TEXAS ADVANCED COMPUTING CENTER

WWW.TACC.UTEXAS.EDU

TEXAS The University of Texas at Austin

Disease Propagation Model

S. Charlie Dey, Director of Training and Professional Development

Victor Eijkhout, Research Scientist

Science in the Cloud, 2019

Disease Propagation Modeling

Goal: build an explicit simulation

We will maintain an explicit description of all the people in a population, and track each of their status.

We will use a simple model where a person can be:

- sick: when they are sick, they can infect other people;
- susceptible: they are healthy, but can be infected;
- recovered: they have been sick, but no longer carry the disease, and can not be infected for a second time;
- inoculated: they are healthy, do not carry the disease, and can not be infected.

We're going to start simple: any sick person is infectious.

We always start with just one person infected. The program will then track the population from day to day, running indefinitely until none of the population is sick. Since there is no re-infection, the run will always end.

TACC

Disease Propagation Modeling

We start by writing code that models a single person.
The main methods serve to infect a person, and to track their state.
We need to have some methods for inspecting that state.
Let's say "Joe" is infected on day 4, and is infected for 6 days
The intended output will look something like:

```
On day 1, Joe is susceptible
On day 2, Joe is susceptible
On day 3, Joe is susceptible
On day 4, Joe is sick (6 day(s) to go)
On day 5, Joe is sick (5 day(s) to go)
On day 6, Joe is sick (4 day(s) to go)
On day 7, Joe is sick (3 day(s) to go)
On day 8, Joe is sick (2 day(s) to go)
On day 9, Joe is sick (1 day(s) to go)
On day 10, Joe is recovered
```

Disease Propagation Modeling

```
joe = Person()

while (not joe.is_inoculated):
    joe.update()
    print ("Joe is " + joe.current_status, end="")
    if (joe.is_infected):
        print (" " + str(joe.days_sick) + " to go", end="")
    print()
    bad_luck = rnd.random()
    if (bad_luck > .90 and not joe.is_inoculated):
        joe.infect(5)
```

TACC

Disease Propagation Modeling, Person Object

```
class Person(object):
    def __init__(self):
        self.current_status = "well"
        self.days_sick = 0
        self.is_inoculated = False
        self.is_stable = True
        self.is_infected = False

    def infect(self, days):
        self.days_sick = days
        self.is_infected = True
        if (self.is_inoculated):
            self.is_infected = False
            self.days_sick = 0
```

```
def update(self):
     if (self.current_status == "well"
and
          self.is_infected == True):
          self.current_status = "sick"
          self.is_stable = False
     elif (self.current_status ==
     "sick"):
          self.days_sick =
               self.days_sick-1

     if (self.days_sick == 0):
          self.current_status =
               "recovered"
          self.is_inoculated = True
          self.is_stable = True
          self.is_infected = False
```

TACC

Disease Propagation Modeling

You will then expand on your code:

- **you will build a Population, which will be a list People objects**
- **make one person sick**
- **code random interactions within the population, allowing the disease to spread**
- **keep the simulation running until everyone is stable**
    - **record the number of your population**
    - **percentage that was immunized at the beginning**
    - **number of days until the entire population was healthy**
- **begin immunizing a percentage of your population, and rerun**

Disease Propagation Modeling, Population Object

```
class Population(object):
    import random as rnd
    def __init__(self, number):
        self.People = []
        bad_luck = self.rnd.randint(0,number-1)
        for i in range(0,number):
            p = Person()
            self.People.append(p)

        self.People[bad_luck].infect(5)
```

```
    def number_of_sick(self):
        sick_count = 0
        for p in self.People:
            if (p.is_infected):
                sick_count = sick_count+1

        return sick_count
    def update(self):
        for p in self.People:
            p.update()
            bad_luck = self.rnd.random()

            if (bad_luck > .9 and not
                    p.is_inoculated and not
                    p.is_infected):
                p.infect(5)
```

Disease Propagation Modeling

This is a research project, so you'll need to run various datasets and save the data to files analyze the runs, draw intelligent conclusions

Record how long the disease runs through the population with various population sizes

With a fixed number of contacts and probability of transmission, how is this number of function of the percentage that is vaccinated?

Investigate the matter of ‘herd immunity’: if enough people are vaccinated, then some people who are not vaccinated will still never get sick. Let’s say you want to have this probability over 95 percent. Investigate the percentage of inoculation that is needed for this as a function of the contagiousness of the disease.