

2024학년도 2학기

문제해결프로그래밍 강의 12주차

조기필

2024.11.28



지난시간 복습

1. 코로나-19 데이터를 이용한 β 추정하기

문제 3 : COVID-19 수학 모델

단위 시간당 접촉

환자들의 감염 전파기간

$$\mathcal{R}_0 = \alpha \times p \times \frac{1}{\gamma} = \frac{\beta}{\gamma}$$

접촉 당 감염될 확률

질병이 사라지는 조건 $\longleftrightarrow R_0 < 1$



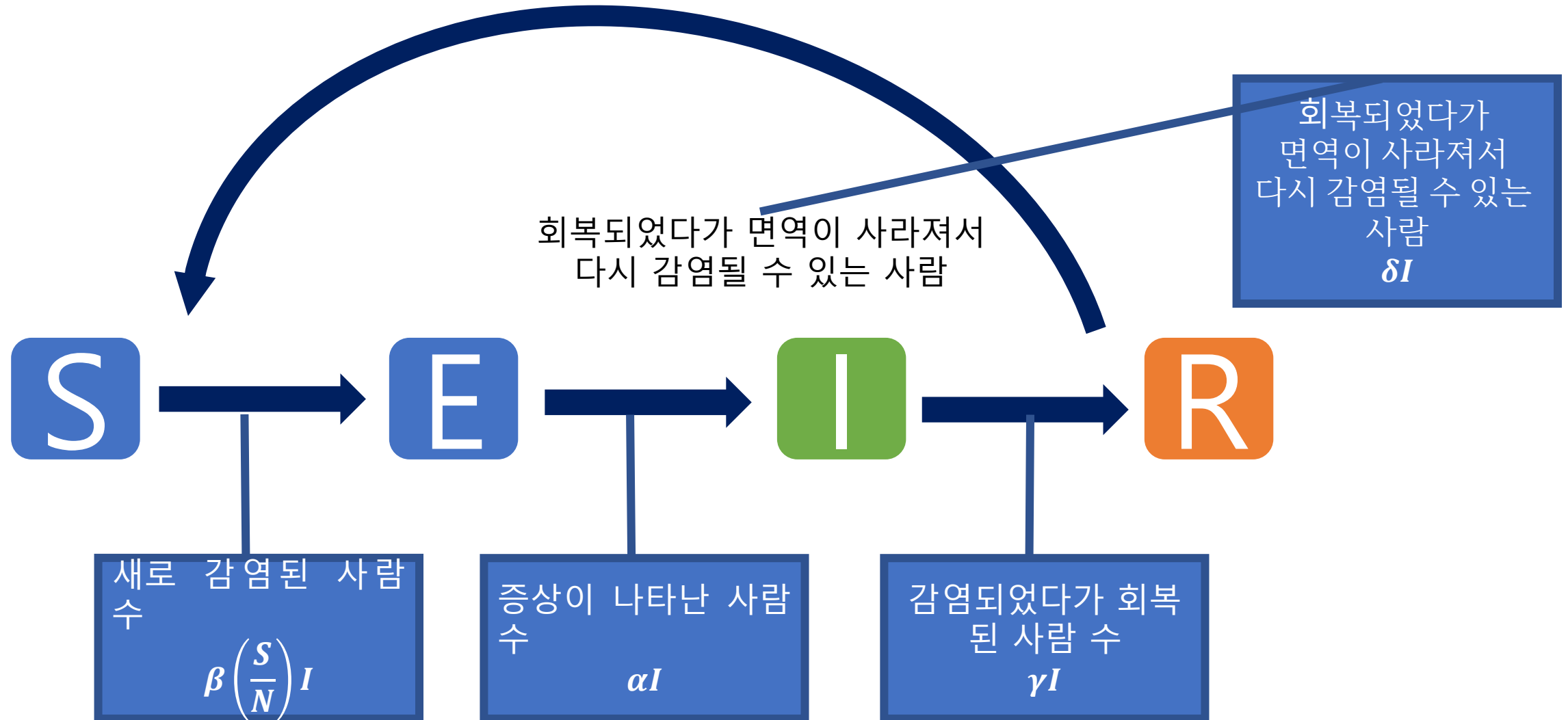
γ 는 회복률은 질병자체의 고유한 회복기간이 필요하므로 인위적으로 낮추기 어려움
그래서, R_0 를 1보다 작게 만들기 위해 현재 β (감염상수)가 얼마인지를 알아야 한다.

문제 3 : COVID-19 수학 모델

COVID-19 감염자 데이터 (2020년 8월)

| 날짜 | 3일 | 4일 | 5일 | 6일 | 7일 | 8일 | 9일 | 10일 | 11일 | 12일 | 13일 | 14일 | 15일 | 16일 |
|--------------------|----|----|----|----|----|----|----|-----|-----|-----|-----|-----|-----|-----|
| 감염된 사람 수 (명) | 3 | 13 | 13 | 23 | 9 | 30 | 30 | 17 | 23 | 35 | 47 | 85 | 154 | 267 |

문제 3 : COVID-19 수학적 모델



문제 3 : COVID-19 수학 모델

SEIR 모델 함수화

```
def f(y_t):  
    S, E, I, R = y_t  
    dS = -beta*S*I/N + delta*R  
    dE = beta*S*I/N - alpha*E  
    dI = alpha*E - gamma*I  
    dR = gamma*I - delta*R  
    output = np.array([dS, dE, dI, dR])  
    return output
```

SEIR 모델 (잠복기가 추가된 모델)

$$\frac{d}{dt}S(t) = -\beta S(t) \frac{I(t)}{N} + \delta R(t)$$

$$\frac{d}{dt}E(t) = \beta S(t) \frac{I(t)}{N} - \alpha E(t)$$

$$\frac{d}{dt}I(t) = \alpha E(t) - \gamma I(t)$$

$$\frac{d}{dt}R(t) = \gamma I(t) - \delta R(t)$$

문제 3 : COVID-19 수학 모델

β 를 실제 신규 감염자수에 맞게 추정하기 위해서

모델의 신규감염자 수(β) \approx 실제 신규 감염자수

```
pred_cases = np.zeros(len(time))
```

→ 모델의 신규 감염자 수를 정의하자.

```
pred_cases[0] = 10
```

→ 모델의 첫 시간의 초기 감염자 수가 10이므로 초기값을 10으로 지정하자.

문제 3 : COVID-19 수학 모델

```
for i in range(n):  
    y[i+1,:] = rk4(f, y[i,:], h)  
    pred_cases[i+1] = alpha*y[i,1]
```

모델의 신규 감염자 수는 $\alpha E(t)$ 로 나타낼 수 있다.

SEIR 모델 (잠복기가 추가된 모델)

$$\frac{d}{dt}S(t) = -\beta S(t) \frac{I(t)}{N} + \delta R(t)$$

$$\frac{d}{dt}E(t) = \beta S(t) \frac{I(t)}{N} - \alpha E(t)$$

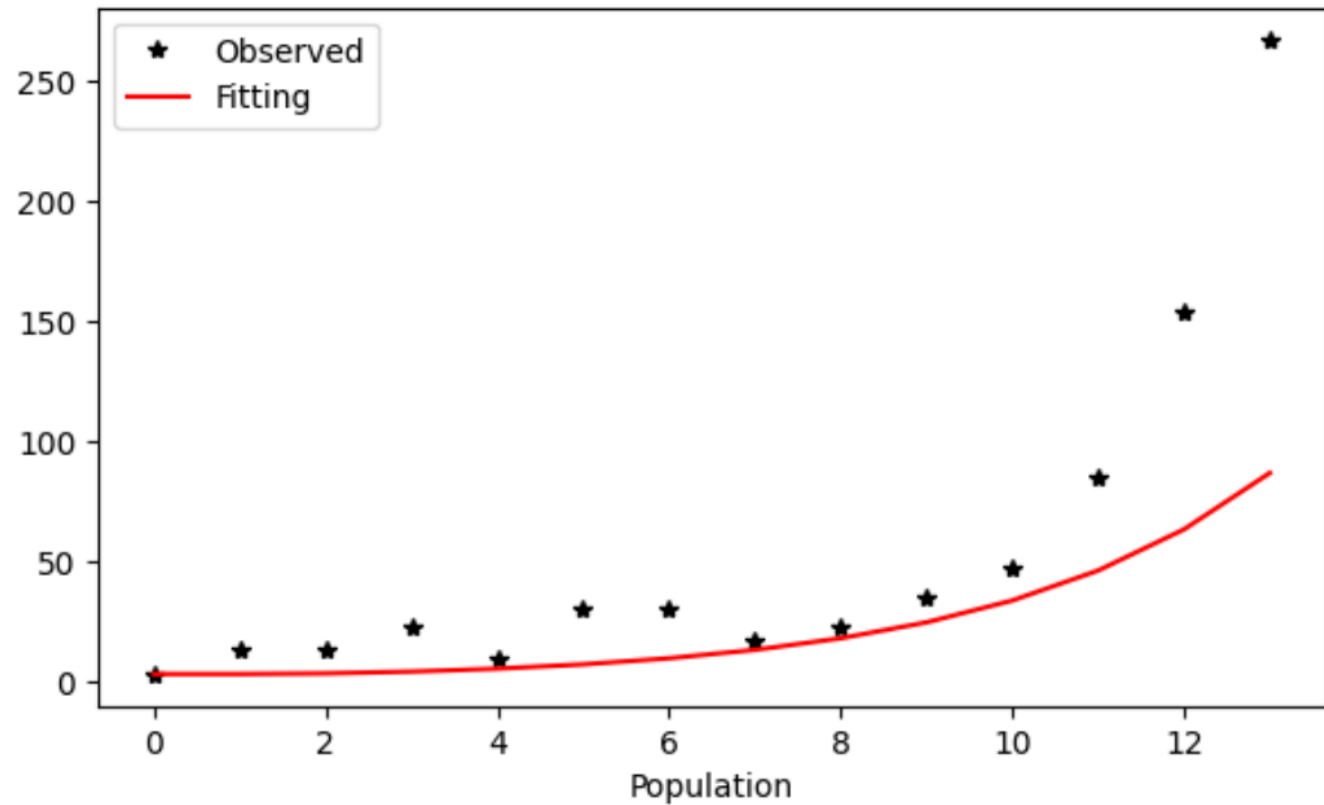
$$\frac{d}{dt}I(t) = \alpha E(t) - \gamma I(t)$$

$$\frac{d}{dt}R(t) = \gamma I(t) - \delta R(t)$$

문제 3 : COVID-19 수학 모델

모델의 신규감염자 수와 실제 신규 감염자수를 비교해보자.

```
plt.figure(figsize=(7,4))  
plt.plot(time,data['Cases'],'*k')  
plt.plot(time,pred_cases,'r')  
plt.xlabel('time')  
plt.ylabel('Population')  
plt.legend(('Observed','Fitting'),loc='best')  
plt.show()
```



문제 3 : COVID-19 수학 모델

그럼 어떻게 데이터에 가장 알맞은 β 를 추정할 수 있을까?



데이터의 신규 감염자수와 모델의 감염자 수의 차이를 최소화



최소값을 찾는 문제 (최적화 문제)

문제 3 : COVID-19 수학 모델

최적화 문제의 수학적 표현

$$\min f(X)$$

$$\text{subject to } g_i(x) \leq 0, \quad i = 1, 2, \dots, m$$

$$h_j(x) = 0, \quad j = 1, 2, \dots, p$$

$$x_l \leq x \leq x_u$$

$$\text{where } X = [x_1, x_2, x_3, \dots, x_n]^T$$

문제 3 : COVID-19 수학 모델

파이썬의 scipy.optimize 패키지

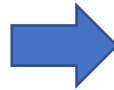
| Types of optimization problem | | Function | Method |
|-------------------------------|-----------------------------|------------------------------------------------------|------------------------------------------------------------------------------|
| Local Opt. (국소) | unconstrained (비제약) | minimize | Nelder-Mead BFGS Newton-CG trust-ncg trust-krylov trust-exact |
| | constrained (제약) | | trust-constr <u>SLSQP</u> |
| Global Opt. (전역) | derivative-based (미분 적용) | <u>basinhopping</u> shgo | - |
| | metahuristic | brute(?) differential_evolution dual_annealing | - |

문제 3 : COVID-19 수학 모델

Minimize 함수 사용 방법

핵심

```
scipy.optimize.minimize(  
    fun, → 목적함수  
    x0, → 초깃값  
    args=( ), → 초깃값 외에 목적함수에 전달할 매개변수  
    method=None, → 최적화 해 찾기 종류  
    jac=None, hess=None, hessp=None,  
    bounds=None, → 경계값  
    보조 constraints=( ), → 제약조건  
    tol=None, callback=None, options=None )
```



```
sol = minimize(obj_fun, x0, method='SLSQP')
```

문제 3 : COVID-19 수학 모델

```
sol
```

```
message: Optimization terminated successfully  
success: True  
status: 0  
  fun: -4.056172844149885  
    x: [-1.071e-01]  
  nit: 6  
  jac: [ 9.382e-04]  
nfev: 14  
njev: 6
```

x: 최적화 해

success: 최적화에 성공하면 True 반환

status: 종료 상태. 최적화에 성공하면 0 반환

message: 메시지 문자열

fun: x 위치에서의 함수의 값

jac: x 위치에서의 자코비안(그레디언트) 벡터의 값

hess_inv: x 위치에서의 헤시안 행렬의 역행렬의 값

nfev: 목적함수 호출 횟수

njev: 자코비안 계산 횟수

nhev: 헤시안 계산 횟수

nit: x 이동 횟수

문제 3 : COVID-19 수학 모델

```
bound = (0,3)
sol_2 = minimize(obj_fun, x0, method='SLSQP', bounds=(bound,))
```

범위가 하나지만 심표를 붙이고
괄호를 닫는다.

문제 3 : COVID-19 수학 모델

항상 우리가 원하는 최소화 점을 찾지 못할 수 있다.

$x_0=1.5$ 일때,

```
x0 = 1.5
```

```
sol = minimize(obj_fun, x0, method='SLSQP')
```

를 실행시켜보자. 결과가 같은가?



NO!!

문제 3 : COVID-19 수학 모델

minimize와 basinhopping의 결과를 비교해보자.

```
from scipy.optimize import basinhopping
```

```
sol_2 = basinhopping(obj_fun, x0)
```

```
sol_2
```

```
message: ['requested number of basinhopping iterations completed successfully']
success: True
      fun: -4.056172885244464
       x: [-1.072e-01]
      nit: 100
  minimization_failures: 0
      nfev: 1336
      njev: 668
lowest_optimization_result: message: Optimization terminated successfully.
                           success: True
                           status: 0
                           fun: -4.056172885244464
                           x: [-1.072e-01]
                           nit: 5
                           jac: [ 0.000e+00]
                           hess_inv: [[ 9.339e-02]]
                           nfev: 14
                           njev: 7
```

문제 3 : COVID-19 수학 모델

데이터의 신규 감염자수와 모델의 감염자 수의 차이를 최소화



데이터의 신규 감염자수와 모델의 감염자 수의 MSE를 최소화



MSE를 함수화하여 Minimize 사용

문제 3 : COVID-19 수학 모델

가장 알맞은 β 를 추정해야 함으로 RMSE를 β 에 대한 함수로 나타내자.

1. 모델을 β 에 대한 함수로 변경하자.

```
def f(y_t):  
    S, E, I, R = y_t  
    dS = -beta*S*I/N + delta*R  
    dE = beta*S*I/N - alpha*E  
    dI = alpha*E - gamma*I  
    dR = gamma*I - delta*R  
    output = np.array([dS, dE, dI, dR])  
    return output
```



```
def f_beta(y_t, x):  
    S, E, I, R = y_t  
    dS = (-x*S*I/N + delta*R).item()  
    dE = (x*S*I/N - alpha*E).item()  
    dI = (alpha*E - gamma*I).item()  
    dR = (gamma*I - delta*R).item()  
    return np.array([dS, dE, dI, dR])
```

문제 3 : COVID-19 수학 모델

2. rk4를 β 에 대한 함수로 변경하자.

```
def rk4(f, y_t, h):  
    k1 = f(y_t)  
    k2 = f(y_t+k1*h/2)  
    k3 = f(y_t+k2*h/2)  
    k4 = f(y_t+k3*h)  
    y_t1 = y_t+h*(k1+2*k2+2*k3+k4)/6  
    return y_t1
```



```
def rk4_beta(f, y_t, h, x):  
    k1 = f_beta(y_t, x)  
    k2 = f_beta(y_t+k1*h/2, x)  
    k3 = f_beta(y_t+k2*h/2, x)  
    k4 = f_beta(y_t+k3*h, x)  
    y_t1 = y_t+h*(k1+2*k2+2*k3+k4)/6  
    return y_t1
```

3. MSE 함수 가져오자.

```
from sklearn.metrics import mean_squared_error
```

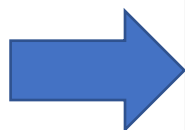
문제 3 : COVID-19 수학 모델

4. 실제 데이터와 모델의 예측 값을 β 에 대한 RMSE 함수를 정의하자.

```
pred_cases = np.zeros(len(time))
```

```
pred_cases[0] = 10
```

```
for i in range(n):  
    y[i+1,:] = rk4(f, y[i,:], h)  
    pred_cases[i+1] = alpha*y[i,1]
```



```
def MSE_beta(x):  
    pred_cases = np.zeros(len(time))  
    pred_cases[0]=10  
    for i in range(n):  
        y[i+1,:] = rk4_beta(f_beta, y[i,:], h, x)  
        pred_cases[i+1] = alpha*y[i,1]  
    result = mean_squared_error(data['Cases'],pred_cases)  
    return result
```

문제 3 : COVID-19 수학 모델

5. RMSE 함수를 목적함수로 만들자.

```
def Obj_beta(x):  
    return MSE_beta(x)
```

6. 목적함수를 Minimize에 적용하자.

```
bound=(0,10)
```

```
param = minimize(Obj_beta, 1.0, method='SLSQP', bounds=(bound,))
```

```
param
```

```
message: Optimization terminated successfully  
success: True  
status: 0  
  fun: 251.92722631402904   최소가 되는 MSE : 251.927  
     x: [ 1.392e+00]  
    nit: 9               최적의  $\beta = 1.392$   
   jac: [ 6.561e-04]  
  nfev: 24  
  njev: 9
```

오늘의 학습내용

1. 사회적 거리두기 단계에 따른 감염 양상은?
2. 사회적 거리두기 몇 단계가 효과적일까?

문제 3 : COVID-19 수학 모델

최종 모델

감염될 수 있는 사람

S



E



감염된 사람

I



H



회복된 사람

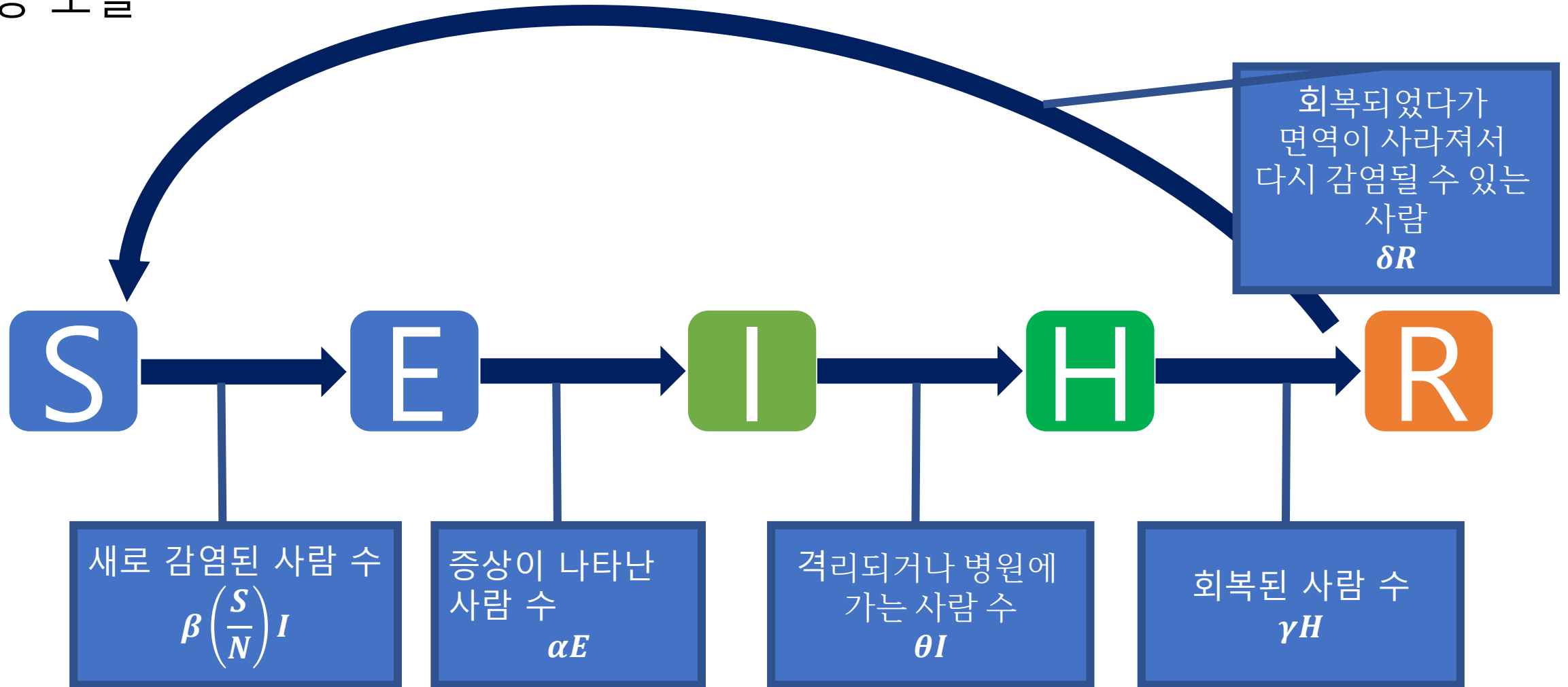
R

잠복기인 사람

격리 또는 병원에 입원한 사람

문제 3 : COVID-19 수학 모델

최종 모델



문제 3 : COVID-19 수학 모델

SEIHR 모델 (병원 및 격리가 추가된 모델)

$$\frac{d}{dt}S(t) = -\beta S(t) \frac{I(t)}{N} + \delta R(t)$$

$$\frac{d}{dt}E(t) = \beta S(t) \frac{I(t)}{N} - \alpha E(t)$$

$$\frac{d}{dt}I(t) = \alpha E(t) - \theta I(t) \longleftarrow \text{신규 감염자}$$

$$\frac{d}{dt}H(t) = \theta I(t) - \gamma H(t)$$

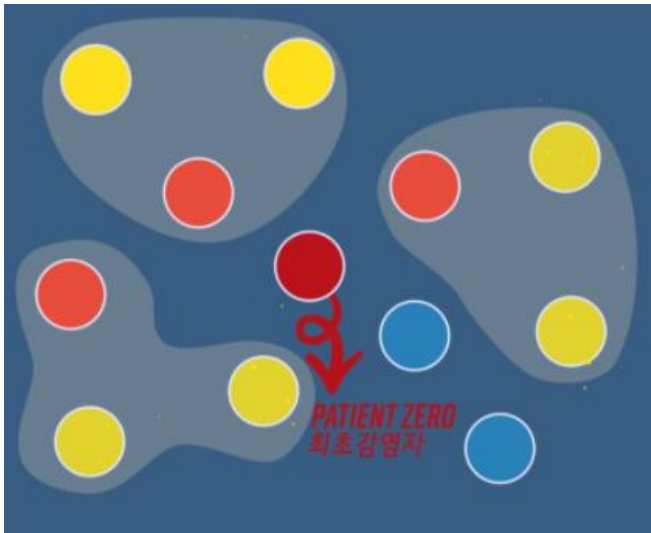
$$\frac{d}{dt}R(t) = \gamma H(t) - \delta R(t)$$

문제 3 : COVID-19 수학 모델

Q. 이미 감염병 발생한 이후라면?

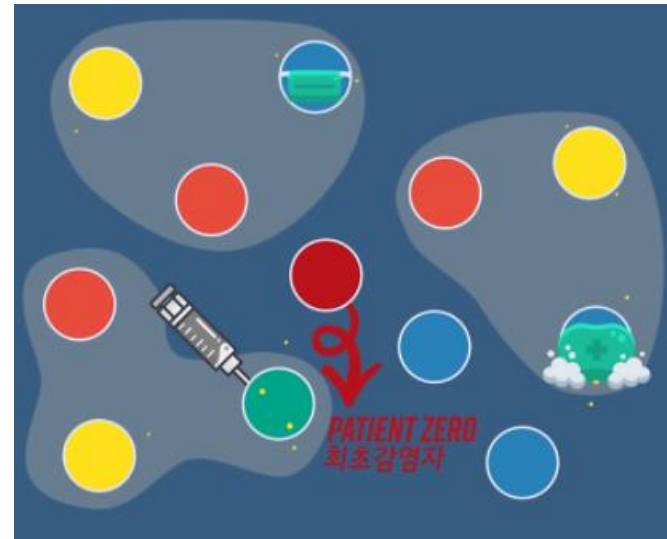
R_t : Effective reproduction number

- 정의: t일의 새롭게 감염되는 환자 평균 수
- 의의: R_t 변화에 따라 방역의 효과 분석 가능



$$R_0 = 3$$

$$R_t = 2 \text{ (6/3)}$$



$$R_0 = 3$$

$$R_t = 1 \text{ (3/3)}$$

문제 3 : COVID-19 수학 모델

$R_t < 1$ 이도록 하기 위해서는

p , 접촉 당 감염될 확률- 감염종류에 따라 다름

- p 를 줄이기 위한 조치: 장갑사용, 수혈 혈액 검사 등
 - * 코로나 19: 마스크, 고글, 장갑 사용, 기침예절, 사람간 2 미터 이상 간격두기

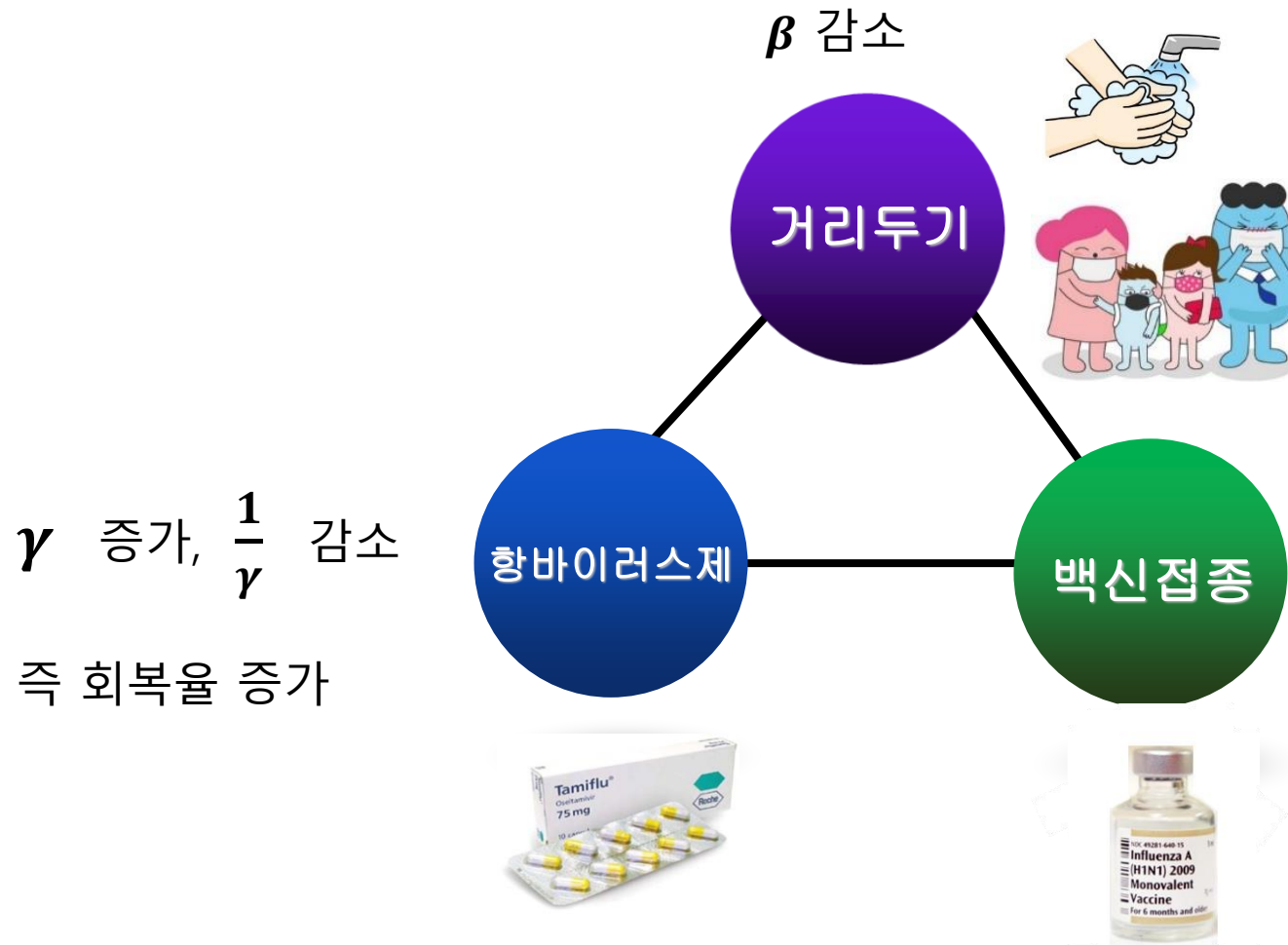
α , 단위시간당 접촉- 감염종류에 따라 다름

- 같은 방, 재채기 거리, 피부접촉
- α 를 줄이기 위한 조치: 격리 (Isolation), 사회적 거리두기 (코로나 19)

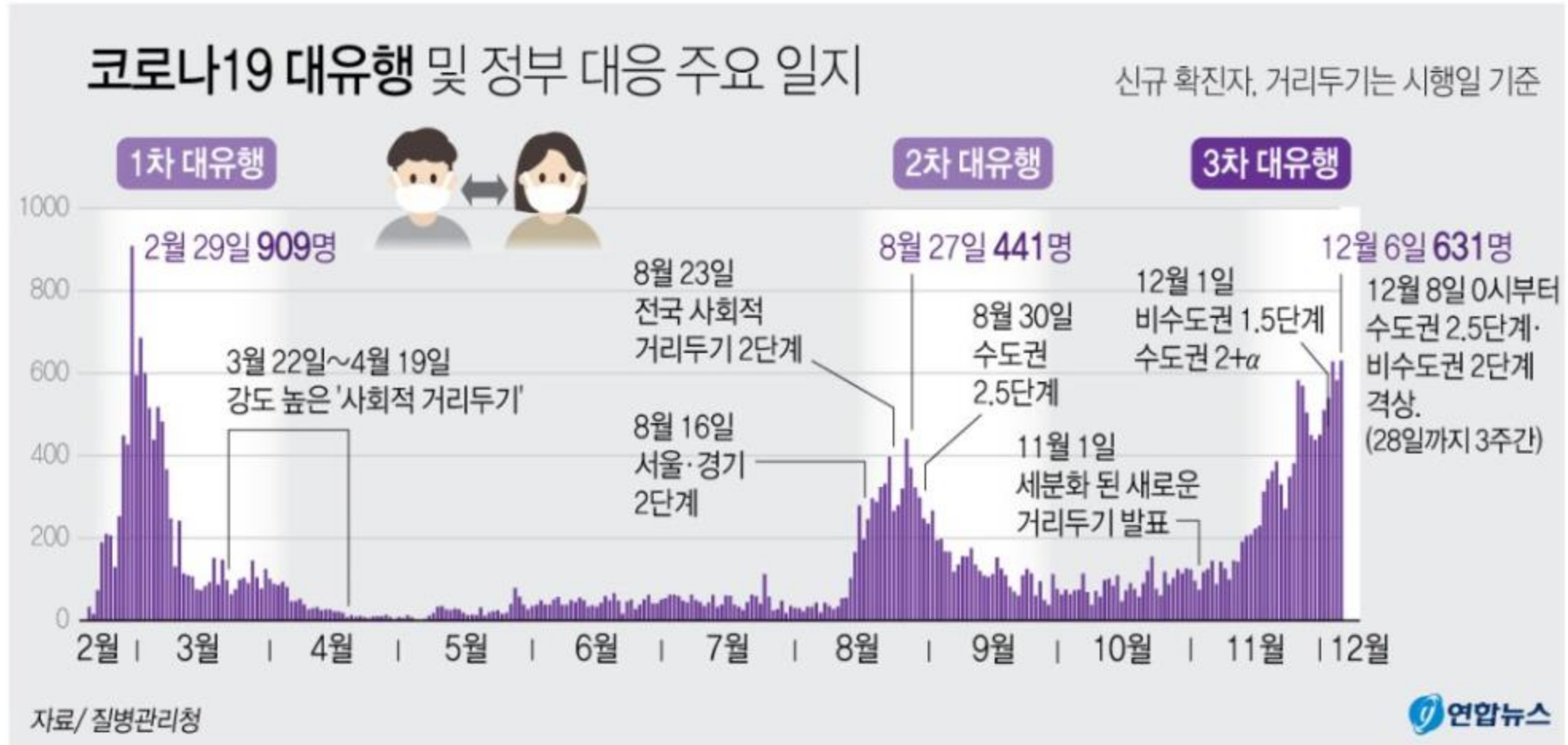
$\frac{1}{\gamma}$, 환자들의 감염 전파기간

- 의료적 조치로 감소
 - * 코로나 19 : 접촉자 추적, 적극적 검사, 격리 (Tracing, Testing, & Isolation)

문제 3 : COVID-19 수학 모델



문제 3 : COVID-19 수학 모델



문제 3 : COVID-19 수학 모델

사회적 거리두기 단계 (SD)에 따라 우리의 모델은 어떤 것이 변화하나? β

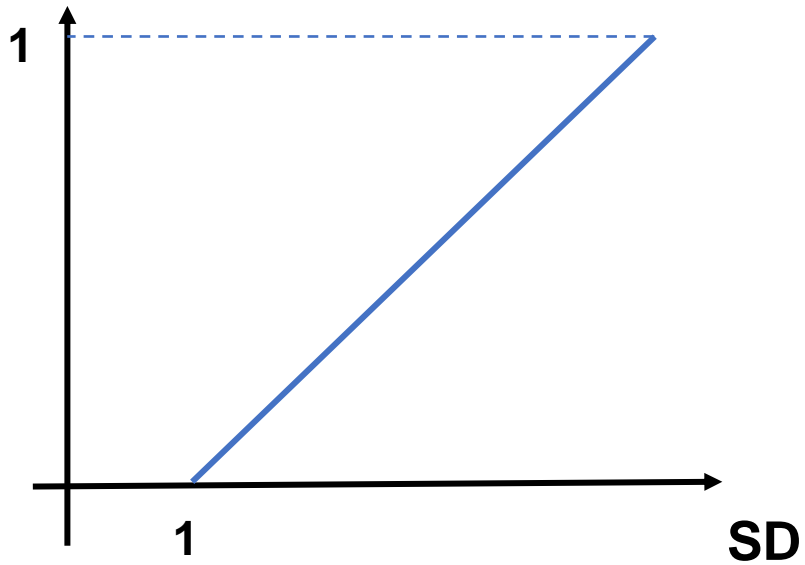
β 와 SD는 어떤 관계가 있는가? SD \uparrow β \downarrow

β 감소율과 SD를 함수 형태로 나타낼 수 있는가?

문제 3 : COVID-19 수학 모델

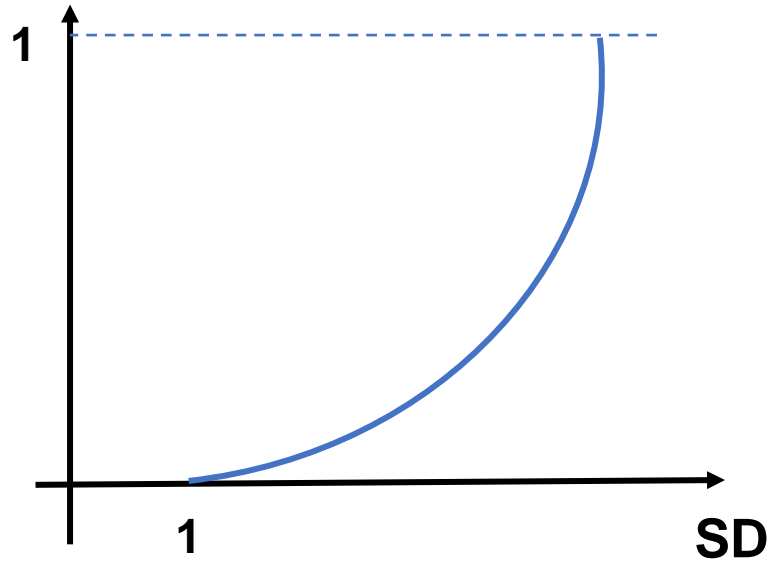
A

β 감소율



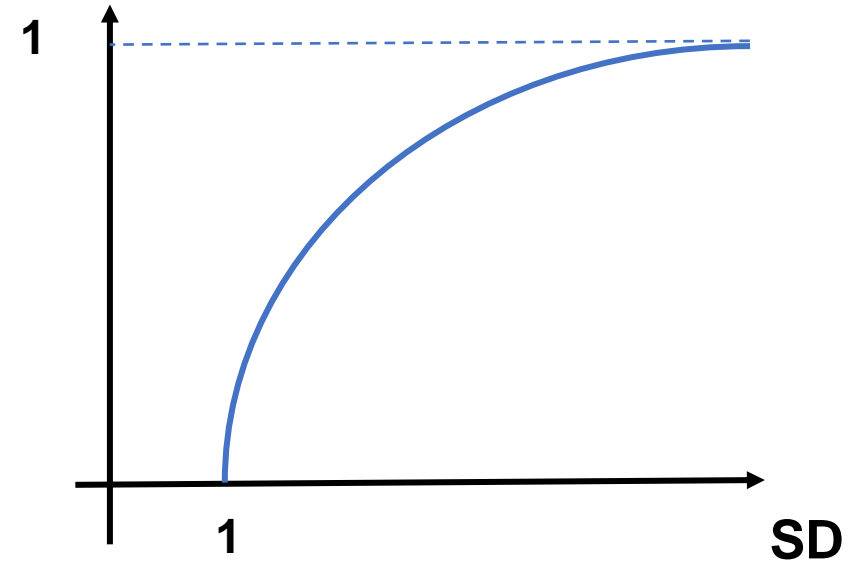
B

β 감소율



C

β 감소율

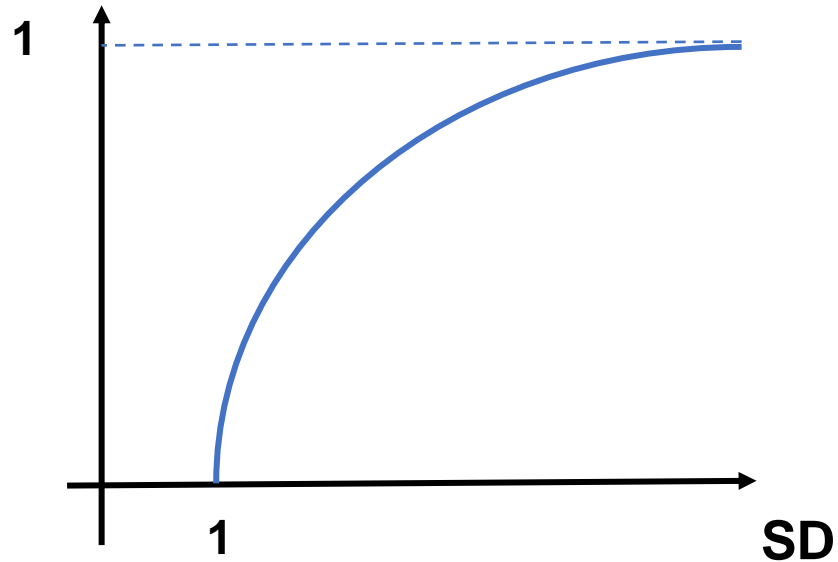


A, B, C 중 가장 이상적인 그림은 어떤 그림인가?

문제 3 : COVID-19 수학 모델

C

β 감소율



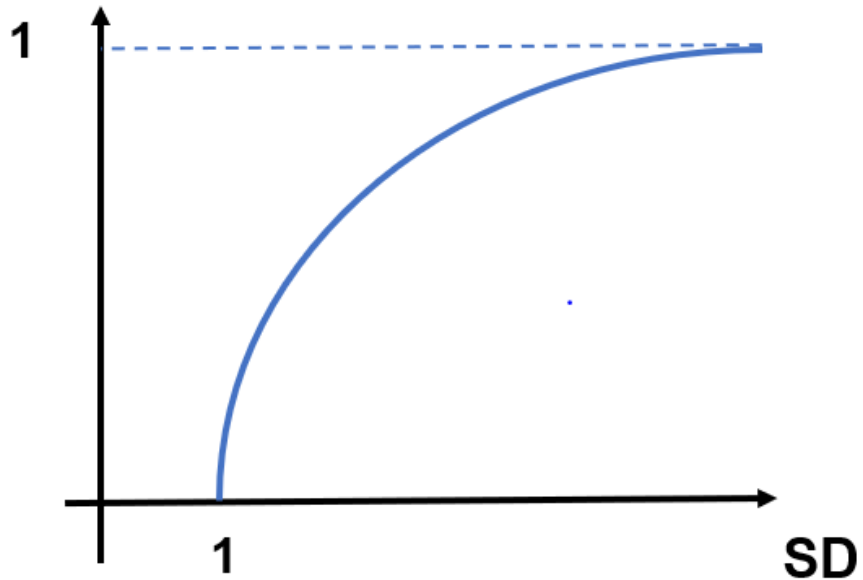
나의 선택은? C

선택의 이유는?

1. 처음에는 작은 거리두기만으로 효과가 클 것이다.
2. 거리두기를 단계가 강화 될수록 쉽고 효과적인 규제가 감소하여 단계 증가의 효과가 갈수록 감소할 것이다.

문제 3 : COVID-19 수학 모델

β 감소율



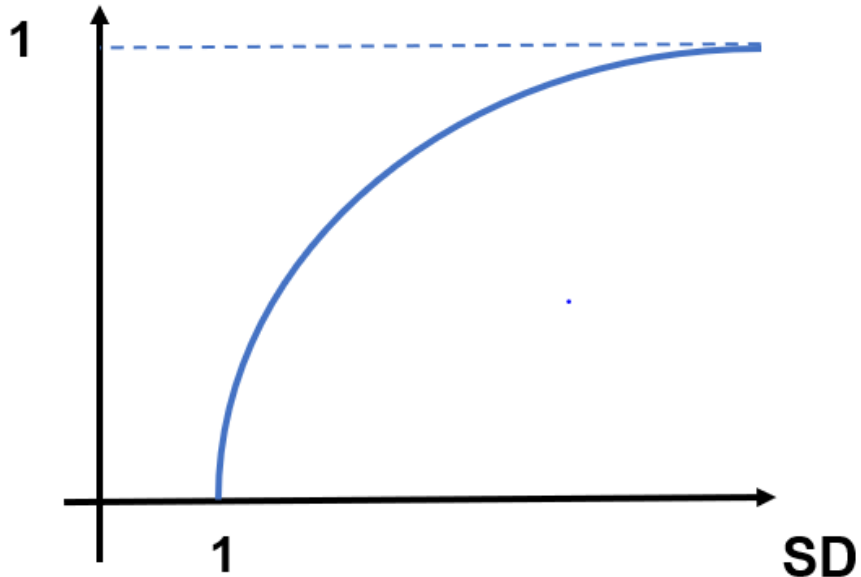
그래프를 선택을 했다면 함수화를 시켜보자.

이 그래프 계형에 어울리는 함수는 어떤 것이 있는가?

- 제곱근(루트) 함수
- 로그 함수
- Sin 함수

문제 3 : COVID-19 수학 모델

β 감소율



Sin 함수로 생각해 보자.

x축은 SD, y축은 β 감소율 이다.

그리고 기본 SD 1단계라고 할 때, 이 제공근 함수는 점 $(SD, \beta \text{ 감소율}) = (1, 0)$ 를 지난다.

그러므로

$$\beta \text{ 감소율} = \sin A(SD - 1)$$

문제 3 : COVID-19 수학 모델

$$\beta \text{ 감소율} = \sin A(\text{SD} - 1)$$

A 값은 어떻게 구할 수 있을까?

우리는 쉽게 사회적 거리두기가 **완전 봉쇄 단계**라고 할 때, β 감소율 = 1
이라는 것을 알 수 있다.

우리 정부는 2020년 8월에 사회적 거리두기를 1~3단계까지 정의했으며,
실제로 시행한 가장 높은 단계가 2.5단계였다.

따라서 우리는 임의로 **3단계가 완전 봉쇄 단계라고 가정**해보자.

문제 3 : COVID-19 수학 모델

$$\beta \text{ 감소율} = \sin A(\text{SD} - 1)$$



$$1 = \sin A(3 - 1)$$

$\sin(x)$ 가 1이 되려면, $x = \frac{\pi}{2}$



$$A(3 - 1) = 2A = \frac{\pi}{2}$$



$$A = \frac{\pi}{4}$$

문제 3 : COVID-19 수학 모델

최종적으로 β 감소율과 SD의 함수는 다음과 같다.

$$\beta \text{ 감소율} = \sin \frac{\pi}{4} (\text{SD} - 1)$$

문제 3 : COVID-19 수학 모델



2주간(8.16~) 서울·경기지역 사회적 거리 두기 2단계 격상

서울시와 경기도의 주민들께서는
앞으로 2주간은 모임이나 외출을 삼가시고,
꼭 필요한 외출 외에는 집에 머물러 주실 것을 부탁드립니다.

01. 집합·모임·행사* 자제 권고 *실내 50인, 실외 100인 이상
02. 클럽 등 일부 고위험시설 추가 방역 수칙 의무화
03. 위험도가 높은 다중이용시설* 핵심 방역수칙 준수 의무화
*학원, 결혼식장, 장례식장, 영화관, 목욕탕 등
04. 실내 국공립시설 이용인원 제한
05. 스포츠 행사 무관중 경기 전환
06. 학교 원격수업 전환 권고 집단발생 지속 발생한 시·군·구
07. 기관·기업 유연·재택근무 등을 통해 근무인원 제한 권고

보건복지부 질병관리본부

수도권 등 사회적 거리 두기 1.5단계 격상 (11.19.(목) 0시부터, 2주간)

수능시험(12.3.) 대비 수능특별방역기간 설정 및 방역관리(11.19.~)

- #1 수도권·강원도 일부 지역, 2주간 식사동반 모임 취소 권고
- 재택근무·점심시간 시차 운영, 시차출퇴근제 권고

- #2 1.5단계 격상 조치에 따른 강화된 방역 조치

- 일반관리시설 이용 인원 4㎡당 1명 제한, 좌석 띄우기 실시
- 중점관리시설 중 유흥시설 춤추기·좌석 간 이동 금지,
노래연습장·공연장 음식섭취 금지
- 국·공립 시설 이용 인원 50% 제한, 스포츠 관람 30% 관중 입장 제한
- 사회복지시설 운영 유지 및 긴급돌봄 등 제공
- 위험도 높은 집합·모임(집회·시위, 대규모 콘서트, 학술행사, 축제 등)
100인 미만 인원 제한
- 종교활동 좌석 30% 이내 인원 제한, 소모임·식사 등 금지

보건복지부

질병관리청

정부에서 일반적으로 사회적 거리두기 전략을 2주 간격으로 정하고 있다.

문제 3 : COVID-19 수학 모델

그럼 SD가 각각

- 1단계
- 1.5단계
- 2단계
- 2.5단계
- 3단계

일 때, 2주간 신규 감염자가 어떻게 될 지 예측해보자.

문제 3 : COVID-19 수학 모델

SD에 따른 신규 감염자 예측 시뮬레이션

문제 3 : COVID-19 수학 모델

새로운 COVID-19 감염자 데이터 (2020년 8월)

정부의 사회적거리두기 적용 시기



| date | Cases |
|------------|-------|
| 2020-08-03 | 3 |
| 2020-08-04 | 13 |
| 2020-08-05 | 13 |
| 2020-08-06 | 23 |
| 2020-08-07 | 9 |
| 2020-08-08 | 30 |
| 2020-08-09 | 30 |
| 2020-08-10 | 17 |
| 2020-08-11 | 23 |
| 2020-08-12 | 35 |
| 2020-08-13 | 47 |
| 2020-08-14 | 85 |
| 2020-08-15 | 154 |
| 2020-08-16 | 267 |
| 2020-08-17 | 188 |
| 2020-08-18 | 235 |
| 2020-08-19 | 283 |
| 2020-08-20 | 276 |
| 2020-08-21 | 315 |
| 2020-08-22 | 315 |
| 2020-08-23 | 386 |
| 2020-08-24 | 258 |
| 2020-08-25 | 264 |
| 2020-08-26 | 307 |
| 2020-08-27 | 434 |
| 2020-08-28 | 359 |
| 2020-08-29 | 308 |
| 2020-08-30 | 283 |

거리두기 적용전 (13일)

거리두기 적용후 (14일)

문제 3 : COVID-19 수학 모델

변경된 모델에 대해서 다시 8월 16일까지 데이터를 활용해서 β 를 추정하자.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
data = pd.read_excel('drive/MyDrive/Colab Notebooks/Problem_solver/COVID19_data_2.xlsx')
```

문제 3 : COVID-19 수학 모델

총시간은 거리두기 적용전 시간(13일)과 적용후 시간(14일)을 합쳐 27일이다.

```
t0 = 0
t1 = 13 ← 거리두기 적용전 시간
tf = 27 ← 총시간
n1 = 13 ← 거리두기 적용전 간격수
n = 27 ← 총 간격수
h = (tf-t0)/n
```

```
time = np.linspace(t0,tf,n+1)
```

```
N = 51840000
beta = 1.0
gamma = 1/14
delta = 1/229
alpha = 1/5
theta = 1/4 ← 증상이 나타난 이후 격리 혹은 병원에 가는데 걸리는 비율 (시간을 평균 4일 가정)
```

문제 3 : COVID-19 수학 모델

초기값 설정

```
H0 = data.loc[0, 'Cases']  
y = np.zeros((n+1,5))  
initial_value = np.array([N-H0*(1+1/theta-1/(theta*alpha)), H0/theta/alpha, H0/theta, H0, 0.0])  
y[0,:] = initial_value
```


문제 3 : COVID-19 수학 모델

SEIHR 모델 함수화

```
def f_beta(y_t, x):  
    S, E, I, H, R = y_t  
    dS = (-x*S*I/N + delta*R).item()  
    dE = (x*S*I/N - alpha*E).item()  
    dI = (alpha*E - theta*I).item()  
    dH = (theta*I - gamma*H).item()  
    dR = (gamma*H - delta*R).item()  
    return np.array([dS, dE, dI, dH, dR])
```

$$\frac{d}{dt}S(t) = -\beta S(t)\frac{I(t)}{N} + \delta R(t)$$

$$\frac{d}{dt}E(t) = \beta S(t)\frac{I(t)}{N} - \alpha E(t)$$


$$\frac{d}{dt}I(t) = \alpha E(t) - \theta I(t)$$

$$\frac{d}{dt}H(t) = \theta I(t) - \gamma H(t)$$

$$\frac{d}{dt}R(t) = \gamma H(t) - \delta R(t)$$

문제 3 : COVID-19 수학 모델

Runge-Kutta method 함수화

```
def rk4_beta(f, y_t, h, x):  
    k1 = f_beta(y_t, x)  
    k2 = f_beta(y_t+k1*h/2, x)  
    k3 = f_beta(y_t+k2*h/2, x)  
    k4 = f_beta(y_t+k3*h, x)  
    y_t1 = y_t+h*(k1+2*k2+2*k3+k4)/6  
    return y_t1
```

문제 3 : COVID-19 수학 모델

8월 16일까지의 실제 신규 감염자수 데이터와 모델의 신규감염자수에 대한 MSE 함수화

```
from scipy.optimize import minimize
from sklearn.metrics import mean_squared_error
```

```
def RMSE_beta(x):
    pred_cases = np.zeros(len(time))
    for i in range(n1):
        y[i+1,:] = rk4_beta(f_beta, y[i,:], h, x)
        pred_cases[i+1] = theta*y[i+1,2]
    result = mean_squared_error(data['Cases'][:n1+1], pred_cases[:n1+1])
    return result
```

```
def Obj_beta(x):
    return RMSE_beta(x)
```


문제 3 : COVID-19 수학 모델

```
bound=(0,10)
param = minimize(Obj_beta, 1.0, method='SLSQP', bounds=(bound,))
```

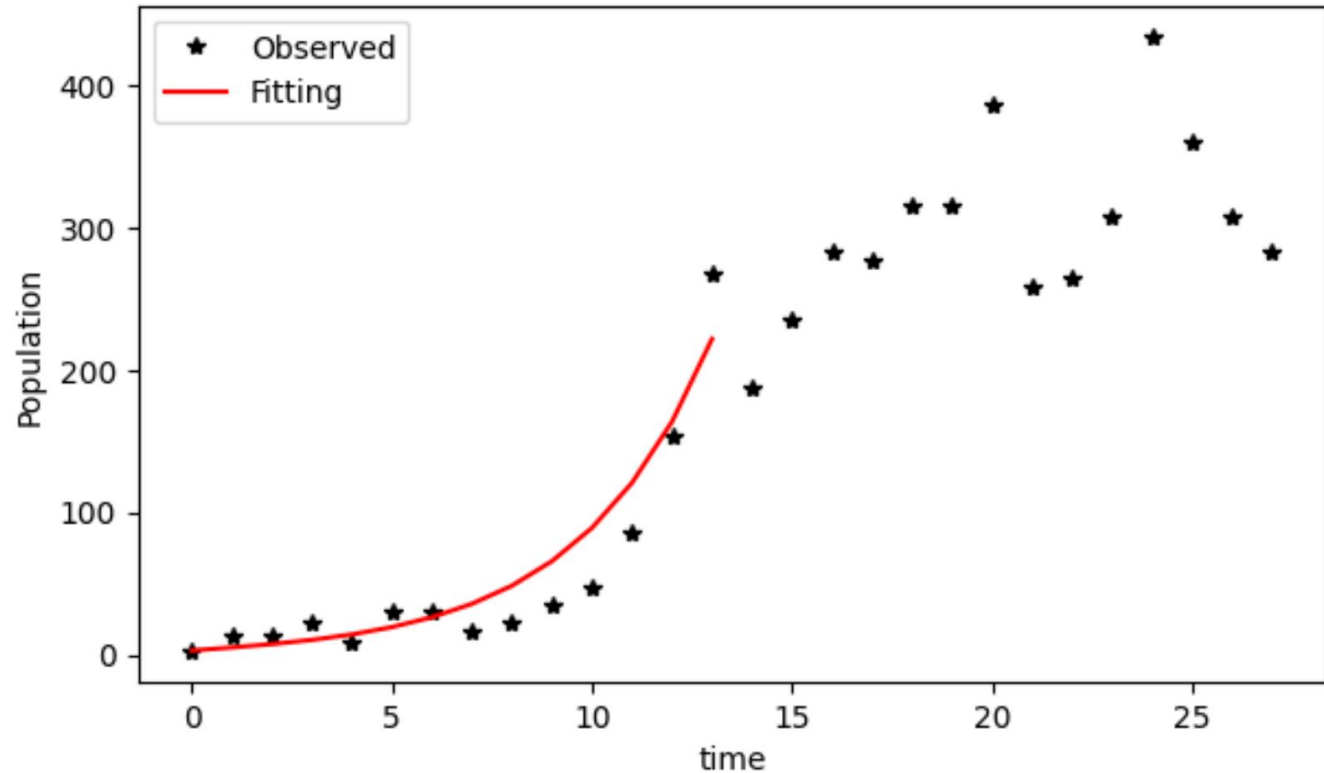
param

```
message: Optimization terminated successfully
success: True
status: 0
  fun: 539.2103316054863
   x: [ 1.396e+00]
  nit: 6
 jac: [ 2.352e-01]
nfev: 16
njev: 6
```

```
pred_cases = np.zeros(len(time))
pred_cases[0]=H0
for i in range(n1):
    y[i+1,:] = rk4_beta(f_beta, y[i,:], h, param.x)
    pred_cases[i+1] = theta*y[i+1,2]
```

문제 3 : COVID-19 수학 모델

```
plt.figure(figsize=(7,4))
plt.plot(time,data['Cases'],'*k')
plt.plot(time[:n1+1],pred_cases[:n1+1],'r')
plt.xlabel('time')
plt.ylabel('Population')
plt.legend(('Observed','Fitting'),loc='best')
plt.show()
```



문제 3 : COVID-19 수학 모델

2주간 예측을 위해 새로운 시간과 추정한 β , 8월 16일의 초기값을 정의한다.

```
beta = param.x
```

```
import math
```

← Sin, π 등 수학 함수를 사용하기 위한 라이브러리

```
def beta_SD(SD):  
    return math.sin(math.pi * (SD.item() - 1) / 4)
```

← β 감소율 함수를 정의하자.

$$\beta \text{ 감소율} = \sin \frac{\pi}{4} (\text{SD} - 1)$$

문제 3 : COVID-19 수학 모델

β 감소율이 반영된 미분방정식 함수를 새로 정의하자.

사회적 거리두기(SD)가 적용되면 기존의 β 가 앞서 정의한

β 감소율 함수가 적용되도록 변형하자.

```
def f_SD(y_t, SD):  
    S, E, I, H, R = y_t  
    dS = (-beta*(1-beta_SD(SD))*S*I/N + delta*R).item()  
    dE = (beta*(1-beta_SD(SD))*S*I/N - alpha*E).item()  
    dI = (alpha*E - theta*I).item()  
    dH = (theta*I - gamma*H).item()  
    dR = (gamma*H - delta*R).item()  
    return np.array([dS, dE, dI, dH, dR])
```

문제 3 : COVID-19 수학 모델

Runge-Kutta method 함수화

```
def rk4_SD(f_SD, y_t, h, SD):  
    k1 = f_SD(y_t, SD)  
    k2 = f_SD(y_t+k1*h/2, SD)  
    k3 = f_SD(y_t+k2*h/2, SD)  
    k4 = f_SD(y_t+k3*h, SD)  
    y_t1 = y_t+h*(k1+2*k2+2*k3+k4)/6  
    return y_t1
```

문제 3 : COVID-19 수학 모델

결과를 result_SD에 데이터프레임 형식으로 저장해보자.

```
result_SD=pd.DataFrame()
```

```
for SD in np.linspace(1.0,3.0,5):  
    result_SD.loc[0,'SD_'+str(SD)] = pred_cases[n1]  
    for num,i in enumerate(range(n1,n)):  
        y[i+1,:]= rk4_SD(f_SD, y[i,:], h, SD) SD적용  
        result_SD.loc[num+1,'SD_'+str(SD)] = theta*y[i+1,2]
```

신규 감염자를 result에 입력

SD를 1부터 0.5간격으로 3까지

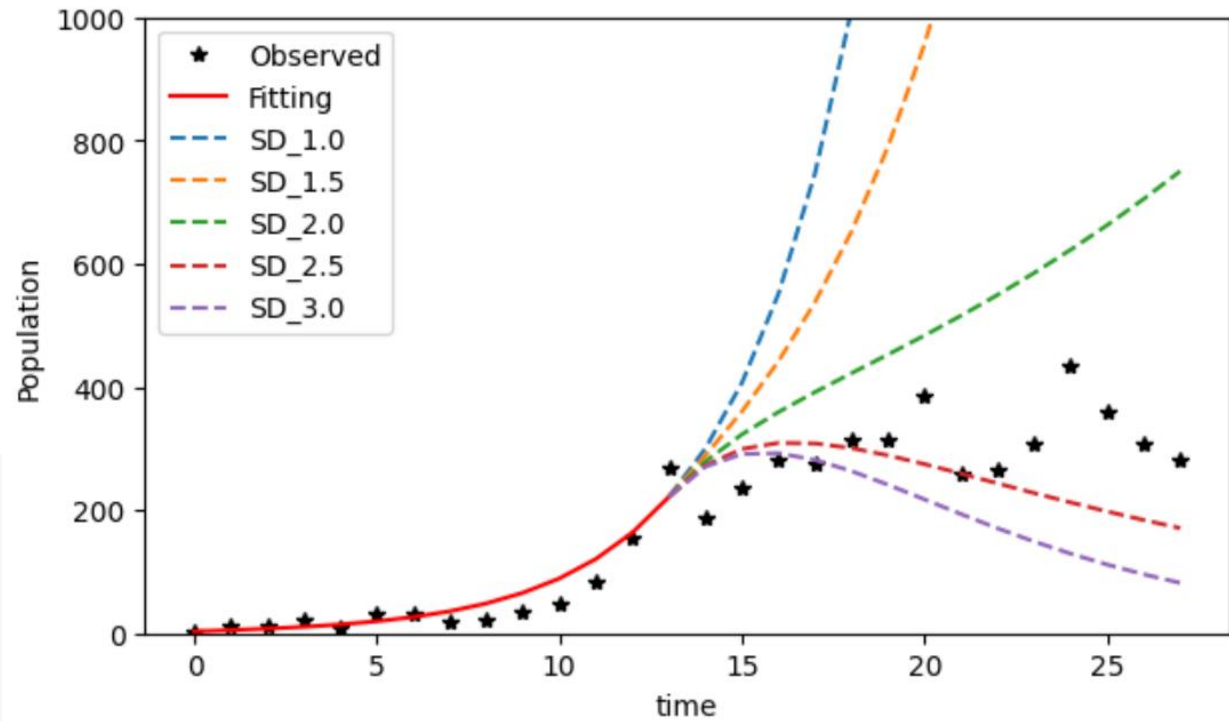
| | SD_1.0 | SD_1.5 | SD_2.0 | SD_2.5 | SD_3.0 |
|----|--------------|-------------|------------|------------|------------|
| 0 | 221.617084 | 221.617084 | 221.617084 | 221.617084 | 221.617084 |
| 1 | 300.311103 | 288.776385 | 279.162658 | 272.823320 | 270.613271 |
| 2 | 406.945810 | 360.341844 | 323.021833 | 299.175282 | 291.004892 |
| 3 | 551.439444 | 441.826789 | 359.273372 | 309.069508 | 292.339601 |
| 4 | 747.229046 | 537.736316 | 391.600087 | 308.240422 | 281.469413 |
| 5 | 1012.517301 | 652.378624 | 422.250705 | 300.592426 | 263.252772 |
| 6 | 1371.959817 | 790.363372 | 452.616722 | 288.772773 | 241.082154 |
| 7 | 1858.947378 | 956.949570 | 483.578425 | 274.565918 | 217.279258 |
| 8 | 2518.690887 | 1158.331769 | 515.712532 | 259.164913 | 193.389930 |
| 9 | 3412.387599 | 1401.915065 | 549.416867 | 243.358124 | 170.403323 |
| 10 | 4622.840768 | 1696.609966 | 584.985309 | 227.657652 | 148.913965 |
| 11 | 6262.027158 | 2053.168751 | 622.652941 | 212.387600 | 129.240914 |
| 12 | 8481.264455 | 2484.581077 | 662.623340 | 197.744702 | 111.514791 |
| 13 | 11484.829472 | 3006.545714 | 705.085178 | 183.839927 | 95.740823 |
| 14 | 15548.120667 | 3638.036320 | 750.222429 | 170.726991 | 81.844072 |

문제 3 : COVID-19 수학 모델

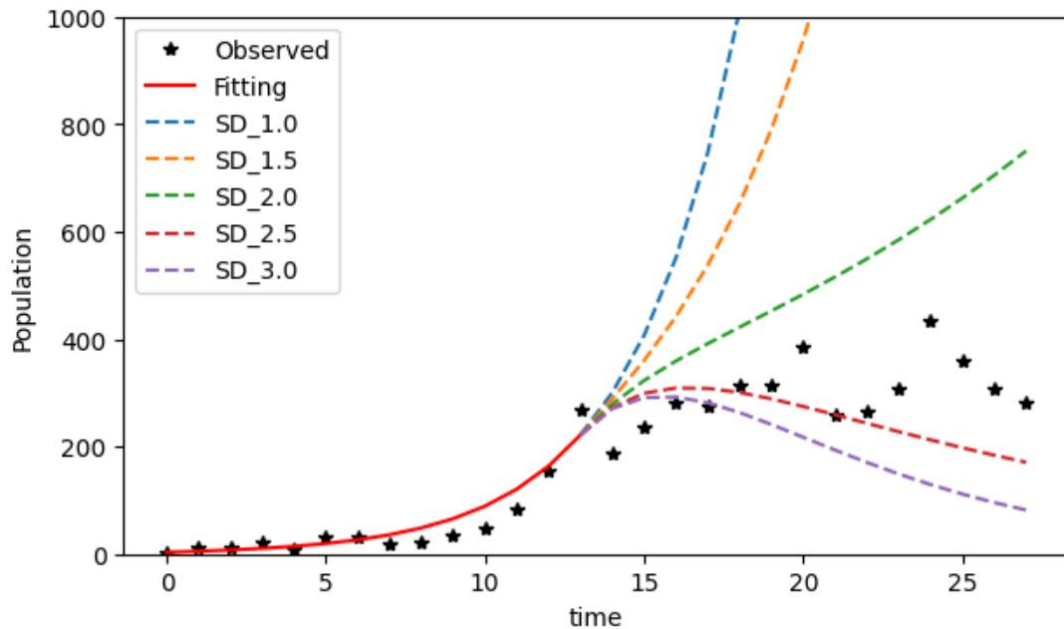
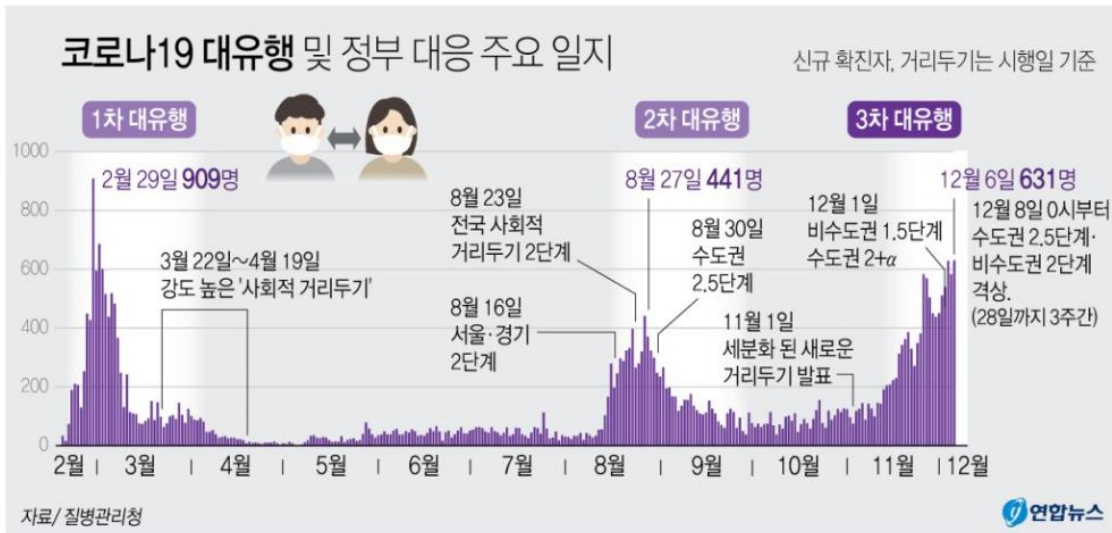
결과를 그림으로 그려보자.

```
plt.figure(figsize=(7,4))  
plt.plot(time,data['Cases'],'*k')  
plt.plot(time[:n1+1],pred_cases[:n1+1],'r')
```

```
for i in np.linspace(1.0,3.0,5):  
    plt.plot(time[n1:],result_SD['SD_'+str(i)], '--')  
plt.xlabel('time')  
plt.ylabel('Population')  
plt.ylim([0,1000])  
plt.legend(('Observed','Fitting','SD_1.0','SD_1.5','SD_2.0','SD_2.5','SD_3.0'),loc='best')  
plt.show()
```



문제 3 : COVID-19 수학 모델



- 실제 사회적 거리두기를 2단계를 시행하고,
- 본격적으로 8월 30일부터 사회적 거리두기 2.5단계를 시행하려고, 하는 단계에서,
- 우리 예측의 2단계와 2.5단계 사이에 감염자가 낮음으로 잘 예측된 결과라고 할 수 있다.

문제 3 : COVID-19 수학 모델

어떻게 가장 적절한 사회적 거리두기 단계를 결정할 수 있을까?

문제 3 : COVID-19 수학 모델

코로나로 인해 발생할 수 있는 피해는?

- 감염으로 인한 피해
- 사망으로 인한 피해



거리두기로 인해 발생할 수 있는 피해는?

- 오프라인 매장 영업 피해 (이용객 감소, 이용시간 감소)
- 관련 중소기업 적자
- 소비 불황
- 정부의 방역체계에 따른 관리 비용



코로나 감염과 사망으로 인한 직접 피해 비용과 거리두기로 인한 피해 비용

어떻게 계산할 수 있을까?

문제 3 : COVID-19 수학 모델

감염

Infected case

직접 의료비 (치료)+
비직접의료비 (역학조사)+
간접비(경제적손실)



Close contacts of a COVID-19 case

간접비

: 감염기간 동안 연령별 밀접접촉자의 경제적 손실 비용

문제 3 : COVID-19 수학 모델

1. 직접 의료비

실제 감염자의 치료에 필요한 비용

- 감염자의 치료 비용
- 코로나-19 진단검사 비용

2. 비의료비

의료비를 제외한 직접비용

- 역학조사비용
- 역학조사관 인건비
- 데이터 구축비
- 육아 및 가사노동 비용

3. 간접비

확진자와 격리대상자가 일하지 못해 발생하는 경제적 비용

- 확진자의 경제적 손실비용
- 격리대상자의 경제적 손실 비용

코로나19로 인한 질병비용은 얼마나 들까

연합뉴스가 코로나19 유행 시나리오를 바탕으로 코로나19 질병비용 분석
(사회적 거리두기로 인한 영업중단, 등교연기 등에 의해 파생된 경제적 손실 제외)

환자 1명당 최소 4,400만원

1명의 코로나19 슈퍼전파자가 4일 후 21명을 집단으로 감염시키고,
이들 21명이 4일 후 3.5명씩 감염시켜 8일간 총 95.5명의 환자가 발생했다고 가정



① 직접 의료비

- 1인당 625만원
- 95.5명이 총 5억9,673만원

무증상·경증환자:
4억6,327만원
중증환자:
1억3,346만원



② 비직접 의료비

- 1인당 430만원
- 총 4억원

역학조사 비용: 620만원
데이터 관리비: 2억7,000만원
육아 및 가사노동 비용:
1억3,100만원



③ 간접비 (노동손실액*)

- 1인당 3,370만원
- 총 32억1,475만원

격리대상자 1인당: 77만원
확진자 1인당: 155만원
→ 확진자 1명당 접촉자
수십명 격리

*확진자와 격리대상자가 일하지 못해 발생한 경제적 손실

자료/ 질병관리본부, 건강보험공단 등



김영은 기자 / 20200518

트위터 @yonhap_graphics 페이스북 tune.y.kr/LeYN1

문제 3 : COVID-19 수학 모델





사망

- 사망자 1명에 따른 경제적 손실
- 통계적 생명 데이터 (Value of a Statistical Life, VSL)
사람들이 건강과 관련하여 만드는 위험/보상 절충을 볼 때 보는 경제적 가치
- 연령 i 사망에 따른 경제적 손실 계산식

$$Cost_i = \sum_{i=1}^5 \frac{\text{평균남은수명}_i}{\text{평균 기대수명}} Death_i * VSL_{USA} \frac{GDP_{per\ capita_{Korea}}}{GDP_{per\ capita_{USA}}}$$

| | 0-19 | 20-34 | 35-49 | 50-64 | 65+ |
|----------------------|-----------|-----------|-----------|-----------|---------|
| 1인당 사망비용(\$/명) | 4,176,111 | 3,214,984 | 2,344,604 | 1,512,463 | 513,667 |
| 그룹별연령평균 (5세단위 계산) | 10.13 | 26.97 | 42.22 | 56.80 | 74.30 |
| 평균기대수명 (2019) | 83.3 | 83.3 | 83.3 | 83.3 | 83.3 |
| 평균남은수명(year) | 73.17 | 56.33 | 41.08 | 26.5 | 9 |

문제 3 : COVID-19 수학 모델

| | 1차 | 2차 | 3차 | 4차 |
|-----------|-----------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------|
| 시기 | 2020년 5월 | 9월 | 2021년 1월 | 3월 |
| 대상 | 전국민 | 소상공인, 특수고용직 | 소상공인, 고용취약계층 | 소상공인, 농어민, 노점상, 여행업 종사자 등 |
| 규모 (원) |  14조3000억 |  7조8000억 |  9조3000억 |  20조6000억 |

재난지원금
 $14.3\text{조} + 7.8\text{조} + 9.3\text{조} + 20.6\text{조} + 8.6\text{조}$
 $= 60.6\text{조}$
 코로나 대략 600일
 $60.6\text{조} / 600\text{일} = 1010\text{억}$

24일 국회는 본회의에서 총 34조9000억 원 규모의 추경안을 통과했다. 이 중 5차 재난지원금으로 쓰이는 금액은 총 8조6000억 원으로, 국민 87.7%에게 1인당 25만 원씩 지급할 예정이다. 1인 가구 기준 연소득 5000만 원 이상의 고소득자는 제외된다. 2021. 7. 26.

문제 3 : COVID-19 수학 모델

현재 우리는 사망자는 다루지 않으므로

감염 비용 + 사회적 비용 을 최소화 시키는 사회적 거리두기 단계가

가장 적절한 사회적 거리두기 단계라고 생각할 수 있다.

문제 3 : COVID-19 수학 모델



감염자 한사람당 비용은 왼쪽표와 같이 정부에서
계산한 값이 있기 때문에 그대로 활용하자.

한사람당 : 4,400만원

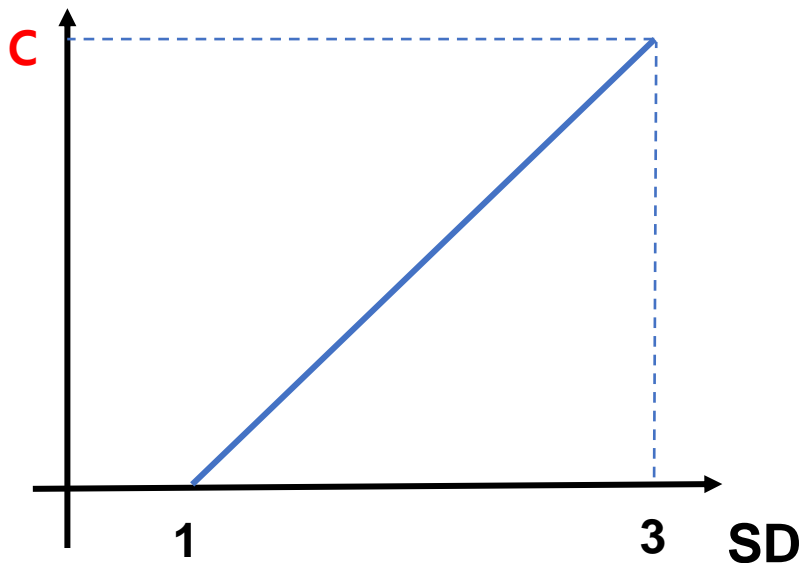
문제 3 : COVID-19 수학 모델

사회적 거리두기 단계에 따른 사회적 비용은? **알 수 없다.**

앞서 거리두기와 β 감소율 관계처럼 가정해보자.

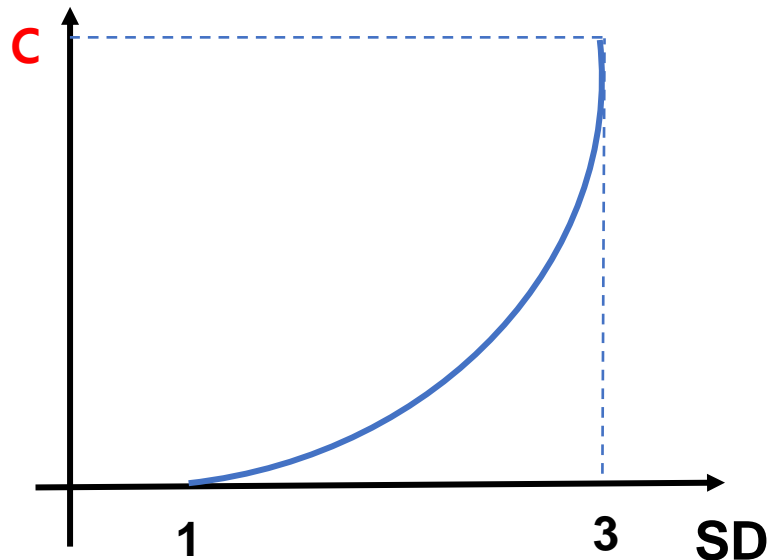
A

거리두기 비용



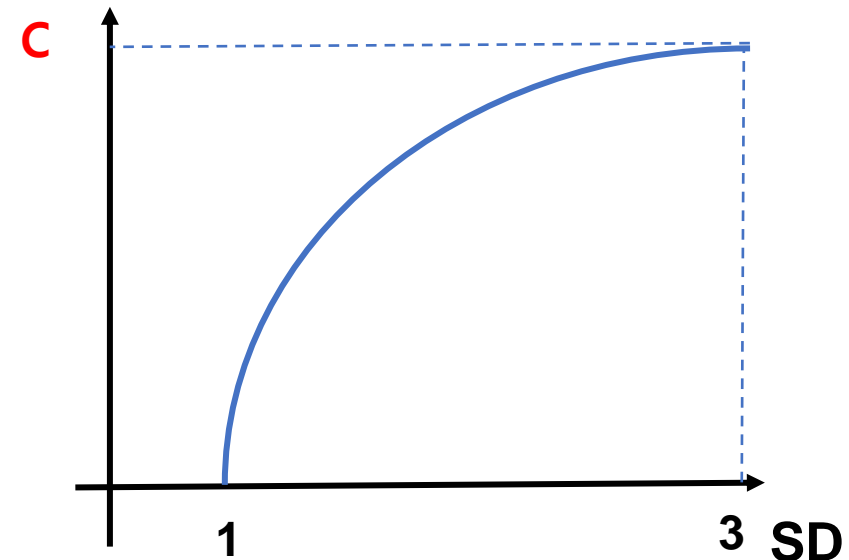
B

거리두기 비용



C

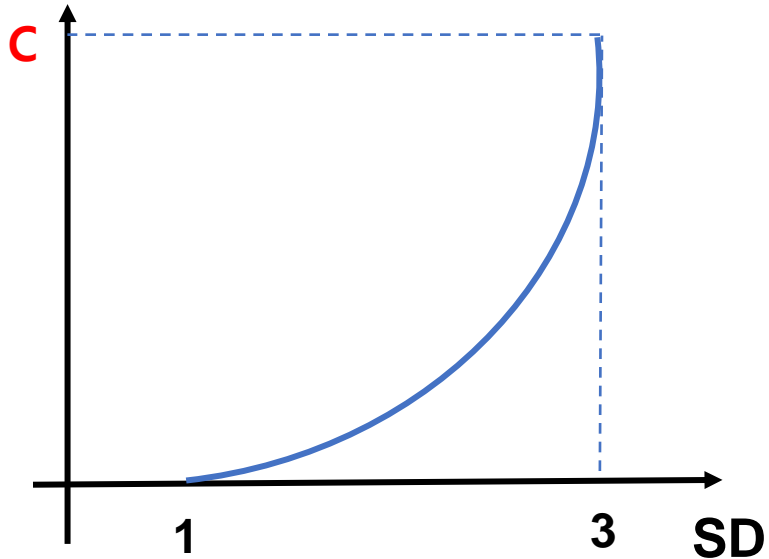
거리두기 비용



문제 3 : COVID-19 수학 모델

B

거리두기 비용



나의 선택은? B

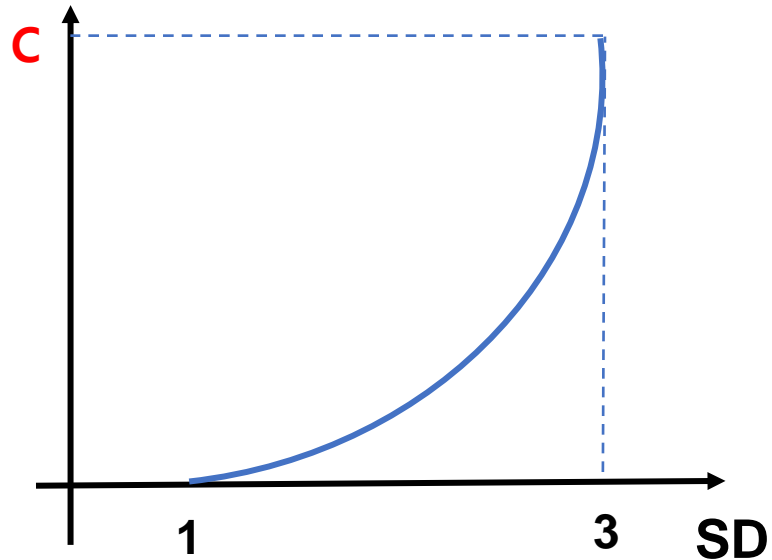
선택의 이유는?

1. 처음에는 작은 거리두기는 적은 비용이 소모될 것이다.
2. 거리두기를 단계가 강화 될수록 소득에 영향을 미치는 거리두기 규제가 많아져 비용 상승률이 갈수록 증가할 것이다.

문제 3 : COVID-19 수학 모델

B

거리두기 비용



그래프를 선택을 했다면 함수화를 시켜보자.

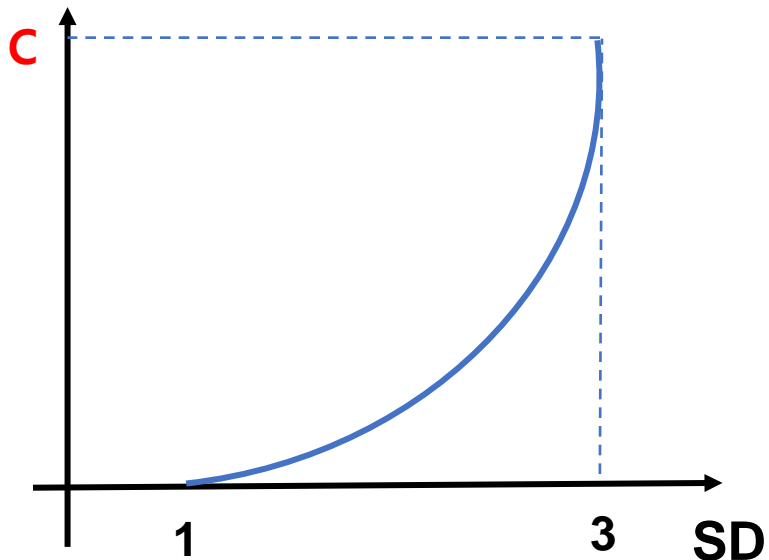
이 그래프 계형에 어울리는 함수는 어떤 것이 있는가?

- 2차 함수
- \arcsin 함수

문제 3 : COVID-19 수학 모델

B

거리두기 비용



2차 함수로 생각해 보자.

X축은 SD, y축은 거리두기 비용이다.

그리고 기본 SD 1단계라고 할 때, 이 제곱근 함수는 점 $(SD, \text{거리두기 비용}) = (1, 0)$ 를 지난다.

또한 3단계 봉쇄비용이 C라고 가정하면,

$(SD, \text{거리두기 비용}) = (3, C)$ 를 지난다. 그때 이 두점을 지나는 2차함수는?

$$\text{거리두기 비용} = \frac{C}{4} (SD - 1)^2$$

문제 3 : COVID-19 수학 모델

그러므로 하루 3단계 SD 비용(C)에 따라 최적의 SD 결과가 달라질 것이다.
그래서 다음과 같이 비용(C)이

- 100억
- 300억
- 500억
- 1000억
- 5000억

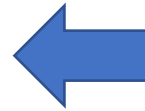
일 때, 2주간 최적의 SD와 그에 따른 신규 감염자 양상을 추정해 보자.

문제 3 : COVID-19 수학 모델

거리두기 비용에 따른 최적의 SD단계 추정 시뮬레이션

문제 3 : COVID-19 수학 모델

```
#C 봉쇄비용이라 가정  
def Cost_SD(C,SD):  
    return C*((SD-1)**2)/4
```



$$\text{거리두기 비용} = \frac{C}{4} (SD - 1)^2$$

문제 3 : COVID-19 수학 모델

목적함수 총비용(Total_cost)을 C와 SD에 관한 함수로 나타내보자.

```
def Total_cost(C,SD):  
    temp = np.zeros(n-n1+1)  
    temp[0]=pred_cases[n1]  
    for num,i in enumerate(range(n1,n)):  
        y[i+1,:] = rk4_SD(f_SD, y[i,:], h, SD)  
        temp[num+1] = theta*y[i+1,2]  
    cost = 0.44*temp.sum()+(n-n1+1)*Cost_SD(C,SD)  
    return cost
```

← 감염 비용 = 0.44(억) * 신규 감염자

← 사회적 비용 = SD시행 일수*1일당 거리두기 비용

```
def Obj_cost(SD):  
    return Total_cost(C,SD)
```


문제 3 : COVID-19 수학 모델

비용과 우리의 결과의 저장공간을 정의하자.

```
Max_cost = [100,300,500,1000,5000]
```

← 5가지 가정한 비용

```
optimal_cost_SD = np.zeros(5)
```

← 5가지 비용에 따른 최적의 SD 저장 공간 확보

```
optimal_result = pd.DataFrame()
```

← 5가지 비용에 따른 최적의 SD일 때, 신규 감염자

문제 3 : COVID-19 수학 모델

C 비용에 따른 최적의 SD를 계산하는 코드를 구현해보자.

```
bound_SD = (1.0,3.0) ← SD는 1-3단계 범위
SD_0 = 2.0 ← 최적화를 위한 SD 초기값
for num, C in enumerate(Max_cost):
    SD_optimal = minimize(Obj_cost, SD_0, method = 'SLSQP', bounds = (bound_SD,))
    optimal_cost_SD[num] = SD_optimal.x.item() ← 최적의 SD 결과저장
    optimal_result.loc[0, 'Cost_'+str(C)] = pred_cases[n1]
    for num,i in enumerate(range(n1,n)):
        y[i+1,:] = rk4_SD(f_SD, y[i,:], h, SD_optimal.x) ← 최적의 SD에 따른 모델 시뮬레이션
        optimal_result.loc[num+1, 'Cost_'+str(C)] = theta*y[i+1,2]
```

Minimize 사용

최적의 SD에 따른 신규 감염자 저장

문제 3 : COVID-19 수학 모델

C 비용에 따른 최적의 SD 결과를 확인해보자.

```
optimal_cost_SD
```

```
array([2.57794387, 2.26288588, 2.11243483, 1.91397657, 1.49950538])
```

100억

300억

500억

1000억

5000억

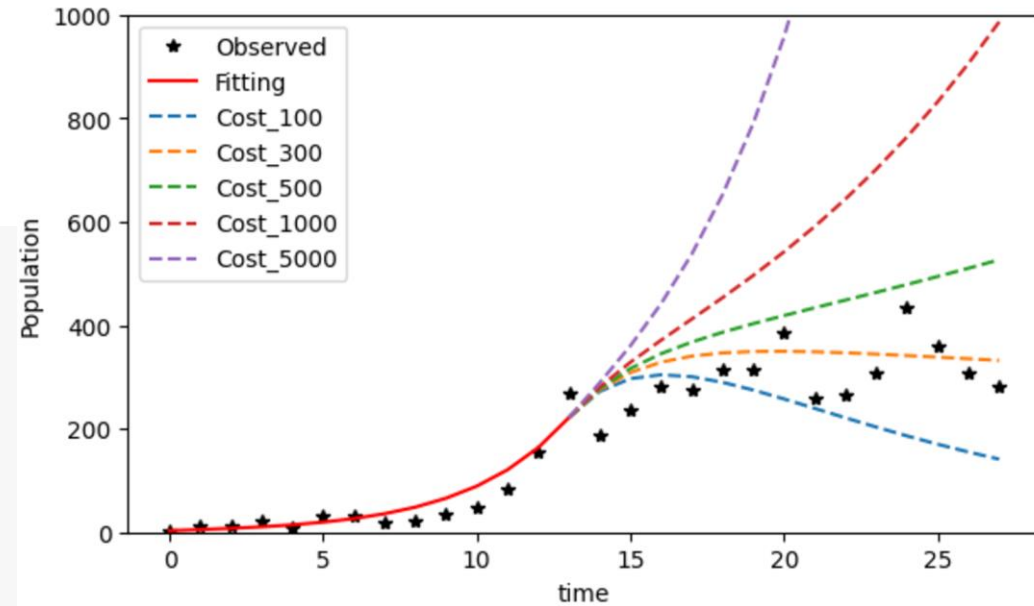
사회적 SD 비용이 비쌀 수록 감염자를 줄이기 위한 강한
거리두기를 하기 주저하게 된다.

문제 3 : COVID-19 수학 모델

C 비용에 따른 최적의 SD를 적용했을때, 신규 감염자 양상을 확인해보자.

```
plt.figure(figsize=(7,4))
plt.plot(time,data['Cases'],'*k')
plt.plot(time[:n1+1],pred_cases[:n1+1],'r')

for C in Max_cost:
    plt.plot(time[n1:],optimal_result['Cost_'+str(C)], '--')
plt.xlabel('time')
plt.ylabel('Population')
plt.ylim([0,1000])
plt.legend(('Observed','Fitting','Cost_'+str(Max_cost[0]),'Cost_'+str(Max_cost[1]),
          'Cost_'+str(Max_cost[2]),'Cost_'+str(Max_cost[3]),'Cost_'+str(Max_cost[4])),loc='best')
plt.show()
```



Thank you

