CX 4230 - Mini Project 1

Eduardo Amorim

February 2023

1 Introduction

Abstract

Our task in this project is to implement and analyze a "Well Mixed" SIR Model and in the end adapt it to become a SIRV Model.

2 SIR Model

The Susceptible-Infected-Recovered Model is a simple yet interesting representation for disease spreading. Even with its simplicity, the applications of it are countless. For this model, there are three types of people:

a. *susceptible* people, who have never had contact with the disease;

b. *infected* people, who are currently contaminated;

c. *recovered* people, who had the disease and developed immunity against it.

The disease can only spread from infected people to susceptible people. An infected person stays sick for some period, but then recovers; once they have recovered, they can never get sick again. Let's assume the behavior of the disease is governed by the following mathematical model:

$$\vec{x} = \vec{x}(t) = \begin{bmatrix} S(t) \\ I(t) \\ R(t) \end{bmatrix}$$

Where $S(t)$, $I(t)$, and $R(t)$ denote the fractions of the population who are susceptible, infected, or recovered, respectively at time t. In addition, note that we assume that no one is born and no one dies, so that the conservation law holds and $S(t) + I(t) + R(t) = 1$, $\forall t \in [0, \infty]$
Suppose the state $\vec{x}$ evolves as an ordinary differential equation,

$$D\vec{x} = \vec{f}(\vec{x})$$

where $\vec{f}(\vec{x})$ is the following vector function:

$$\vec{f}(\vec{x}) = \begin{bmatrix} -\tau S(t)I(t) \\ \tau S(t)I(t) - \frac{I(t)}{\kappa} \\ \frac{I(t)}{\kappa} \end{bmatrix}$$

The parameter $\tau \geq 0$ measures how quickly the disease can spread, with higher values corresponding to faster rates of spread. The parameter $\kappa > 0$ measures how quickly an infected person recovers, with higher values corresponding to slower (longer) recovery times.

2.1 Fixed points of the system:

To find the fixed points, we need to follow the Nullclines and substitute the values that make each of the derivatives equal to zero. For our system we have:

- Nullcline for $Ds = 0$:

$$-\tau S(t)I(t) = 0$$

$$S(t) = 0 \implies Di = -\frac{I(t)}{\kappa},\ Dr = \frac{I(t)}{\kappa}$$

$$I(t) = 0 \implies Di = 0,\ Dr = 0$$

- Nullcline for $Di = 0$:

$$\tau S(t)I(t) - \frac{I(t)}{\kappa} = 0$$

$$I(t) = 0 \implies Ds = 0,\ Dr = 0$$

$$\tau S(t) - \frac{1}{\kappa} = 0 \ \therefore$$

$$S(t) = \frac{1}{\tau\kappa} \implies Ds = \frac{I(t)}{\kappa},\ Dr = \frac{I(t)}{\kappa}$$

- Nullcline for $Dr = 0$:

$$\frac{I(t)}{\kappa} = 0$$

$$I(t) = 0 \implies Ds = 0,\ Di = 0$$

Therefore, we can conclude that we have a fixed point at $\vec{x}_* = (0, 0, 0)$ and at $\vec{x}_* = (\frac{1}{\tau\kappa}, 0, 0), \forall\kappa, \tau$. Now, we will classify the stability of these points.
Let $\vec{x}(t) = \vec{x}_* + s$, for a small $||s|| = s(t)$, then $Ds = f(\vec{x}_* + s)$.
Then, linearize at $\vec{x}_*$ using a multivariate Taylor expansion:

$$f(\vec{x}_* + s) = f(\vec{x}_* + J_f(\vec{x}_*)s$$

$$J_f(\vec{x}_{*1}) = \begin{bmatrix} -\tau I(t) & -\tau S(t) & 0 \\ \tau I(t) & \tau S(t) - \frac{1}{k} & 0 \\ 0 & \frac{1}{k} & 0 \end{bmatrix}$$

Which translates in a fixed line for I(t) = 0. By the eigenvalue/ eigenvector analysis, we can conclude the point $\vec{x} = (0, 0, 0)$ is stable. Analogously, for $\vec{x} = (\frac{1}{\tau\kappa}, 0, 0)$ we have:

$$J_f(\vec{x}_{*2}) = \begin{bmatrix} 0 & -\frac{1}{k} & 0 \\ 0 & 0 & 0 \\ 0 & \frac{1}{k} & 0 \end{bmatrix}$$

By the eigenvalue/ eigenvector analysis, we can conclude the point $\vec{x} = (\frac{1}{\tau\kappa}, 0, 0)$ is unstable.

2.2 Implementation of the model in Python:

- Assume the initial conditions to be: $I(0) = 0.01$, $S(0) = 0.99$, and $R(0) = 0$.

```
import numpy as np
from scipy.integrate import solve_ivp
import matplotlib.pyplot as plt

# Initializing initial conditions:
i0 = 0.01
s0 = 0.99
r0 = 0
x0 = [s0, i0, r0]

# SIR Model
def ode(x, y):
  S, I, R = y
  Ds = -tau*S*I
  Di = -I/kappa + tau*I*S
  Dr = I/kappa
  return [Ds, Di, Dr]
```

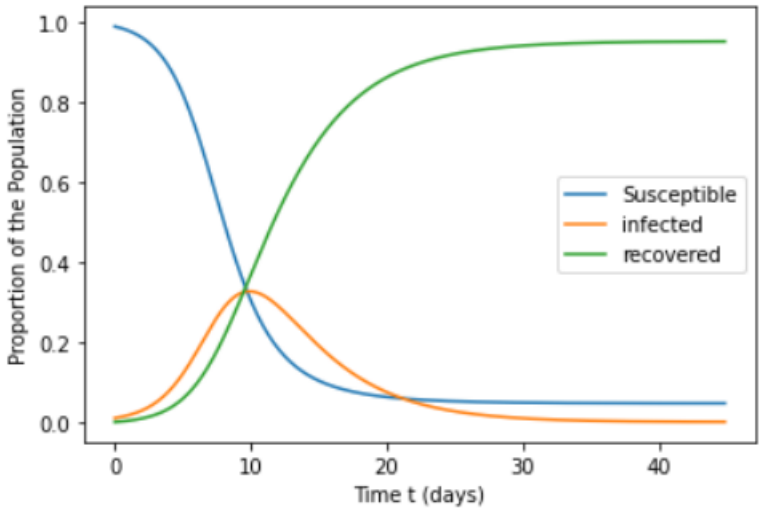
- Let $\tau = 0.8$ and $\kappa = 4$.

```
# First simulation
tau = 0.8
kappa = 4

# Set event for "stopping condition"
def event(x, y):
  result = ode(x, y)
```

```
  return result[2] - 0.0001
event.terminal=True

sol = solve_ivp(ode, (0,100), (0.99, 0.01, 0), method='RK45',
    max_step=0.1, events=[event])
plt.plot(sol.t, sol.y[0].T, label='Susceptible')
plt.plot(sol.t, sol.y[1].T, label='infected')
plt.plot(sol.t, sol.y[2].T, label='recovered')
print(sol.t_events[0])
plt.legend()
plt.show()
```




```
Stopping time: 44.87
Peak infected: 9.90 days, wtih proportion of population: 0.33
```

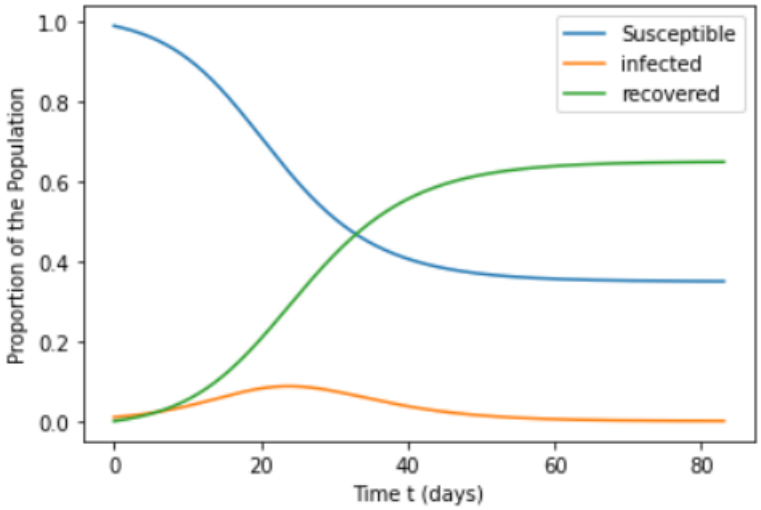
The stopping time calculated is 44.87.

- Let $\tau = 0.4$ and $\kappa = 4$.

```
# Second simulation
tau = 0.4
kappa = 4

sol = solve_ivp(ode, (0,100), (0.99, 0.01, 0), method='RK45',
    max_step=0.1, events=[event])

plt.plot(sol.t, sol.y[0].T)
plt.plot(sol.t, sol.y[1].T)
plt.plot(sol.t, sol.y[2].T)
print(sol.t_events[0])
plt.legend()
plt.show()
```




```
Stopping time: 83.08
Peak infected: 23.79 days, wtih proportion of population: 0.09
```

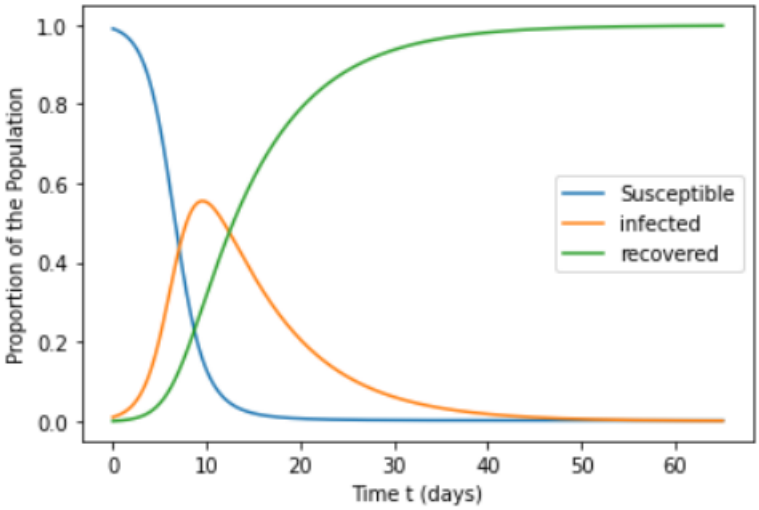
The stopping time calculated is: 83.08.

- Let $\tau = 0.8$ and $\kappa = 8$.

```python
# Third simulation
tau = 0.8
kappa = 8

sol = solve_ivp(ode, (0,100), (0.99, 0.01, 0), method='RK45',
    max_step=0.1, events=[event])

plt.plot(sol.t, sol.y[0].T)
plt.plot(sol.t, sol.y[1].T)
plt.plot(sol.t, sol.y[2].T)
print(sol.t_events[0])
```




```
Stopping time: 65.06
Peak infected: 9.56 days, wtih proportion of population: 0.56
```

The stopping time calculated is: 65.06.

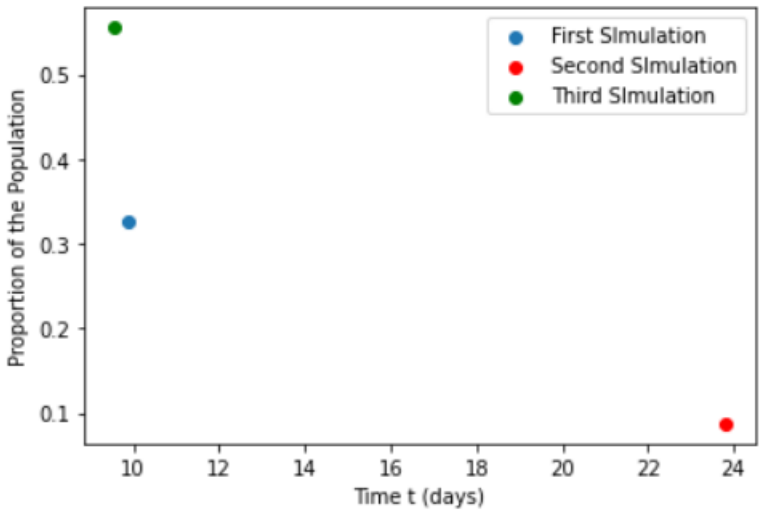
2.3 Summary of the findings:

We can see the strong influence that the choice of the constants τ and κ have on the development in time of our system.

It is particularly interesting to see how the stopping time stretches harshly when the recovery time is divided by two. We go from 44.87 in Simulation 1 days to 83.08 in Simulation 2, while keeping the velocity of the spread constant. When we make our rate of spread bigger, while having a longer recovery time, in Simulation 3, we can see that while the infected proportion is much larger, the overall time to eradication is not as big as Simulation 2.

The peak infected proportion is an interesting point in time since it represents the largest possible proportion of people that are simultaneously infected. It is a quantity that is directly proportional to both τ and κ as an increase in those will affect positively the proportion of infected at the peak. Thus, we can maximize the peak of infected by maximizing the product of τ and κ.

Furthermore, the relationship of τ and the stopping time while keeping κ constant is very interesting. We can see that diseases with a smaller rates of spreading tend to take a longer time to naturally eradicate.



3 SIRV Model

The Susceptible-Infected-Recovered-Vaccinated is a natural upgrade for the SIR Model. Suppose there is a vaccine that can be given to susceptible people. A susceptible person who receives a vaccine can no longer become infected. However, this vaccine is very expensive, so you cannot vaccinate all susceptible people. Our task, is to develop a "policy" of vaccination. That is, introduce a fourth state variable, $V(t)$, that is the fraction of the population who are vaccinated at time t, such that the conservation law, $S(t)+I(t)+R(t)+V(t) = 1$, holds. Let's assume the new behavior of the disease is governed by the following mathematical model:

$$\vec{x} = \vec{x}(t) = \begin{bmatrix} S(t) \\ I(t) \\ R(t) \\ V(t) \end{bmatrix}$$

In addition, the state $\vec{x}$ still evolves as an ODE.
Our goal is to determine a good "policy" $\frac{dV}{dt}$ that has a beneficial (from our human perspective) impact for the development of the disease.

3.1 Discussion of the policy:

The key to plan a vaccination policy is to understand its relations with the other variables in our system. The quantity $V(t)$, that represents the number of people vaccinated in one day, is independent of $R(t)$ since they do not affect each other. However, if there is vaccination, it means that there will be less people

in the susceptible and, by consequence, the infected proportion. Therefore, we know that these three quantities are inversely proportional.
When developing a "policy", the most natural idea is to note that if you vaccinate people, $S(t)$ should decrease and $V(t)$ should increase by the same amount. Therefore, we can update the rate of change of susceptible individuals to:

$$-\tau S(t)I(t) - \mu V(t), \mu \geq 0$$

Therefore, the states evolve through time by

$$D\vec{x} = \vec{f}(\vec{x})$$

where $\vec{f}(\vec{x})$ is the following vector function:

$$\vec{f}(\vec{x}) = \begin{bmatrix} -\tau S(t)I(t) - \mu V(t) \\ \tau S(t)I(t) - \frac{I(t)}{\kappa} \\ \frac{I(t)}{\kappa} \\ \mu V(t) \end{bmatrix}$$

However, this model is not enough to describe our system, since our rate of vaccination (Dv) is not known beforehand. Hence, we can use the fact that the quantities Dv, $S(t)$, and $I(t)$ are inversely proportional to construct the following relation:

$$\frac{1}{Dv} \propto \frac{1}{S(t)} + \frac{1}{I(t)} \therefore$$

$$Dv \propto \frac{S(t)I(t)}{S(t) + I(t)}$$

We can include that in our state evolution model by adding a constant ν to regulate the vaccination behavior. Then, we have:

$$\vec{f}(\vec{x}) = \begin{bmatrix} -\tau S(t)I(t) - \nu \frac{S(t)I(t)}{S(t)+I(t)} \\ \tau S(t)I(t) - \frac{I(t)}{\kappa} \\ \frac{I(t)}{\kappa} \\ \nu \frac{S(t)I(t)}{S(t)+I(t)} \end{bmatrix}$$

3.2 Implementation in Python

```
nu = 0.3
def odeVac(x, y):
  S, I, R, V = y
  Ds = -tau*S*I - (nu*S*I)/(I+S)
```

```
  Di = -I/kappa + tau*I*S
  Dr = I/kappa
  Dv = (mu*S*I)/(I+S)
  return [Ds, Di, Dr, Dv]
```

```
def eventVac(x, y):
  result = odeVac(x, y)
  return result[2] - 0.0001
eventVac.terminal=True

def peakInfectedVac(x, y):
  result = odeVac(x, y)
  return result[1]
```

1. Simulation 1

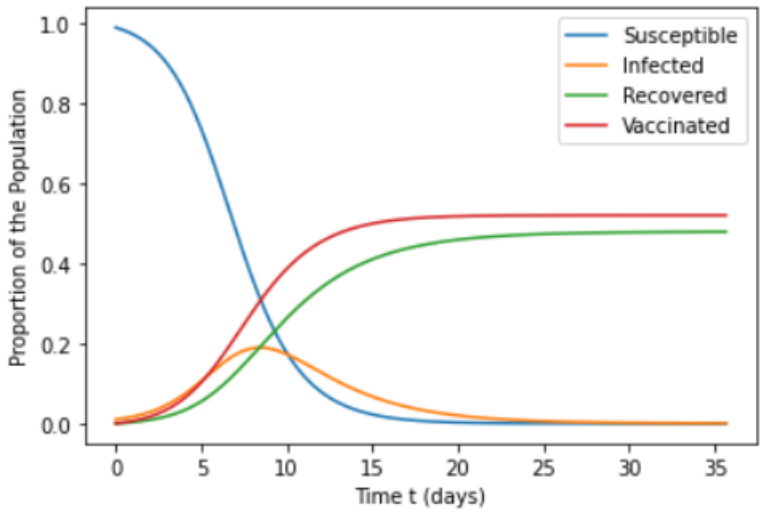
```
# SIRV Simulation 1

# Update the constants
tau = 0.8
kappa = 4

sol = solve_ivp(odeVac, (0,100), (0.99, 0.01, 0, 0), method='
    RK45',max_step=0.1, events=[eventVac, peakInfectedVac])
plt.plot(sol.t, sol.y[0].T, label='Susceptible')
plt.plot(sol.t, sol.y[1].T, label='Infected')
plt.plot(sol.t, sol.y[2].T, label='Recovered')
plt.plot(sol.t, sol.y[3].T, label='Vaccinated')
plt.xlabel('Time t (days)')
plt.ylabel('Proportion of the Population')
plt.legend()
plt.show()

stop = sol.t_events[0]
peak = sol.t_events[1]
yPeak = sol.y_events[1][0][1]

print('Stopping time: ' + '{:.2f}'.format(float(stop)))
print('Peak infected: ' + '{:.2f}'.format(float(peak)) + '
    days, wtih proportion of population: ' + '{:.2f}'.format(
    float(yPeak)))
```




```
Stopping time: 35.65
Peak infected: 8.43 days, wtih proportion of population: 0.19
```

Interesting notes:

→ Eradication time decreased from 44.87 days to 35.65 days;

→ Peak proportion of infected decreased from 0.33 to 0.19.

→ Peak infected day decreased from 9.90 days to 8.43 days.

2. Simulation 2

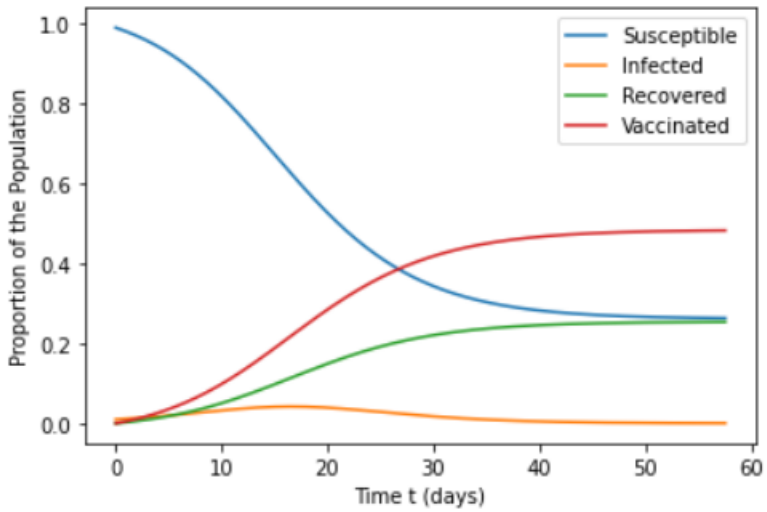
```
# SIRV Simulation 2

# Update constants
tau = 0.4
kappa = 4

sol1 = solve_ivp(odeVac, (0,100), (0.99, 0.01, 0, 0), method='
    RK45',max_step=0.1, events=[eventVac, peakInfectedVac])
plt.plot(sol1.t, sol1.y[0].T, label='Susceptible')
plt.plot(sol1.t, sol1.y[1].T, label='Infected')
plt.plot(sol1.t, sol1.y[2].T, label='Recovered')
plt.plot(sol1.t, sol1.y[3].T, label='Vaccinated')
plt.xlabel('Time t (days)')
plt.ylabel('Proportion of the Population')
plt.legend()
plt.show()

stop = sol1.t_events[0]
peak = sol1.t_events[1]
yPeak = sol1.y_events[1][0][1]
```

```
20
21 print('Stopping time: ' + '{:.2f}'.format(float(stop)))
22 print('Peak infected: ' + '{:.2f}'.format(float(peak)) + '
       days,  wtih proportion of population: ' + '{:.2f}'.format(
       float(yPeak)))
```




```
Stopping time: 57.56
Peak infected: 16.57 days,  wtih proportion of population: 0.04
```

Interesting notes:

→ Eradication time decreased from 83.08 days to 57.56 days;

→ Peak proportion of infected decreased from 0.09 to 0.04;

→ Peak infected day decreased from 23.79 days to 16.57 days.

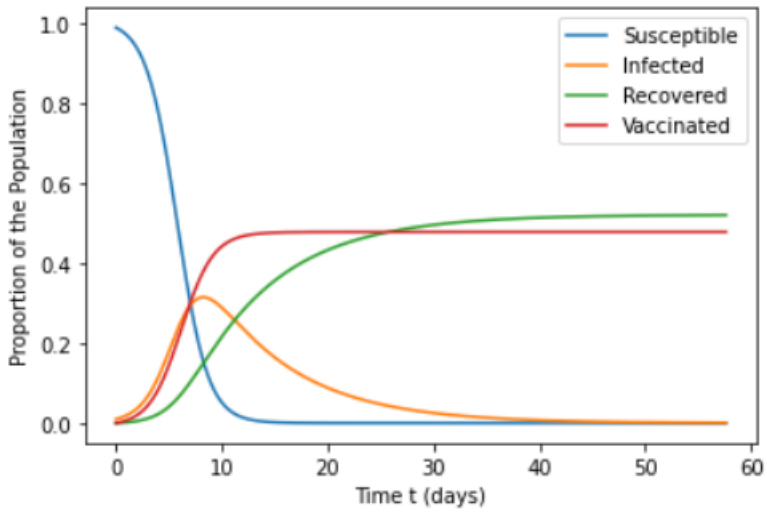
3. Simulation 3

```
1 # SIRV Simulation 3
2
3 # Update the constants
4 tau = 0.8
5 kappa = 8
6
7 solVac3 = solve_ivp(odeVac, (0,100), (0.99, 0.01, 0, 0),
      method='RK45',max_step=0.1, events=[eventVac,
      peakInfectedVac])
8 plt.plot(solVac3.t, solVac3.y[0].T, label='Susceptible')
9 plt.plot(solVac3.t, solVac3.y[1].T, label='Infected')
```

```
plt.plot(solVac3.t, solVac3.y[2].T, label='Recovered')
plt.plot(solVac3.t, solVac3.y[3].T, label='Vaccinated')
plt.xlabel('Time t (days)')
plt.ylabel('Proportion of the Population')
plt.legend()
plt.show()

stop = solVac3.t_events[0]
peak = solVac3.t_events[1]
yPeak = solVac3.y_events[1][0][1]

print('Stopping time: ' + '{:.2f}'.format(float(stop)))
print('Peak infected: ' + '{:.2f}'.format(float(peak)) + '
    days, wtih proportion of population: ' + '{:.2f}'.format(
    float(yPeak)))
```




```
Stopping time: 57.62
Peak infected: 8.23 days, wtih proportion of population: 0.32
```

Interesting notes:

→ Eradication time decreased from 65.06 days to 57.52 days;

→ Peak proportion of infected decreased from 0.56 to 0.32;

→ Peak infected day decreased from 9.56 days to 8.23 days.

4 Conclusion

We can conclude that the vaccination policies are an interesting addition to the SIR Model. It was a great learning experience.

5 References

Dianne O'Leary. Scientific computing with case studies. SIAM, 2009.
DOI: 10.1137/9780898717723