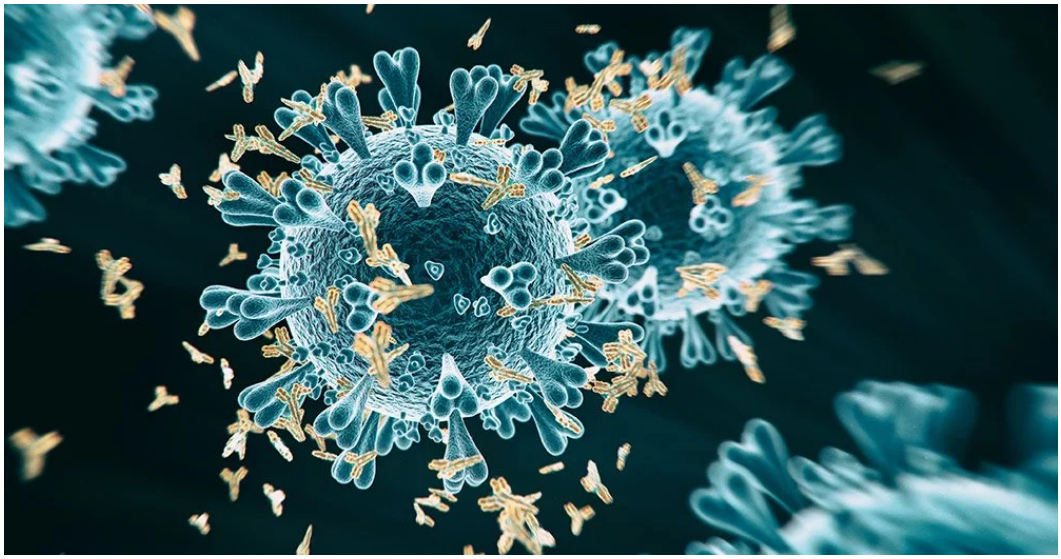
Corona Simulation



Ein Modell als Hilfestellung von Entscheidungen in der Corona-Krise

Sommersemester 2021
Problemlösung Algorithmen und Datenstrukturen

Kollaborateure: Jeroshan Empeerious und Salih Tuncer

1 Einleitung

1.1 Motivation

Durch die Corona-Krise hat sich das Leben verändert. Arbeiten von zu Hause, Schulunterricht per Videokonferenzen, virtuelles Treffen mit Freunden. Die Anzahl der sozialen Kontakte sinkt immer weiter, während die Fallzahlen steigen. Es ist eine Frage der Moral, ob und wie man sich im realen Leben noch treffen kann. Eine gute Hilfe bei der Entscheidung kann ein Algorithmus sein, der dabei hilft zu verstehen, wie das Treffen einer bestimmter Anzahl von Personen den Inzidenzwert beeinflussen kann.

1.2 Problembeschreibung als Auftraggeber

Es soll ein Programm entworfen werden, welches mithilfe eines Algorithmus' zur Berechnung des Inzidenzwertes und somit zum Entscheidungsproblem beiträgt. Diese ist eine wichtige Kennziffer, um darzustellen, wie hoch die Zahl der Infizierten in Relation zur Bevölkerung ist. So soll man daraus schlussfolgern können, inwiefern das Treffen mit einem oder mehreren Freunden - ohne Auswirkungen auf die gesundheitliche Krise - ausgeführt werden kann.

1.3 Veranschaulichung des ersten Ansatzes zur Problemlösung

Man könnte davon ausgehen, dass wir lediglich ein exponentielles Wachstum visualisieren müssen, jedoch stecken viele Faktoren dahinter, wie sich das Virus verbreitet.

- Inkubationszeit: laut dem RKI[1] liegt die aktuelle Inkubationszeit zwischen fünf bis sechs Tagen.
- Fallsterblichkeitsrate: diese liegt jüngsten Umfragen nach bei circa 2.75%[2].
- Krankheitsdauer: der Frankfurter Rundschau[3] zufolge ist das Virus bis zu 18 Tage ansteckend.

---

[1] „Epidemiologischer Steckbrief zu SARS-CoV-2 und COVID-19" (RKI, 2021, „5. Inkubationszeit und serielles Intervall")

[2] „Letalitätsrate beim Coronavirus (COVID-19) in den am stärksten betroffenen Ländern" (Statista, 2021)

[3] „Corona-Infektion: Wie lange ist man ansteckend? Forschende kommen der Antwort auf die Spur" (Frankfurter Rundschau, 2021)

- Kontakte/Person: einer Studie[4] nach hat der durchschnittliche Deutsche einen erweiterten Freundeskreis aus 42 Personen. Wir gehen davon aus, dass man jeden Freund*in durchschnittlich zweimal im Jahr trifft. Aus der Rechnung $\frac{365}{(2 \cdot 42)} = 4.35$ leiten wir ab, dass sich eine Person alle vier Tage mit einer anderen trifft.

Diese Parameter sollen durch eine Datei in unseren Algorithmus eingeschleust werden. Für uns ist dies wichtig, da sich die Statistiken auf jüngsten Begebenheiten basieren und sich jederzeit ändern können. Durch graphische, als auch tabellarische Ansichten soll das Programm auf die Entwicklung des Corona-Virus hinweisen und notfalls auch Besserungen oder Warnungen angeben.

2 Version 1

2.1 Datenstruktur

```
class Simulation():

    def __init__(self, config: dict[str, float]):
        # Wir initialisieren unsere Konfigurationen in der
        # Simulationsklasse. Die Virus-Kette wird ebenso
        # dokumentiert. Wir haben in Kapitel 1.3 bereits
        # eine Formel kreiert, wie häufig sich der/die
        # Deutsche*in im Jahr mit Freunden*innen trifft.
        # Diese Formel wird hier nochmal berechnet

    # Representation wird aufgerufen, sobald das Objekt
    # versucht wird, ausgegeben zu werden.
    def __repr__(self):
        # Hierbei werden die Konfigurationen der Klasse
        # vernünftig formatiert ausgegeben.

    # Hauptmerkmal des ganzen Programmes
    def simulate(self):
        # Abhängig von der Simulationsdauer gegeben in
        # der Konfiguration, als auch der Lebensdauer
        # des Viruses, wird das Verhalten des Virus' simuliert.
```

[4] „Deutsche haben 3,7 enge Freunde – Offene Kommunikation und Fürsorge in einer Freundschaft am wichtig“ (YouGov, 2018)

```
class Virus():

    def __init__(self, creation_date: int, duration_of_infection:
float):
        # Das Entstehungsdatum der Virus-Kette dient als
        # eindeutige ID. Die Lebenszeit wird ebenso dokumentiert.

    # Wird aufgerufen, sobald das Objekt entfernt wird.
    def __del__(self):
        # Eine Ausgabe der verstorbenen Virus-Kette

    def day_ends(self):
        # Es vergeht ein Tag für den Virus.

    def is_vanished(self):
        # Schaut, ob der Virus schon verstorben ist.

# Wird beim Start des Programmes ausgeführt.
def main():
    # Hier wird die Konfigurations-Datei eingelesen
    # und ausgewertet.Mit dieser Konfiguration,
    # initialisieren wir unseren Simulator.
```

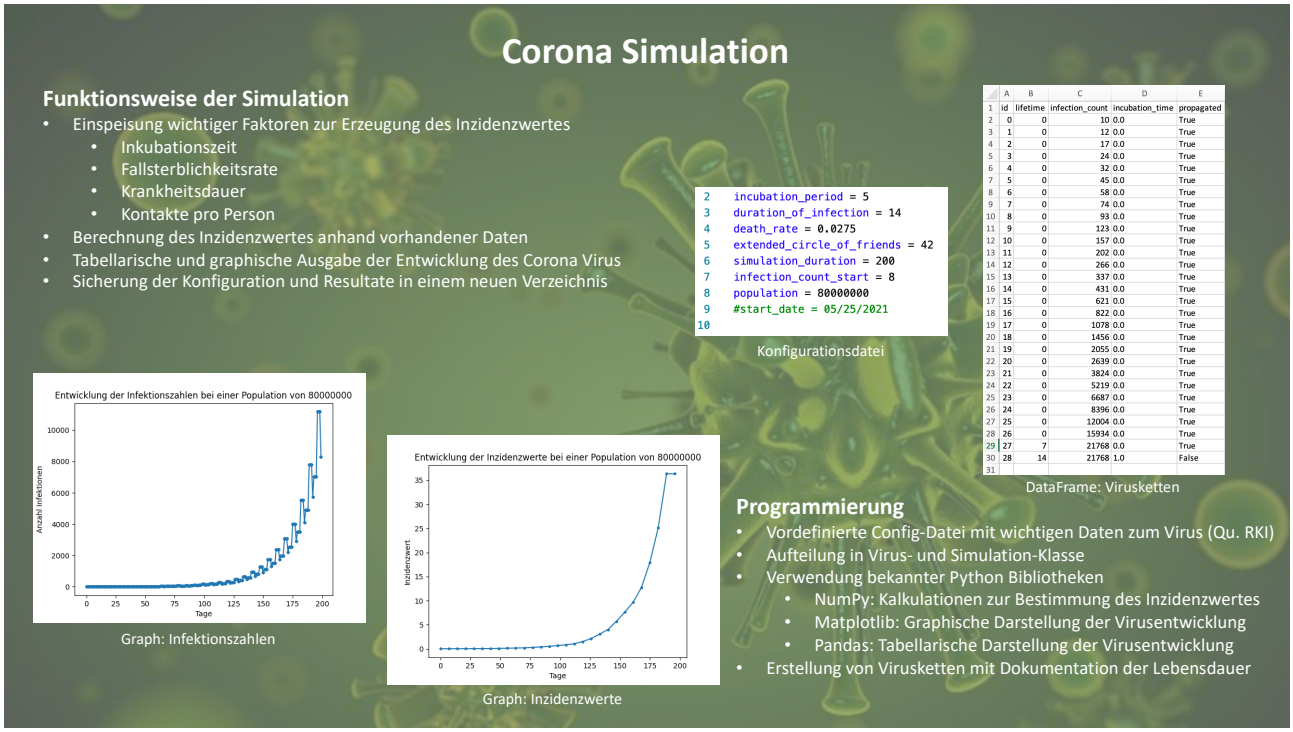
2.2 Bibliotheken

- NumPy: Zum Dokumentieren unserer Viren-Ketten, aber auch für Kalkulationen und andere Vorgehen verwenden wir die Python-Bibliothek NumPy. Es verbindet die Schnelligkeit der Programmiersprache C und die einfache Bedienung von Python und ist mittlerweile eine Grundlage für viele andere Bibliotheken.
- Matplotlib: Zudem möchten wir gerne die Bibliothek Matplotlib verwenden, um die Ergebnisse durch Graphen zu visualisieren, da der Mensch ein visuell-orientiertes Lebewesen ist.
- Seaborn: Sollten uns die Visualisierungen der Bibliothek jedoch nicht genügen, ziehen wir in Betracht, die Bibliothek Seaborn zu beanspruchen. Die Konfiguration dieser ist aufwändiger, aber führt zu besseren Visualisierungen.
- Pandas: Ebenso möchten wir gerne die Daten in Tabellen darstellen. Dafür eignet sich die Bibliothek Pandas besonders gut. Dort wird nämlich eine Datenstruktur namens

DataFrame angeboten, welche eine äußerst gute Darstellung der Daten in Tabellenformat anbietet. Mit DataFrames kann man zudem jegliche Kalkulationen als auch Manipulationen an den Daten vornehmen.

- PyGt5.QtWidgets: Wird zur Darstellung der GUI verwendet, um die Anwendung für den Benutzer mit Hilfe einer grafischen Oberfläche attraktiver zu machen

3 Plakat



4 Version 2

4.2 Datenstruktur

```
class Simulation:

    def get_total_active_infections_count(self) -> int:
        # gibt Anzahl aller bisherigen Infektionen aufsummiert
          zurück

    def get_total_infections_count(self) -> int:
        # gibt Anzahl aktiver Infektionen zurück
```

```
 def get_incidence_values(self) -> int:
        # kalkuliert Inzidenz-Wert in Abhängigkeit der Bevölkerung
          und der aktiven Infektionszahlen

   def get_mortality(self) -> int:
        # gibt Anzahl verstorbener Menschen zurück

    def get_seven_day_incidence(self) -> int:
        # gibt die 7-Tages-Inzidenz zurück

    def get_seven_days_total_infections_count(self) -> int:
        # gibt Infektionszahlen der letzten sieben Tage zurück

    def spread_infection_history(self) -> int:
        # verteilt die Infektionszahlen realistisch über die
          Tage, um keine lineare Entwicklung darzustellen


# Namen von Virus zu Infection_Chain umgeändert
class Infection_Chain:
    # gibt an, ob sich der Virus bereits entfaltet hat
    propagated: bool

    def propagate(self):
        # Infektions-Kette wird erst beachtet und simuliert,
          sobald die Inkubationszeit vorbei ist


def plot_data(_infection_history, _incidence_values):
    # gibt einen Plot aus in Abhängigkeit der Infetionszahlen und
     der Inzidenz-Werte

def analyze_as_data_frame(_virus_chain):
    # formt alle Infection_Chain-Klassen in DataFrames (Tabellen)
     um, um sie daraufhin in einer .csv-Dateie zu speichern

if __name__ == '__main__':
```

```
    # erstellt nun auch einen Ordner, in denen die Plots, die
     Konfiguraitons-Datei und die csv-Datei gespeichert werden
```

4.3 Programm-Anleitung

Vor dem Starten des Programmes muss eine Config-Datei in das Programm-Verzeichnis eingefügt werden. Diese Datei wird als *config.cfg* gespeichert und muss folgende Werte definieren:

`incubation_period:` Inkubationszeitraum
`duration_of_infection:` Dauer der Infektion
`death_rate:` Sterberate
`extended_circle_of_friends:` Anzahl der Personen im Freundeskreis
`simulation_duration:` Dauer der Simulation

Anschließend navigiert man über das Terminal in das Programm-Verzeichnis. Das Programm lässt sich entweder im Terminal öffnen oder auch in einer grafischen Benutzeroberfläche.

Klassischer Start im Terminal:

python3 CoronaSimulation.py --mode=cmd

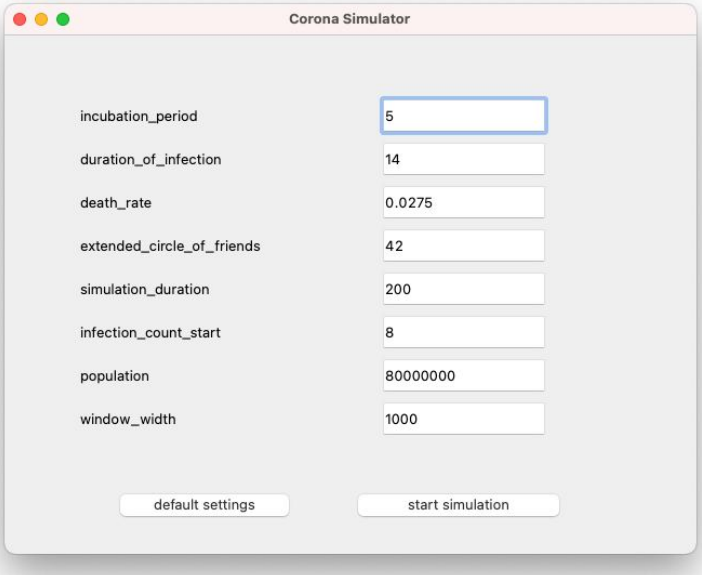
Start in der grafischen Benutzeroberfläche:

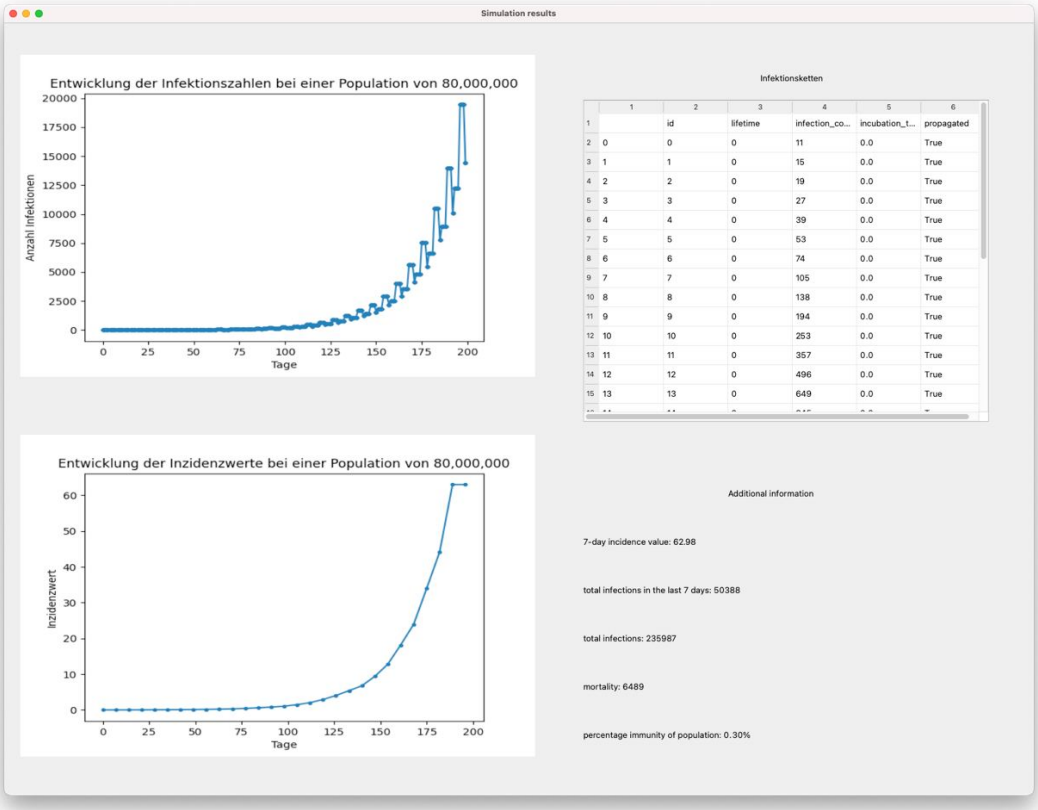
python3 CoronaSimulation.py --mode=gui

Das Programm läuft den Algorithmus durch und berechnet anhand der vordefinierten Werte aus der Config-Datei den Verlauf der Infektionszahlen und Inzidenzwerte und stellt diese schließlich graphisch und tabellarisch dar.

Wird das Programm klassisch im Terminal ausgeführt, wird die Berechnung automatisch mit den vordefinierten Werten aus der Config-Datei ausgeführt. In der GUI-Anwendung ist es möglich, vor Ausführen des Algorithmus die Werte manuell anzupassen. Außerdem lässt sich die Fenstergröße der Anwendung über das Feld *window_width* anpassen. Je nach ausgewählter Größe passen sich auch die Grafiken und Tabellen dem Fenster an. Möchte man lieber die vordefinierten Werte nutzen, kann man die Eingabe über den Button *default settings* wieder zurücksetzen.

Durch Klicken auf *start simulation* wird dann das Fenster geöffnet, welches nach Berechnung die Ergebnisse der Simulation darstellt.





5 Zusammenfassender Projektbericht

Seit Beginn der Kollaboration in diesem Projekt ist uns aufgefallen, dass durch die Produktivität mit der Zeit gestiegen ist und wir die erarbeiteten Informationen durch das gemeinsame Brainstorming somit nutzen konnten, die Software immer weiter auszubauen, sodass sie nun anwenderfreundlich und nutzbereit ist. Dafür war die Strukturierung der Information von oberster Priorität, um einen klaren Durchblick während der Arbeit zu behalten. Unsere Erfahrungen im Bereich der Python-Programmierung, Python-Bibliotheken und auch die Nutzung von Latex zur Erstellung von Projektmappen sind stark gestiegen, durch Recherchieren, um die entstandenen Probleme zu lösen. Das Kommentieren der Codes hat dafür gesorgt, schneller einen Überblick zu kriegen, wie sich eine Änderung in einer Funktion auf die Ausführung des Programms auswirkt, sodass wir daran von Anfang an zur Sicherung einer hohen Code-Qualität gedacht haben.

Abschließend kann festgestellt werden, dass wir unseren eigenständig gesetzten Zeit- und Qualitätsziel übertroffen haben und das Ziel somit mit Erfolg erreicht wurde. Durch eine effektive und motivierte Zusammenarbeit können wir rückblickend sagen, dass die Kommunikation eine wichtige Rolle dabei spielt, die Ziele zu erreichen. Auch sind wir durch die sehr ausführliche und detailreiche Dokumentation sehr zufrieden mit unseren erbrachten Leistungen und das Ergebnis der monatelangen Arbeit.

6 Quellenverzeichnis und Literatur

- „Epidemiologischer Steckbrief zu SARS-CoV-2 und COVID-19“ (RKI, 2021, „5. Inkubationszeit und serielles Intervall“)
- „Letalitätsrate beim Coronavirus (COVID-19) in den am stärksten betroffenen Ländern“ (Statista, 2021)
- „Corona-Infektion: Wie lange ist man ansteckend? Forschende kommen der Antwort auf die Spur“ (Frankfurter Rundschau, 2021)
- „Deutsche haben 3,7 enge Freunde – Offene Kommunikation und Fürsorge in einer Freundschaft am wichtig“ (YouGov, 2018)