Multipatch Epidemic Model for Spatial Spread of Vector borne

Diseases

# Project goal

To form a basis for precautionary measures for disease spread based on the prediction of rainfall and temperature.

# Concept

The model in paper is SEIR (Susceptible, Exposed, Infected, Recovered) disease spread model.

Contagious disease spread rate in a region depends on these factors:

1. Region human (hosts) population factors
   1. Birth rate (Host recruitment), natural death rate, death due to diseases
2. Region’s mosquito (vector) population factors
   1. Egg deposition rate, maturation rate, death rate
   2. Factors dependent on rainfall and temperature
   3. Limited by the vector carrying capacity of a region.
3. SEIR dynamics of disease already in region, both for vectors and hosts.
4. Interaction strength between hosts, transmission probability factors, biting rate of vectors
5. Migration rate from and to other regions.

# Introduction

The equations describing dynamics of model at a given period of time are ordinary differential equations. The initial situational conditions for every region is provided and for a period of maxTime years (in the implementation it is 10 years), for an adaptive time step period all the model parameter values are calculated.

# Input data

1. **Population dynamics of regions (Input\_cities.csv)**



1. **Q.csv (matrix describing fraction of population leaving the region)**
2.  **Migration.csv (l and r parameters in single matrix)**
3. **Monthly mean temperatures (AfricaMonthlyTemperatures.csv)**



1. **Monthly mean Precipitation (AfricaMonthlyPrecipitation.csv)**



Source for temperature and precipitation:

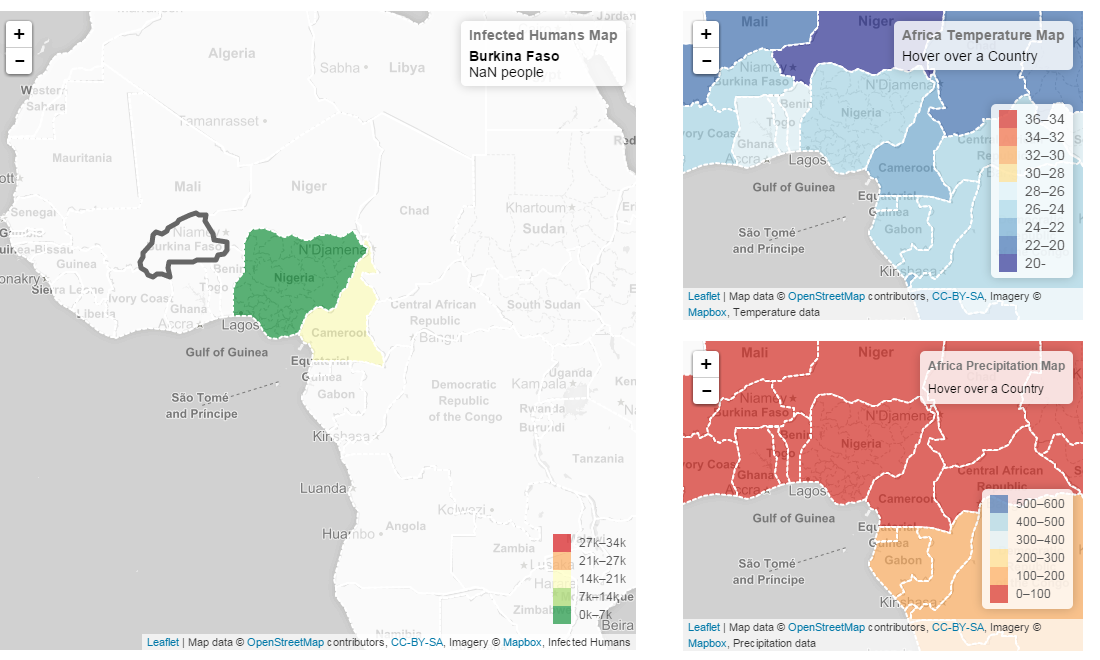
<http://data.worldbank.org/data-catalog/cckp_historical_data>

# Visualization elements

**Line Chart ([SEIR] vs Time)**

A line chart describing the trend SEIR over the years is presented. Multiple cities are plot on the same graph for comparison. Our model can describe when does a disease die out, or when it become epidemic and require precautionary measures to counter it.

**Cartographical display**

We show on the map of Africa how the disease spread over the years in the regions/countries under our study. Once the page is completely loaded, time and temperature sliders can be used to interactively observe the trend of disease spread in countries. Two other smaller cartographs showing temperature and precipitation states of countries at the time under study are shown beside the infected humans map.

# Code flowchart

includes libraries, lays out layout

Reads input parameters for all regions

Create threads and calls newSpawn() passing on the required data

Mutithreaded environment to calculate SEIR vs time objects.

One time call to function to create Time\_h2 object which will be used in linecharts.

Creates line chart of Ih, Sh, Eh, Rh vs time(years) for all cities. Also create line chart of Temperature & precipitation vs time(in months)

Creates Africa Map, with color code for intensity of disease infection in region, temperature and precipitation cartographs of time shown in sliders

One time initialization of map layers of temperature and precipitation maps.

Initialize sliders and their event listeners.

Draw lenged for line charts showing color codes for cities/countries.

Start

Input\_cities.csv

Index.html

Migration.csv

CSVread()

q.csv

CalculateAll()

AfricanMonthlyTemperatures.csv

newSpawn()

AfricanMonthlyPrecipitation.csv

InitializeTimeh2()

RhlineChart()

IhlineChart()

ShlineChart()

EhlineChart()

TempLineChart()

PrepLineChart()

AfricaMapInit()

calc\_execTime()

AfricaMap()

tempMap()

PrepMap()

PrepMapInit()

tempMapInit()

Sliders()

draw\_legend()

calc\_execTime();

AfricaMap();

drawMapLegen);

Stop

**Implementation**

JavaScript is used extensively for implementation of this project. These are the external libraries used:

1. D3.js
2. Numeric.js
3. Parallel.js
4. Leaflet.js
5. Eval.js
6. jQuery
7. D3.slider.js
8. Colorbrewer.js

The most numerically intensive part of the project which also forms the basis of this model is computed in calculate.js (calculateAll() function). Differential equations are solved using ODE solver provided in numeric.js. numeric.dopri is an adaptive 4th order Dorman-Prince Runge Kutta solver which computes values of Infected humans at adaptive time step intervals until the provided maxTime is reached. As the number of iterations are high, usually it takes great amount of time to compute for multiple cities, thus we have used parallelization of computation to use multiple cores of modern computers. This led to increase of ~4X for 5 city/region model. For this we used already available Parallel.js which spawns different threads. D3 was used to create line graph and also forms the basis of other visualization elements indirectly. Sliders were created using slider.js and African Maps were presented using leaflet.js libraries.

**Result**

**Conclusion**

**References**