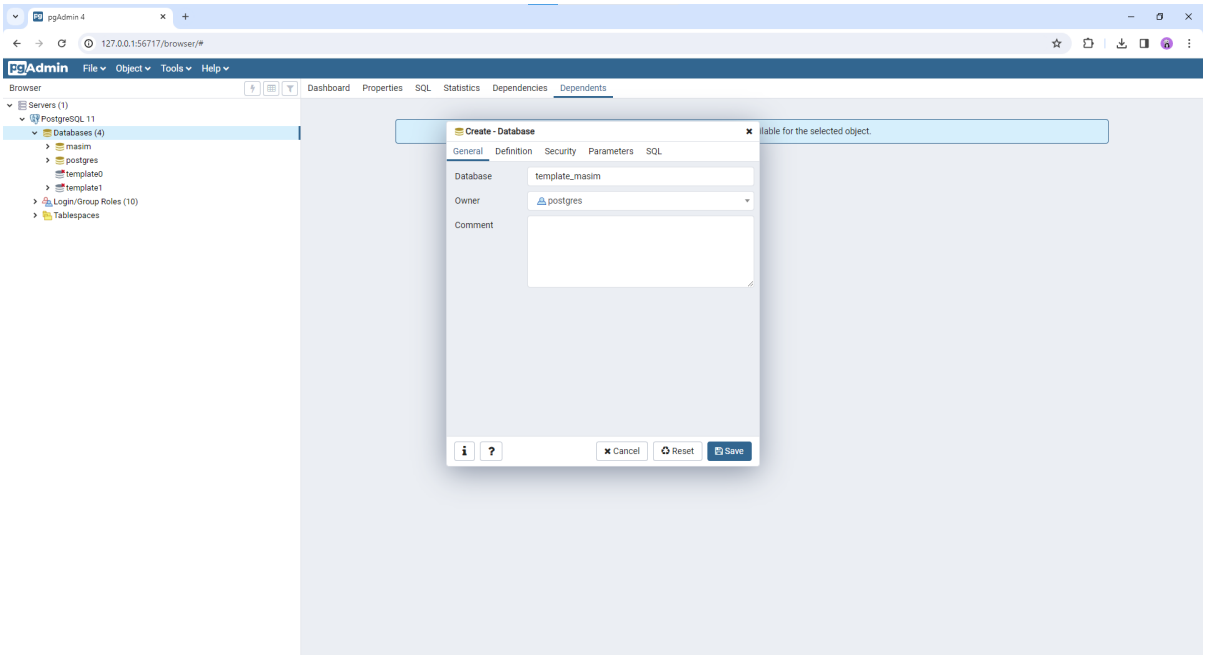


Figure 59: (ref:fig-pgadmin-create)

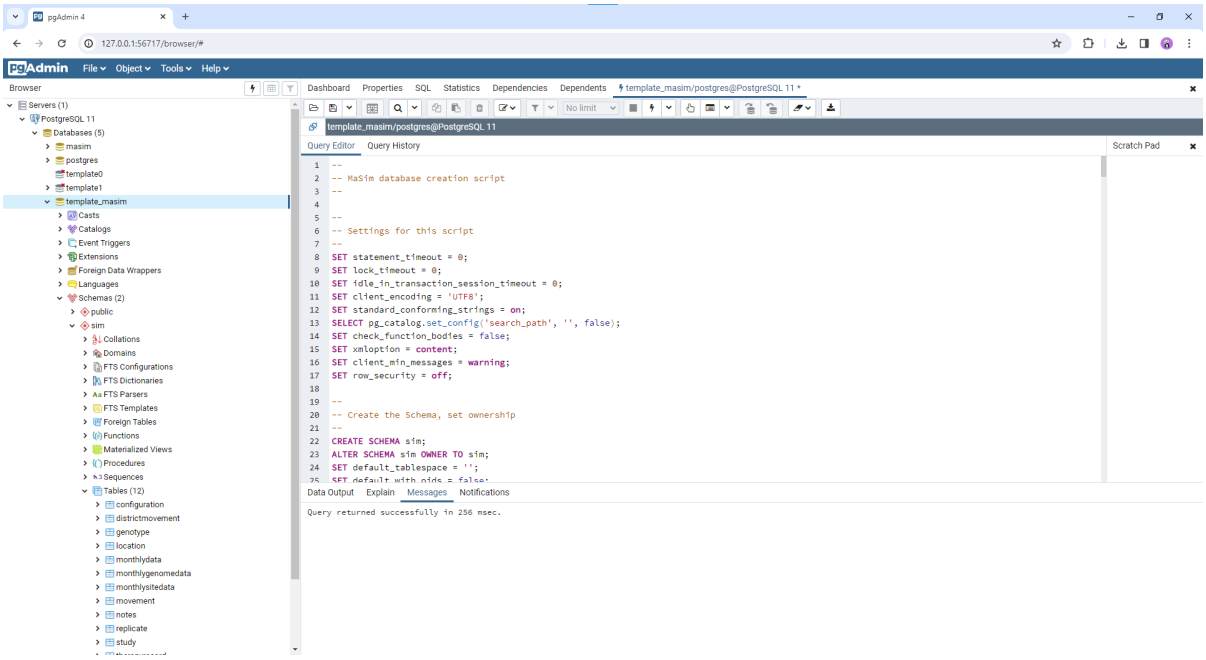


Figure 60: (ref:fig-pgadmin-script)

11.4 Backing Up Databases

In the event that a database backup is required then it is recommended to use the pgAdmin control panel to start the process via `Tools > Backup` when a database is selected. A dialog will appear that will allow a file name can then be supplied (ex., `backup.bak`) and by default the file will be written to `/var/lib/pgadmin/storage/USERNAME/backup.bak` where `USERNAME` is the pgAdmin login of the user that started the process. This is a protected directory so it is recommended to login to the server and move the backup to a different directory. This can be done as follows[35]:

```
cd ~
sudo su
mv /var/lib/pgadmin/storage/USERNAME/backup.bak .
exit
```

Once moved, `rsync` can be used to move the file:

```
rsync -a USERNAME@SERVER:backup.bak .
```

Where `USERNAME` is your login for the server, and `SERVER` is either the name or IP address of the server. Since `rsync` runs over `ssh` it defaults to port 22, but the port can be changed via:

```
rsync -rvz -e 'ssh -p PORT' -a USERNAME@SERVER:backup.bak .
```

For larger back-up files, the `--progress` and `--stats` arguments can be used to monitor progress. Note that in addition to the network speed, how fast it takes to sync the file will also depend upon the medium you are writing to, so high speed storage is recommended for large files in addition to a fast network speed.

11.5 Restoring Databases

Assuming that the database was dumped per the previous section, the database can then be restored as follows:

```
pg_restore -C -d DATABASE backup.bak
```

Where `DATABASE` is the name of the database to be *created* on the server.

12 Technicalities

Due to the complex nature of the simulation, there are a number of technical decisions that may impact the model performance or results. This chapter contains the technical documentation concerning these decisions.

12.1 Random Numbers

A single random number generator is used for the entire life cycle of the simulation The simulation class `Random` acts as a wrapper for GNU Scientific Library Random Number Generation and Random Number Distributions. The simulation uses the Mersenne Twister 19937 generator (`gsl_rng_mt19937`), an implementation of the Mersenne twister (Matsumoto and Nishimura 1998), which produces 32-bit numbers with a state size of 19937 bits. Unless a seed is provided by the configuration, the random number generator instance is seeded by `std::random_device` which produces a non-deterministic, uniformly-distributed integer value.

[35] Note that this requires using `sudo` to switch to the `root` administrative user, the amount of time spent in this mode should be limited to the minimal amount necessary to get the job done

12.2 Internal Events

Internal events are scheduled events with the simulation that are cannot be produced as a result of the YAML configuration. These are automatically created by the simulation when appropriate and may be of interest in understanding how the simulation works or how it can be modified.

12.2.1 `ProgressToClinicalEvent`

The `ProgressToClinicalEvent` is scheduled when an individual is infected by a parasite and the infection will progress to a clinical infection, and possible death. Upon the event triggering, the simulation first checks to see if the conditions for a clinical infection are still present (i.e., parasites still in their system) and that another clinical infection did not already trigger. If the conditions are acceptable then parasite density is increased to clinical levels, and the determination to seek treatment occurs. If the individual seeks treatment, then it is administered and record keeping takes place and the `TestTreatmentFailureEvent` is scheduled. Otherwise, recording keeping takes place and the individual immune response is replied upon to clear the infection. In the event that the dies due to the infection, the death is immediately recorded and any treatment is logged as a failure by `ReportTreatmentFailureDeathEvent` at the treatment failure testing day (`tf_testing_day`).

12.2.2 `ReportTreatmentFailureDeathEvent`

The `ReportTreatmentFailureDeathEvent` is produced by the `ProgressToClinicalEvent` when an individual receives treatment for malaria but dies. This event ensures that treatment failures are logged appropriately for the expected treatment time frame.

12.2.3 `TestTreatmentFailureEvent`

The `TestTreatmentFailureEvent` is created when an individual with clinical case of malaria elects to receive treatment and is scheduled to trigger on the configured treatment failure testing day (`tf_testing_day`), which is typically 28 days. Treatment failures are determined by testing to see if the parasite that caused the clinical case has a parasite density above the detectable limit, as configured by the `log_parasite_density_detectable` field. The results of this test are logged for both the individual's location and the associated therapy.

13 Reporters

The simulation exposes a number of different reporters that can be specified via the command line switch `-r` and listed via the `--lr` switch. Reporters currently write data either to a text file (CSV or TSV formatted) or to the database back-end.

Primary reporters write extensive amounts of information from the simulation, in contrast, *secondary* reporters tend to be limited in either what is reported, or have limitations on their use.

13.1 Reporter Types

13.1.1 Database Reporter (`DbReporter`)

Primary Reporter
Output: Database

The DbReporter, short for Database Reporter, uses the PostgreSQL database as the back-end and respects the record_genome_db setting in the configuration file when determining if data genotype data should be stored. If genotype data is stored, it is aggregated at the cellular level and can result in large databases.

13.1.2 Database Reporter by District (DbReporterDistrict)

Primary Reporter
Output: Database

The DbReporterDistrict is a derivative of the DbReporter, except genotype data will be aggregated to the district level.

13.1.3 Genotype Carriers Reporter (GenotypeCarriers)

Specialist Reporter
Output: Database

The Genotype Carriers Reporter (GenotypeCarriers) is a specialist reporter that works in tandem with either of the database reporters and expects to be run after all other reporting has been completed, which can be done by invoking the reporter as the last option in the switch (e.g., -r DbReporterDistrict,GenotypeCarriers). When the reporter is triggered it will report count of the first allele mutation on the second locus (i.e., 580Y, 561H, or 469Y).

When the reporter initializes it will check for the presence of the genotypecarriers column in sim.monthlysitedata which is where the absolute count of genotype carriers will be stored during model executing. If the column is not present, it can be added using the following SQL:

```
ALTER TABLE sim.monthlysitedata
ADD genotypecarriers integer;
```

13.1.4 Therapy Record Reporter (TherapyRecord)

Specialist Reporter
Output: Database

The Therapy Record Reporter (TherapyRecord) is a specialist reporter that works in tandem with either of the database reporters and expects to be run at any point after the primary reporter has completed, which can be done by invoking the reporter following the main reporter in the switch (e.g., -r DbReporterDistrict,TherapyRecord,GenotypeCarriers). The reporter starts reporting data after the start_collect_data_data is reached and will report the total number of treatments that completed in the current month along with the total numbers of success or failures.[36] The data is recorded in the sim.therapyrecord table with the locationid field corresponding to either the cell id, or the district id, in accordance with the primary reporter used (i.e., DbReporter or DbDistrictReporter).

While the sim.therapyrecord will be created when the canonical template database is cloned, but may not exist in legacy databases. As such, if the reporter is passed as a parameter to the simulation, when the reporter initializes it will check for the presence of the sim.therapyrecord table and will call the simulation to terminate if the table is not present.

[36] The total number of successes plus the total number of failures should always equal the total number completed.

13.1.5 Monthly Reporter (`MonthlyReporter`)

Primary Reporter
Output: TSV File

The Monthly Reporter `MonthlyReporter` is the original reporter used by the simulation to report the simulation status and produces two tab separated values (TSV) file as described below.

13.1.6 Seasonal Immunity Reporter (`SeasonalImmunity`)

Specialist Reporter
Output: CSV File

The Seasonal Immunity Reporter (`SeasonalImmunity`) is a specialist reporter that scans all locations and individuals and aggregates summary data at the climatic zone level. This is useful for determining the role that seasonal transmission patterns has upon the genotype evolution and immune response to the parasite. The reporter is hard coded for the first allele mutation on the second locus, so while it reports 580Y in the output, 561H could be captured, depending upon model configuration.

13.2 Reporter Data Files

13.2.1 Monthly Reporter

The `MonthlyReporter` generates two data files in tab separated values (TSV) format with group separators indicated by the sentinel value `-1111` where n is a zero-indexed location id.

Table 3: **monthly_reporter_*n*.txt** - Summary data for the model generated at the end of each simulated month.

Block	Column Number(s)	Description
Summary	1	Model time, number of days elapsed
	2	Model time, calendar date as system time
	3 - 5	Model time, calendar date (Year, Month, Day)
	6	Seasonal factor
	7	Treatment coverage, probability to be treated (0 - 1)
	8	Treatment coverage, probability to be treated (0 - 10)
	9	Population size
	10	Group separator
EIR and PfRP	$1 + (n * 5)$	EIR by location per year
	$2 + (n * 5)$	Group separator
	$3 + (n * 5)$	Blood slide prevalence, *Pf*PR < 5
	$4 + (n * 5)$	Blood slide prevalence, *Pf*PR 2 - 2
	$5 + (n * 5)$	Blood slide prevalence, *Pf*Pr all
Infections by Location	1	Group separator
	$2 + n$	Number of new infections by location
Treatments by Location	1	Group separator
	$2 + n$	Number of treatments by location
Clinical Episodes by Location	1	Group separator
	$2 + n$	Number of clinical episodes by location
Genotype Frequency	1	Group separator
	...	See genotype frequency discussion

Table 4: **summary_reporter_*n*.txt** Summary that is generated after the model has completed execution.

Block	Column Number(s)	Description
Summary I	1	Random seed value
	2	Number of locations
	3	*Beta* value
	4	Population size
EIR and PfRP	$1 + (n * 5)$	EIR by location per year
	$2 + (n * 5)$	Group separator
	$3 + (n * 5)$	Blood slide prevalence, *Pf*PR < 5
	$4 + (n * 5)$	Blood slide prevalence, *Pf*PR 2 - 2
	$5 + (n * 5)$	Blood slide prevalence, *Pf*Pr all
Summary II	1	Treatment strategy id
	2	Percent treatment failures per year

13.2.2 Genotype Frequency

Genotype frequency is a complex entry that follows a similar output structure, but is generated differently depending upon the approach selected.

Genotype frequency, by weighted number of parasite-positive individuals

```
FOR EACH location
  FOR EACH state per person
    FOR EACH age class per person
      IF the person has no parasites THEN CONTINUE
      Update the total count
      Count the number of genotypes per person
      Adjust the count for the region
    END
  END
  // Formats the genotype per location as TSV
  FOR EACH genotype
    OUTPUT << weighted value << '\t'
  END
END
OUTPUT << "-1111\t"
// Format the summary genotype results as TSV
FOR EACH genotype
  OUTPUT << weighted value << 't'
END
OUTPUT << "-1111\t"
OUTPUT << Total count
```

14 MaSim Configuration File

The simulation uses YAML to load configuration settings for the simulation. While the simulation was previously forward compatible (i.e., older files would work with newer versions), with the transition to version 4.0 new nodes were added or deprecated that have resulted in a divergence between 3.x and 4.0 on wards.

As a matter of convention, the YAML key is generally indicated with **bold** text. The data type (e.g., integer, string, etc.) or the possible values are indicated in parentheses following the key name with any default indicated in **bold**. Generally this document is organized such that headings are organized by operational impact upon the simulation, followed by simple key-value pairs as the first entry in each section, followed by more complex entities as subheadings. Within subsections, the YAML keys should be in alphabetical order, although closely coupled keys (e.g., `number_of_age_classes` and `age_structure`) will break this pattern with the first key that should be read appearing first.

14.1 Model Operation

The following nodes govern how the model executes in terms of simulation execution.

connection_string (string) : The connection string for the PostgreSQL database that stores the simulation data.

days_between_notifications (integer) : The number of model days that should elapse between status updates to the console.

record_genome_db (Boolean) : Indicates that genome data should be recorded to the database when using the `DbReporter` reporter class. Note that recording genomic data to the database will cause the database to quickly inflate in size. It is recommended that this setting only be used when genomic data needs to be retrieved.

14.2 Model Configuration

The following nodes contain the settings for the simulation.

artificial_rescaling_of_population_size (double) : A scaling value that should be applied to the population size in a given location. Defaults to 1.0, but 0.25 is commonly applied when geospatial data is used that maps locations to current populations.

birth_rate (float) : The number of births per thousand individuals in the simulation, expressed as a decimal (i.e., 42 births per 1000 is entered as 0.042).

initial_age_structure (integer array) : Used to initialize the population structure at model initialization (time zero).

initial_seed_number (integer) : The seed value that should be used by the random number generator. The default value of zero (0) indicates that the seed will be generated at model execution time based upon the number of milliseconds since the Unix epoch.

number_of_age_classes (integer) : The size of the `age_structure` array.
age_structure (integer array) : An array of integer values that corresponds to the oldest age that defines a break in the age structure. This age structure is used for reporting and age-specific mortality calculations.
death_rate_by_age_class (float array) : A float array of values that corresponds to the all-causes death rate for the simulation withe same index correspondence as `age_structure`. Typically, supplied as a malaria adjusted value.
mortality_when_treatment_fail_by_age_class (float array) : A float array of values that corresponds to the death rate when treatment fails, using the same index correspondence as `age_structure`.

number_of_tracking_days (integer) : The number of days to take the total number of parasites in the population.

report_frequency (integer) : The number of model days that various reporters will save data, and data aggregation events will trigger.

start_collect_data_day (integer) : The number of model days that should elapse before data collection begins (e.g., number of clinical episodes, number of deaths, etc.)

start_of_comparison_period (date string, YYYY/mm/dd) : The calendar date upon which the simulation should start calculating the number of treatment failures (NTF), artemisinin monotherapy usage (AMU), and useful therapeutic life (UTL). Note that as of version 4.0 the AMU results are considered to be *deprecated* and will be removed at a later date.

starting_date (date string, YYYY/mm/dd) : The start date of the simulation, model days elapsed will be indexed from this date.
ending_date (date string, YYYY/mm/dd) : The end date of the simulation.

14.3 Simulation Geography

14.3.1 raster_db

This node contains data related to the spatial organization of the model to include population distributions and raster files.

Usage
The use of the `raster_db` node will override the use of the `location_db` and errors or inconsistencies will occur if both are used at the same time. All the raster files must have the same header information as defined in the Esri ASCII raster format and the same number of defined pixels. A `std::runtime_error` may be generated if the number of data pixels in a given raster exceeds, or is less than a previously loaded raster.

Note that while any arbitrary value may be used for the `NODATA_VALUE` it is recommended that the standard value of `-9999` be used.

```
raster_db:
  beta_raster: "beta.asc"
  district_raster: "district.asc"
  location_raster: "location.asc"
  population_raster: "population.asc"
  travel_raster: "travel.asc"
  ecoclimatic_raster: "ecoclimatic.asc"
  pr_treatment_under5: "pr_treatment_under5.asc"
  pr_treatment_over5: "pr_treatment_over5.asc"

  cell_size: 5
  age_distribution_by_location: [[0.0]]
  p_treatment_for_less_than_5_by_location: [0.0]
  p_treatment_for_more_than_5_by_location: [0.0]

  beta_by_location: [0.0]
```

beta_raster (string) : The beta parameter (float) used for each cell in the model, overrides the value set in `beta_by_location`.
district_raster (string) : Contains the canonical identification values (integer) that should be for each cell in the model. The simulation follows the same convention as ArcGIS Pro in that the identification values should be sequential from one to n and will verify this when loading the ASC file. Zero indexed districts are supported and consistency is checked (i.e., ten districts would be numbered 0 to 9).
location_raster (string) : May contain arbitrary numeric values and is used by the simulation to convert the X-Y coordinates to their linear values. Only needed if it is the *only* raster file provided.
population_raster (string) : Contains the number of individuals (integer) in each cell within the simulation.
travel_raster (string) : Contains the friction surface (float) that is used for determining the likelihood of travel to a given cell.

ecoclimatic_raster (string) : Contains the ecoclimatic zone value for each cell. The ecoclimatic zone value should be a zero-indexed integer that corresponds to the `seasonal_info` value to use.
pr_treatment_under5 (string) : Contains the access to treatment (float) for individuals under five years (60 months) for each cell in the simulation, overrides `p_treatment_for_less_than_5_by_location`.
pr_treatment_over5 (string) : Contains the access to treatment (float) for individuals over five years (60 months) for each cell in the simulation, overrides `p_treatment_for_more_than_5_by_location`.

cell_size (float) : The size of each cell along one axis, in kilometers.
age_distribution_by_location : An array of arrays which contains floating point values corresponding to the age distribution (as defined by `age_structure`).
p_treatment_for_less_than_5_by_location, **p_treatment_for_more_than_5_by_location** : (*Deprecated in Version 4.0*) Contains the access to treatment for individuals under or over (respectively) five years (60 months). If the array length is one, then the value supplied (float) is used for all locations in the model. Otherwise, each element in the array is presumed to correspond to a different location.

beta_by_location : (*Array deprecated in Version 4.0, single value recommended or raster*) Contains the beta value (float) to be used in the simulation. If the array length is one, then the value supplied (float) is used for all locations in the model. Otherwise, each element in the array is presumed to correspond to a different location.

14.3.2 seasonal_info

This setting governs the malaria season in the model and operates differently depending upon the setting and version of the simulation.

14.3.2.1 Equation based

An equation based model of seasonal variation in transmission is provided where parameters must be fit to the following equation:

$$multiplier = base + \left(a \cdot sin^{+}\left(\frac{b \cdot \pi \cdot (t - \phi)}{365}\right)\right)$$

Once the equation is fit, the following YAML is used for the configuration:

```yaml
seasonal_info:
  enable: true
  mode: equation
  equation:
    raster: false
    base: [0.0]
    a: [0.0]
    b: [0.0]
    phi: [0.0]
```

mode (**equation** | rainfall) : (*Optional*) Indicates the node that should be used for the seasonality, namely based upon the equation based model, or by using rainfall data. In the event that a value is not supplied, the simulation will default to the equation based model.
enable (true | false) : Enables or disables seasonality in the simulation.
raster (true | false) : Indicates that a raster file should be used to set the correct rate for each pixel. When true each index in the array corresponds to an array index used by the cells in the raster. Otherwise the first value is used for all pixels.
base, **a**, **b phi** (double array) : Arrays of one to n `double` values that indicate the the variables in the seasonal period equation.

14.3.2.2 Rainfall Based

The rainfall based approach assumes that a model has already been fit and the adjustment values are stored in a CSV file.

```
seasonal_info:
  enable: true
  mode: rainfall
  rainfall:
    filename: filename.csv
    period: 365
```

mode (equation | **rainfall**) : Required in order to load the rainfall data.
filename (string) : The CSV file that contains the adjustment that should be applied to the beta. Each adjustment should be supplied on a single line in the file.
period (integer) : The period of time before the pattern in the CSV file should repeat, generally 365 days is expected.

14.3.3 spatial_model

The `spatial_model` setting defines movement model that transfers individuals between model cells. All spatial models used in the simulation have the same basic definition structure:

```
spatial_model:
  name: "ModelName"
  ModelName:
    paramter: value
```

where the `name` may be of the type `Marshall`, `Wesolowski`, or `WesolowskiSurface` which each have their own configuration values.

14.3.3.1 Marshall Movement Model

The Marshall movement model is based upon the gravity model described in Marshall et al. (2018) which presumes that the probability of a trip is defined by the proportional probability of movement from i to j such that $P(j|i) \propto N_j^\tau k(d_{i,j})$ where the kernel is defined by $k(d_{i,j}) = \left(1 + \frac{d_{i,j}}{\rho}\right)^{-\alpha}$ from the following configuration:

```
spaital_model:
  name: "Marshall"
  Marshall:
    tau: 1.0
    alpha: 1.0
    log_rho: 1.1
```

tau (double) : The calibrated value for τ.
alpha (double) : The calibrated value for α.
log_rho (double) : The calibrated $log_{10}(\rho)$ value.

14.3.3.2 Wesolowski Movement Model

THe Wesolowski movement model is based upon the gravity model described in Wesolowski et al. (2015) where the amount of travel to N_{ij} is defined by $N_{ij} = \frac{pop_i^{\alpha} pop_j^{\beta}}{d(i,j)^{\gamma}} \kappa$ from the following configuration:

```
spaital_model:
  name: "Wesolowski"
  Wesolowski:
    kappa: 1.0
    alpha: 1.0
    beta: 1.0
    gamma: 1.0
```

kappa (double) : The calibrated value for κ.
alpha (double) : The calibrated value for α.
beta (double) : The calibrated value for β.
gamma (double) : The calibrated value for γ.

14.3.3.3 Wesolowski Movement Model with Travel Surface

This movement model is similar to the generic Wesolowski movement model, except a travel surface penalty is applied to N_{ij} such that $N'_{ij} = \frac{N_{ij}}{(1+t_i+t_j)}$ using the `travel_raster` loaded as part of the `raster_db` and the following configuration:

```
spaital_model:
  name: "WesolowskiSurface"
  WesolowskiSurface:
    kappa: 1.0
    alpha: 1.0
    beta: 1.0
    gamma: 1.0
```

kappa (double) : The calibrated value for κ.
alpha (double) : The calibrated value for α.
beta (double) : The calibrated value for β.
gamma (double) : The calibrated value for γ.

14.4 Individual Immunity and Infection Response

allow_new_coinfection_to_cause_symtoms (*true* | false) : Flag to indicate if an asymptomatic host that is bitten and infected by a new parasite clone may present with new symptoms. Note the spelling of `symtoms` in the configuration.

transmission_parameter (float) : Governs how likely malaria is to be transmitted to a naive individual in the sporozoite challenge.

14.4.1 parasite_density_level

The `parasite_density_level` setting contains several sub-values that govern individual behavior or state due to the total number of parasites that are present in the individual's blood stream. When setting the parasite density for the detectable levels note that 10 per μl is the middle of the bounds for detection under

Giemsa-stained thick blood film under laboratory conditions (4 - 20 parasites/μl) while 50 per μl is the lower bounds for detection under field conditions (50 - 100 parasites/μl) (Wongsrichanalai et al. 2007). Generally, a higher detection limit for the PfPR will require a higher transmission for a given PfPR than a lower detection level.

```
parasite_density_level:
  # corresponds to 100 total parasites (0.00002 per ul)
  log_parasite_density_cured:          -4.699

  # corresponds to 50,000 total parasites (0.01 per ul)
  log_parasite_density_from_liver:     -2.000

  # corresponds to 1,000 parasites per microliter of blood
  log_parasite_density_asymptomatic:    3

  # corresponds to 20,000 parasites per microliter of blood (total 10^11)
  log_parasite_density_clinical:        4.301

  # corresponds to 2,000 parasites per microliter of blood (total 10^10)
  log_parasite_density_clinical_from:   3.301

  # corresponds to 200,000 parasites per microliter of blood (total 10^12)
  log_parasite_density_clinical_to:     5.301

  # corresponds to 10 parasites per microliter of blood
  log_parasite_density_detectable:      1.000

  # corresponds to 50 parasites per microliter of blood
  log_parasite_density_detectable_pfpr: 1.699

  # corresponds to 2,500 parasites per microliter of blood
  log_parasite_density_pyrogenic:       3.398
```

log_parasite_density_cured (double) : When an individual is considered to be **cured** of a specific parasite colony.
log_parasite_density_from_liver (double) : Governs the lower bound for the number of parasites following the initial infection.
log_parasite_density_asymptomatic (double) : Threshold at which an individual is asymptomatic of malaria.
log_parasite_density_clinical (double) : Governs the upper bound for the number of parasites following the initial infection.
log_parasite_density_clinical_from (double) : Governs the lower bound of parasites that an individual may be inflicted with when progressing to clinical via the `ProgressToClinicalEvent` event.
log_parasite_density_clinical_to (double) : Governs the upper bound of parasites that an individual may be inflicted with when progressing to clinical via the `ProgressToClinicalEvent` event.
log_parasite_density_detectable (double) : Sets the threshold for the number of parasites that an individual may have present in their blood when tested to check if the prescribed treatment failed.
log_parasite_density_detectable_pfpr (double) : Sets the threshold for the number of parasites that an individual may have present in their blood when tested to see if a *detectable* level is presented. This value is used to inform calculations for the PfPR in the simulation.
log_parasite_density_pyrogenic (double) : (**UNUSED**) Sets the threshold for when fever may present as a symptom.

14.4.2 immune_system_information

The `immune_system_information` node contains parameters that are used to control the response of the individual immune system and is one of the mechanisms by which the simulation can be calibrated to match a given country.

```
immune_system_information:
  # Immune function parameters
  b1: 0.00125
  b2: 0.0025

  # Duration of infection parameters
  duration_for_naive: 300
  duration_for_fully_immune: 60

  # Population initialization parameters
  mean_initial_condition: 0.1
  sd_initial_condition: 0.1

  # Probability bounds for clinical symptoms
  max_clinical_probability: 0.99

  # Immunity acquisition parameters
  immune_inflation_rate: 0.01
  age_mature_immunity: 10
  factor_effect_age_mature_immunity: 0.3

  # Immunity function parameters
  immune_effect_on_progression_to_clinical: 12
  midpoint: 0.4
```

b1 (double) : Rate of immune function increase when parasitaemic
b2 (double) : Rate of immune function decrease when not parasitaemic
duration_for_naive (double) : Duration, in days, of infection when naive.
duration_for_fully_immune (double) : Duration, in days, of infection when fully immune.
mean_initial_condition (double) : Mean initial immune function of population at initialization.
sd_initial_condition (double) : Standard deviation of initial immune function of population at initiation.
max_clinical_probability (double) : Maximum probability of clinical symptoms as a result of a new infection.
immune_inflation_rate (double) : Yearly age-dependent faster acquisition of immunity between ages 1 to 10.
age_mature_immunity (double) : Age at which the immune function is mature, i.e., age at which the immune acquisition model switches from child to adult.
factor_effect_age_mature_immunity (double) : Adjustment to the curve of immune acquisition under the age indicated by `age_mature_immunity`, parameter kappa in supplement to Nguyen et al. (2015).
immune_effect_on_progression_to_clinical (double) : Slope of the sigmoidal probability versus immunity function, parameter z in supplement to Nguyen et al. (2015).
midpoint (double) : Adjusts the midpoint of the slope of the sigmoidal probability versus immunity function, parameter z in supplement to Nguyen et al. (2015).

14.5 Treatments

The following nodes govern how drugs and treatments are managed by the simulation.

tf_testing_day (integer) : The number of days following the administration of a therapy that a individual should be tested for treatment failure.

14.5.1 drug_db

This setting is used to configure the various drugs used in the configuration and is structured as an array of drugs, which contain the specific setting for that particular compound. **Note** that while the simulation assumes that the compounds will be ordered from 0 to n, it is the responsibility of the user to ensure that they are assigned and ordered correctly.

```
drug_db:
  0:
    name: "ART"       # Artemisinin
    half_life: 0.0
    maximum_parasite_killing_rate: 0.999
    n: 25
    age_specific_drug_concentration_sd: [0.4,0.4,0.4,0.4,0.4,0.4,0.4,0.4,0.4,
                                         0.4,0.4,0.4,0.4,0.4,0.4]
    age_specific_drug_absorption: [0.7,0.7,0.85,0.85,0.85,0.85,0.85,0.85,0.85,
                                   0.85,1.0,1.0,1.0,1.0,1.0]
    mutation_probability: 0.005
    affecting_loci: [2]
    selecting_alleles: [[1]]
    k: 4
    EC50:
      ..0..: 0.75
      ..1..: 1.2
```

name (string) : The name of the compound.
half_life (double) : The compound's half-life in the body, in days.
maximum_parasite_killing_rate (double) : The percentage of parasites that the compound will kill in one day if an individual has the highest possible drug concentration.
n (integer) : The slope of the linear portion of the concentration-effect curve.
age_specific_drug_concentration_sd (double array) : The actual drug concentration, per individual, that will be drawn from a normal distribution with a mean of one and this standard deviation.
age_specific_drug_absorption (double array) : The percentage of the drug that is absorbed into the bloodstream, based upon the age of the individual. When not supplied the default value is one for all age groups.
mutation_probability (double) : The probability that exposure to the drug will result in a mutation in the parasite to resist it.
affecting_loci (integer array) : The index of the loci of alleles where drug resistance may form (see genotype_info).
selecting_alleles (integer matrix) : The index of the alleles where drug resistance may form (see genotype_info).
k (double) : Controls the change in the mutation probability when drug levels are intermediate. For example, k=0.5 is a simple linear model where mutation probability decreases linearly with drug concentration; whereas k=2 or k=4 are a piecewise-linear model where mutation probability increases from high concentrations to intermediate concentrations, and then decreases linearly from intermediate concentrations to zero.
EC50 (array of key-value pairs) : The drug concentration which produces 50% of the parasite killing achieved at maximum-concentration, format is a string that describes the relevant genotypes (see genotype_info), followed by the concentration where 1.0 is the expected starting concentration.

14.5.2 therapy_db

This setting is used to define the various therapies that will be used in the simulation and two variations are supported: simple therapies that consist of one or more drugs (defined using the the `id` from the `drug_db`) given over a number of days, and complex therapies that consist of one or more therapies (defined using the `id` of the previously defined therapy) given over a regimen.

```
therapy_db:
  # Artemisinin combination therapy (ACT) - artemether-lumefantrine (AL), three days
  0:
    drug_id: [0, 1]
    dosing_days: [3]
  # ACT - AL, one day
  1:
    drug_id: [0, 1]
    dosing_days: [1]

  # Complex therapy, AL dosed three days, one day off, with one final dose (3-1-1)
  2:
    therapy_ids: [0, 1]
    regimen: [1, 5]

  # Artemisinin combination therapy (ACT) - artemether-lumefantrine (AL),
  # five days with specified compliance
  3:
    drug_id: [0, 1]
    dosing_days: [5]
    # Probability that an individual will complete exactly this many days of treatment
    pr_completed_days: [0.1, 0.1, 0.2, 0.2, 0.4]
```

14.5.2.1 Simple Therapies

drug_id (integer array) : One or more integers that correspond to the defined identification numbers (i.e., array index) in the `drug_db`.
dosing_days (integer) : The number of days that the drug combination should be given for.
pr_completed_days (float array) : (*Optional*) The probability that the individual will comply with the course of treatment where 1 indicates they will always take it; otherwise, $0 < n < 1$ is the probability that they will stop on that day. In the event that the field is not supplied then it is assumed that the individual will always comply with the therapy.

14.5.2.2 Complex Therapies

Complex therapies consist of multiple simple therapies that are dosed over several days and may contain gaps in the dosing. Prior to Version 4.1.1 patient compliance with the therapy was determined by the `p_compliance` which generally assumed full compliance with the course of treatment. With complex therapies, noncompliance is currently not support and such configurations will produce an error.

therapy_ids (integer array) : One or more integers that correspond to the defined therapies.
regimen (integer array) : A one-index list of the days that the corresponding therapy should be given.

14.5.3 strategy_db and initial_strategy_id

This setting describes the treatment strategy applied when an individual presents with a clinical case of malaria and typically consists of at least two strategies: an initial 'baseline' that is consistent with

the national first-line therapy and one or more interventions that represent the various drug policy approaches that are being applied. Following model initialization and burn-in following the strategy defined in `initial_strategy_id`, a new strategy can be employed using the `change_treatment_strategy` which takes effect upon execution of the event.

Within the `strategy_db` the indexing should begin at zero although the `initial_strategy_id` allows for any defined strategy to be used. The `type` field used for defining the strategies comes from the `IStrategy::StrategyTypeMap` and are thus case-sensitive. Failure to supply a valid strategy will result in the `StrategyBuilder` logging a `WARNING` before returning `nullptr`.

```yaml
strategy_db:
  0:
    name: Baseline
    type: MFT
    therapy_ids: [ 0 ]
    distribution: [ 1 ]
  1:
    name: ExampleMFT
    type: MFT
    therapy_ids: [ 1, 2 ]
    distribution: [ 0.5, 0.5 ]
  2:
    name: ExampleCycling
    type: Cycling
    therapy_ids: [ 1, 2 ]
    cycling_time: 90
initial_strategy_id: 0
```

initial_strategy_id (integer) : The integer index of the treatment strategy to use from day zero of the simulation.

14.5.3.1 Cycling Strategy

A cycling strategy calls for the rotation between two or more single first-line therapies (e.g., artemether–lumefantrine and artesunate–amodiaquine) on a strict fixed schedule (e.g., every 90 days as opposed to quarterly). As implemented the simulation supports an unlimited number of days and will return to the first therapy defined in `therapy_ids`. Bookkeeping related to when cycling events occur are automatically managed by the simulation as part of the `Model::daily_update` function.

name (string) : A string name used to refer to the treatment in console output and logs.
type (string) : Always `Cycling`
therapy_ids (integer array) : The therapies to deploy, based upon the index of the therapies defined in `therapy_db`.
cycling_time (integer) : The number of days that a therapy is used before switching.

14.5.3.2 Multiple First Line Therapies (MFT)

The multiple first-line therapies (MFT) approach calls for two or more therapies to be deployed nationally with an individual randomly receiving one at time of treatment, which they then use for the entire course of treatment. As a matter of convention, when a single first-line therapy is deployed, the MFT strategy *should* still be used in the configuration over the deprecated `SFT` (single first-line therapy) strategy and the `MFTStrategy` class works correctly when only a single therapy is defined.[37]

[37] The `SFT` strategy is preserved for use with the DxG Generator, but should not be used in simulations.

name (string) : A string name used to refer to the treatment in console output and logs.
type (string) : Always `MFT`
therapy_ids (integer array) : The therapies that may be given, based upon the index of the therapies defined in `therapy_db`.
distribution (float array) : The probability that an individual will receive the therapy corresponding to the same index where the total of all values provided *must* equal one.

14.5.4 District MFT Strategy

The district MFT strategy approach allows for different MFT approach to be targeted on a regional bases based upon the district id provided. These district ids should align with those defined in the `district_raster`. Upon initialization of the strategy when the configuration file is read, the strategy is checked for errors which includes: all district ids are used, all therapies actually exist, and the total sum of the distribution is 100%.

```
strategy_db:
  0:
    name: Baseline
    type: MFT
    therapy_ids: [ 0 ]
    distribution: [ 1 ]
  1:
    name: ExampleDistrictMFT
    type: DistrictMFT
    definitions:
      0:
        district_ids: [ 1, 2, 3 ]
        therapy_ids: [ 1, 2 ]
        distribution: [ 0.75, 0.25 ]
      1:
        district_ids: [ 4, 5, 6 ]
        therapy_ids: [ 1, 2 ]
        distribution: [ 0.5, 0.5 ]
      2:
        district_ids: [ 7, 8, 9 ]
        therapy_ids: [ 1, 2 ]
        distribution: [ 0.25, 0.75 ]
```

name (string) : A string name used to refer to the treatment in console output and logs.
type (string) : Always `DistrictMFT`
definitions : Indicates the start of a new list of definitions numbered zero to *n*.
district_ids (integer array) : The ids of the districts where the therapies may be given.
therapy_ids (integer array) : The therapies that may be given, based upon the index of the therapies defined in `therapy_db`.
distribution (float array) : The probability that an individual will receive the therapy corresponding to the same index where the total of all values provided *must* equal one.

14.6 Policy Interventions

While most of the policy interventions can be implemented using the therapies deployed, and switching them via events such as `change_treatment_strategy`, some of the more complex strategies require more in-depth configuration or are closely coupled with how the simulation operates.

14.6.1 Regular administration of prophylactic therapy (Untested)

The regular administration of prophylactic therapy, or RAPT protocol, is a speculative approach to malaria control that calls for individual to take an artemisinin combination therapy (ACT) periodically without the presentation of clinical symptoms for malaria. The protocol presumes an individual will remember the month to take then next ACT, and at some point during that month a check will be performed to see if they take the therapy. The probability is based upon the probability that an individual in their age group will seek treatment and the probability of compliance with the RAPT protocol (i.e., $Pr(RAPT) = Pr(Treatment) \cdot Pr(Compliance)$). In the event that the individual already took an ACT in the past 28 days (determined by checking for `TestTreatmentFailureEvent`), they will not take the ACT regardless.

Implementation of this protocol requires that events be scheduled for at model initialization, although the point at which the individuals start taking the ACTs is determined by the configuration. Due to the computational overhead involved with the RAPT protocol, the simulation execution time is longer. If the `rapt_config` entry is not present in the configuration file, the event will not be enabled within the simulation.

NOTE that the RAPT protocol is highly experimental and the event has not been extensively tested.

```yaml
rapt_config:
  period: 12
  therapy_id: 1
  compliance: 0.7
  age_start: 18
  start_day: 2022/08/18
```

period (int): the interval, in months, between RAPT doses.
therapy_id (int): corresponds to id of the treatment defined in the `therapy_db` to take.
compliance (float): the probability that the individual will comply with the RAPT protocol.
age_start (int): the age, in years, that the individual will start checking for compliance with the RAPT protocol.
start_day (date string, YYYY/mm/dd): the date at which the RAPT protocol will take effect, after this date, individuals will start checking for compliance.

14.7 Genotype Information

14.7.1 genotype_info

This node defines the genotype database that is used by the simulation for the mutation pathways, looking up the drug sensitivity and resistance, and the cost that the parasite incurs due to the mutation. The node itself is made up of multiple loci which in turn have multiple allele attached to them. The loci are zero indexed and values should not be repeated; however, while the alleles are zero indexed, the index count restarts with each locus.

```yaml
genotype_info:
  loci:
    - locus_name: "pfcrt"
      position: 0
      alleles:
        - value: 0
          allele_name: "K76"
          short_name: "K"
          can_mutate_to: [1]
          mutation_level: 0
          daily_cost_of_resistance: 0.0
```

```
        - value: 1
          allele_name: "76T"
          short_name: "T"
          can_mutate_to: [0]
          mutation_level: 1
          daily_cost_of_resistance: 0.0005
    - locus_name: "K13 Propeller"
      position: 1
      alleles:
        - value: 0
          allele_name: "C580"
          short_name: "C"
          can_mutate_to: [1]
          mutation_level: 0
          daily_cost_of_resistance: 0.0
        - value: 1
          allele_name: "580Y"
          short_name: "Y"
          can_mutate_to: [0]
          mutation_level: 1
          daily_cost_of_resistance: 0.0005
```

locus_name (string) : the name of the locus, which in turn will have `alleles` attached to it.
position (integer) : the index of the locus, zero-indexed.
alleles (YAML) : the YAML node that the allele associated with this locus will be attached to.
value (int) : the index of the allele within this locus, zero indexed.
allele_name (string) : the name of the allele.
short_name (character) : the character code that is assigned to the allele.
can_mutate_to (integer array) : an array of the indexes of the alleles that this position can mutate to.
mutation_level (integer) : the mutation level of the allele, the higher the number, the more mutations are necessary to reach the allele (i.e., a mutation level of three indicates that the allele must mutate from A to B to C before reaching D).
daily_cost_of_resistance (float) : the fitness cost to the parasite to maintain the mutation.

14.8 Events

The `events` setting is used to list the various events that will be loaded and run during the model. The `name` field dictates which event will be parsed and all the data for the `info` field following will be provided to the loader function.

14.8.1 annual_beta_update_event

The annual beta update event increases or decreases the beta for each cell in the model using the formula $beta' = beta + (beta \cdot rate)$ and clamps the lower bounds for the beta a zero.

```
events:
  - name: annual_beta_update_event
    info:
      - day: 2020/10/26
        rate: -0.025
```

day (date string, YYYY/mm/dd) : the date when the update will first occur.
rate (float) : the rate of change for the beta for the cells.

14.8.2 annual_coverage_update_event

The annual coverage update event increases the coverage for each cell in the model by reducing the coverage gap by a fixed value provided by rate. If the model is run for a long enough period of time, the presumption should be that the coverage will reach 100%.

```
events:
  - name: annual_coverage_update_event
    info:
      - day: 2020/09/02
        rate: 0.025
```

day (date string, YYYY/mm/dd) : The date when the update will first occur.
rate (float) : The rate of reduction in the coverage for the cells.

14.8.3 change_circulation_percent_event

This event changes the `circulation_percent` value of the `circulation_info` setting block. The percentage of population that moves to a new location on a daily basis is changed, with the change remaining in effect for the remainder of the simulation.

```
events:
  - name: change_circulation_percent_event
    info:
      - day: 2023/12/22
        circulation_percent: 0.0042
```

day (date string, YYYY/mm/dd) : The date when the update will first occur.
circulation_percent (float) : The new daily percentage of the population that will move to a new cell.

14.8.4 change_treatment_coverage

Change the current treatment coverage to be either `SteadyTCM`, `InflatedTCM`, or `LinearTCM`. Passing an invalid treatment coverage name will result in a run time error, and the YAML depends on the treatment coverage model named. Since the simulation defaults to the `SteadyTCM` it is recommended that if there are geographic scale differences in treatment coverage that the `pr_treatment_under5` and `pr_treatment_over5` rasters be used via the `raster_db` setting.[38]

14.8.4.1 Steady Treatment Coverage Model

The steady treatment coverage model is the simulation default and maintains a fixed treatment coverage unless updated by the `annual_coverage_update_event`.

[38]This event is largely provided for legacy functionality or very small simuations since the ablity to supply rasters via the event is not supported.

```yaml
events:
  - name: change_treatment_coverage
    info:
      - type: SteadyTCM
        day: 2023/09/26
        p_treatment_less_than_5: [0.1]
        p_treatment_more_than_5: [0.1]
```

type (string): Must be `SteadyTCM` to use this treatment coverage model.
day (date string, YYYY/mm/dd) : The date when this treatment coverage model will begin.
p_treatment_less_than_5 (float array) : An array of floating point values that correspond to the treatment coverage for under five (< 5) in each cell.
p_treatment_more_than_5 (float array) : An array of floating point values that correspond to the treatment coverage for under five (< 5) in each cell.

14.8.4.2 Inflated Treatment Coverage Model

```yaml
events:
  - name: change_treatment_coverage
    info:
      - type: InflatedTCM
        day: 2023/09/26
        annual_inflation_rate: 0.03
        p_treatment_less_than_5: [0.1]
        p_treatment_more_than_5: [0.1]
```

type (string): Must be `InflatedTCM` to use this treatment coverage model.
day (date string, YYYY/mm/dd) : The date when this treatment coverage model will begin.
annual_inflation_rate (float) : The inflation rate to apply to the treatment converge until it reaches 100%.
p_treatment_less_than_5 (float array) : An array of floating point values that correspond to the treatment coverage for under five (< 5) in each cell.
p_treatment_more_than_5 (float array) : An array of floating point values that correspond to the treatment coverage for under five (< 5) in each cell.

14.8.4.3 Linear Treatment Coverage Model

The linear treatment coverage model increases the treatment coverage by a linear amount, calculated at run time, with a monthly update based upon the start and end dates.

```yaml
events:
  - name: change_treatment_coverage
    info:
      - type: LinearTCM
        from_day: 2023/09/26
        to_day: 2024/01/01
        p_treatment_for_less_than_5_by_location_from: [0.1]
        p_treatment_for_more_than_5_by_location_from: [0.1]
        p_treatment_for_less_than_5_by_location_to: [0.3]
        p_treatment_for_less_than_5_by_location_to: [0.2]
```

type (string): Must be `LinearTCM` to use this treatment coverage model.
from_day (date string, YYYY/mm/dd) : The date when the update will start to occur.
to_day (date string, YYYY/mm/dd) : The date when the updates will end.
p_treatment_for_less_than_5_by_location_from (float array) : An array of floating point values that correspond to the initial treatment coverage for under five (< 5) in each cell.
p_treatment_for_more_than_5_by_location_from (float array) : An array of floating point values that correspond to the initial treatment coverage for under five (< 5) in each cell.
p_treatment_for_less_than_5_by_location_to (float array) : An array of floating point values that correspond to the initial treatment coverage for under five (< 5) in each cell.
p_treatment_for_less_than_5_by_location_to (float array) : An array of floating point values that correspond to the initial treatment coverage for under five (< 5) in each cell.

14.8.5 change_treatment_strategy

Change the treatment strategy that is currently deployed in the simulation to the one defined in `strategy_db` with the given index. Simple error checking is performed upon model initialization and the `strategy_id` must be within bounds of the `strategy_db`.

```
events:
  - name: change_treatment_strategy
    info:
      - day: 2024/1/1
        strategy_id: 1
```

day (date string, YYYY/mm/dd) : The date when the update will first occur.
strategy_id (integer) : The index of the strategy defined in `strategy_db`.

14.8.6 importation_periodically_random_event

Over the course of the month indicated (January:1 - December:12) introduce the infection into the model at a random location, selected by a draw that is weighted by the population. Each day an infection is introduced based upon a uniform draw against `count / [days in month]`. Once started, this event continues until model termination.

```
events:
  - name: importation_periodically_random_event
    info:
      - day: 2021/08/01
        genotype_id: 1
        count: 10
        log_parasite_density: 4.301
      - day: 2021/09/01
        genotype_id: 2
        count: 20
        log_parasite_density: 3.0
```

day (date string, YYYY/mm/dd) : The first day of the month for which the event will occur.
genotype_id (int) : The id of the genotype to be introduced.
count (int) : The number of cases to be introduced in the month.
log_parasite_density (double) : the log density of the parasite to be imported.

14.8.7 introduce_mutant_event

On the specified date, find infected individuals and force the parasite genotype from the given wild type to mutation specified (e.g., C580 to 580Y). This operation will fill the difference between the input fraction and the current frequency of the genotype in the population. **Note** that while this is a one time event, it is recommended that the event be invoked multiple times prior to any policy interventions acting upon a given mutation frequency.

```
events:
  - name: introduce_mutant_event
    info:
      - day: 2021/12/27
        district: 9
        fraction: 0.01
        locus: 2
        mutant_allele: 1
```

day (date string, YYYY/mm/dd) : The model date when this event should occur.
district (int) : The district id for where the mutation event should occur.
fraction (int) : The target frequency of the mutation.
locus (int) : The genotype database locus index of the desired mutation.
mutant_allele (int) : The genotype database allele index of the mutation allele that will be applied.

14.8.8 introduce_mutant_raster_event

Similar to the `introduce_mutant_event` in that infected individuals will be switched from teh wild type to the mutant genotype indicated; however, the selection of individuals is based upon the raster provided as opposed to a district number.

This event follows the "fail fast" paradigm and as part of event initialization during model initialization the raster is loaded and checked to ensure that the cells indicated correspond to populated locations. Determination of cell locations uses the same algorithm as all other raster processing in the simulation, and as a result simulated cells must be labeled as either zero (no mutations) or one (mutations).

```
events:
  - name: introduce_mutant_raster_event
    info:
      - day: 2021/12/27
        raster: locations.asc
        fraction: 0.01
        locus: 2
        mutant_allele: 1
```

day (date string, YYYY/mm/dd) : The model date when this event should occur.
raster (string) : An ASC file indicating the locations where the mutations should take place. A cell value of zero indicates no mutations, while a value of one indicates that mutations should occur.
fraction (int) : The target frequency of the mutation.
locus (int) : The genotype database locus index of the desired mutation.
mutant_allele (int) : The genotype database allele index of the mutation allele that will be applied.

14.8.9 rotate_treatment_strategy_event

When initially called, will change the current treatment strategy that is currently deployed in the simulation to the one defined in the `strategy_db` with the given index; however, upon the defined number of `years` it

will change to the second one. This is followed by a regular rotation back and forth based upon the defined strategies and number of `years`. The event uses a fixed value of 365 days for one year, and rotations must be at least one year apart. Simple error checking is performed upon model initialization to ensure that values are within bounds.

```
  - name: rotate_treatment_strategy_event
    info:
      - day: 2023/12/14
        years: 2
        first_strategy_id: 1
        second_strategy_id: 2
```

day (date string, YYYY/mm/dd) : The date when the first rotation will occur. **years** (int) : The number of years between rotations, where a year is 365 days. **first_strategy_id** (int) : The `strategy_db` id of the first treatment strategy to change to. **first_strategy_id** (int) : The `strategy_db` id of the second treatment strategy to change to.

14.8.10 turn_off_mutation

Turn off all mutations in the model, recommended during the model burn-in.

```
events:
  - name: turn_off_mutation
    info:
      - day: 2020/09/04
```

day (date string, YYYY/mm/dd) : the date when the event will occur.

14.8.11 turn_on_mutation

Turn on all mutations in the model, or the mutations for individual drugs.

```
events:
  - name: turn_on_mutation
    info:
      - day: 2020/09/04
        drug_id: 0
        mutation_probability: 0.005
```

day (date string, YYYY/mm/dd) : the date when the event will occur.
drug_id (integer) : the id of the drug, as defined in the `drug_db` or `-1` to apply the value to all drugs.
mutation_probability (float) : the mutation probability to use.

14.8.12 update_beta_raster_event

Update all beta values in the simulation to those in the raster file.

```
events:
  - name: update_beta_raster_event
    info:
      - day: 2023/02/24
        beta_raster: "beta_2023.asc"
```

day (date string, YYYY/mm/dd) : the date when the event will occur.
beta_raster (string) : The beta parameter (float) used for each cell in the model, overrides the current value in the cell.

14.8.13 update_ecozone_event

Update all the cells matching the original ecozone to the new ecozone.

```
events:
  - name: update_ecozone_event
    info:
      - day: 2021/01/23
        from: 0
        to: 1
```

day (date string, YYYY/mm/dd) : the date when the event will occur.
from (integer) : the id of the ecozone, defined in `seasonal_info`, that is the original ecozone.
to (integer) : the id of the new ecozone, defined in `seasonal_info`, that will be applied to matching cells.

Appendicies

A Data Sources

In addition to national data sources (e.g., National Malaria Control Programs, Census Reports, etc.) the following data sources may be useful in parameterization of the simulation for a country or as a broader resource for malaria modeling.

A.1 Simulation Data

The DHS Program: Demographic and Health Surveys - https://dhsprogram.com/

Sponsored by USAID, the DHS Program conducts periodic demographic and health surveys (DHS survey[39]) in countries throughout the world, with malaria endemic countries well represented. Typically surveys are conducted at a greater frequency than national census and health questions relevant to malaria are included.

Malaria Atlas Project - https://malariaatlas.org/

Sponsored by Telethon Kids Institute and Curtin University, the Malaria Atlas Project (MAP) is considered one of the reference sources for malaria prevalence projections with the malaria modeling community. As such, MAP is a good starting point for preparing a model calibration; however, the data should be cross referenced with any locally acquired data since MAP projections can be off due to publishing delays or how the spatial interpolation model used by MAP works.

Spatial Data Repository: The DHS Program - https://spatialdata.dhsprogram.com/home/

Sponsored by USAID and managed by the DHS Program, the Spatial Data Repository is another means of approaching the data gathered during a DHS survey. This can also be a useful site for finding shapefiles that define districts within a country.

The World Bank: Data Catalog - https://datacatalog.worldbank.org/home

While lacking malaria data, the World Bank's Data Catalog can be a very useful resource for shapefiles (e.g., national and sub-national boundaries) as well as some statistical data. Data is typically published under an open data license (e.g., Creative Commons Attribution 4.0), although it should be double checked to ensure that all of the relevant attribution data is present.

World Population Prospects - https://population.un.org/wpp/

A key part of preparing the simulation for calibration is ensuring that the simulated population is a reasonable proxy for the real population, this includes ensuring that the mortality rate are set correct. In the event that data is not available from a canonical source within the country itself, the United Nation's World Population Prospects is a reasonable source to start with.

[39] While this turn of phrase is redundant — much like a NIC card (network interface card card) — this is a common way of referring to the surveys in the malaria modeling community.

A.2 Research Data

ACT Watch: Evidence for Malaria Medicine Policy - https://www.actwatch.org/

Managed by Population Services International (PSI), this website hosts data from from the ACTwatch and FPwatch projects. The ACTwatch project typically has some useful data, but is much more geographically constrained compared to the DHS Program. However, the site also has the antimalarial database, which is a comprehensive listing of all antimalarials, typically with photographs of the packaging as well.

African Journals Online (AJOL) - https://www.ajol.info/index.php/ajol

African Journals Online (AJOL) is a non-profit that receives funding from the Swedish International Development Cooperation Agency (Sida) and the Wellcome Trust, and acts as an aggregator of all African research journals. This includes journals that are not index by other sources such as Google Scholar. While the quality of individual journals can be a bit uneven, AJOL does provide resources assessing the journal quality. Typically as long as good scientific judgement is applied when reviewing a manuscript, this can be a very good resource for gathering additional data on the malaria situation in a given country.

Famine Early Warning Systems Network (FEWS NET) - https://fews.net/

Proper nutrition can play an important role in malaria in terms of ensuring that drugs are properly absorbed by the body, but also in supporting the immune systems ability to clear or resist an infection. While not a commonly used reference for malaria modeling, this can be a useful resource for helping to understand the context, and also includes a fair amount of geospatial data can be potentially used to inform the simulation.

FAO Map Catalog - https://data.apps.fao.org/map/catalog/srv/eng/catalog.search

The Map Catalog of the Food and Agriculture Organization (FAO) of the United Nations is situationally useful for acquiring geospatial data that can be used as part of analysis or in preparing cartographic products for reports. However, care should be taken since it can be unclear what the underling license of a given geospatial product is, which can interfere with publishing.

The Humanitarian Data Exchange (HBX) - https://data.humdata.org/

The Humanitarian Data Exchange (HBX) is managed by the United Nations Office for the Coordination of Humanitarian Affairs (OCHA) and is another useful site for geospatial data, with the majority covered by an open data license that allow for use in published materials.

B Genotype Information

Genotypes are encoded into the application via the input file which contains a YAML entry for `genotype_info`. This structure is organized as `loci` with the associated `alleles` that may be mutated during model execution. The following table outlines loci and alleles that are included in the sample configuration file.

Table 5: Genotype encoding scheme

Locus	Allele	Short Name	Description
PfCRT	K76	K	Chloroquine sensitivity, Amodiaquine sensitive, Lumefantrine resistance
	76T	T	Chloroquine resistance, Amodiaquine resistance, Lumefantrine sensitive
PfMDR1	N86 Y184 one copy of pfmdr1	NY–	Lumefantrine resistance, Amodiaquine sensitive
	86Y Y184 one copy of pfmdr1	YY–	Amodiaquine resistance
	N86 184F one copy of pfmdr1	NF–	lumefantrine resistance
	86Y 184F one copy of pfmdr1	YF–	Amodiaquine resistance, Lumefantrine resistance
	N86 Y184 2 copies of pfmdr1	NYNY	lumefantrine resistance, Amodiaquine sensitive, Mefloquine resistance
	86Y Y184 2 copies of pfmdr1	YYYY	Amodiaquine resistance, Mefloquine resistance
	N86 184F 2 copies of pfmdr1	NFNF	lumefantrine resistance, Mefloquine resistance
	86Y 184F 2 copies of pfmdr1	YFYF	Amodiaquine resistance, Lumefantrine resistance, Mefloquine resistance
K13 Propeller	C580	C	Artemisinin sensitive
	580Y	Y	Artemisinin resistance
Plasmepsin 2-3	Plasmepsin 2-3 one copy	1	Piperaquine sensitive
	Plasmepsin 2-3 2 copies	2	Piperaquine resistant
Hypothetical locus for multiple use	naïve	x	Experimental use in the model
	mutant	X	Experimental use in the model

The short name field requires additional note since genotype results generated by the model are based upon the short names. For example:

KNY–C1x

Indicates that the parasite has the K76 allele from the PfCRT locus (K), N86 Y184 one copy of pfmdr1 (NY–), C580 from K13 Propeller locus (C), Plasmepsin 2-3 one copy (1), and is a naïve copy of the hypothetical locus (x).

C Penn State Specifics

C.1 Building on ICDS-ACI, Roar Collab

To build the simulation to run on the ICDS-ACI Roar Collab cluster the first time it is necessary to perform a number of configuration steps. After logging on to the interactive environment (`submit.hpc.psu.edu`) and cloning this repository, run `config.sh` which will prepare the build environment. As part of the process a build script will be created at `build/build.sh` that will ensure the environment is set correctly to compile the code base.

When the `config.sh` script is done running, note that it will recommend changes to the `.bashrc` which can be edited via `vi ~/.bashrc` and adding lines similar to the following:

```
# Configure run time environment
module use /storage/icds/RISE/sw8/modules
module load gsl
LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/storage/home/USERNAME/work/build_env/postgres/lib
```

Running the `config.sh` script will report the relevant lines at the end of the script.

C.2 Cluster Runs

The Roar Supercomputer Users' Guide providers a good overview for running single replicates on the cluster; however, when running batches it is recommended to script out the process. When replicates need to be run with a variety of settings (e.g., sensitivity analysis) some scripts present in PSU-CIDD-MaSim-Support under the `bash` directory can be used to parse a CSV formatted list of replicates to be run. In addition to the `calibrationLib.sh` file the support repository, the following files need to be created for this:

1) A runner script which will be queued on the cluster as a job, typically named `run.sh` or similar in project repositories:

```
#!/bin/bash
source ./calibrationLib.sh
runReplicates 'replicates.csv' '[USERNAME]'
```

2) The Portable Batch System (PBS) file that defines the job for `run.sh`:

```
#!/bin/bash

#PBS -A [ALLOCATION]
#PBS -l nodes=1:ppn=1:rhel7:stmem
#PBS -l pmem=4gb
#PBS -l walltime=120:00:00

#PBS -m ea

cd $PBS_O_WORKDIR
./run.sh
```

When defining the PBS file note the low memory usage (`pmem`) and high `walltime`. Since the job will only be responsible for running this script, only a limited amount of resources is needed. However, the total batch of jobs may run for quite some time, so the wall clock time is likely to be quite high.

3) The CSV file that defines the replicates to be run, where the first column is the PBS file for the replicate and the second column is the count:

```
bfa-slow-no-asaq.job,1
bfa-fast-no-asaq.job,1
bfa-rapid.job,1
```

While the `runReplicates` command executes, the number of jobs per user account is limited to the `LIMIT` defined in `calibrationLib.sh` (99 by default). When the limit is reached the script will sleep and periodically awaken to check to see if more jobs can be queued.

C.3 WSL on the PSU VPN

One minor problem that may occur while on the PSU VPN is that `psu.edu` domains are not resolved correctly. This is usually due to the `/etc/resolv.conf` file not being updated correctly by WSL. To manually update the file, launch `PowerShell` and run:

```
Get-DnsClientServerAddress -AddressFamily IPv4 | Select-Object -ExpandPropert ServerAddresses
```

Then copy the addresses to `/etc/resolv.conf` so you have something similar to:

```
nameserver 192.168.1.1
nameserver 192.168.1.2
nameserver 192.168.1.3
search psu.edu
```

References

Bomblies, Arne. 2012. "Modeling the Role of Rainfall Patterns in Seasonal Malaria Transmission." *Climatic Change* 112 (3): 673–85. https://doi.org/10.1007/s10584-011-0230-6.

Cameron, Ewan, Katherine E. Battle, Samir Bhatt, Daniel J. Weiss, Donal Bisanzio, Bonnie Mappin, Ursula Dalrymple, et al. 2015. "Defining the Relationship Between Infection Prevalence and Clinical Incidence of Plasmodium Falciparum Malaria." *Nature Communications* 6 (1): 8170. https://doi.org/10.1038/ncomms9170.

Collins, Robert O. 2004. "The Ilemi Triangle." *Annales d'Éthiopie*, 5–12. https://www.persee.fr/doc/ethio_0066-2127_2004_num_20_1_1065.

Division of National Malaria Programme (DNMP) [Kenya], and ICF. 2021. "Kenya Malaria Indicator Survey 2020." Nairobi, Kenya; Rockville, Maryland, USA: DNMP; ICF. https://dhsprogram.com/pubs/pdf/MIS36/MIS36.pdf.

Fick, Stephen E., and Robert J. Hijmans. 2017. "WorldClim 2: New 1-Km Spatial Resolution Climate Surfaces for Global Land Areas." *International Journal of Climatology* 37 (12): 4302–15. https://doi.org/10.1002/joc.5086.

Hay, Simon I., Carlos A. Guerra, Andrew J. Tatem, Peter M. Atkinson, and Robert W. Snow. 2005. "Urbanization, Malaria Transmission and Disease Burden in Africa." *Nature Reviews Microbiology* 3 (1): 81–90. https://doi.org/10.1038/nrmicro1069.

Marshall, John M., Sean L. Wu, Hector M. Sanchez C., Samson S. Kiware, Micky Ndhlovu, André Lin Ouédraogo, Mahamoudou B. Touré, Hugh J. Sturrock, Azra C. Ghani, and Neil M. Ferguson. 2018. "Mathematical Models of Human Mobility of Relevance to Malaria Transmission in Africa." *Scientific Reports* 8 (1): 7713. https://doi.org/10.1038/s41598-018-26023-1.

Matsumoto, Makoto, and Takuji Nishimura. 1998. "Mersenne Twister: A 623-Dimensionally Equidistributed Uniform Pseudo-Random Number Generator." *ACM Trans. Model. Comput. Simul.* 8 (1): 3–30. https://doi.org/10.1145/272991.272995.

Nguyen, Tran Dang, Piero Olliaro, Arjen M Dondorp, J Kevin Baird, Ha Minh Lam, Jeremy Farrar, Guy E Thwaites, Nicholas J White, and Maciej F Boni. 2015. "Optimum Population-Level Use of Artemisinin Combination Therapies: A Modelling Study." *The Lancet Global Health* 3 (12): e758–66. https://doi.org/10.1016/S2214-109X(15)00162-X.

Pascual, Mercedes, Bernard Cazelles, M. J. Bouma, L. F Chaves, and K Koelle. 2008. "Shifting Patterns: Malaria Dynamics and Rainfall Variability in an African Highland." *Proceedings of the Royal Society B: Biological Sciences* 275 (1631): 123–32. https://doi.org/10.1098/rspb.2007.1068.

Weiss, Daniel J, Tim C D Lucas, Michele Nguyen, Anita K Nandi, Donal Bisanzio, Katherine E Battle, Ewan Cameron, et al. 2019. "Mapping the Global Prevalence, Incidence, and Mortality of Plasmodium Falciparum, 2000–17: A Spatial and Temporal Modelling Study." *The Lancet* 394 (10195): 322–31. https://doi.org/10.1016/S0140-6736(19)31097-9.

Wesolowski, Amy, Wendy Prudhomme O'Meara, Nathan Eagle, Andrew J. Tatem, and Caroline O. Buckee. 2015. "Evaluating Spatial Interaction Models for Regional Mobility in Sub-Saharan Africa." *PLOS Computational Biology* 11 (7): e1004267. https://doi.org/10.1371/journal.pcbi.1004267.

Wongsrichanalai, Chansuda, Mazie J. Barcus, Sinuon Muth, Awalludin Sutamihardja, and Walther H. Wernsdorfer. 2007. "A Review of Malaria Diagnostic Tools: Microscopy and Rapid Diagnostic Test (RDT)." *The American Journal of Tropical Medicine and Hygiene* 77 (6_Suppl): 119–27. https://doi.org/10.4269/ajtmh.2007.77.119.

World Health Organization. 2022. *World Malaria Report 2022*. Geneva: World Health Organization. https://www.who.int/teams/global-malaria-programme/reports/world-malaria-report-2022.

Zupko, Robert J., Tran Dang Nguyen, J. Claude S. Ngabonziza, Michee Kabera, Haojun Li, Thu Nguyen-Anh Tran, Kien Trung Tran, Aline Uwimana, and Maciej F. Boni. 2023. "Modeling Policy Interventions for Slowing the Spread of Artemisinin-Resistant Pfkelch R561H Mutations in Rwanda." *Nature Medicine* 29 (11): 2775–84. https://doi.org/10.1038/s41591-023-02551-w.

Zupko, Robert J., Tran Dang Nguyen, Anyirékun Fabrice Somé, Thu Nguyen-Anh Tran, Jaline Gerardin, Patrick Dudas, Dang Duy Hoang Giang, et al. 2022. "Long-Term Effects of Increased Adoption of Artemisinin Combination Therapies in Burkina Faso." *PLoS Global Public Health* 2 (2): e0000111.

https://doi.org/10.1371/journal.pgph.0000111.