UnB

CIC0203 - Computação Experimental - TA - 2022.2 - Tarefa T7 - Novo Aprimoramento de uma Simulação URL Read-only Overleaf: https://www.overleaf.com/read/hjrptnncxxrm

Isaque Augusto da Silva Santos (seraphritt)

Brasília, 2023-01-20 17:09:57-03:00

Lista de tarefas pendentes

Sumário

Lista de Figuras

Lista de Tabelas

Resumo

Este documento contém o produto da tarefa especificada no título deste documento, conforme as orientações em https://www.overleaf.com/read/cytswcjsxxqh.

Parte I

Estudos Empíricos Exploratórios

Capítulo 1

T7 - Novo aprimoramento de uma Simulação: Laboratório e Experimento virus_on_network - Parte 2, por Isaque Augusto da Silva Santos (seraphritt)

1.1 Introdução

Este capítulo apresenta a construção e uso do laboratório de simulações virus_on_network para a realização de experimentos que tem por objetivo investigar a hipótese causal letalidade-mortalidade que relaciona variáveis independentes e variáveis dependentes, supostamente presente nos estudos bibliométricos por mim.

É composto por mais cinco seções:

1. Descrição do fenômeno real;
2. Apresentação do laboratório de simulações;
3. Apresentação de análises exploratórias dos dados de experimentos realizados com o uso do laboratório;
4. Discussão sobre *insights* obtidos após os experimentos; e
5. Conclusões.

1.2 O Fenômeno do Mundo Real

O fenômeno do experimento em questão é a infecção por vírus e seu contágio ao haver contato entre pessoas. Esse é um acontecimento corriqueiro que está presente em qualquer sociedade na qual seres humanos compartilham o mesmo espaço físico (DELIKHOON et al., 2021), (LEUNG, 2021).

Como pode-se ver na imagem 1.1 a seguir, esse fenômeno de contágio pode ocorrer de múltiplas formas.

Figura 1.1: Publicidade do Ministério da Saúde sobre o contágio do Coronavírus

As pessoas são os multiagentes e o contato entre elas é representado pelas arestas do grafo.

1.3 O Laboratório virus_on_network

O laboratório virus_on_network é um modelo genérico que simula a transmissão de um vírus em uma rede e pode ser utilizado tanto para simular uma infecção viral como uma infecção

por um vírus em uma rede de computadores, como pode ser visto na descrição original do modelo (F. STONEDAHL; WILENSKY, 2008).

Nesse ínterim, levando em conta o momento atual de preocupação com a prevenção de doenças virais como a Covid-19, escolhi usar esse modelo para simular uma epidemia.

1.3.1 O Conceito da Simulação

O código base utilizado está presente no repositório do *GitHub* denominado *projectmesa/mesa-examples/Virus_on_network* link para o repositório.

A arquitetura do modelo consiste em representar cada ser humano como um agente, criando assim uma rede de na qual os agentes se conectam. Portanto, as pessoas são os vértices e o contato entre elas é representado pelas arestas do grafo. Isso posto, existem quatro variáveis dependentes e oito variáveis independentes nesse modelo que serão especificadas posteriormente.

A simulação cria um ambiente que retrata o espalhamento de um vírus em uma sociedade. Por meio dessa simulação é possível notar relações entre as variáveis independentes e a alteração dos valores das variáveis dependentes, podendo significar uma relação de causa-efeito.

Os fenômenos que possivelmente podem estar relacionados com uma maior transmissão como, por exemplo, a concentração populacional (quantidade de vértices), o frequência de contato (quantidade de arestas) entre as pessoas, chance de contágio do vírus (variável virus_spread_chance), chance de recuperação após infecção (variável recovery_chance) e chance de letalidade (variável virus_letal_chance) foram explorados durante a simulação.

1.3.2 O Simulador

No código estão presentes as seguintes variáveis:

virus_letal_chance representa a chance de um vírus fazer com que o agente mude seu estado para morto (*dead*).

num_nodes contém o número de agentes presentes no modelo.

avg_node_degree representa o número médio de arestas entre os agentes, ou seja, o número médio de pessoas que podem estar infectadas caso um agente tenho o estado de *infected*.

initial_outbreak_size define a quantidade de agentes que iniciam com o estado de *infected*.

virus_spread_chance estipula a possibilidade de contágio do vírus.

virus_check_frequency define a possibilidade de mudar o estado de um agente (*infected* ou *dead*).

recovery_chance representa a chance de um agente com o estado *infected* mudar seu estado para *suscetible*.

gain_resistance_chance contém a chance de um agente mudar seu estado para *resistant*.

stable contém o valor que define se a simulação está estável ou não. Se o valor dessa variável for 0 significa que não está estável (False), se for 1 significa estabilidade da simulação (True). A condição de estabilidade definida foi: se houver 0 agentes com estado *Infected*, a simulação será considerada estável.

A seguinte imagem 1.2 mostra interface gráfica do modelo:

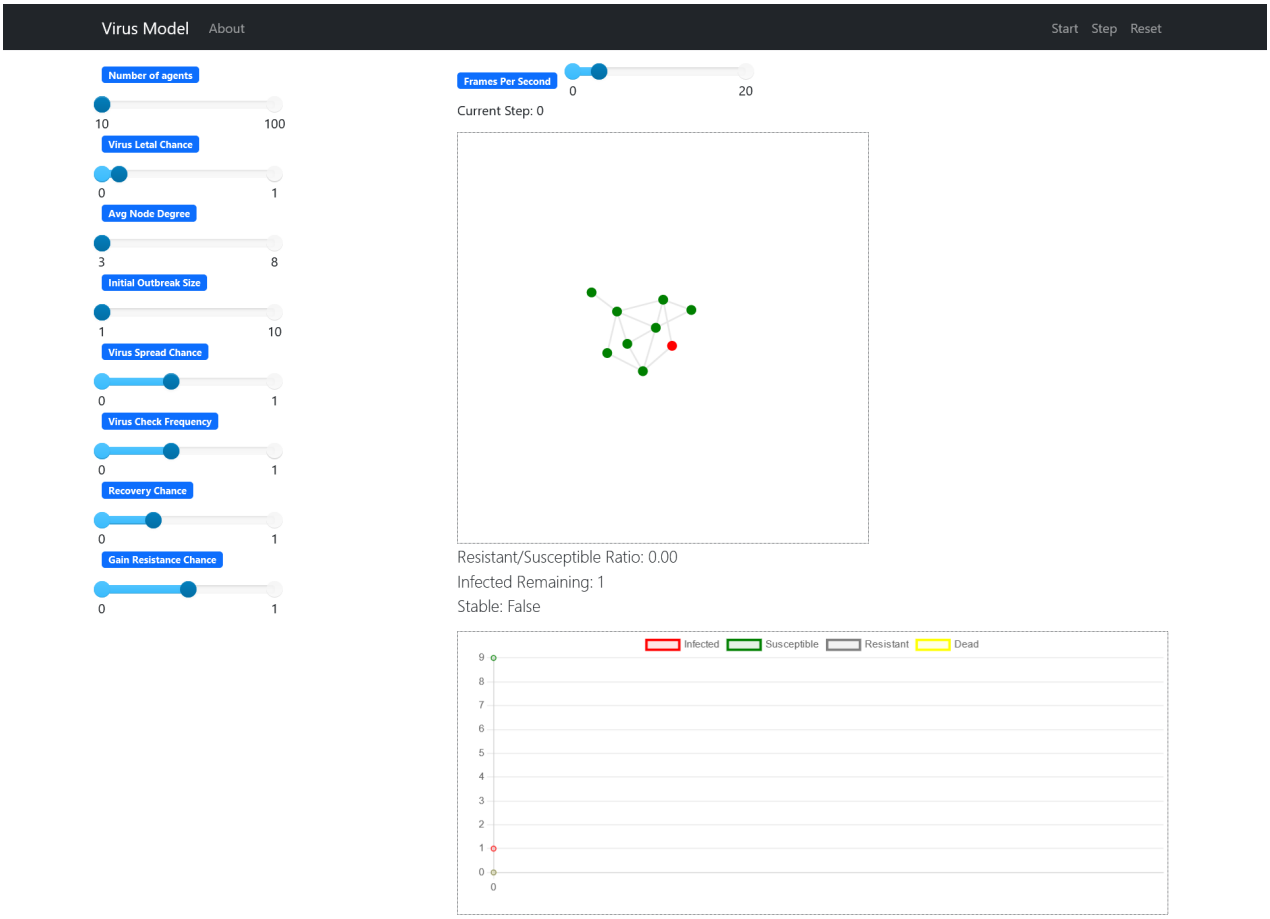

Figura 1.2: Interface gráfica do modelo

1.3.2.1 Variáveis Independentes ou de Controle

São as seguintes as variáveis Independentes ou de Controle, manipuláveis na interface gráfica do simulador:

Number of agents define a quantidade de agentes na simulação, ou seja, o quantidade de vértices no grafo, em um intervalo de 10 a 100.

Virus Letal Chance define a probabilidade do vírus mudar a cor do agente de vermelho (*infected*) para amarelo (*dead*), em um intervalo de 0 a 1.

Avg Node Degree é a quantidade média de arestas que cada vértice possui, em um intervalo de 3 a 8.

Initial Outbreak Size controla a quantidade de agentes na cor vermelha (*infected* no primeiro passo da simulação, em um intervalo de 1 a 10.

Virus Spread Chance define a probabilidade dos vizinhos de um agente com a cor vermelha fazer com que seus vizinhos também fiquem com a cor vermelha, ou seja, infectados, em um intervalo de 0 a 1.

Virus Check Frequency controla a probabilidade da simulação mudar o estado de um agente para *infected* ou dead, em um intervalo de 0 a 1.

Recovery Chance define a probabilidade de um agente ter sua cor mudada para verde (*suscetible*) dado que está com a cor vermelha (*infected*), em um intervalo de 0 a 1.

Gain Resistance Chance controla a probabilidade de um agente ter sua cor mudada para cinza (*resistant*), em um intervalo de 0 a 1.

1.3.2.2 Variáveis Dependentes

São as seguintes as variáveis Dependentes, cujos valores são coletados e apresentados na interface gráfica do simulador:

Infected apresenta a quantidade de agentes infectados.

Suscetible apresenta a quantidade de agentes suscetíveis a serem infectados.

Resistant apresenta a quantidade de agentes resistentes ao vírus, ou seja, mesma que tenham contato com um agente infectado, esses agentes resistentes não serão infectados.

Dead apresenta a quantidade de agentes mortos. Esses agentes não trasmitem o vírus.

Stable representa a estabilidade do modelo.

1.3.3 A Hipótese Causal

A minha hipótese causal surgiu de questionamentos sobre a relação entre a letalidade e a transmissão de um vírus

Será que um vírus com alta chance de letalidade consegue infectar uma grande quantidade de pessoas?

Vírus com menor chance de letalidade necessariamente infectam um número maior de pessoas quando comparado a vírus extremamente letais?

Ademais, chegou-se a seguinte hipótese causal : "A letalidade de um vírus não é a única variável independente responsável por uma grande quantidade de mortes por causa viral.". Observação: Considera-se "grande quantidade" um número próximo ou igual a quantidade de agentes suscetíveis à infecção viral.

A hipótese causal foi obtida com o auxílio dos artigos (GEOGHEGAN et al., 2016) e (BELAY; MONROE, 2014).

1.3.4 O Código do Simulador

Com o intuito de comprovar a veracidade da hipótese causal, adicionei uma variável independente *Virus Letal Chance* e uma variável dependente Dead para aprimorar o modelo e assim simular a infecção por vírus com chances de letalidade diferentes.

As classes presentes no simulador são:

State, classe responsável por controlar o estado de cada agente e possui os atributos *SUSCETIBLE = 0*, *INFECTED = 1*, *RESISTANT = 2* e *DEAD = 3*.

VirusAgent, classe resposável por declarar os agentes.

VirusOnNetwork, classe resposável pelo modelo em si, bem como a coleta de dados. Essa classe instancia os agentes.

A criação e o modelo de agentes pode ser vista no trecho de código a seguir:

```
class VirusOnNetwork(mesa.Model):
    """A virus model with some number of agents"""

    def __init__(
        self,
        virus_letal_chance=0.1, #mod
        num_nodes=10,
        avg_node_degree=3,
        initial_outbreak_size=1,
        virus_spread_chance=0.4,
        virus_check_frequency=0.4,
        recovery_chance=0.3,
        gain_resistance_chance=0.5,
    ):
        self.virus_letal_chance = virus_letal_chance
        self.num_nodes = num_nodes
```

```
        prob = avg_node_degree / self.num_nodes
        self.G = nx.erdos_renyi_graph(n=self.num_nodes, p=prob)
        self.grid = mesa.space.NetworkGrid(self.G)
        self.schedule = mesa.time.RandomActivation(self)
        self.initial_outbreak_size = (
            initial_outbreak_size if initial_outbreak_size <= num_nodes else num_node
        )
        self.virus_spread_chance = virus_spread_chance
        self.virus_check_frequency = virus_check_frequency
        self.recovery_chance = recovery_chance
        self.gain_resistance_chance = gain_resistance_chance

        self.datacollector = mesa.DataCollector(
            {
                "Infected": number_infected,
                "Susceptible": number_susceptible,
                "Resistant": number_resistant,
                "Dead": number_dead,
            }
        )

        # Create agents
        for i, node in enumerate(self.G.nodes()):
            a = VirusAgent(
                i,
                self,
                State.SUSCEPTIBLE,
                self.virus_spread_chance,
                self.virus_check_frequency,
                self.recovery_chance,
                self.gain_resistance_chance,
                self.virus_letal_chance,
            )
            self.schedule.add(a)
            # Add the agent to the node
            self.grid.place_agent(a, node)

        # Infect some nodes
        infected_nodes = self.random.sample(list(self.G), self.initial_outbreak_size)
        for a in self.grid.get_cell_list_contents(infected_nodes):
            a.state = State.INFECTED
```

```
        self.running = True
        self.datacollector.collect(self)

    def resistant_susceptible_ratio(self):
        try:
            return number_state(self, State.RESISTANT) / number_state(
                self, State.SUSCEPTIBLE
            )
        except ZeroDivisionError:
            return math.inf

    def step(self):
        self.schedule.step()
        # collect data
        self.datacollector.collect(self)

    def run_model(self, n):
        for i in range(n):
            self.step()


class VirusAgent(mesa.Agent):
    def __init__(
        self,
        unique_id,
        model,
        initial_state,
        virus_spread_chance,
        virus_check_frequency,
        recovery_chance,
        gain_resistance_chance,
        virus_letal_chance,
    ):
        super().__init__(unique_id, model)

        self.state = initial_state

        self.virus_spread_chance = virus_spread_chance
        self.virus_check_frequency = virus_check_frequency
        self.recovery_chance = recovery_chance
```

```
        self.gain_resistance_chance = gain_resistance_chance
        self.virus_letal_chance = virus_letal_chance

    def try_to_infect_neighbors(self):
        neighbors_nodes = self.model.grid.get_neighbors(self.pos, include_center=Fals
        susceptible_neighbors = [
            agent
            for agent in self.model.grid.get_cell_list_contents(neighbors_nodes)
            if agent.state is State.SUSCEPTIBLE
        ]
        for a in susceptible_neighbors:
            if self.random.random() < self.virus_spread_chance and a.state is not Sta
                a.state = State.INFECTED

    def try_gain_resistance(self):
        if self.random.random() < self.gain_resistance_chance:
            self.state = State.RESISTANT

    def try_remove_infection(self):
        # Try to remove
        if self.random.random() < self.recovery_chance:
            # Success
            self.state = State.SUSCEPTIBLE
            self.try_gain_resistance()
        else:
            # Failed
            self.state = State.INFECTED

    def try_check_situation(self):
        if self.random.random() < self.virus_check_frequency:
            # Checking...
            if self.state is State.INFECTED:
                self.try_remove_infection()
            if self.state is State.INFECTED and self.random.random() < self.virus_let
            # letal chance of the virus is enough, agent is considered dead
            # dead agents cannot transmit the virus
                self.state = State.DEAD

    def step(self):
        if self.state is State.INFECTED:
            self.try_to_infect_neighbors()
```

```
        self.try_check_situation()
```

O comportamento do modelo e dos agentes a cada passo pode ser visto nos seguintes trechos de código:

```
def number_state(model, state):
    return sum(1 for a in model.grid.get_all_cell_contents() if a.state is state)


def number_infected(model):
    return number_state(model, State.INFECTED)


def number_susceptible(model):
    return number_state(model, State.SUSCEPTIBLE)


def number_resistant(model):
    return number_state(model, State.RESISTANT)

def number_dead(model): #mod
    return number_state(model, State.DEAD)


 def step(self):
        self.schedule.step()
        # collect data
        self.datacollector.collect(self)

    def run_model(self, n):
        for i in range(n):
            self.step()

def try_to_infect_neighbors(self):
        neighbors_nodes = self.model.grid.get_neighbors(self.pos, include_center=False)
        susceptible_neighbors = [
            agent
            for agent in self.model.grid.get_cell_list_contents(neighbors_nodes)
            if agent.state is State.SUSCEPTIBLE
        ]
        for a in susceptible_neighbors:
            if self.random.random() < self.virus_spread_chance and a.state is not State.DEAD
```

```
                a.state = State.INFECTED

    def try_gain_resistance(self):
        if self.random.random() < self.gain_resistance_chance:
            self.state = State.RESISTANT

    def try_remove_infection(self):
        # Try to remove
        if self.random.random() < self.recovery_chance:
            # Success
            self.state = State.SUSCEPTIBLE
            self.try_gain_resistance()
        else:
            # Failed
            self.state = State.INFECTED

    def try_check_situation(self):
        if self.random.random() < self.virus_check_frequency:
            # Checking...
            if self.state is State.INFECTED:
                self.try_remove_infection()
            if self.state is State.INFECTED and self.random.random() < self.virus_let
            # letal chance of the virus is enough, agent is considered dead
            # dead agents cannot transmit the virus
                self.state = State.DEAD

    def step(self):
        if self.state is State.INFECTED:
            self.try_to_infect_neighbors()
        self.try_check_situation()
```

1.4 Os Experimentos Realizados

Foram feitos 2 experimentos principais para verificar a veracidade da hipótese causal:

Experimento 1 com a variável *Virus Letal Chance* com o valor máximo e as outras variáveis com um valor aleatório. Esse experimento tem como objetivo verificar se a variável *Virus Letal Chance* é dominante sobre as outras variáveis e por si só faz com que a quantidade de mortes aumente significativamente.

Experimento 2 com a variável *Virus Letal Chance* com o valor máximo e *Virus Spread Chance* com valor máximo e *Recovery Chance* com valor mínimo, com o intuito de achar alguma

configuração de variáveis na qual o número de mortes seja ampliado significativamente.

Além desses 2 experimentos principais, foram feitos diversos outros experimentos modificando o valor de todas as variáveis para encontrar outras configurações que fariam com que o número de mortes fosse o mínimo ou o máximo.

Devido a quantidade de variáveis, houve uma certa dificuldade em achar relações de causa e efeito de uma variável independente isolada com uma variável dependente. Isso deve-se ao fato da complexidade do fenômeno de infecção viral.

Ademais, notou-se a necessidade de simplificar o modelo, tendo em vista também essa complexidade dita anteriormente. Isso posto, variáveis como a chance de mutabilidade do vírus e de reinfecção, por exemplo, não foram adicionadas.

A imagem 1.3 a seguir apresenta o gráfico obtido após o experimento 1.

Figura 1.3: Gráfico obtido no experimento 1

Adicionou-se também um novo arquivo com o nome *experimentos_estatisticos.py*, com o intuito de auxiliar na coleta dos dados das simulações, como pode ser visto na próxima seção.

1.4.1 Os Dados Coletados

A coleta dos dados das simulações foram feitas seguindo uma sequência de critérios para facilitar as conclusões sobre os dados encontrado. Esses critérios foram:

As variáveis independentes *Number of agents*, *Avg Node Degree*, *Initial Outbreak Size*, *Virus Spread Chance*, *Virus Check Frequency*, *Recovery Chance* e *Gain Resistance Chance* tiveram seus valores definidos como constantes, ou seja, durante a série de experimentos feitos para coleta de dados o valor dessas variáveis não muda.

Number of agents com valor constante de 10.

Avg Node Degree com valor constante de 3.

Initial Outbreak Size com valor constante de 1.

Virus Spread Chance com valor constante de 0,4.

Virus Check Frequency com valor constante de 0,4.

Recovery Chance com valor constante de 0,3.

Gain Resistance Chance com valor constante de 0,5.

*F*oram feitos experimentos com números de replicações diferentes para obter maior grau de certeza. Portanto, houve experimentos com 50, 100, 200 e 400 replicações.

*C*omo a hipótese causal refere-se á letalidade e sua singularidade no aumento do número de mortes, somente a variável *Virus Letal Chance* deve seu valor modificado entre os experimentos, com um intervalo de 0.1, indo de 0 (valor mínimo) até 1 (valor máximo.

Variáveis presentes nos histogramas:

Frequency: O número de ocorrências de determinado

Dead Agents O número de agentes com o estado *Dead*.

O primeiro histograma 1.4 foi feito com 50 replicações e é possível notar a grande concentração de ocorrências de 0 agentes com estado *Dead*, ou seja, a quantidade mínima de agentes. Semelhantemente, nota-se o mesmo resultado com o número de replicações igual a 100 1.5, 200 1.6 e 400 1.7. Com isso pode-se concluir que a modificação da letalidade do vírus, por si só, não consegue aumentar significadamente o número de mortes, fazendo com que esse número seja próximo da quantidade máxima de agentes.

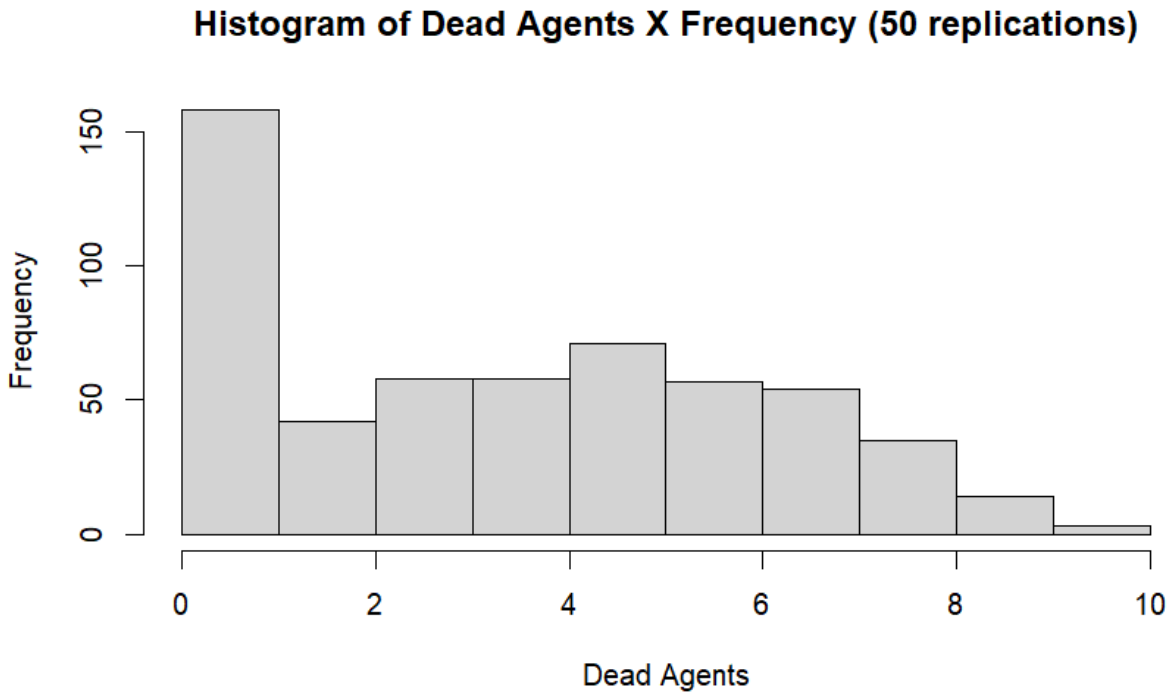

Figura 1.4: Histograma obtido a partir dos dados do experimento realizado com 50 replicações

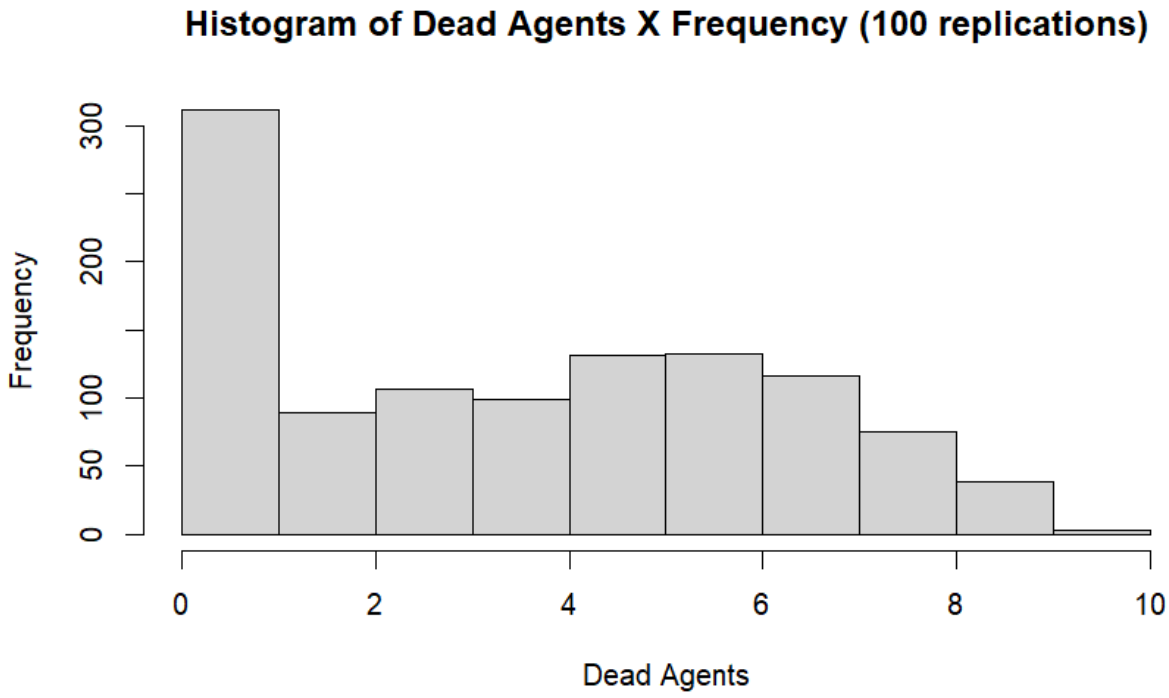

Figura 1.5: Histograma obtido a partir dos dados do experimento realizado com 100 replicações

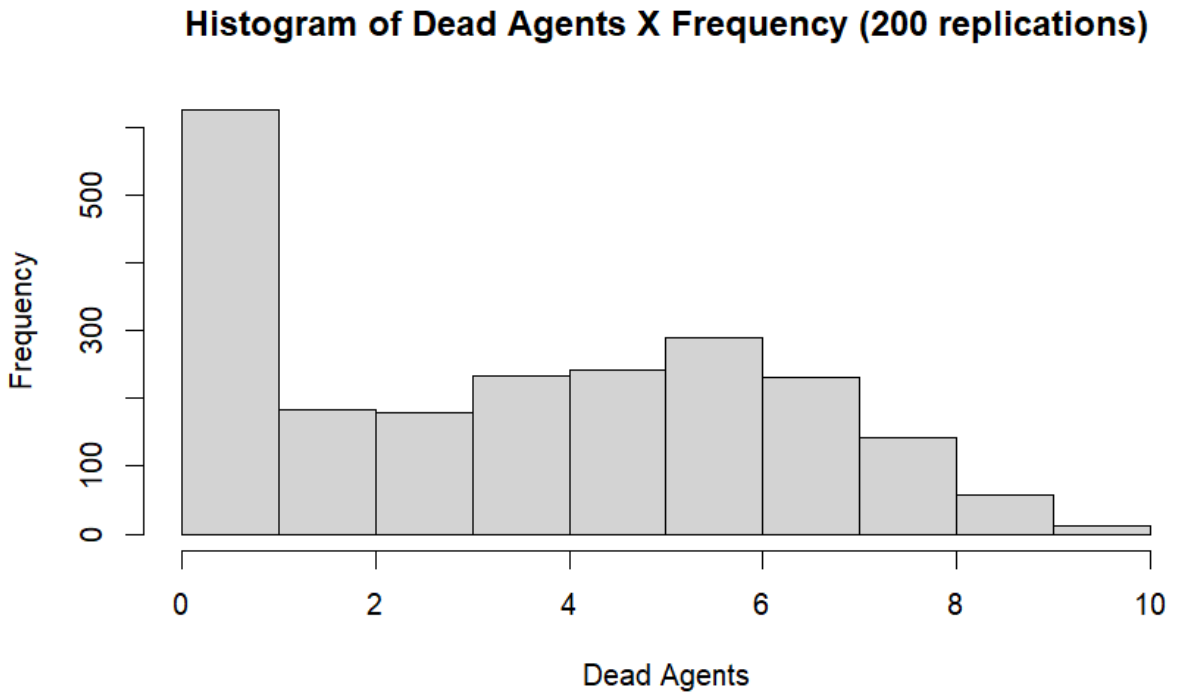

Figura 1.6: Histograma obtido a partir dos dados do experimento realizado com 200 replicações

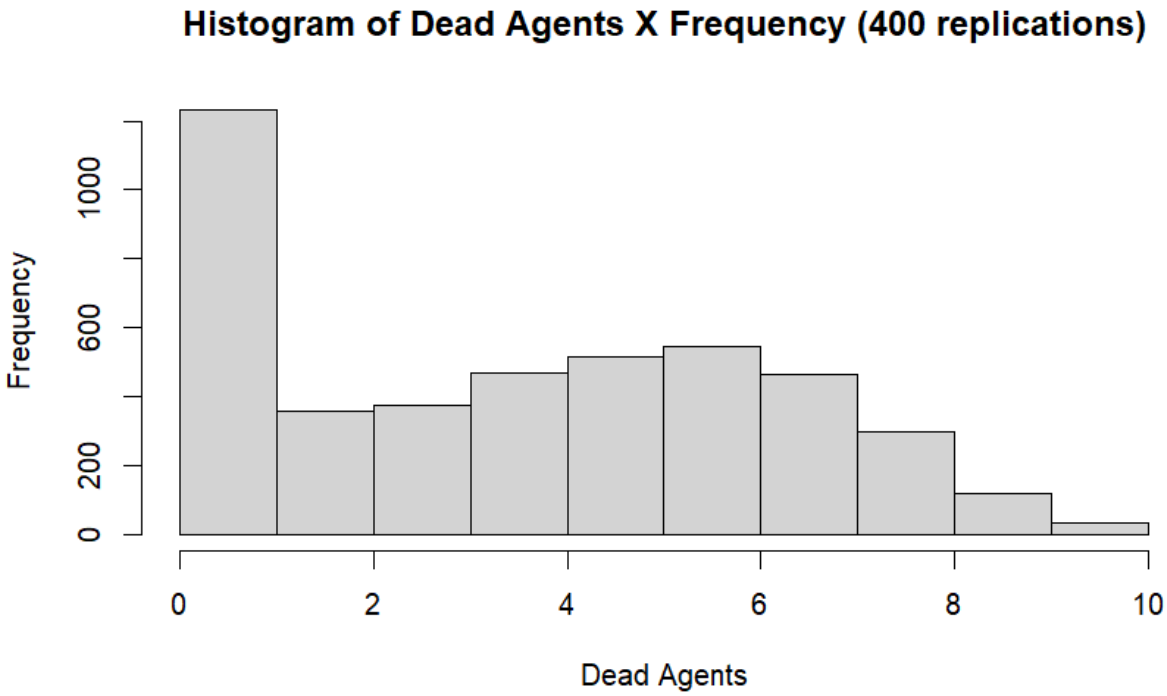

Figura 1.7: Histograma obtido a partir dos dados do experimento realizado com 400 replicações

Variáveis presentes nos*boxplots*:

Virus Lethal Chance: A probabilidade de um agente com estado *Infected* ter seu estado mudado para *Dead*.

Dead Agents O número de agentes com o estado *Dead*.

Os boxplots com 50 1.8, 100 1.9, 200 1.10 e 400 1.11 replicações também demonstram dados que concluem que a letalidade do vírus sozinha não maximiza a quantidade de mortes. Nota-se que mesmo com a letalidade em valores altos, como 70%, 80%, 90% e até 100%, os valores se concentram entre 4 e 5 agentes com estado *Dead*.

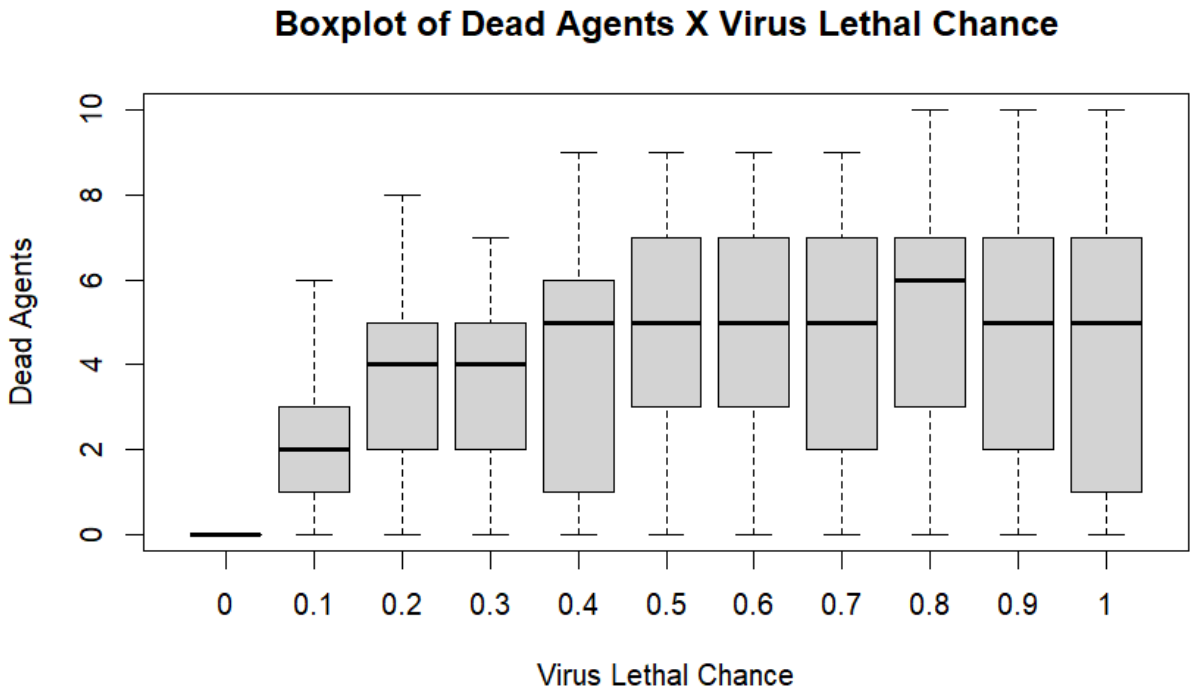

Figura 1.8: *Boxplot* obtido a partir dos dados do experimento realizado com 50 replicações

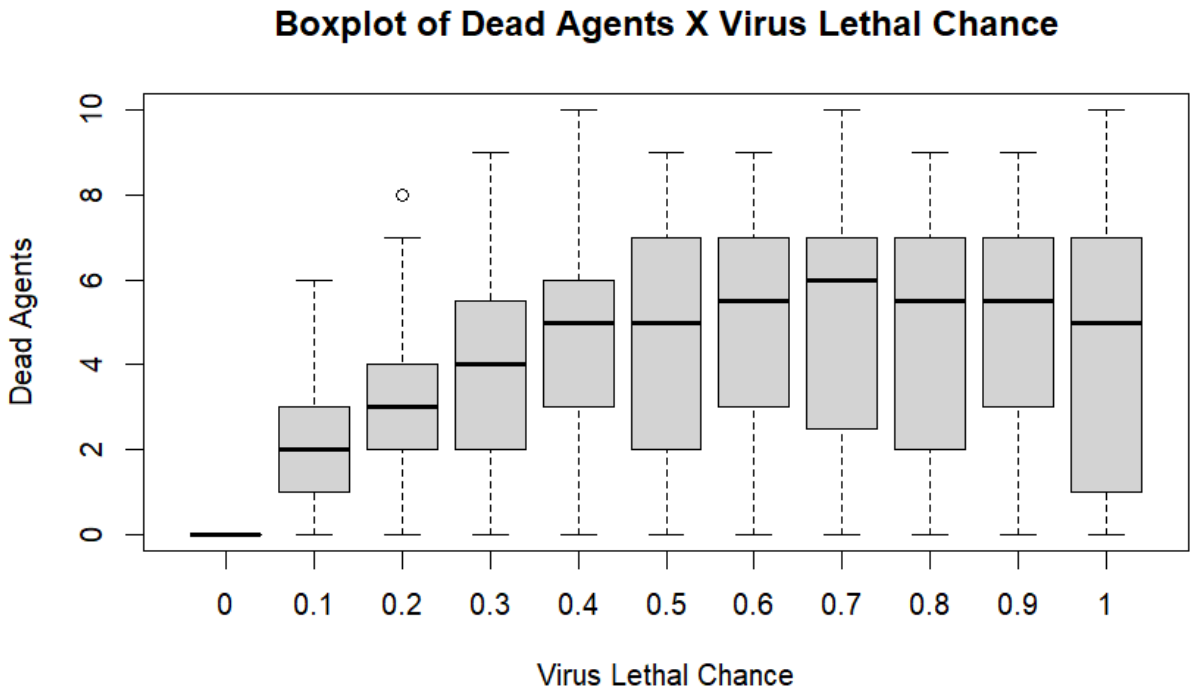

Figura 1.9: *Boxplot* obtido a partir dos dados do experimento realizado com 100 replicações

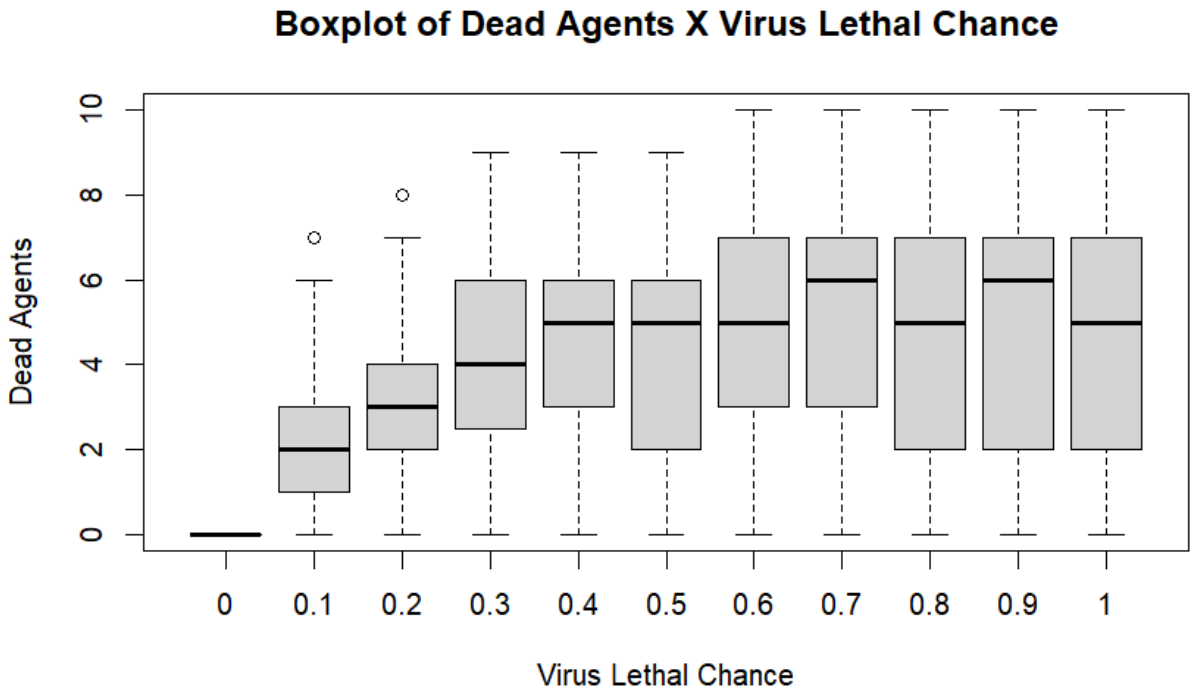

Figura 1.10: *Boxplot* obtido a partir dos dados do experimento realizado com 200 replicações

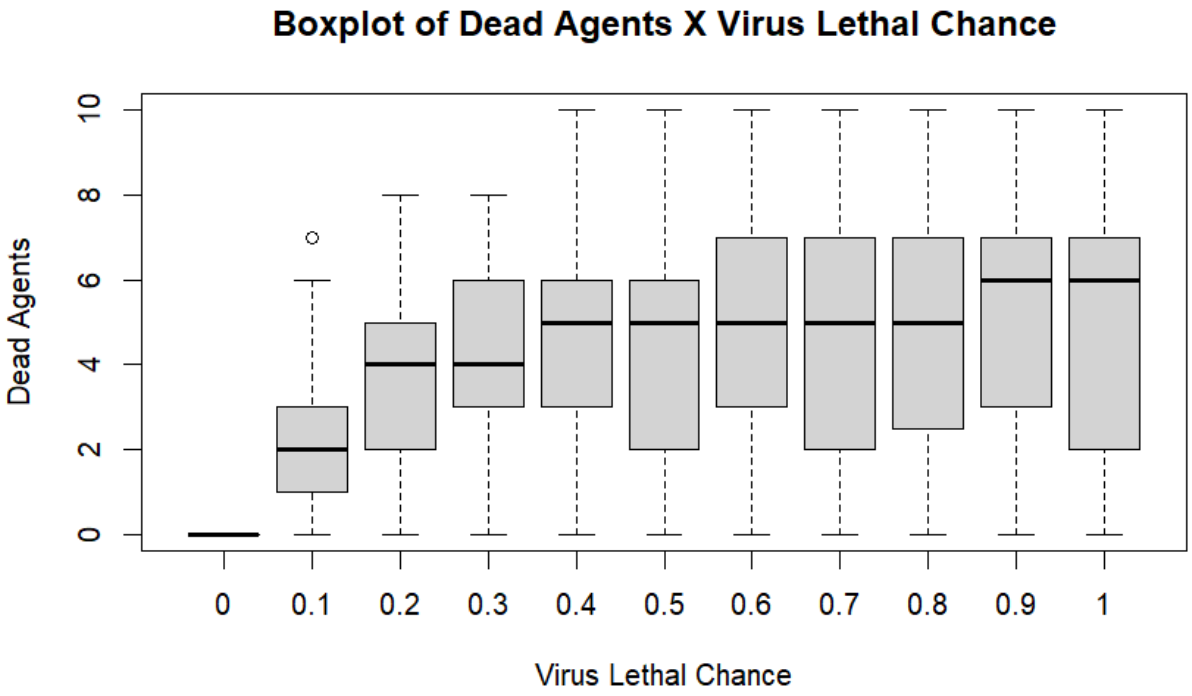

Figura 1.11: *Boxplot* obtido a partir dos dados do experimento realizado com 400 replicações

A seguir pode-se ver uma parte de uma tabela 1.1 (as tabelas completas possuem ao menos 220 linhas) usada para a geração dos histogramas e dos *boxplots*.

Variáveis presentes na tabela:

Virus Letal Chance: A probabilidade de um agente com estado *Infected* ter seu estado mudado para *Dead*.

Infected: Número de agentes com estado *Infected*.

Suscetible: Número de agentes com estado *Suscetible*.

Resistant: Número de agentes com estado *Resistant*.

Dead: Número de agentes com estado *Dead*.

Stable: Variável que define a estabilidade da simulação, sendo que 0 significa não estável e 1 significa estável.

virus_letal_chance	Infected	Susceptible	Resistant	Dead	Stable
0.0	0	3	7	0	1
0.1	0	3	4	3	1
0.2	0	4	1	5	1
0.30000000000000004	0	3	3	4	1
0.4	0	0	3	7	1
0.5	0	1	2	7	1
0.6000000000000001	0	0	6	4	1
0.7000000000000001	0	1	4	5	1
0.8	0	1	2	7	1
0.9	0	0	0	10	1
1.0	0	9	0	1	1
0.0	0	1	9	0	1
0.1	0	3	5	2	1
0.2	0	1	4	5	1
0.30000000000000004	0	1	3	6	1
0.4	0	2	1	7	1
0.5	0	5	2	3	1
0.6000000000000001	0	0	2	8	1
0.7000000000000001	0	3	2	5	1

Tabela 1.1: Tabela reduzida obtida a partir dos dados do experimento com 50 replicações

1.5 Discussão e *insights* preliminares sobre as hipóteses

Considerando a simplificação do fenômeno modelado em comparação com o fenômeno real de infecção viral, nota-se que em vista dos resultados obtidos nos diversos experimentos, inicialmente comprova-se a hipótese causal.

O experimento principal 1 prova que a letalidade por si só não consegue dominar sobre as outras variáveis e não causa um grande número de mortes. A exemplo, se a taxa de transmissão for baixa(com valores entre 0 e 0.1), mesmo a letalidade sendo máxima, o número de mortes não é suficientemente grande.

Ademais, o experimento principal 2 apresenta uma configuração de variáveis na qual o número de mortes é próximo ao máximo e essa configuração depende dos valores de 3 variáveis.

Com o auxílio dos dados coletados nesse experimento, pode-se reafirmar por meio dos histogramas e *boxsplots* a comprovação da hipótese causal.

1.6 Conclusão

Ao finalizar o experimento foi possível obter a resposta para os dois questionamentos que originaram a hipótese causal. Em relação a primeira pergunta, foi possível concluir que um vírus com alta letalidade pode infectar e consequentemente causar a morte de uma grande quantidade de pessoas se houver outras condições para isso, como por exemplo uma pequena taxa de resistência ao vírus e uma grande taxa de transmissão.

Já em relação a segunda pergunta, interpreta-se que não necessariamente existe essa relação de superioridade da quantidade de pessoas infectadas com um vírus menos letal, uma vez que foi achada uma configuração onde um vírus com letalidade máxima infectou um número bem próximo a quantidade de agentes.

Entretanto, existem várias questões não respondidas devido a simplificação do modelo. Por exemplo, indagações como quais outras variáveis influenciam na infecção viral (mutabilidade do vírus, resistência a antivirais, etc.) ou como os métodos de isolamento e de higiene pessoal influem na quantidade de infectados.

Bibliografia

BELAY, Ermias D; MONROE, Stephan S. Low-incidence, high-consequence pathogens. *Emerg Infect Dis*, v. 20, n. 2, p. 319–321, fev. 2014. Place: United States. ISSN 1080-6059. DOI: 10.3201/eid2002.131748. Disponível em: <http://dx.doi.org/10.3201/eid2002.131748>. Citado na p. 11.

DELIKHOON, Mahdieh et al. Modes of Transmission of Severe Acute Respiratory Syndrome-Coronavirus-2 (SARS-CoV-2) and Factors Influencing on the Airborne Transmission: A Review. *International Journal of Environmental Research and Public Health*, v. 18, n. 2, 2021. ISSN 1660-4601. DOI: 10.3390/ijerph18020395. Citado na p. 6.

F. STONEDAHL; WILENSKY, U. *NetLogo Models Library: Sample Models/Networks (back to the library)*. English. 2008. Disponível em: <http://ccl.northwestern.edu/netlogo/models/VirusonaNetwork>. Acesso em: 5 jan. 2023. Citado na p. 8.

GEOGHEGAN, Jemma L et al. Virological factors that increase the transmissibility of emerging human viruses. *Proc Natl Acad Sci U S A*, v. 113, n. 15, p. 4170–4175, mar. 2016. Place: United States. ISSN 1091-6490. DOI: 10.1073/pnas.1521582113. Disponível em: <http://dx.doi.org/10.1073/pnas.1521582113>. Citado na p. 11.

LEUNG, Nancy H. L. Transmissibility and transmission of respiratory viruses. *Nature Reviews Microbiology*, v. 19, n. 8, p. 528–545, ago. 2021. ISSN 1740-1534. DOI: 10.1038/s41579-021-00535-6. Disponível em: <https://doi.org/10.1038/s41579-021-00535-6>. Citado na p. 6.