Epidemic Control

Epidemic Control

As our next case study we will consider an epidemic control problem

- Let's assume we are at early stages of an epidemic
- ...And we want to do our best to control while we wait for a cure/vaccine

Epidemic Control as Optimization

Technically, this is an optimization problem

- We need to decide which actions to take
- ...Subject to a variety of constraints (e.g. socio economical impact)
- ...So that the total number of infected is minimized

But how do we evaluate the impact of our actions?

Epidemic Control as Optimization

Epidemical dynamics can be simulated

- We can use differential equations
- We can use multi-agent models
- We can use network models to account for connections

However, even if simulators are defined via well-known rules and equations

...Using these in declarative optimization is typically very difficult

- A multi-agent simulator may have too many agents
- A differential equation may introduce too many non-linearities
- ...And both may need to run for too many steps

Black-box optimization is an option, but it cannot deal easily with constraints

Optimizing over ML Models

We will tackle this problem by combining ML and optimization

In particular, we will use ML to model part of our optimization problem

- In particular, we will learn a ML model as usual
- Then we will find a way encode it into a given optimization technology
- Finally, we will optimiza as usual

The approach was formalized in 2011 (the main reference is more recent)

...And it is designed to enable optimization over complex real-world systems

Our Simulator

For our use case we will use a SIR model as a simulator

SIR models are a type of compartmental model

- The population is divided into three groups (compartments)
- ...I.e. Susceptibles, Infected, Recovered

The classical SIR model is a dynamic system

- The size of the three groups evolves over time
- According to an Ordinary Differemtial Equation (ODE)

An ODE is a differential equation in the form:

$$\dot{y} = f(y, t)$$

- y is a (vector) variable representing the system state
- $f(y, t)$ defines the gradient of the state

SIR Model

In the case of the SIR model, we have:

$$\dot{S} = -\beta \frac{1}{N} S I$$

$$\dot{I} = \beta \frac{1}{N} S I - \gamma I$$

$$\dot{R} = \gamma I$$

Where:

- S, I, R refer to the size of each component
- N is the population size (i.e. $N = S + I + R$
- ...β is the infection rate and γ the recovery rate
- ...And the ratio $R_0 = \beta/\gamma$ is called basic reproductive number

SIR Model

In the case of the SIR model, we have:

$$\dot{S} = -\beta \frac{1}{N} S I$$

$$\dot{I} = \beta \frac{1}{N} S I - \gamma I$$

$$\dot{R} = \gamma I$$

We have that:

- S decreases proportionally to the product SI
- I grows by the same rate, and decreases proportionally to its size I
- R grows proportionally to I

Individiduals "flow" from S to I, and then to R

Running Our Simulator

A SIR model simulator is available in our `util` module

In order to run it, we need to define test values for all parameters

In [10]:
```
S0, I0, R0 = 0.99, 0.01, 0.0
beta, gamma = 0.1, 1/14
tmax = 365
```

- We consider a normalized population ($N = 1$)
- Initially, 1% of the population is infected
- γ is the inverse of the average recovery time (14 days)
- We simulate for one year

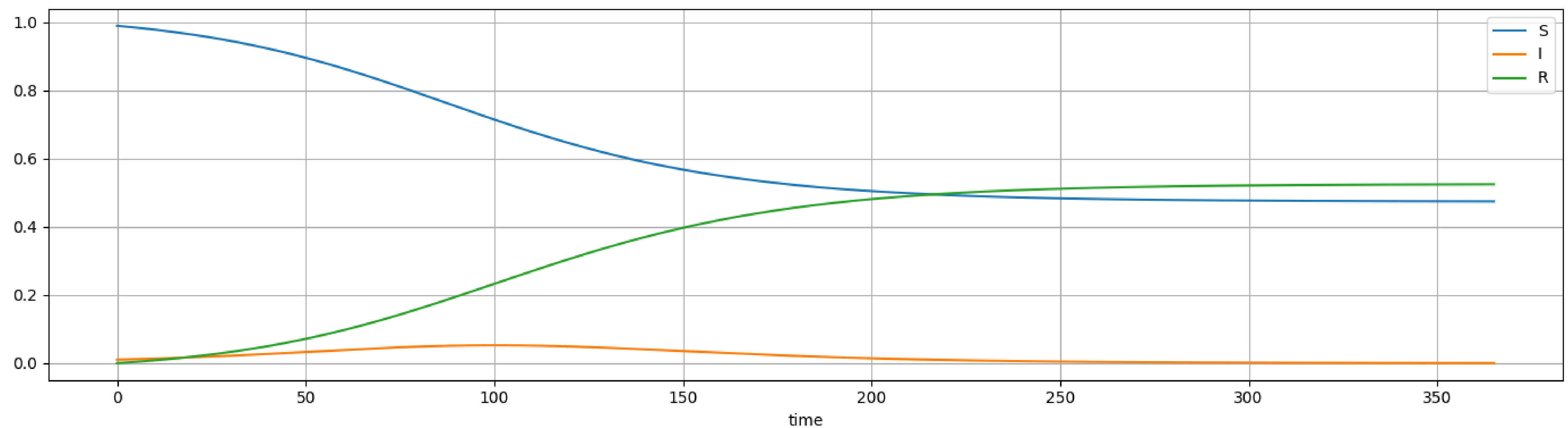
The value of R_0 determines whether we have a proper epidemic behavior

- If $R_0 > 1$ infections grow before falling, otherwise they only decrease
- We have $R_0 = \beta/\gamma = 1.4$, i.e. a true epidemic behavior

Running Our Simulator

Let's plot the dynamics for one year

In [11]:
```
data = util.simulate_SIR(S0, I0, R0, beta, gamma, tmax=tmax, steps_per_day=1)
util.plot_df_cols(data, figsize=figsize)
```



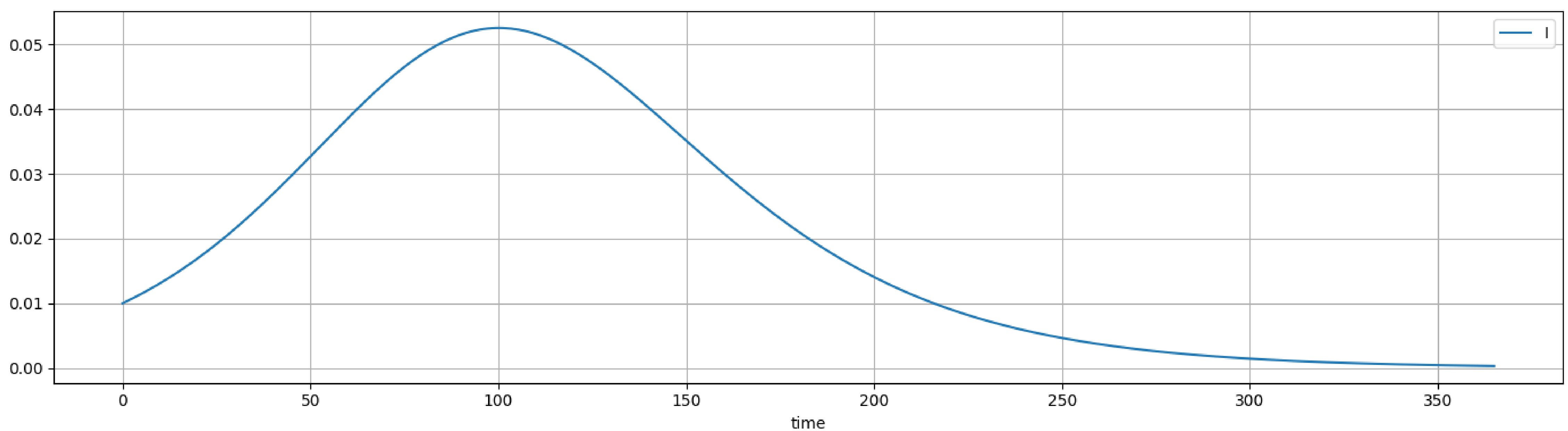
- The S compartment monotonically decreases
- The R compartment monotonically increases

Running Our Simulator

Let's focus on the infected curve

The number of infected grows, before decreasing again

In [12]:
```
util.plot_df_cols(data[['I']], figsize=figsize)
```



Running Our Simulator

Let's focus on the infected curve

The number of infected grows, before decreasing again

In [13]:
```
util.plot_df_cols(data[['I']], figsize=figsize)
```