In [259…
```
import scipy.optimize as optimization
import numpy as np
import matplotlib.pyplot as plt
from numpy.polynomial import polynomial as P
import scipy
from scipy import stats
from matplotlib import pyplot
import os
from numpy import arange
import plotly.express as px
import pandas as pd

#mention all libraries imported and how plotly and pandas were installed
```

In [260…
```
print("Name: Rohan Kadam")
print("Student ID: 20334092")
```

```
Name: Rohan Kadam
Student ID: 20334092
```

In [261…
```
input_file = "C:\Rohan\COVIDData.txt"
days, cases = np.loadtxt(input_file, skiprows=1, unpack=True)

#loading file in
```

In [262…
```
np.set_printoptions(suppress=True)
print(days)
print(cases)
print(len(days))
print(len(cases))

#checking if both sets of data match
```

```
[  1.   2.   3.   4.   5.   6.   7.   8.   9.  10.  11.  12.  13.  14.
  15.  16.  17.  18.  19.  20.  21.  22.  23.  24.  25.  26.  27.  28.
  29.  30.  31.  32.  33.  34.  35.  36.  37.  38.  39.  40.  41.  42.
  43.  44.  45.  46.  47.  48.  49.  50.  51.  52.  53.  54.  55.  56.
  57.  58.  59.  60.  61.  62.  63.  64.  65.  66.  67.  68.  69.  70.
  71.  72.  73.  74.  75.  76.  77.  78.  79.  80.  81.  82.  83.  84.
  85.  86.  87.  88.  89.  90.  91.  92.  93.  94.  95.  96.  97.  98.
  99. 100. 101. 102. 103. 104. 105. 106. 107. 108. 109. 110. 111. 112.
 113. 114. 115. 116. 117. 118. 119. 120. 121. 122. 123. 124. 125. 126.
 127. 128. 129. 130. 131. 132. 133. 134. 135. 136. 137. 138. 139. 140.
 141. 142. 143. 144. 145. 146. 147. 148. 149. 150. 151. 152. 153. 154.
 155. 156. 157. 158. 159. 160. 161. 162. 163. 164. 165. 166. 167. 168.
```

```
 169. 170. 171. 172. 173. 174. 175. 176. 177. 178. 179. 180. 181. 182.
 183. 184. 185. 186. 187. 188. 189. 190. 191. 192. 193. 194. 195. 196.
 197. 198. 199. 200. 201. 202. 203. 204. 205. 206. 207. 208. 209. 210.
 211. 212. 213. 214. 215. 216. 217. 218. 219. 220. 221. 222. 223. 224.
 225. 226. 227. 228. 229. 230. 231. 232. 233. 234. 235. 236. 237. 238.
 239. 240. 241. 242. 243. 244. 245. 246. 247. 248. 249. 250. 251. 252.
 253. 254. 255. 256. 257. 258. 259. 260. 261. 262. 263. 264. 265. 266.
 267. 268. 269. 270. 271. 272. 273. 274. 275. 276. 277. 278. 279. 280.
 281. 282. 283. 284. 285. 286. 287. 288. 289. 290. 291. 292. 293. 294.
 295. 296. 297. 298. 299. 300. 301. 302. 303. 304. 305. 306. 307. 308.
 309. 310. 311. 312. 313. 314. 315. 316. 317. 318. 319. 320. 321. 322.
 323. 324. 325. 326. 327. 328. 329. 330. 331. 332. 333. 334. 335. 336.
 337. 338. 339. 340. 341. 342. 343. 344. 345. 346. 347. 348. 349. 350.
 351. 352. 353. 354. 355. 356. 357. 358. 359. 360. 361. 362. 363. 364.
 365. 366. 367. 368. 369. 370. 371. 372. 373. 374. 375. 376. 377. 378.
 379. 380. 381. 382. 383. 384. 385. 386. 387. 388. 389. 390. 391. 392.
 393. 394. 395. 396. 397. 398. 399. 400. 401. 402. 403. 404. 405. 406.
 407. 408. 409. 410. 411. 412. 413. 414. 415. 416. 417. 418. 419. 420.
 421. 422. 423. 424. 425. 426. 427. 428. 429. 430. 431. 432. 433. 434.
 435. 436. 437. 438. 439. 440. 441. 442. 443. 444. 445. 446. 447. 448.
 449. 450. 451. 452. 453. 454. 455. 456. 457. 458. 459. 460. 461. 462.
 463. 464. 465. 466. 467. 468. 469. 470. 471. 472. 473. 474. 475. 476.
 477. 478. 479. 480. 481. 482. 483. 484. 485. 486. 487. 488. 489. 490.
 491. 492. 493. 494. 495. 496. 497. 498. 499. 500. 501. 502. 503. 504.
 505. 506. 507. 508. 509. 510. 511. 512. 513. 514. 515. 516. 517. 518.
 519. 520. 521. 522. 523. 524. 525. 526. 527. 528. 529. 530. 531. 532.
 533. 534. 535. 536. 537. 538. 539. 540. 541. 542. 543. 544. 545. 546.
 547. 548. 549. 550. 551. 552. 553. 554. 555. 556. 557. 558. 559. 560.
 561. 562. 563. 564. 565. 566. 567. 568. 569. 570. 571. 572. 573. 574.
 575. 576.]
[   1.    1.    4.    7.    5.    1.    2.    3.   10.    9.   27.   20.
   39.   40.   54.   69.   74.  191.  126.  102.  121.  219.  204.  235.
  255.  302.  294.  200.  295.  325.  212.  402.  424.  331.  390.  748.
  365.  464.  705.  696.  553.  430.  527.  548.  657.  629.  597.  630.
  445.  401.  388.  629.  936.  577.  377.  701.  386.  229.  376.  359.
  221.  343.  330.  266.  211.  265.  137.  156.  219.  236.  139.  107.
  159.  426.  129.   92.   64.   88.   51.   64.   76.  115.   76.   57.
   59.   37.   73.   46.   39.   59.   66.   77.   10.   47.   38.   28.
   24.   25.    9.    9.   19.    8.   13.   46.    8.   18.   14.    8.
   16.   13.   22.    6.    4.   10.    5.   11.   11.   23.    3.   24.
   11.    6.   15.    9.   11.   18.    4.   24.   11.   22.   25.   23.
   17.   11.   32.   14.   21.   34.   21.    9.    6.   36.   17.    7.
   20.   24.   12.   11.   40.   14.   85.   38.   45.   53.   46.   44.
   48.   69.   98.  174.   68.   57.   34.   40.   92.   67.  200.   66.
   56.  190.   54.  136.   80.  156.   61.  147.   92.  164.   93.  128.
  141.   42.   53.  217.   89.   95.   98.  231.  138.  102.  307.   84.
  196.  211.  159.  255.  208.  357.  254.  240.  253.  274.  396.  188.
  334.  234.  324.  326.  248.  430.  390.  363.  429.  442.  470.  613.
  364.  518.  432.  611.  506.  617. 1012.  817.  825.  811. 1095. 1205.
```

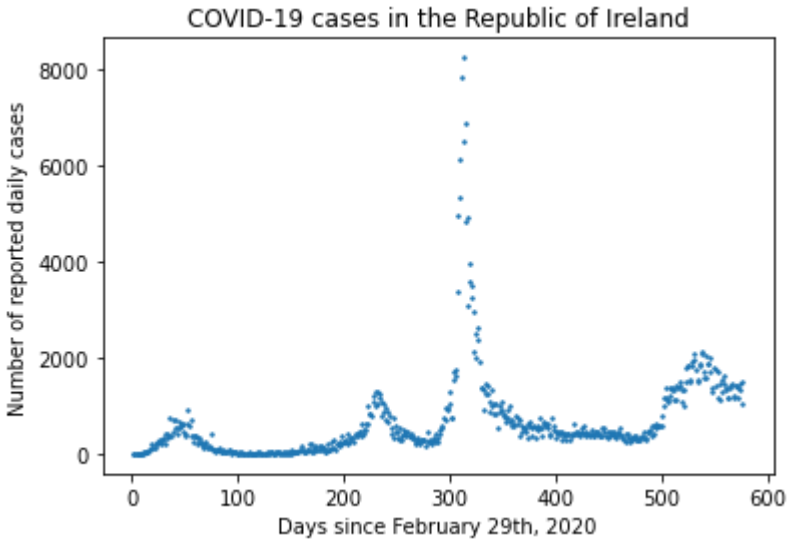
```
 1000. 1276. 1283. 1031. 1269. 1167. 1066.  777.  859. 1025.  939.  720.
  675.  866.  772.  416.  552.  767.  322.  444.  591.  449.  335.  542.
  270.  270.  362.  395.  482.  456.  378.  456.  366.  379.  429.  330.
  344.  318.  252.  226.  269.  335.  206.  243.  299.  306.  269.  270.
  183.  265.  456.  301.  242.  215.  227.  310.  313.  247.  429.  264.
  329.  431.  484.  582.  527.  764.  727.  970.  938.  922. 1025. 1296.
  744.  765. 1546. 1720. 1620. 1754. 3394. 4962. 6110. 5325. 7836. 6521.
 8248. 4842. 6888. 4929. 3086. 3569. 3955. 3498. 3231. 2944. 2121. 2001.
 2488. 2608. 2371. 1910. 1378. 1372.  928. 1335. 1466. 1254. 1414. 1247.
 1062.  879. 1013. 1318. 1047.  827. 1024.  829.  556. 1006.  867.  921.
 1078.  788.  821.  744.  650.  901.  763.  988.  679.  686.  575.  574.
  613.  776.  738.  612.  687.  369.  566.  462.  522.  539.  525.  437.
  311.  631.  592.  646.  543.  384.  575.  349.  557.  582.  507.  525.
  769.  520.  371.  683.  606.  584.  624.  604.  539.  368.  411.  761.
  591.  511.  457.  320.  443.  423.  400.  473.  455.  303.  394.  358.
  431.  309.  420.  420.  269.  403.  390.  401.  617.  434.  461.  429.
  437.  426.  371.  474.  545.  569.  402.  453.  383.  418.  393.  434.
  408.  514.  381.  379.  448.  456.  425.  461.  375.  379.  365.  526.
  470.  533.  384.  447.  349.  353.  445.  432.  461.  456.  367.  370.
  337.  407.  465.  529.  416.  313.  377.  271.  259.  398.  319.  431.
  315.  242.  283.  329.  373.  313.  393.  288.  284.  294.  348.  304.
  380.  443.  340.  305.  351.  452.  448.  512.  448.  562.  365.  397.
  581.  534.  631.  581.  576.  600.  589.  783.  994. 1173. 1377. 1179.
 1071. 1110. 1378. 1189. 1386. 1345. 1126. 1345. 1120. 1408. 1361. 1501.
 1427. 1098. 1352. 1015. 1314. 1491. 1782. 1828. 1837. 1522. 1508. 1819.
 1903. 1978. 2074. 1758. 1558. 1496. 1861. 1818. 2098. 2125. 1688. 1592.
 1571. 2051. 1866. 1875. 1997. 1706. 1293. 1382. 1789. 1751. 1414. 1703.
 1180. 1144. 1470. 1545. 1292. 1620. 1466. 1346. 1394. 1181. 1185. 1413.
 1392. 1456. 1224. 1154. 1423. 1432. 1355. 1163. 1335. 1459. 1049. 1499.]
576
576
```

In [304…

```
# TASK 1 recreating graph given
plt.title("COVID-19 cases in the Republic of Ireland")
plt.ylabel("Number of reported daily cases")
plt.xlabel("Days since February 29th, 2020")
plt.scatter(days, cases, s = 2)

#plt.figure(figsize=(20, 20))
#recreating graph given, information given from data set, gives us insight into the trends present
```

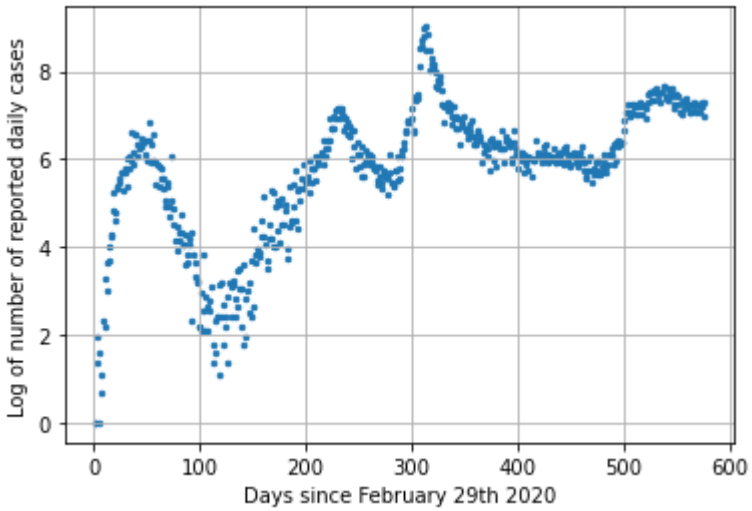
Out[304…

```
<matplotlib.collections.PathCollection at 0x177969bb760>
```



In [353…

```
#TASK 2 - seeing how straight line fits could potentially be applied for each wave

plt.scatter(days, np.log(cases), s = 6)
plt.xlabel("Days since February 29th 2020")
plt.ylabel("Log of number of reported daily cases")
plt.grid()
plt.show()

#how linear fits could potentially be applied
#why do a log on y axis and how does this help
#now much easier to identify trends
```



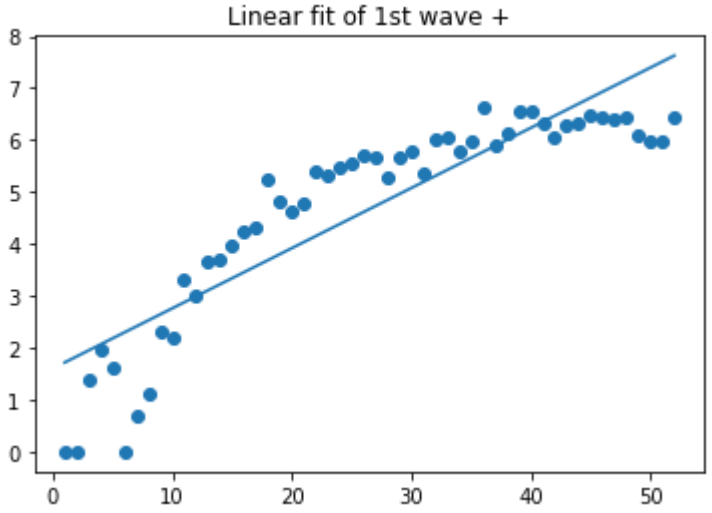
In [306…

```
import plotly.express as px
import pandas as pd
norm_data = pd.DataFrame({'days':days, 'cases':cases})
fig = px.line(norm_data)
fig.show()

#why did you try search this way of doing it
#benefit of this
#how did you install package
```

In [307…

```
linearfit1_function = stats.linregress(days[0:52], np.log(cases[0:52]))
linearfit1 = (linearfit1_function.slope*days[0:52] + linearfit1_function.intercept)
plt.scatter(days[0:52], np.log(cases[0:52]))
plt.title("Linear fit of 1st wave +")
plt.plot(days[0:52], linearfit1)
intercept1 = linearfit1_function.intercept
linearintercept1 = "a = "
print(lineintercept1 + str(intercept1))
slope1 = linearfit1_function.slope
lineslope1 = "b ="
print(lineslope1 + str(slope1))

#now finding slope and intercept for each fit line
#what do these tell us about the fit lines
#is the fit line skewed at all
#what does + and - mean
```

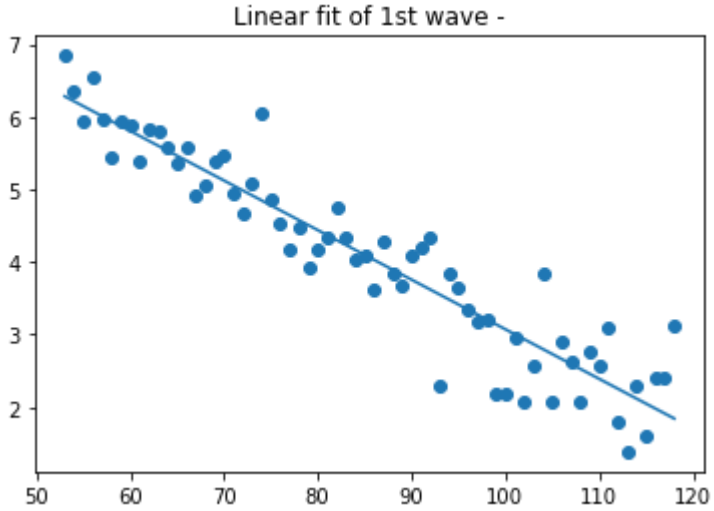
```
a = 1.6016822884770465
b =0.1158551490941329
```



In [308…

```
linearfit2_function = stats.linregress(days[52:118], np.log(cases[52:118]))
linearfit2 = (linearfit2_function.slope*days[52:118] + linearfit2_function.intercept)
plt.scatter(days[52:118], np.log(cases[52:118]))
plt.title("Linear fit of 1st wave -")
plt.plot(days[52:118], linearfit2)
intercept2 = linearfit2_function.intercept
linearintercept2 = "a = "
print(lineintercept2 + str(intercept2))
slope2 = linearfit2_function.slope
lineslope2 = "b ="
print(lineslope2 + str(slope2))
```

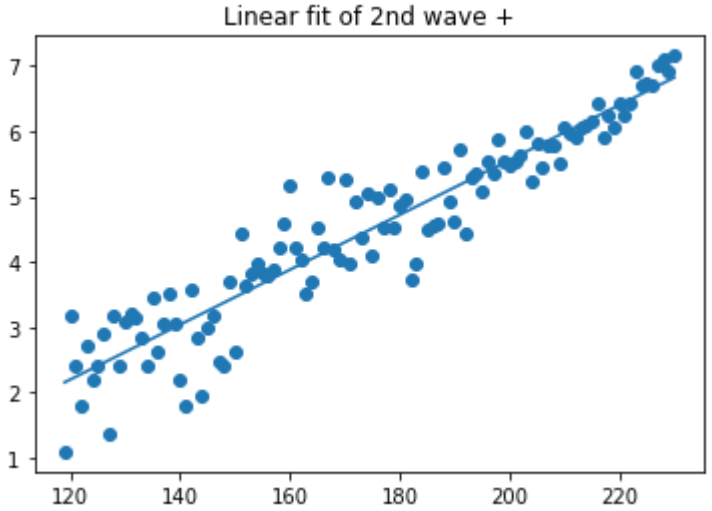
```
a = 9.90000023115569
b =-0.06827931132514443
```



In [313…

```
linearfit3_function = stats.linregress(days[118:230], np.log(cases[118:230]))
linearfit3 = (linearfit3_function.slope*days[118:230] + linearfit3_function.intercept)
plt.scatter(days[118:230], np.log(cases[118:230]))
plt.title("Linear fit of 2nd wave +")
plt.plot(days[118:230], linearfit3)
intercept3 = linearfit3_function.intercept
linearintercept3 = "a = "
print(lineintercept3 + str(intercept3))
slope3 = linearfit3_function.slope
lineslope3 = "b ="
print(lineslope3 + str(slope3))
```

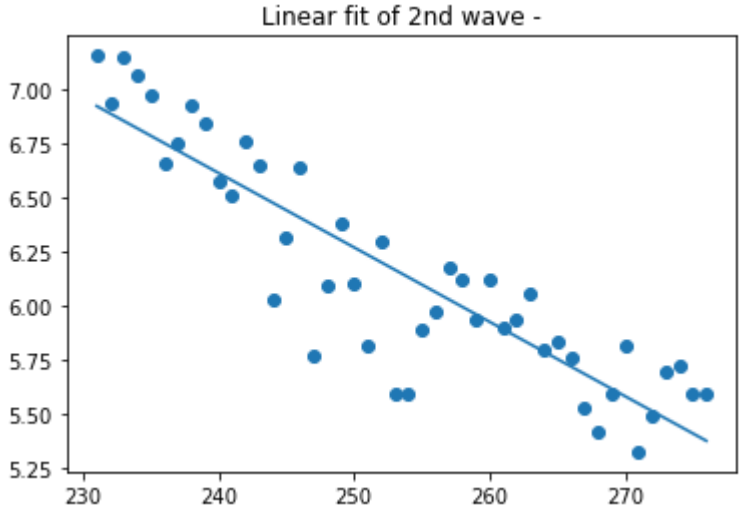
```
a = -2.833469818797857
b =0.041954134690244164
```



In [355…

```
linearfit4_function = stats.linregress(days[230:276], np.log(cases[230:276]))
linearfit4 = (linearfit4_function.slope*days[230:276] + linearfit4_function.intercept)
plt.scatter(days[230:276], np.log(cases[230:276]))
plt.title("Linear fit of 2nd wave -")
plt.plot(days[230:276], linearfit4)
intercept4 = linearfit4_function.intercept
linearintercept4 = "a = "
print(lineintercept4 + str(intercept4))
slope4 = linearfit4_function.slope
lineslope4 = "b ="
print(lineslope4 + str(slope4))
```

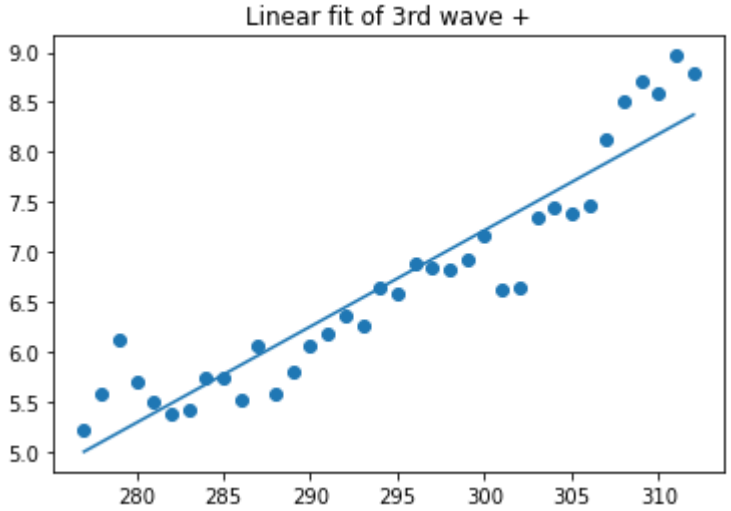
```
a = 14.858700960290992
b =-0.03434945590956145
```



In [315…

```
linearfit5_function = stats.linregress(days[276:312], np.log(cases[276:312]))
linearfit5 = (linearfit5_function.slope*days[276:312] + linearfit5_function.intercept)
plt.scatter(days[276:312], np.log(cases[276:312]))
plt.title("Linear fit of 3rd wave +")
plt.plot(days[276:312], linearfit5)
intercept5 = linearfit5_function.intercept
lineintercept5 = "a = "
print(lineintercept5 + str(intercept5))
slope5 = linearfit5_function.slope
lineslope5 = "b ="
print(lineslope5 + str(slope5))
```

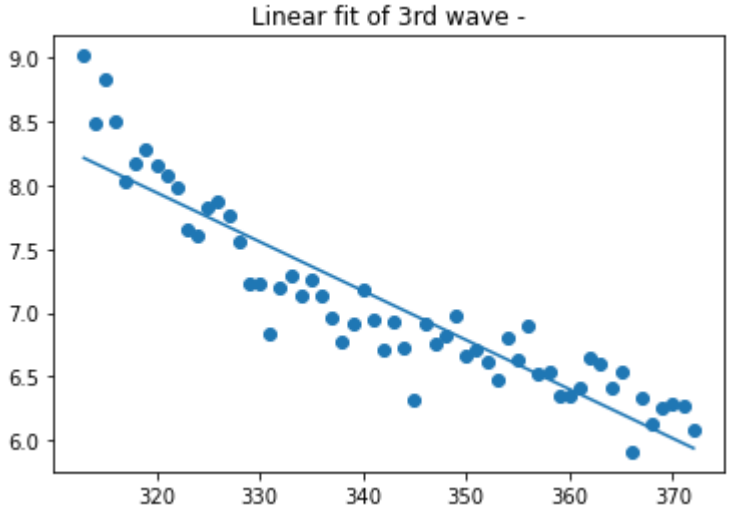
```
a = -21.736076238263905
b =0.09650781512829676
```



In [316…

```
linearfit6_function = stats.linregress(days[312:372], np.log(cases[312:372]))
linearfit6 = (linearfit6_function.slope*days[312:372] + linearfit6_function.intercept)
plt.scatter(days[312:372], np.log(cases[312:372]))
plt.title("Linear fit of 3rd wave -")
plt.plot(days[312:372], linearfit6)
intercept6 = linearfit6_function.intercept
lineintercept6 = "a = "
print(lineintercept6 + str(intercept6))
slope6 = linearfit6_function.slope
lineslope6 = "b ="
print(lineslope6 + str(slope6))
```

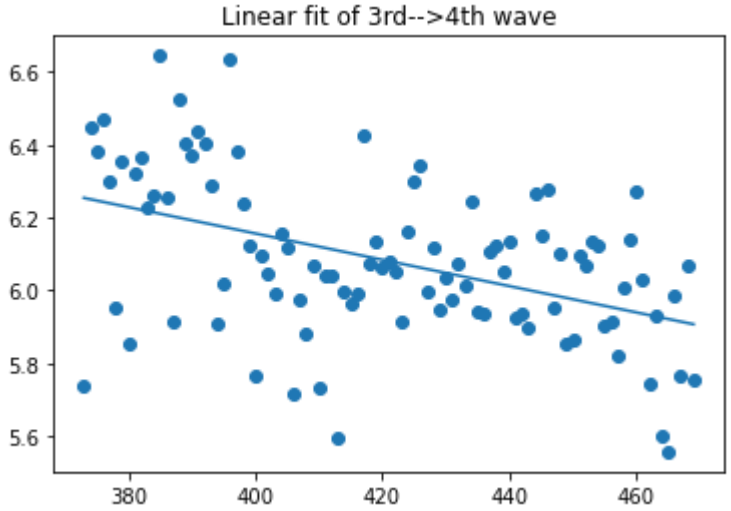
```
a = 20.290570954522313
b =-0.03858181981649011
```



In [318…

```
linearfit7_function = stats.linregress(days[372:469], np.log(cases[372:469]))
linearfit7 = (linearfit7_function.slope*days[372:469] + linearfit7_function.intercept)
plt.scatter(days[372:469], np.log(cases[372:469]))
plt.title("Linear fit of 3rd-->4th wave")
plt.plot(days[372:469], linearfit7)
intercept7 = linearfit7_function.intercept
lineintercept7 = "a = "
print(lineintercept7 + str(intercept7))
slope7 = linearfit7_function.slope
lineslope7 = "b ="
print(lineslope7 + str(slope7))
```

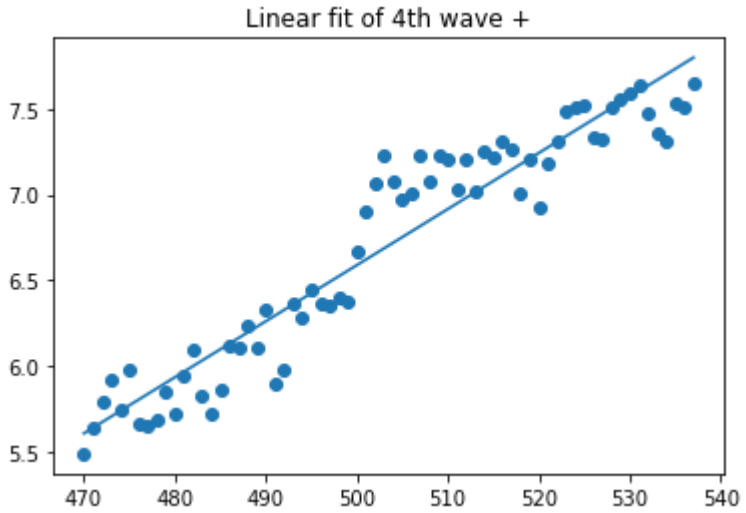
```
a = 7.602660580576682
b =-0.0036155029932723346
```



In [319…

```
linearfit8_function = stats.linregress(days[469:537], np.log(cases[469:537]))
linearfit8 = (linearfit8_function.slope*days[469:537] + linearfit8_function.intercept)
plt.scatter(days[469:537], np.log(cases[469:537]))
plt.title("Linear fit of 4th wave +")
plt.plot(days[469:537], linearfit8)
intercept8 = linearfit8_function.intercept
lineintercept8 = "a = "
print(lineintercept8 + str(intercept8))
slope8 = linearfit8_function.slope
lineslope8 = "b ="
print(lineslope8 + str(slope8))

#why did you decide to take these points as a linear fit
#describe what was happening i.e plauteuing in spread
```

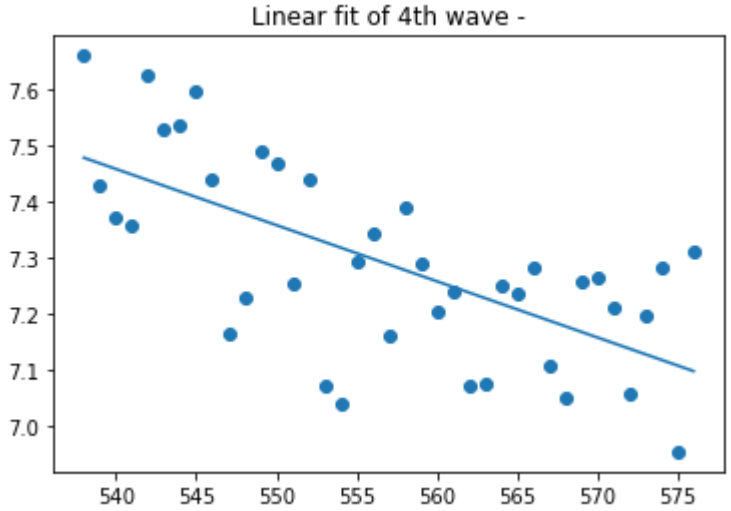
```
a = -9.757928868409273
b =0.03269260492317787
```



In [320…

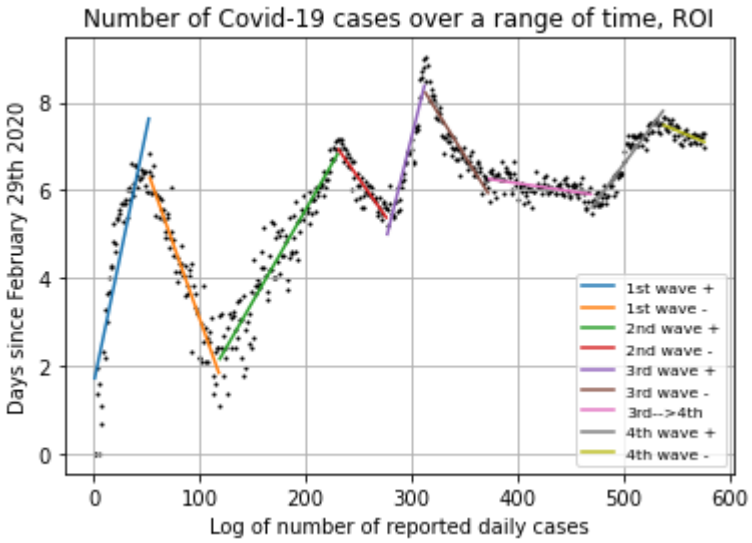
```
linearfit9_function = stats.linregress(days[537:576], np.log(cases[537:576]))
linearfit9 = (linearfit9_function.slope*days[537:576] + linearfit9_function.intercept)
plt.scatter(days[537:576], np.log(cases[537:576]))
plt.title("Linear fit of 4th wave -")
plt.plot(days[537:576], linearfit9)
intercept9 = linearfit9_function.intercept
lineintercept9 = "a = "
print(lineintercept9 + str(intercept9))
slope9 = linearfit9_function.slope
lineslope9 = "b ="
print(lineslope9 + str(slope9))
```

```
a = 12.861896057072073
b =-0.010004576248892208
```



In [348…

```
plt.plot(days[0:52], linearfit1, label='1st wave +')
plt.plot(days[52:118], linearfit2, label='1st wave -')
plt.plot(days[118:230], linearfit3, label='2nd wave +')
plt.plot(days[230:276], linearfit4, label='2nd wave -')
plt.plot(days[276:312], linearfit5, label='3rd wave +')
plt.plot(days[312:372], linearfit6, label='3rd wave -')
plt.plot(days[372:469], linearfit7, label='3rd-->4th')
plt.plot(days[469:537], linearfit8, label='4th wave +')
plt.plot(days[537:576], linearfit9, label='4th wave -')
plt.scatter(days[0:576], np.log(cases[0:576]), color ='black', s = 1.1)
plt.title("Number of Covid-19 cases over a range of time, ROI")
plt.xlabel("Log of number of reported daily cases")
plt.ylabel("Days since February 29th 2020")
plt.legend(loc=4, fontsize = 7.8)
plt.grid()
plt.show()
```



In [322…

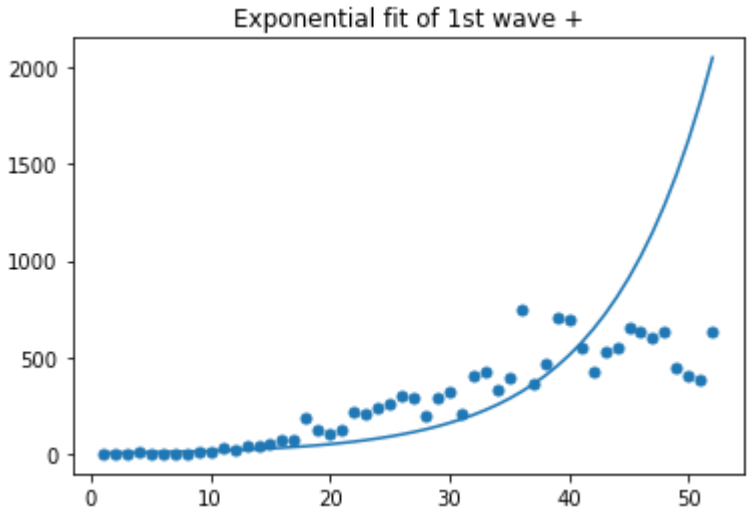
```
no1 = np.e**(intercept1 + slope1*0)
no2 = np.e**(intercept2 + slope2*52)
no3 = np.e**(intercept3 + slope3*118)
no4 = np.e**(intercept4 + slope4*230)
no5 = np.e**(intercept5 + slope5*276)
no6 = np.e**(intercept6 + slope6*312)
no7 = np.e**(intercept7 + slope7*372)
no8 = np.e**(intercept8 + slope8*469)
no9 = np.e**(intercept9 + slope9*537)
print(no1)
print(no2)
print(no3)
print(no4)
print(no5)
print(no6)
print(no7)
print(no8)
print(no9)

#how background colours etc affected clarity of graph
#how linear fit can help us analyse these graphs
```

```
4.961371866475987
572.1928255909049
8.307162335269078
1051.871356465713
134.30062229617883
```

```
3839.29168592591
522.0135091154974
263.7239311229675
1789.0474590573285
```

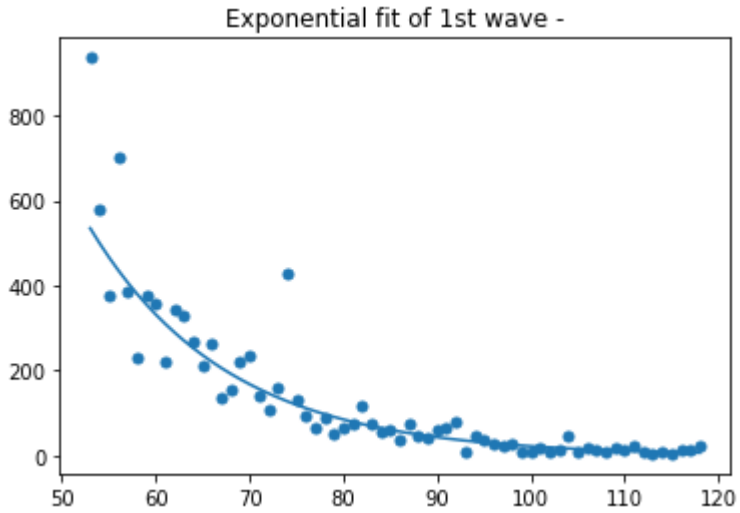
In [332…

```
exponentialfit1 = no1*np.e**((slope1)*(days[0:52] - 0))
plt.scatter(days[0:52], cases[0:52], s = 25)
plt.title("Exponential fit of 1st wave +")
plt.plot(days[0:52], exponentialfit1)
plt.show()

#now why exponential fit is used here and why does it help
#mention how different ranges are in each graph and it is not standardized
```

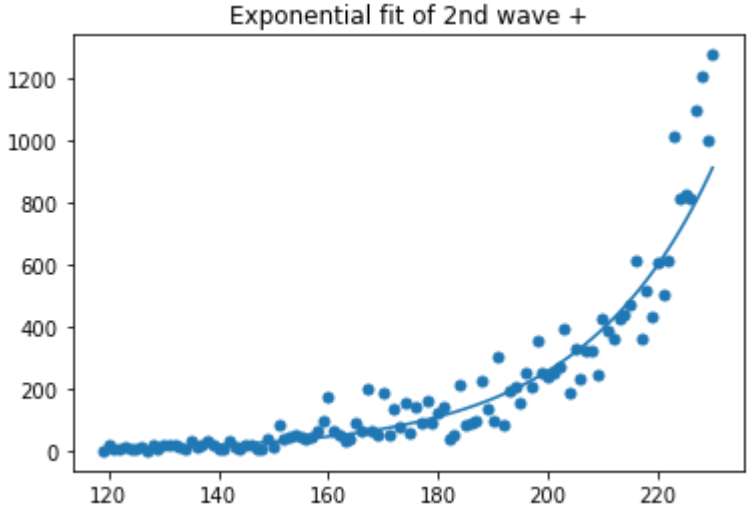


In [333…

```
exponentialfit2 = no2*np.e**((slope2)*(days[52:118] - 52))
plt.scatter(days[52:118], cases[52:118], s = 25)
plt.title("Exponential fit of 1st wave -")
plt.plot(days[52:118], exponentialfit2)
plt.show()
```

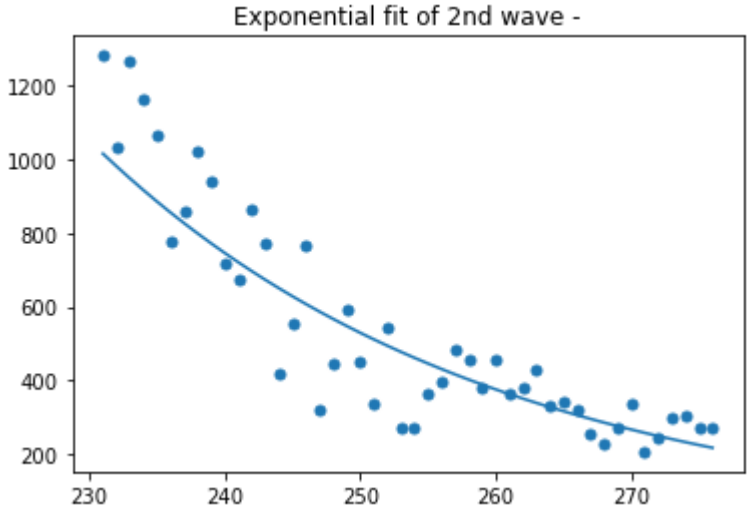


In [335…
```
exponentialfit3 = no3*np.e**((slope3)*(days[118:230] - 118))
plt.scatter(days[118:230], cases[118:230], s = 25)
plt.title("Exponential fit of 2nd wave +")
plt.plot(days[118:230], exponentialfit3)
plt.show()
```



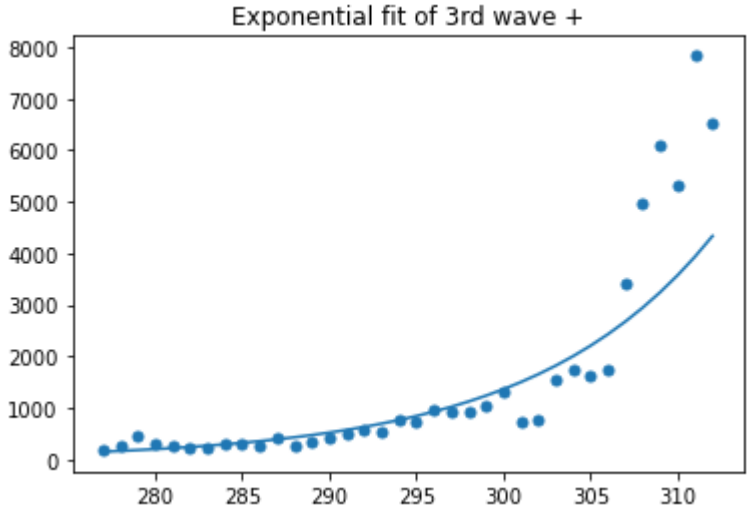
In [337…
```
exponentialfit4 = no4*np.e**((slope4)*(days[230:276] - 230))
plt.scatter(days[230:276], cases[230:276], s = 25)
plt.title("Exponential fit of 2nd wave -")
```

```
plt.plot(days[230:276], exponentialfit4)
plt.show()
```



In [339…

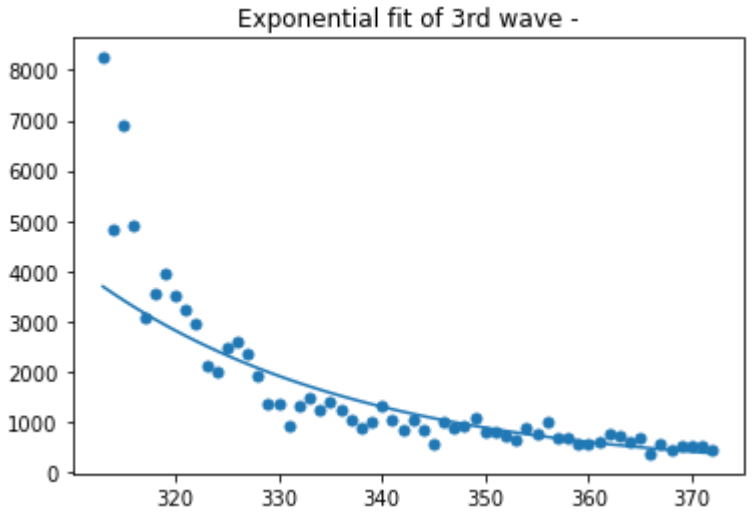
```
exponentialfit5 = no5*np.e**((slope5)*(days[276:312] - 276))
plt.scatter(days[276:312], cases[276:312], s = 25)
plt.title("Exponential fit of 3rd wave +")
plt.plot(days[276:312], exponentialfit5)
plt.show()
```



In [341…

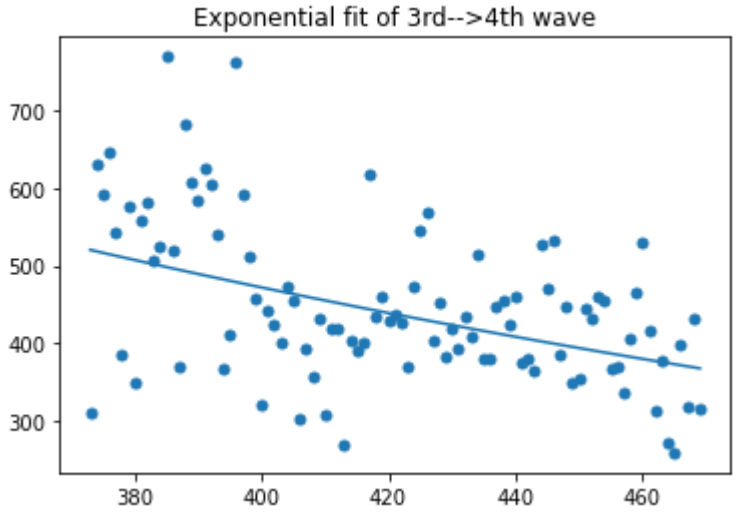
```
exponentialfit6 = no6*np.e**((slope6)*(days[312:372] - 312))
plt.scatter(days[312:372], cases[312:372], s = 25)
```

```
plt.title("Exponential fit of 3rd wave -")
plt.plot(days[312:372], exponentialfit6)
plt.show()
```



In [344…

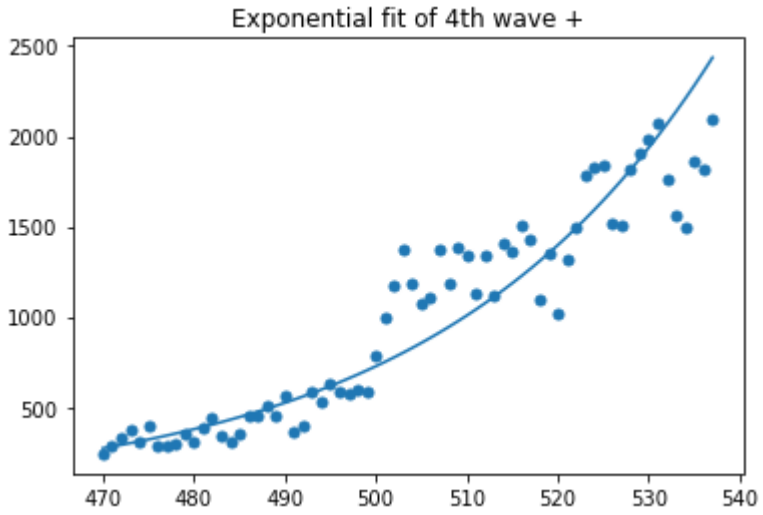
```
exponentialfit7 = no7*np.e**((slope7)*(days[372:469] - 372))
plt.scatter(days[372:469], cases[372:469], s = 25)
plt.title("Exponential fit of 3rd-->4th wave")
plt.plot(days[372:469], exponentialfit7)
plt.show()
```



In [343…

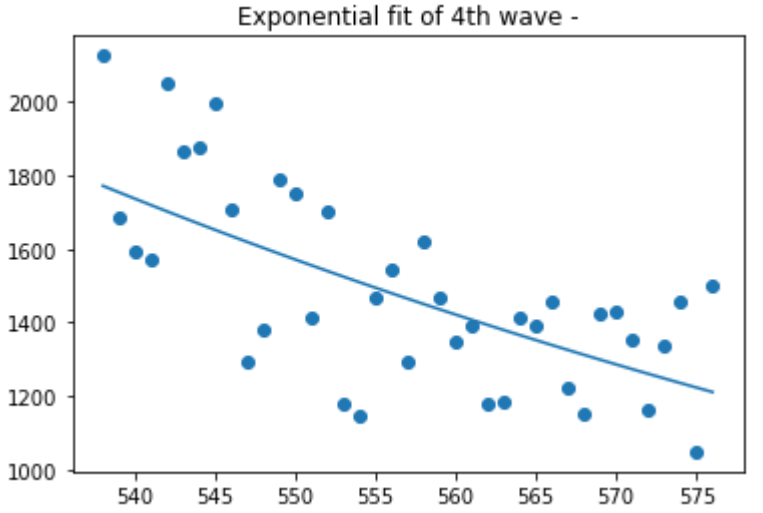
```
exponentialfit8 = no8*np.e**((slope8)*(days[469:537] - 469))
```

```
plt.scatter(days[469:537], cases[469:537], s = 25)
plt.title("Exponential fit of 4th wave +")
plt.plot(days[469:537], exponentialfit8)
plt.show()
```

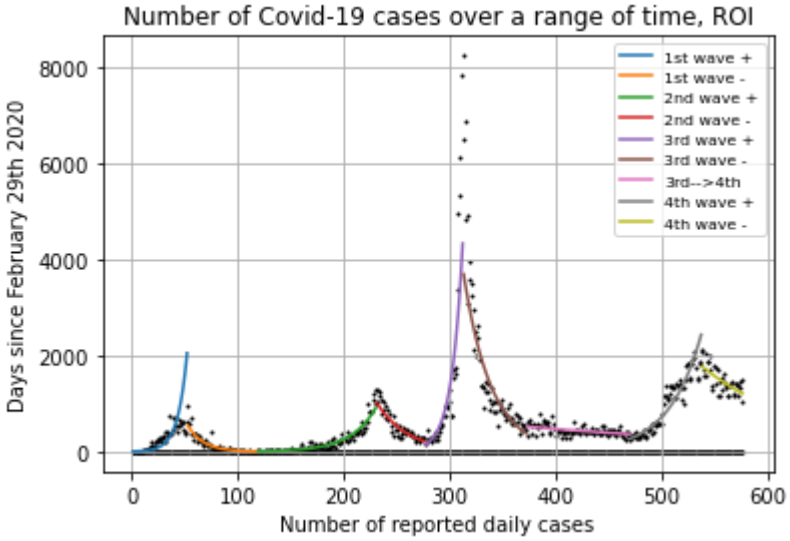


In [345…

```
exponentialfit9 = no9*np.e**((slope9)*(days[537:576] - 537))
plt.scatter(days[537:576], cases[537:576])
plt.title("Exponential fit of 4th wave -")
plt.plot(days[537:576], exponentialfit9)
plt.show()
```



In [352…
```
plt.scatter(days, cases, color = 'black', s = 1.1)
plt.plot(days[0:52], exponentialfit1, label='1st wave +')
plt.plot(days[52:118], exponentialfit2, label='1st wave -')
plt.plot(days[118:230], exponentialfit3, label='2nd wave +')
plt.plot(days[230:276], exponentialfit4, label='2nd wave -')
plt.plot(days[276:312], exponentialfit5, label='3rd wave +')
plt.plot(days[312:372], exponentialfit6, label='3rd wave -')
plt.plot(days[372:469], exponentialfit7, label='3rd-->4th')
plt.plot(days[469:537], exponentialfit8, label='4th wave +')
plt.plot(days[537:576], exponentialfit9, label='4th wave -')
plt.scatter(days[0:576], np.log(cases[0:576]), color ='black', s = 1.2)
plt.title("Number of Covid-19 cases over a range of time, ROI")
plt.xlabel("Number of reported daily cases")
plt.ylabel("Days since February 29th 2020")
plt.legend(loc=1, fontsize = 7.8)
plt.grid()
plt.show()

#why are we plotting this graph, mention what its telling us
#what type of wave could be applied to 4th wave, possible sign wave
#do a sin fitting below
#mention polynomial of odd degree, multiplicity etc
#although in the case I have made linear fit works quite well
```



In [10]:
```
plt.scatter(days[450:576], cases[450:576])
plt.plot()
plt.show()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-10-9346558924de> in <module>
----> 1 plt.scatter(days[450:576], cases[450:576])
      2 plt.plot()
      3 plt.show()

NameError: name 'plt' is not defined
```

In [ ]: