Analysis of Cumulative COVID-19 Cases,Hospitalizations, and Deaths in NYC

04/01/2022

Xiao Ma(xm2276), Yida Wang(yw3774), Ruiqi Yan(ry2417), Hao Zheng(hz2770), Peilin Zhou(pz2281)

Introduction

Objectives The rapid spreading of COVID-19 has been a serious threat to global health, economic, and the social systems for the last two years and its impact is still ongoing. Studying the current status and using a model to mimic the growth of confirmed cases, deaths, and hospitalizations of COVID-19 are unquestionably necessary for predicting the future dynamics of this virus and planning for strategic responses. This project focuses on the cumulative confirmed COVID-19 cases, deaths, and hospitalization counts in each borough of New York City since February 29th, 2020, which is the date of the start of outbreak in NYC. The objectives of the project include applying the widely-used Richard growth function to model the cumulative counts as different pandemic waves for each borough. To obtain estimates of the parameters of the growth function that capture the real patterns, an optimization algorithm based on Newton-Raphson is developed. Then, a comparison is made between the fitted curves of each pandemic wave and borough and the observations based on the data.

Data Description Data used for this project is obtained from NYC open data. It contains

the daily counts of confirmed COVID-19 cases, deaths, and hospitalizations in Bronx, Brooklyn, Manhattan, Queens, and Staten Island from February 29th, 2020 to February 24th 2022. For the purpose of this project, the cumulative counts are calculated by summing up the daily observations. 4 pandemic waves are identified by plotting the daily counts by day as a direct indication as shown in Fig 1. A clear pattern of the trajectory in time is observed. The cutting point of each wave is inferred by the declination of each wave. The name of each wave is given based on the variant detected at that approximate time period. Therefore, 4 pandemic waves are named as alpha, beta, delta, and omicron. Alpha wave starts from February 29th, 2020 to August 1st, 2020; beta wave starts from August 2nd 2020 to June 25th, 2021; delta wave starts from June 26th, 2021 to November 21st, 2021; omicron wave starts from November 22nd, 2021 to the latest date in the data, February 24th, 2022.

Model and Optimization Algorithms

Richard Growth Function The Richard Growth Curve Model, also known as the generalized logistic curve, is a four-parameter, nonlinear S-shaped function and its utility is the ability to describe a variety of growing processes. Let N(t) be a population at time t, the function takes the form

$$N(t) = \frac{a}{\{1 + d\exp\{-k(t - t_0)\}\}^{1/d}}.$$

The four parameters in the model include a, k, d, and t0. In epidemiological modeling, the Richard curve can be understood as describing the infection trajectories in time. In terms of modeling the outbreak of COVID-19, N(t) represents the cumulative counts of cases at time t; t represents the time since infection takes place; a is the maximum N(t) it can reach (the maximum cumulative counts across time); d is a shape parameter which controls the flexibility of the curve; k is the slope of inflection, in other words, the infection rate; t0 is the time at an inflection where the curve changes from convex to concave.

Newton-Raphson is a multivariate generalization of Newton's method, which focuses on

improving a function of one variable interactively. First, set initial point $\boldsymbol{\theta}_0$, evaluate the approximated gradient $\bigtriangledown f(\theta)$ using

$$\bigtriangledown f(\boldsymbol{\theta}_0) + \bigtriangledown^2 f(\boldsymbol{\theta}_0)(\boldsymbol{\theta} - \boldsymbol{\theta}_0)$$

Next by setting this equation to 0, we can solve the solution as the next point. Then iterate until convergence for the sequence of θ is achieved. The ith $\boldsymbol{\theta}$ is

$$\boldsymbol{\theta}_i = \boldsymbol{\theta}_{i-1} - [\bigtriangledown^2 f(\boldsymbol{\theta}_{i-1})]^{-1} \bigtriangledown f(\boldsymbol{\theta}_{i-1})$$

Then, we modified Newton-Raphson by including a step-halving method and redirection. Step-halving is to cut the potential step (λ) in half until non-decreasing log-likelihood is reached in each iteration; Redirection is to check the Hessian matrix is negative definite at each step, if not, replace it with a similar negative definite matrix such as $\bigtriangledown^2 f(\boldsymbol{\theta}_{i-1}) - \gamma I$, with γ being sufficiently large to generate negative definite matrix.

Broyden-Fletcher-Goldfarb-Shanno Method Sometime it is frustrating to solve inverse of the matrix hessian matrix in high dimension Newton Raphson, so the method BFGS(Broyden-Fletcher-Goldfarb-Shanno Method) is introduced to approximate $\bigtriangledown^2 f(\boldsymbol{\theta}_i)^{-1}$. That is to update the surrogate B_i through the formula

$$B_i = (I - \frac{S_{i-1}Y_{i-1}^\intercal}{Y_{i-1}^\intercal S_{i-1}})B_{i-1}(I - \frac{Y_{i-1}S_{i-1}^\intercal}{Y_{i-1}^\intercal S_{i-1}}) + \frac{S_{i-1}S_{i-1}^\intercal}{Y_{i-1}^\intercal S_{i-1}}$$

at each ith iteration, where $S_i = \boldsymbol{\theta}_i - \boldsymbol{\theta}_{i-1}$, $Y_i = \bigtriangledown f(\boldsymbol{\theta}_i) - \bigtriangledown f(\boldsymbol{\theta}_{i-1})$.

Assumptions In order to apply optimization algorithm to fit the cumulative confirmed cases, hospitalization and death of Covid-19 with Richard growth model, we assume that the cumulative count Y_i of each wave is normally distributed with mean, $N(\boldsymbol{\theta}; t_i)$, and variance, σ^2. While $N(\boldsymbol{\theta}; t_i)$ is the Richard growth model in day t_i with parameter $\boldsymbol{\theta} = (a, k, d, t_0)$, θ is fixed within each wave and differs across waves. t_i is the number of days from the

beginning of each pandemic wave.

Given the normality assumption and the observed cumulative count Y_i, the log-likelihood function with respect to $\boldsymbol{\theta}$ is

$$-\frac{\sum_{i=1}^{n}(Y_i - N(\boldsymbol{\theta}; t_i))^2}{2\sigma^2} - n\log\left(\sqrt{2\pi\sigma^2}\right)$$

Assuming σ^2 is known and constant, $\boldsymbol{\theta}$ are independent of σ^2. Then, σ^2 could be ignored and $\boldsymbol{\theta}$ that maximizes the likelihood function is equivalent to the one maximizing $-\sum_{i=1}^{n}\left(Y_i - N(\boldsymbol{\theta}, t_i)\right)^2$.

Initial Value Before the optimization algorithm is implemented, it is necessary to have a good guess of the parameters that are close to the true values to initiate the iteration, an improper starting value could result in non-convergence, slow-convergence or wrong convergence. According to the geometric meanings and the mathematical relationship in the growth model, we develop a strategy, which utilizes observations to make the reasonable guess of a, k, d and t_0.

- The guess of a is the maximum of Y_i, since it stands for the upper bound of $N(\boldsymbol{\theta}, t_i)$. t_0 is the day in inflection, so the growth is fastest and the number of daily cases is highest in t_0 day. The initial value of t_0 should be the day that has maximum daily cases.

- When $t_i = t_0$,

$$N(t_0) = \frac{a}{(1+d)^{1/d}} \Rightarrow 1 + d - \left(\frac{a}{N(t_0)}\right)^d = 0$$

  The most reasonable estimation of $N(t_0)$ is cumulative count on t_0, Y_0. Then, we apply the bi-section algorithm with the initial guess of a and t_0 to find the root of $1 + d - \left(\frac{a}{Y_0}\right)^d$ as the initial d. Since d is commonly between 0 and 1 but cannot be 0, the default starting interval of bi-section algorithm is 0.1 and 1. If the function in lower and upper bound have the same sign, the algorithm returns the lower bound; if the default interval could not produce the good guess of d and cause the non-convergence

in iterations, different intervals could be used to find a better guess for d.

- As a, t_0 and d are known, the growth model could be transformed into a linear model:

$$\log\left(\frac{\left(\frac{a}{N(t_i)}\right)^d - 1}{d}\right) = -kt_i + kt_0$$

The slope of this model is $-k$. After replacing $N(t_i)$ with observed Y_i and fitting a linear regression between t_i and the logarithm on the left, the opposite of the slope is our initial guess of k. Only observed Y_i smaller than initial a is used, otherwise, the logarithm will be undefined.

Implementation The optimization is only implemented on the nonzero cumulative observations. The modified Newton-Raphson algorithm in the dimension of 4 is utilized. Moreover, a matrix B_i calculated by the BFGS method surrogates the inverse of the Hessian matrix to avoid the computational burden of solving inverse and improve the efficiency. For convenience, the negative identity matrix is the initial surrogate matrix.
In each iteration, the algorithm checks whether the surrogate matrix is negative definite and performs redirection if not. Then, we apply newton-raphson with step-halving to ensure that log-likelihood is non-decreasing with updated $\boldsymbol{\theta}_{i+1}$. Lastly, the surrogate matrix is updated through the BFGS formula before the next iteration.

Results

Confirmed Case By observing the comparison plot (Fig 2) between the fitted curve and the actual cases, the predicted curves are pretty good. The predicted curve and the cumulated confirmed cases curve are almost overlapping. Alpha, beta and delta have relatively flattened cumulative case curves, Beta and omicron have a better "S-shape." Beta and Omicron have rapidly increased in cumulative cases. One of the reasons is the weather, winter facilitates the virus to spread. More importantly, omicron moves fast, it spreads

swiftly through populations, and infections develop quickly in individuals. Although delta has shorter incubation on average, most of the public are well-prepared. People wear masks and some of them already get the vaccine.

Between boroughs we find that each borough shares similar trends of cumulative cases. In alpha, Bronx has the highest cumulative cases. In beta, delta and omicron, Brooklyn has the highest cumulative cases.

From the estimate parameters table (Table 1), as mentioned above, a is the largest population could reach. k is the growth rate. d is the shape parameter and t_0 is the time at incubation. Among four waves, omicron has the largest upper bound of people who could be infected and the largest growth rate. The growth rate of the delta and omicron are very close. Regardless of boroughs, the alpha wave has the smallest inflection point so it reached the peak of daily counts earliest; the inflection of the beta wave came the latest. Among boroughs in alpha, although Staten Island has the lowest upper bound of confirmed cases, it has the highest growth rate. In the beta, delta, and omicron wave, Bronx has the highest growth rate; Brooklyn has lowest growth rate in alpha and delta, Staten Island has lowest growth rate in beta and Manhattan has lowest growth rate in omicron. Brooklyn always has the highest upper bond since it has the largest population among boroughs (The Bronx 1,472,654; Brooklyn 2,736,074; Manhattan 1,694,251; Queens 2,405,464; Staten Island 495,747 according to census 2020).

Deaths For the cumulative deaths, we can see from the daily death count (Fig 3) that the variant alpha caused more deaths than the other 3 variants, and Brooklyn and Queen have more deaths than the other boroughs. For the cumulative death count (Fig 4), the variant alpha has a clear S-shape, meaning it has a steeper peak than the other three. On the contrary, you can merely see the s-shape for delta. For the same wave, borough Bronx, Brooklyn and Queens have higher cumulative death counts. For the same variant, the waves seem to be quite similar across different boroughs.

The fitted fatality curves of each pandemic wave and borough (red lines) based on the estimated parameters of the Richard's function show good agreement with the actual patterns as illustrated in Fig 5. As each pandemic wave has leveled off, the fitted fatality curve is well-adjusted correspondingly. From the plots, it is also observed that the inflection point where the curve changes from convex to concave comes rather quickly for the alpha wave comparatively, meaning that the growth rate of death comes to decay in this wave.

Estimates of the parameters in Richard's function for cumulative death counts are listed in table 2. The growth rate of death, k, is higher in the Bronx in the first three waves. This might be due to the insufficient access to health care facilities. Also, the growth rate is highest in the last wave across boroughs. The inflection is estimated to happen more rapidly in the alpha wave and relatively slower in the beta wave since the value of t0 indicates the inflection point. The counts of death are uncomparable across boroughs because of the varied population size. However, by alternating the count to rate, it is possible to draw some insights from comparison. The estimated maximum case fatality rate is calculated by the estimated value of maximum death counts over the maximum confirmed case counts, which is the parameter a in the Richard curve. As shown in Table 3, the omicron wave is estimated to have a relatively lowest case fatality rate and Manhattan has the lowest value among the boroughs during the omicron wave. The estimates are highest for the alpha wave and this is reasonable since it is the beginning of the outbreak when the severity of the virus is unaware of and the necessary medical supports are inadequate. In contrast, as the variant changes to omicron, a significant decrease is observed across boroughs.

Hospitalization In this part, we divided the date into 4 curves(alpha, beta, delta, omicron) using the same date points in the death and case part. According to the daily hospitalized count plot(Fig 1), we can observe that the alpha and delta have the highest number of hospitalizations which is more than 50000, while delta has the lowest number of that,only 10000. Although the omicron has the second lowest peak in total number, it grows very

sharply. The daily hospitalized counts are also divided into each borough(Fig 6). From the panel plots, it's clear to see that the curves look very similar across each borough.

The detailed parameter estimates table for hospitalized cases are shown in Table 4. k indicates the increase rate of hospitalized cases, a is the peak for the number of hospitalized cases and t_0 is the inflection point of the curves. As the table showed, the omicron period has the highest increase rate instead of the alpha period. Although the beta and delta periods' waves seem differently in the figure, they have similar increase rates (k values). Staten Island has the lowest increase rate except in alpha, in which wave Brooklyn has the lowest increase rate while Bronx always has the largest or the second largest increase rate. Brooklyn has the largest or the second largest peak in all four periods. Staten Island is the first one to reach infection point in the delta period, which means it has the shortest time to have the largest number of cases. Comparing between the waves, we can find that the alpha reaches the inflection point very fast while the beta has the longest days to reach the peak.

Similar to what has been done in analyzing the death cases, the case hospitalization rate is calculated using the same formula in case fatality rate. We can get the new table of adjusted a (table 5) which indicates the proportion of people hospitalized among those who got covid-19. We can observe that the rate of hospitalization has a decreasing trend from alpha wave to omicron wave and the omicron wave is almost 0.1 times of the rate of alpha wave. Bronx has the highest case hospitalization rate in the omicron period (0.0500) among all the boroughs, while NYC only for 0.0323 at the same time. In the alpha wave, Manhattan draw the highest case hospitalization rate (0.2681). Meanwhile, Staten Island only for 0.1625.

Based on the fit graphs (Fig 7), we can see our model fits well, even though there is a very slight difference between the real one in Brooklyn during the alpha period.

Conclusion

Findings In conclusion, the four waves behave quite differently and represent the four different variants for COVID-19: alpha, beta, delta and omicron. We use the modified Newton-Raphson algorithm with step-halving, redirection and BFGS method to fit different curves for different boroughs. For confirmed cases, alpha have a longer incubation period while omicron has a high peak and a sharply increase; For hospitalization, alpha, beta both have a high peak, omicron have a low peak but a sharply increase; For deaths, inflection happens more quickly for the alpha wave across boroughs and the beta wave has longer lag phase relatively. The growth rate of death counts is highest in the last wave but its maximum case fatality rate is the lowest across boroughs and other waves. In general, the modified Newton-Raphson algorithm performs reasonably well in fitting the COVID-19 data with the Richard growth model.

Limitation One disadvantage of the algorithm is that it is very sensitive to the selection of starting values, especially the shape parameter d. We could make reasonable and close guesses of a, k and t_0, but for the initial guess of d, a slight deviation from the true value could easily result in a long execution time or non-convergence. In the future, we might need to modify our strategy about guessing the shape parameter, d. In addition, our parameter estimation is limited to our current data of COVID-19 as as in-sample analysis. It is implemented to help us understand the mathematical framework of Richard Curve and its application in epidemiological modeling, but the model may not be predictive of further epidemic change in COVID-19 as the out-sample validation is not performed for this model.

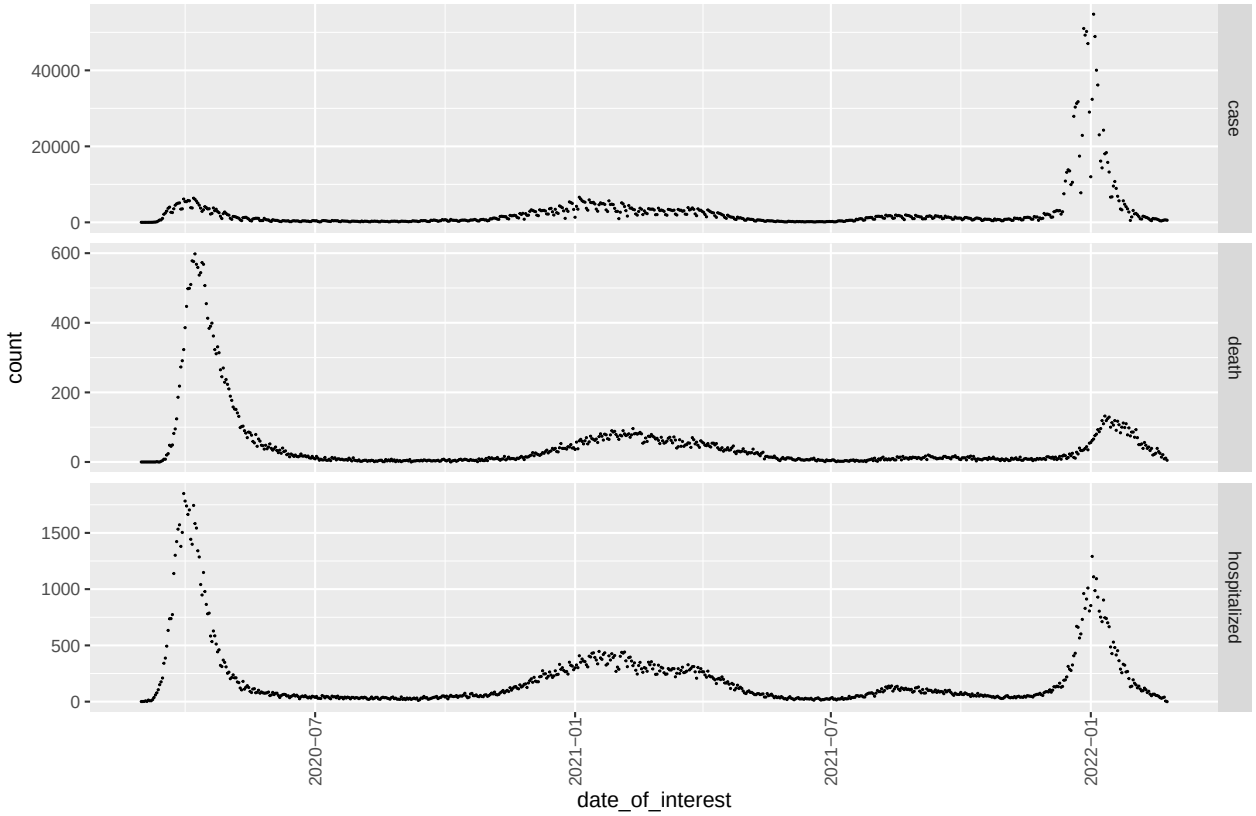
Contribution:

Xiao Ma responsible for confirmed cases comparison and the corresponding part for PPT and report. Yida Wang finished the data of the hospitalization part including code, PPT and report. Ruiqi Yan developed the strategy of initial guess and implementation of optimization algorithms in code, report and presentation, fitted the confirmed cases using

the algorithm and generated the table and figure for confirmed cases part. Hao Zheng fitted the curves for the cumulative death count of Bronx, Brooklyn and Manhattan, explaining the algorithms used, generalized description of death curve and conclusion. Peilin Zhou was responsible for fitting fatality curves for Queens and Staten Island, generating comparison plots and estimate table for the cumulative death counts, and writing introduction, richard's curve, and the result of comparison in the death section of the report.

Appendix I Figures and Tables

Fig.1 Trajectories of Daily Counts in Time



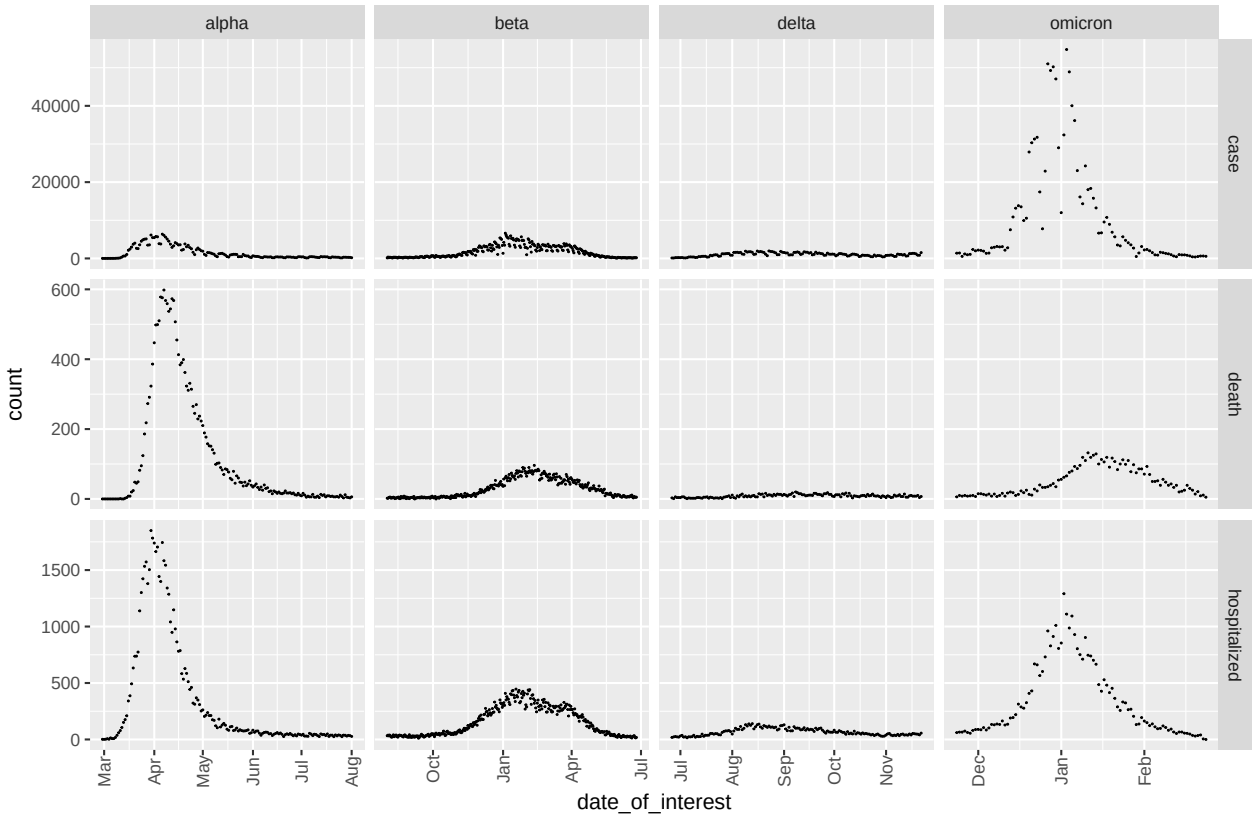


Table 1: Estimates of Parameters for Cumulative Case

	alpha				beta				delta				omicron			
Borough	**a**	**k**	**d**	**t0**	**a**	**k**	**d**	**t0**	**a**	**k**	**d**	**t0**	**a**	**k**	**d**	**t0**
NYC	214896	0.056	-0.168	33	584029	0.026	0.732	176	173483	0.019	-0.335	57	985059	0.145	0.791	38
Bronx	48438	0.060	-0.19	29	103621	0.030	0.884	178	21088	0.027	-0.188	56	162944	0.173	0.798	39
Brooklyn	60578	0.039	-0.404	24	177569	0.026	0.848	179	65847	0.017	-0.361	62	298991	0.135	0.701	38
Manhattan	27379	0.043	-0.278	31	83873	0.027	0.861	176	32659	0.019	-0.352	56	188762	0.115	0.496	35
Queens	66060	0.056	-0.188	31	168444	0.026	0.629	176	39717	0.021	-0.32	56	270766	0.155	0.831	39
Staten Island	14119	0.066	-0.192	26	50936	0.019	0.117	158	15265	0.021	-0.26	57	64203	0.151	0.942	39

Fig.2 Cumulative Case: Fitted Curves vs Actual Patterns

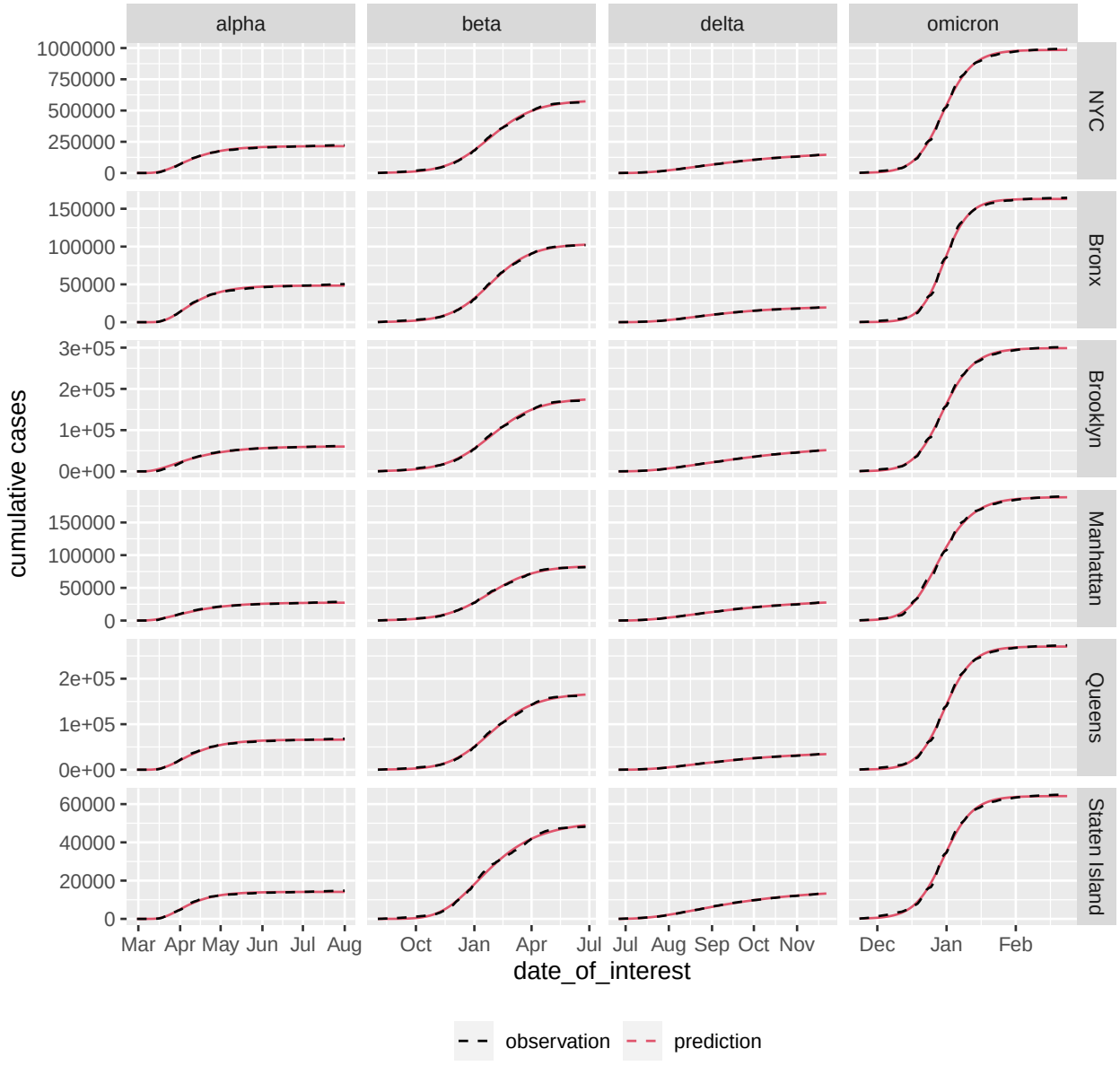


Fig.3 Daily Death Count in Each Borough

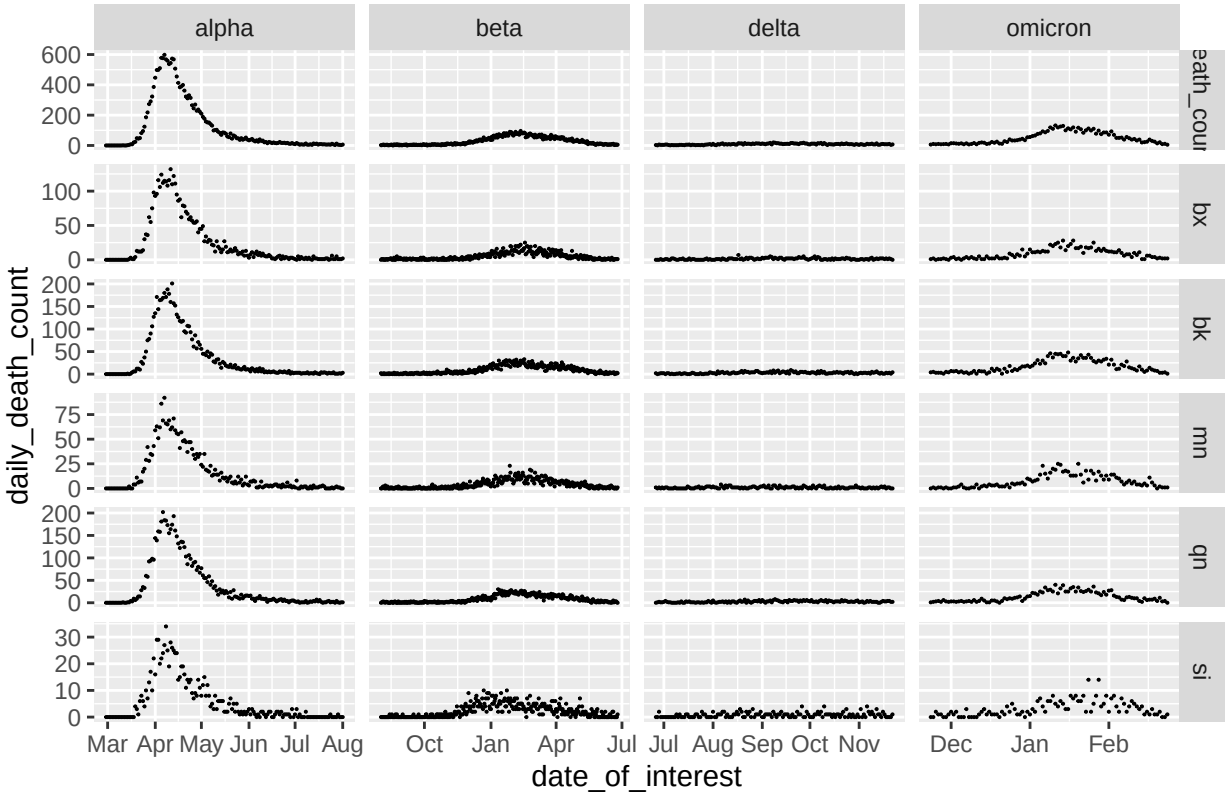


Fig.4 Cumulative Death Count in Each Borough

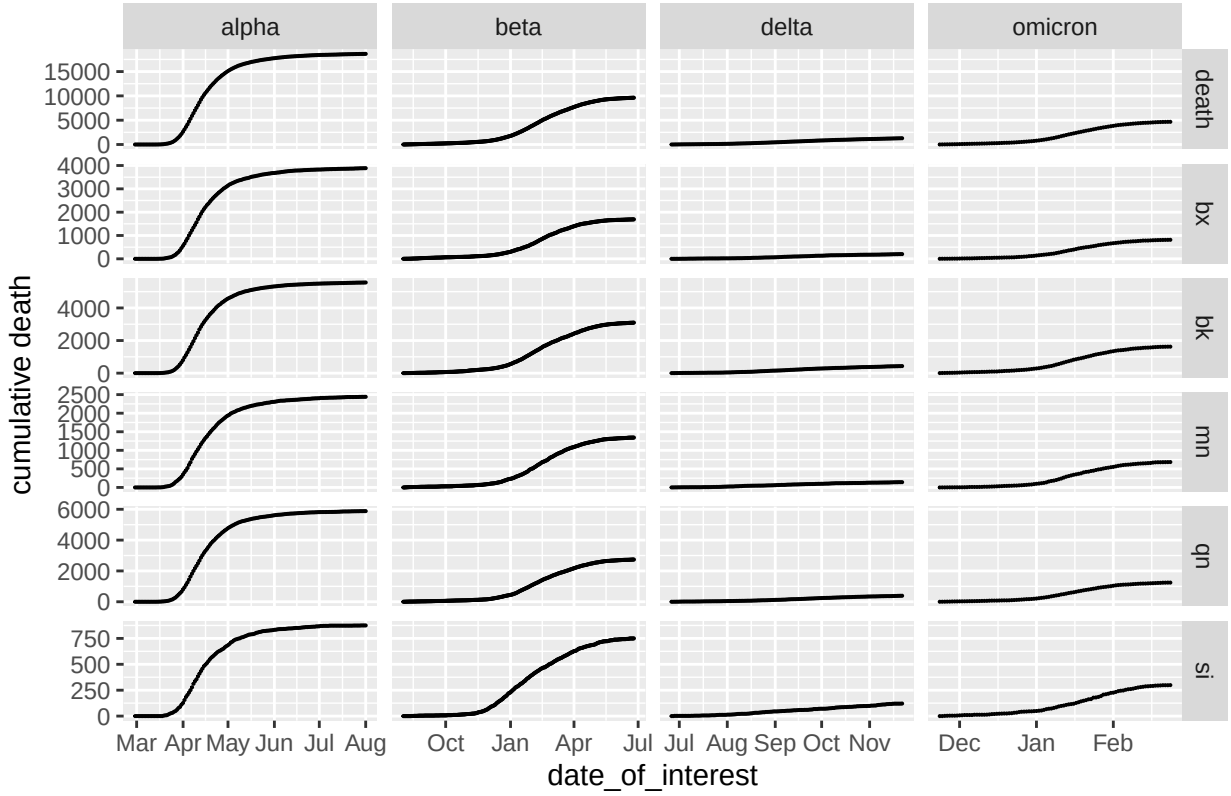


Table 2: Estimates of Parameters for Cumulative Death Counts

	alpha				beta				delta				omicron			
Borough	**a**	**k**	**d**	**t0**	**a**	**k**	**d**	**t0**	**a**	**k**	**d**	**t0**	**a**	**k**	**d**	**t0**
NYC	18621	0.064	-0.1807	28	9828	0.030	0.90261	198	1492	0.026	0.2317	80	4753	0.103	1.1999	57
Bronx	3822	0.069	-0.1956	24	1694	0.040	1.59619	204	209	0.038	0.6073	77	830	0.103	1.1574	57
Brooklyn	5473	0.075	-0.1621	25	3187	0.028	0.84765	199	488	0.027	0.1885	78	1650	0.106	1.3198	57
Manhattan	2438	0.057	-0.2906	23	1358	0.034	1.05481	198	171	0.019	-0.2214	62	704	0.091	0.7591	55
Queens	5877	0.060	-0.2125	27	2810	0.029	0.72999	195	436	0.033	0.6833	88	1255	0.108	1.2788	57
Staten_Island	871	0.058	-0.3659	18	794	0.018	-0.06518	151	284	0.007	-0.5393	92	312	0.109	1.7555	61

Fig.5 Fitted Curves vs Actual Patterns

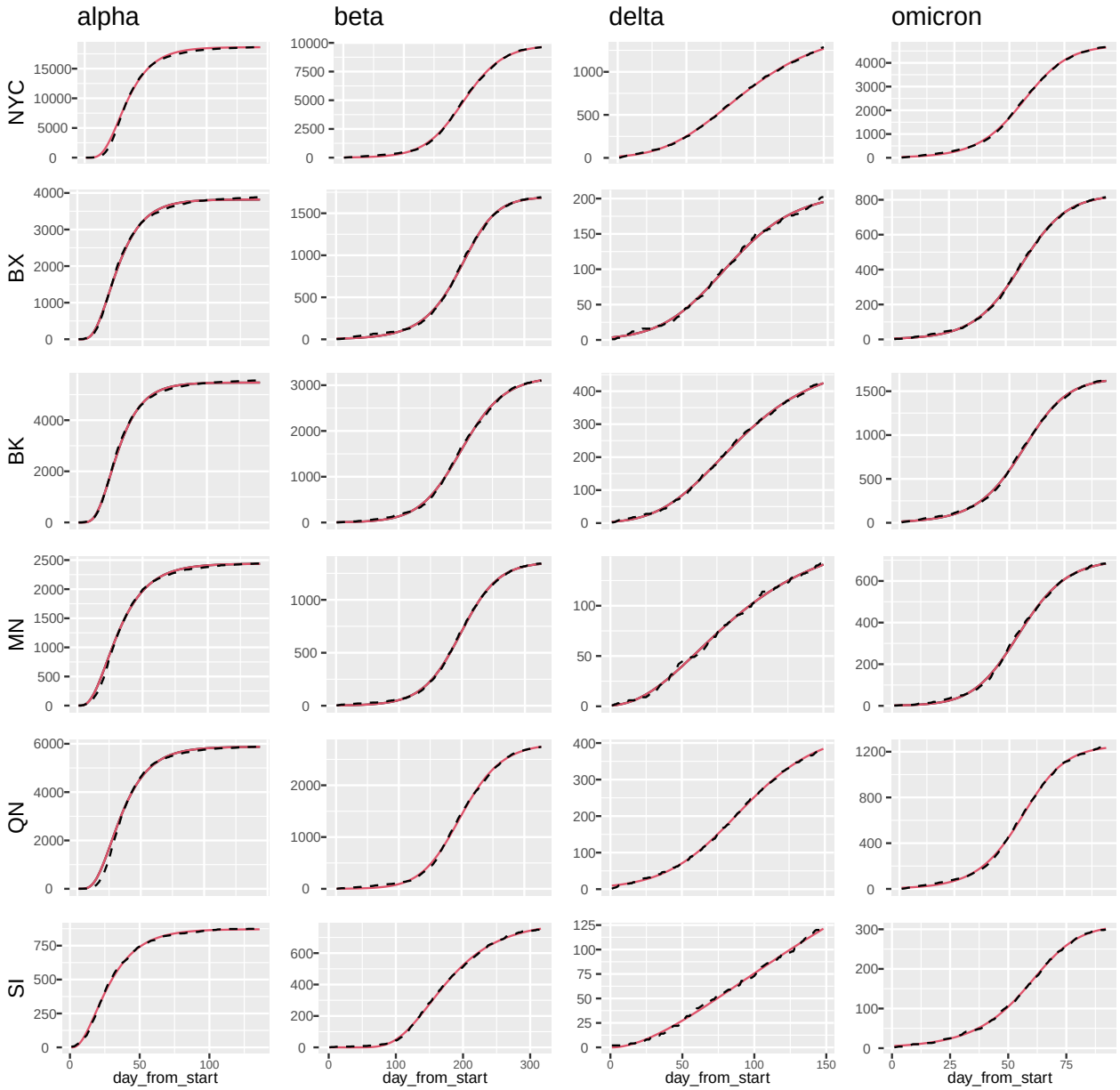


Table 3: Maximum Case Fatality Rate

	Case Fatality Rate			
Borough	alpha	beta	delta	omicron
Bronx	0.0789	0.0163	0.0099	0.0051
Brooklyn	0.0903	0.0179	0.0074	0.0055
Manhattan	0.0890	0.0162	0.0052	0.0037
Queens	0.0890	0.0167	0.0110	0.0046
Staten Island	0.0617	0.0156	0.0186	0.0049

Fig.6 Cumulative Hospitalization Count in Each Borough

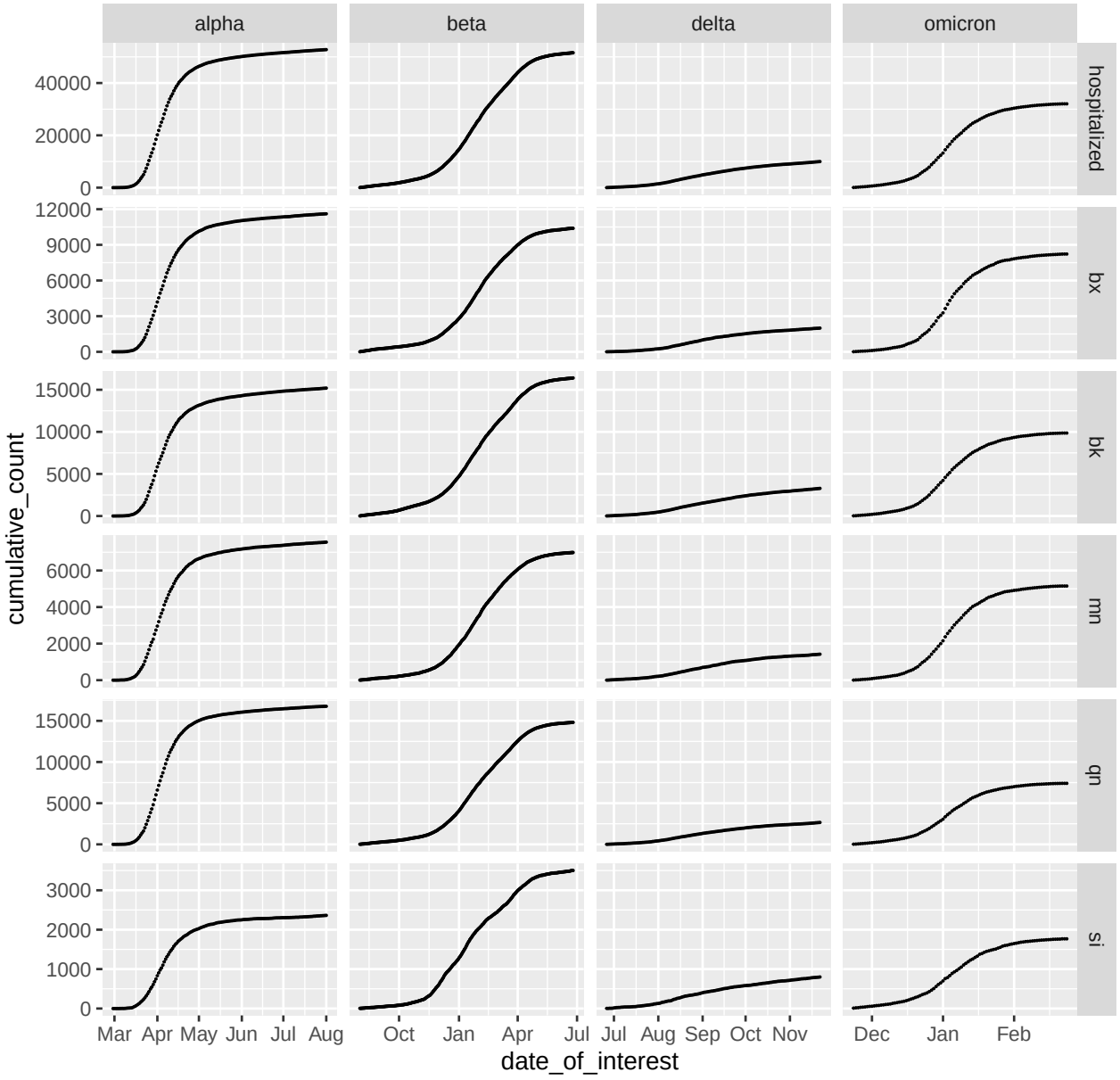


Table 4: Estimates of Parameters for Cumulative Hospitalization

	alpha				beta				delta				omicron			
Borough	**a**	**k**	**d**	**t0**	**a**	**k**	**d**	**t0**	**a**	**k**	**d**	**t0**	**a**	**k**	**d**	**t0**
NYC	51156	0.076	-0.104	31	52815	0.029	1.003	186	10660	0.027	-0.082	58	31865	0.109	0.764	42
Bronx	11295	0.077	-0.096	31	10481	0.034	1.281	189	2061	0.031	-0.032	59	8154	0.115	0.671	41
Brooklyn	14534	0.091	0.203	34	16836	0.028	1.125	189	3631	0.024	-0.129	60	9824	0.104	0.671	41
Manhattan	7347	0.075	-0.133	28	7088	0.031	1.002	185	1483	0.032	0.129	61	5124	0.113	0.748	42
Queens	15999	0.088	-0.1	27	15295	0.027	0.887	186	2832	0.026	-0.103	57	7372	0.111	0.972	43
Staten Island	2294	0.070	-0.164	27	3738	0.018	0.093	159	901	0.020	-0.304	53	1768	0.101	0.992	44

Fig.7 Fitted Curves vs Actual Patterns

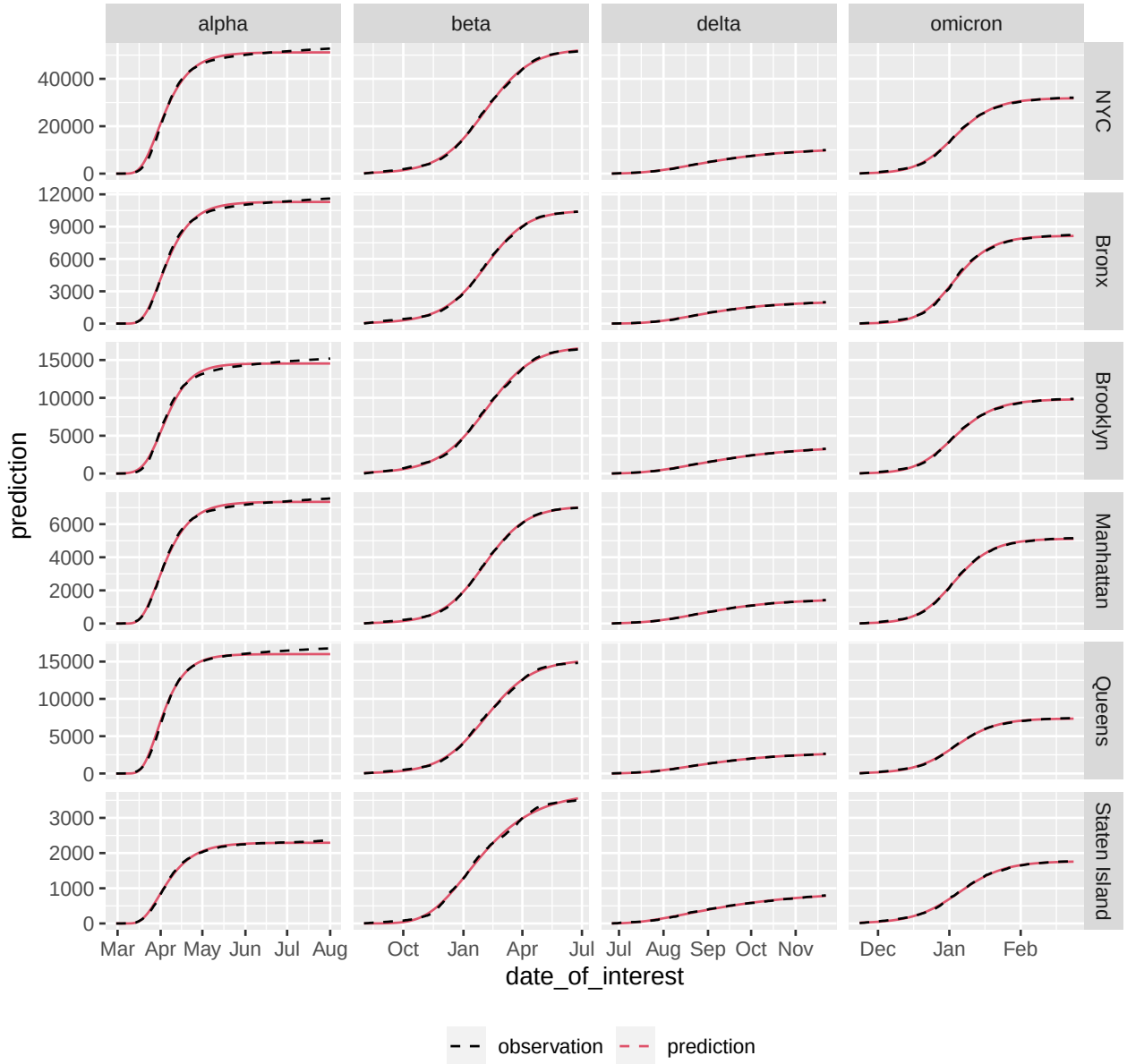


Table 5: Maximum Case Hospitalization Rate

	alpha	beta	delta	omicron
NYC	0.2382	0.0902	0.0615	0.0323
Bronx	0.2303	0.1045	0.0977	0.0500
Brooklyn	0.2475	0.0945	0.0558	0.0329
Manhattan	0.2681	0.0843	0.0456	0.0271
Queens	0.2471	0.0905	0.0708	0.0275
Staten_Island	0.1625	0.0731	0.0592	0.0275

Appendix II: R code

Overview

```
day_by_day = read_csv("data-by-day.csv") %>%
  janitor::clean_names() %>%
  mutate(date_of_interest = as.Date(date_of_interest, "%m/%d/%y"))


p1 = day_by_day %>%  pivot_longer(case_count:death_count,
                names_to = "type",
                values_to = "count",
                names_prefix = "_count") %>%
  mutate(type = str_remove(type, "_count")) %>%
  ggplot(aes(x = date_of_interest, y = count))+
  geom_point(cex = 0.1) +
  facet_grid(type~.,
             scales = "free") +
  theme(axis.text.x = element_text(angle = 90))



 day_by_day = day_by_day %>% mutate(
         wave = case_when(date_of_interest <= as.Date("08/01/20", "%m/%d/%y") ~ "alpha",
                          date_of_interest > as.Date("08/01/20", "%m/%d/%y") &
                          date_of_interest <= as.Date("06/25/21", "%m/%d/%y") ~ "beta",
                          date_of_interest > as.Date("06/25/21", "%m/%d/%y") &
                          date_of_interest <= as.Date("11/22/21", "%m/%d/%y") ~ "delta",
                          date_of_interest > as.Date("11/22/21", "%m/%d/%y") ~ "omicron"
         wave = factor(wave))
```

```
p2 = day_by_day %>%
  select(date_of_interest:death_count, wave) %>%
  pivot_longer(case_count:death_count, names_to = "type",
               values_to = "count") %>%
  mutate(type = str_remove(type, "_count")) %>%
  ggplot(aes(x = date_of_interest, y = count)) +
  geom_point(cex = 0.05) +
  facet_grid(type~wave,
             scales = "free") +
  theme(axis.text.x = element_text(angle = 90))

grid.arrange(p1, p2, nrow = 2, top = "Fig.1 Trajectories of Daily Counts in Time")

day_by_day = day_by_day %>% group_by(wave) %>%
  mutate(cum_case = cumsum(case_count),
         cum_death = cumsum(death_count),
         cum_hospitalized = cumsum(hospitalized_count))
```

Task 1.1

```
f_0 <- function(x, c){1+x-c^x}
find_zero <- function(tol, lower, upper, constant){
  a = lower
  b = upper
  c = (a+b)/2
  if(f_0(a,constant)*f_0(b,constant)>0){
```

```
    return(a)
  }
  while(abs(f_0(c, constant)) > tol){
    if(f_0(c, constant)*f_0(a, constant) > 0){
      a = c
    } else {
      b = c
    }
    c = (a+b)/2
  }
  return(c)
}


initial_guess <- function(index_max, dat, tol, lower = 0.1, upper = 1){
  a0 = max(dat[,1])
  d0 = find_zero(tol, lower, upper, constant = a0/dat[index_max,1])


  y = log(((a0/dat[(dat[,1]!=a0),1])^d0-1)/d0)
  x = dat[(dat[,1]!=a0),2]
  k0 = -coef(lm(y~x))[2]
  t0 = dat[index_max, 2]
  return(c(a0,k0, d0, t0))
}
```

Task 1.2

```
growth_f = function(theta, ti) {
  a = theta[1]
  k = theta[2]
  d = theta[3]
  t0 = theta[4]
  a*(1+d*exp(-k*(ti-t0)))^(-1/d)
}



logliklihood = function(theta, dat){
  yi = dat[,1]
  growth = growth_f(theta, dat[,2])
  return(-sum((yi-growth)^2))
}


gradient = function(theta, dat){
  a = theta[1]
  k = theta[2]
  d = theta[3]
  t0 = theta[4]
  ti = dat[,2]
  yi = dat[,1]
  growth = growth_f(theta,ti)
  exp_f = exp(k*(ti-t0))
  b = 1/(exp_f+d)
```

```
  dbk = -b^2*(ti-t0)*exp_f
  da = sum(2*(yi-growth)*(1+d*exp_f^(-1))^(-1/d))
  dk = sum(2*(yi-growth)*growth*(ti-t0)*b)
  dd = sum(2*(yi-growth)*growth*(log(1+d*exp_f^(-1))-b*d)/d^2)
  dt0 = sum(2*(yi-growth)*-growth*k*b)

  return(c(da,dk,dd,dt0))
}



search_theta_2 = function(start, tol, dat, maxiter) {
  cur = start
  I = diag(4)
  H_cur = -I
  i = 0
  loglik = logliklihood(cur, dat)
  prevlik = loglik - tol-2
  grad_cur = gradient(cur, dat)
  while(tol < abs(loglik-prevlik) && i < maxiter){
    gamma = max(eigen(H_cur)$values, 0)
    H_mod = H_cur - gamma*diag(4)
    lambda = 1
    theta = cur - H_mod %*% grad_cur
    theta_lik = logliklihood(theta, dat)
    while(is.na(theta_lik)){
      lambda = lambda/2
```

```
      theta = cur - lambda*H_mod %*% grad_cur
      theta_lik = logliklihood(theta, dat)
    }
    while(theta_lik < loglik){
      lambda = lambda/2
      theta = cur - lambda*H_mod %*% grad_cur
      theta_lik = logliklihood(theta, dat)
      while(is.na(theta_lik)){
        lambda = lambda/2
        theta = cur - lambda*H_mod %*% grad_cur
        theta_lik = logliklihood(theta, dat)
      }
    }
    prev = cur
    cur = theta
    S_i = cur-prev
    grad_prev = grad_cur
    grad_cur = gradient(cur, dat)
    Y_i = grad_cur - grad_prev
    scaler = as.vector(t(Y_i)%*%S_i)
    H_cur = (I-S_i%*%t(Y_i)/scaler)%*%H_cur%*%(I-Y_i%*%t(S_i)/scaler)+S_i%*%t(S_i)/scale
    prevlik = loglik
    loglik = logliklihood(cur, dat)
    i = i+1
  }
  return(c(cur, i))
}
```

Confirmed Cases

```
day_by_day_case = day_by_day %>% select(date_of_interest, wave, case_count,
                      bx_case_count, bk_case_count, mn_case_count,
                      qn_case_count, si_case_count)


day_by_day_case = day_by_day_case %>%
  group_by(wave) %>%
  mutate(cum_case = cumsum(case_count),
         cum_bx = cumsum(bx_case_count),
         cum_bk = cumsum(bk_case_count),
         cum_mn = cumsum(mn_case_count),
         cum_qn = cumsum(qn_case_count),
         cum_si = cumsum(si_case_count))


day_by_day_case %>%
  pivot_longer(cum_case:cum_si, names_to = "region", names_prefix = "cum_",
               values_to = "cumulative_count") %>%
  mutate(
    region = recode(region, case = "NYC", bx = "Bronx", bk = "Brooklyn", mn = "Manhattan
    region = factor(region, levels = c("NYC", "Bronx", "Brooklyn", "Manhattan", "Queens"
  ggplot(aes(x = date_of_interest, y = cumulative_count, col=wave)) +
  geom_point(cex = 0.05)+
  facet_grid(region~wave,
             scales = "free")
```

NYC

```
cum_case_prediction =
  day_by_day_case %>%
  filter(cum_case != 0) %>%
  group_by(wave) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(wave, case_count, cum_case, date_from_start, date_of_interest) %>%
  nest(data = c("case_count", "cum_case", "date_from_start", "date_of_interest")) %>%
  ungroup() %>%
  mutate(
    index_max = map_dbl(.x = data, ~which.max(.x$case_count)),
    dat = map(.x = data, ~cbind(.x$cum_case, .x$date_from_start)),
    start = map2(.x = index_max, .y = dat, ~initial_guess(.x, .y, tol, lower = 0.0001)),
    theta = map2(.x = start, .y = dat, ~search_theta_2(.x, tol, .y, 1000000)),
    prediction_case = map2(.x = theta, .y = dat, ~growth_f(.x, .y[,2]))) %>%
  select(wave, data, prediction_case, theta)
```

```
prediction_case = cum_case_prediction %>% unnest(cols = c(data, prediction_case)) %>%
  select(date_of_interest, prediction_case)


case_theta = cum_case_prediction %>% select(wave, theta) %>%
  mutate(a = map_dbl(.x = theta, ~.x[1]),
         k = map_dbl(.x = theta, ~.x[2]),
         d = map_dbl(.x = theta, ~.x[3]),
         t0 = map_dbl(.x = theta, ~.x[4]),
         r = map_dbl(.x = theta, ~.x[5]),
         boro = "NYC") %>%
```

```
  select(-theta)


day_by_day_case_prediction_1 = day_by_day_case %>%
  left_join(prediction_case, by = "date_of_interest") %>%
  mutate(prediction_case = ifelse(is.na(prediction_case), 0, prediction_case))
```

Bronx

```
cum_case_bx_prediction =
  day_by_day_case %>%
  filter(cum_bx != 0) %>%
  group_by(wave) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(wave, bx_case_count, cum_bx, date_from_start, date_of_interest) %>%
  nest(data = c("bx_case_count", "cum_bx", "date_from_start", "date_of_interest")) %>%
  ungroup() %>%
  mutate(
    index_max = map_dbl(.x = data, ~which.max(.x$bx_case_count)),
    dat = map(.x = data, ~cbind(.x$cum_bx, .x$date_from_start)),
    start = map2(.x = index_max, .y = dat, ~initial_guess(.x, .y, tol)),
    theta = map2(.x = start, .y = dat, ~search_theta_2(.x, tol, .y, 1000000)),
    prediction_bx = map2(.x = theta, .y = dat, ~growth_f(.x, .y[,2])))
```

```
prediction_bx = cum_case_bx_prediction %>%
  unnest(cols = c(data, prediction_bx)) %>%
  select(date_of_interest, prediction_bx)


case_theta_bx = cum_case_bx_prediction %>% select(wave, theta) %>%
```

```
  mutate(a = map_dbl(.x = theta, ~.x[1]),
         k = map_dbl(.x = theta, ~.x[2]),
         d = map_dbl(.x = theta, ~.x[3]),
         t0 = map_dbl(.x = theta, ~.x[4]),
         r = map_dbl(.x = theta, ~.x[5]),
         boro = "Bronx") %>%
  select(-theta)


day_by_day_case_prediction_2 = day_by_day_case_prediction_1 %>%
  left_join(prediction_bx, by = "date_of_interest") %>%
  mutate(prediction_bx = ifelse(is.na(prediction_bx), 0, prediction_bx))
```

Brooklyn

```
cum_case_bk_prediction =
  day_by_day_case %>%
  filter(cum_bk != 0) %>%
  group_by(wave) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(wave, bk_case_count, cum_bk, date_from_start, date_of_interest) %>%
  nest(data = c("bk_case_count", "cum_bk", "date_from_start", "date_of_interest")) %>%
  ungroup() %>%
  mutate(
    index_max = map_dbl(.x = data, ~which.max(.x$bk_case_count)),
    dat = map(.x = data, ~cbind(.x$cum_bk, .x$date_from_start)),
    start = map2(.x = index_max, .y = dat, ~initial_guess(.x, .y, tol)),
    theta = map2(.x = start, .y = dat, ~search_theta_2(.x, tol, .y, 1000000)),
```

```
    prediction_bk = map2(.x = theta, .y = dat, ~growth_f(.x, .y[,2])))
```

```
prediction_bk = cum_case_bk_prediction %>%
  unnest(cols = c(data, prediction_bk)) %>%
  select(date_of_interest, prediction_bk)


case_theta_bk = cum_case_bk_prediction %>% select(wave, theta) %>%
  mutate(a = map_dbl(.x = theta, ~.x[1]),
         k = map_dbl(.x = theta, ~.x[2]),
         d = map_dbl(.x = theta, ~.x[3]),
         t0 = map_dbl(.x = theta, ~.x[4]),
         r = map_dbl(.x = theta, ~.x[5]),
         boro = "Brooklyn") %>%
  select(-theta)


day_by_day_case_prediction_3 = day_by_day_case_prediction_2 %>%
  left_join(prediction_bk, by = "date_of_interest") %>%
  mutate(prediction_bk = ifelse(is.na(prediction_bk), 0, prediction_bk))
```

Manhattan

```
cum_case_mn_prediction =
  day_by_day_case %>%
  filter(cum_mn != 0) %>%
  group_by(wave) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(wave, mn_case_count, cum_mn, date_from_start, date_of_interest) %>%
  nest(data = c("mn_case_count", "cum_mn", "date_from_start", "date_of_interest")) %>%
```

```
  ungroup() %>%
  mutate(
    index_max = map_dbl(.x = data, ~which.max(.x$mn_case_count)),
    dat = map(.x = data, ~cbind(.x$cum_mn, .x$date_from_start)),
    start = map2(.x = index_max, .y = dat, ~initial_guess(.x, .y, tol, lower = 1, upper
    theta = map2(.x = start, .y = dat, ~search_theta_2(.x, tol, .y, 1000000)),
    prediction_mn = map2(.x = theta, .y = dat, ~growth_f(.x, .y[,2])))
```

```
prediction_mn = cum_case_mn_prediction %>%
  unnest(cols = c(data, prediction_mn)) %>%
  select(date_of_interest, prediction_mn)


case_theta_mn = cum_case_mn_prediction %>% select(wave, theta) %>%
  mutate(a = map_dbl(.x = theta, ~.x[1]),
         k = map_dbl(.x = theta, ~.x[2]),
         d = map_dbl(.x = theta, ~.x[3]),
         t0 = map_dbl(.x = theta, ~.x[4]),
         r = map_dbl(.x = theta, ~.x[5]),
         boro = "Manhattan") %>%
  select(-theta)


day_by_day_case_prediction_4 = day_by_day_case_prediction_3 %>%
  left_join(prediction_mn, by = "date_of_interest") %>%
  mutate(prediction_mn = ifelse(is.na(prediction_mn), 0, prediction_mn))
```

Queens

```
cum_case_qn_prediction =
  day_by_day_case %>%
  filter(cum_qn != 0) %>%
  group_by(wave) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(wave, qn_case_count, cum_qn, date_from_start, date_of_interest) %>%
  nest(data = c("qn_case_count", "cum_qn", "date_from_start", "date_of_interest")) %>%
  ungroup() %>%
  mutate(
    index_max = map_dbl(.x = data, ~which.max(.x$qn_case_count)),
    dat = map(.x = data, ~cbind(.x$cum_qn, .x$date_from_start)),
    start = map2(.x = index_max, .y = dat, ~initial_guess(.x, .y, tol, lower = 1/1000000
    theta = map2(.x = start, .y = dat, ~search_theta_2(.x, tol, .y, 1000000)),
    prediction_qn = map2(.x = theta, .y = dat, ~growth_f(.x, .y[,2])))
```

```
prediction_qn = cum_case_qn_prediction %>%
  unnest(cols = c(data, prediction_qn)) %>%
  select(date_of_interest, prediction_qn)


case_theta_qn = cum_case_qn_prediction %>% select(wave, theta) %>%
  mutate(a = map_dbl(.x = theta, ~.x[1]),
         k = map_dbl(.x = theta, ~.x[2]),
         d = map_dbl(.x = theta, ~.x[3]),
         t0 = map_dbl(.x = theta, ~.x[4]),
         r = map_dbl(.x = theta, ~.x[5]),
         boro = "Queens") %>%
```

```
  select(-theta)


day_by_day_case_prediction_5 = day_by_day_case_prediction_4 %>%
  left_join(prediction_qn, by = "date_of_interest") %>%
  mutate(prediction_qn = ifelse(is.na(prediction_qn), 0, prediction_qn))
```

Staten Island

```
cum_case_si_prediction =
  day_by_day_case %>%
  filter(cum_si != 0) %>%
  group_by(wave) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(wave, si_case_count, cum_si, date_from_start, date_of_interest) %>%
  nest(data = c("si_case_count", "cum_si", "date_from_start", "date_of_interest")) %>%
  ungroup() %>%
  mutate(
    index_max = map_dbl(.x = data, ~which.max(.x$si_case_count)),
    dat = map(.x = data, ~cbind(.x$cum_si, .x$date_from_start)),
    start = map2(.x = index_max, .y = dat, ~initial_guess(.x, .y, tol, upper = 1.5)),
    theta = map2(.x = start, .y = dat, ~search_theta_2(.x, tol, .y, 1000000)),
    prediction_si = map2(.x = theta, .y = dat, ~growth_f(.x, .y[,2])))
```

```
prediction_si = cum_case_si_prediction %>%
  unnest(cols = c(data, prediction_si)) %>%
  select(date_of_interest, prediction_si)


case_theta_si = cum_case_si_prediction %>% select(wave, theta) %>%
```

```
  mutate(a = map_dbl(.x = theta, ~.x[1]),
         k = map_dbl(.x = theta, ~.x[2]),
         d = map_dbl(.x = theta, ~.x[3]),
         t0 = map_dbl(.x = theta, ~.x[4]),
         r = map_dbl(.x = theta, ~.x[5]),
         boro = "Staten Island") %>%
  select(-theta)


day_by_day_case_prediction = day_by_day_case_prediction_5 %>%
  left_join(prediction_si, by = "date_of_interest") %>%
  mutate(prediction_si = ifelse(is.na(prediction_si), 0, prediction_si))
```

Results

```
prediction = day_by_day_case_prediction %>%
  select(date_of_interest, prediction_case:prediction_si) %>%
  pivot_longer(prediction_case:prediction_si, names_to = "boro", names_prefix = "predict
               values_to = "prediction") %>%
  mutate(boro = recode(boro, case = "NYC", bx = "Bronx", bk = "Brooklyn", mn = "Manhatta
         boro = factor(boro, levels = c("NYC", "Bronx", "Brooklyn", "Manhattan", "Queens


observation = day_by_day_case_prediction %>%
  select(date_of_interest, cum_case:cum_si) %>%
  pivot_longer(cum_case:cum_si, names_to = "boro", names_prefix = "cum_",
               values_to = "observation") %>%
  mutate(boro = recode(boro, case = "NYC", bx = "Bronx", bk = "Brooklyn", mn = "Manhatta
         boro = factor(boro, levels = c("NYC", "Bronx", "Brooklyn", "Manhattan", "Queens
```

```
prediction %>% left_join(observation, by = c("date_of_interest","boro", "wave")) %>%
  ggplot(aes(x = date_of_interest, y = prediction, col = NULL)) +
  geom_line(col = 2, lwd = 0.5)+
  geom_line(aes(y = observation),lty = 2)+
  facet_grid(boro~wave,
             scales = "free") +
  scale_colour_manual(name = NULL,
values = c( "observation" = "black", "prediction" = 2)) +
  labs(title = "Fig.2 Cumulative Case: Fitted Curves vs Actual Patterns",
       y = "cumulative cases")

labels_1 = c("Borough", rep(c("a", "k", "d", "t0"), 4))

rbind(case_theta, case_theta_bx, case_theta_bk, case_theta_mn, case_theta_qn, case_theta
  as_tibble() %>% relocate(boro) %>%
  mutate(d = as.character(round(d, digits = 3))) %>%
  select(-r) %>%
  pivot_wider(values_from = a:t0, names_from = wave) %>%
  relocate(boro, ends_with("alpha"), ends_with("beta"), ends_with("delta"), ends_with("o
  kbl(format = "latex",align = "c",booktabs = TRUE, col.names = labels_1,
      digits = c(0, rep(c(0,3,5,0),4)),linesep = c("\\hline", "", "", "",
""), centering = T,
caption = " Estimates of Parameters for Cumulative Case") %>%
  row_spec(0,bold=TRUE) %>%
  add_header_above(c(" " = 1, "alpha" = 4, "beta" = 4,"delta" = 4,"omicron" = 4), bold =
  column_spec(1, bold = TRUE, border_right = TRUE, color = "black") %>%
```

```
  kable_styling(latex_options = "scale_down")
```

Death

```
day_by_day_death = day_by_day %>% select(date_of_interest, wave, death_count,
                        bx_death_count, bk_death_count, mn_death_count,
                        qn_death_count, si_death_count)


day_by_day_death = day_by_day_death %>%
  group_by(wave) %>%
  mutate(cum_death = cumsum(death_count),
         cum_bx = cumsum(bx_death_count),
         cum_bk = cumsum(bk_death_count),
         cum_mn = cumsum(mn_death_count),
         cum_qn = cumsum(qn_death_count),
         cum_si = cumsum(si_death_count))


# Plot of Daily Death Count in Each Borough
day_by_day_death %>%
  pivot_longer(death_count:si_death_count,
               names_to = "Borough", values_to = "daily_death_count") %>%
  mutate(Borough = str_remove(Borough, "_death_count"),
          Borough = factor(Borough,
                           levels = c("death_count", "bx", "bk", "mn", "qn", "si"))) %>
  ggplot(aes(x = date_of_interest, y = daily_death_count)) +
  geom_point(cex = 0.05) +
  facet_grid(Borough~wave,
```

```
              scales = "free") +
  labs(title = "Fig.3 Daily Death Count in Each Borough")


# Plot of Cumulative Death Count in Each Borough
day_by_day_death %>%
  pivot_longer(cum_death:cum_si, names_to = "Borough", names_prefix = "cum_",
               values_to = "cumulative_count") %>%
  mutate(Borough = factor(Borough,
                          levels = c("death", "bx", "bk", "mn", "qn", "si"))) %>%
  ggplot(aes(x = date_of_interest, y = cumulative_count)) +
  geom_point(cex = 0.05) +
  facet_grid(Borough~wave,
             scales = "free") +
  labs(title = "Fig.4 Cumulative Death Count in Each Borough", y = "cumulative death")
```

NYC

```
uni_size = 7
tick_size = 0.5
cum_death_prediction =
  day_by_day_death %>%
  filter(cum_death != 0) %>%
  group_by(wave) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(wave, death_count, cum_death, date_from_start, date_of_interest) %>%
  nest(data = c("death_count", "cum_death", "date_from_start", "date_of_interest")) %>%
  ungroup() %>%
```

```
  mutate(
    index_max = map_dbl(.x = data, ~which.max(.x$death_count)),
    dat = map(.x = data, ~cbind(.x$cum_death, .x$date_from_start)),
    start = map2(.x = index_max, .y = dat, ~initial_guess(.x, .y, tol,  0.00001, upper =
    theta = map2(.x = start, .y = dat, ~search_theta_2(.x, tol, .y, 1000000)),
    prediction_death = map2(.x = theta, .y = dat, ~growth_f(.x, .y[,2])))

prediction_death = cum_death_prediction %>%
  unnest(cols = c(data, prediction_death)) %>%
  select(date_from_start, prediction = prediction_death, wave, cum_death)

plt_nyc = vector(mode = "list", length = 4)

plt_nyc[[1]] = prediction_death %>%
  filter(wave == "alpha") %>%
  ggplot(aes(x = date_from_start, y = prediction)) +
  geom_line(col = 2)+
  geom_line(aes(y = cum_death), lty = 2) +
  labs(title = "alpha", y = "NYC")+
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank(),
        axis.text.y = element_text(size = uni_size),
        axis.ticks.y=element_line(size = tick_size))

plt_nyc[[2]] = prediction_death %>% filter(wave == "beta") %>%
  ggplot(aes(x = date_from_start, y = prediction)) +
```

```
  geom_line(col = 2)+
  geom_line(aes(y = cum_death), lty = 2)+
  labs(title = "beta")+
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank(),
        axis.title.y=element_blank(),
        axis.text.y = element_text(size = uni_size),
        axis.ticks.y=element_line(size = tick_size))

plt_nyc[[3]] = prediction_death %>% filter(wave == "delta") %>%
  ggplot(aes(x = date_from_start, y = prediction)) +
  geom_line(col = 2)+
  geom_line(aes(y = cum_death), lty = 2)+
  labs(title = "delta")+
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank(),
        axis.title.y=element_blank(),
        axis.text.y = element_text(size = uni_size),
        axis.ticks.y=element_line(size = tick_size))

plt_nyc[[4]] = prediction_death %>% filter(wave == "omicron") %>%
  ggplot(aes(x = date_from_start, y = prediction)) +
  geom_line(col = 2)+
  geom_line(aes(y = cum_death), lty = 2)+
  labs(title = "omicron")+
```

```
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank(),
        axis.title.y=element_blank(),
        axis.text.y = element_text(size = uni_size),
        axis.ticks.y=element_line(size = tick_size))


death_theta = cum_death_prediction %>% select(wave, theta) %>%
  mutate(a = map_dbl(.x = theta, ~.x[1]),
         k = map_dbl(.x = theta, ~.x[2]),
         d = map_dbl(.x = theta, ~.x[3]),
         t0 = map_dbl(.x = theta, ~.x[4]),
         maxitr = map_dbl(.x = theta, ~.x[5]),
         Borough = "NYC") %>%
  select(-theta) %>%
  pivot_longer(a:t0, names_to = "Estimates", values_to = "value") %>%
  select(value, Wave = wave, Borough, Estimates)
```

Bronx

```
#alpha


day_by_day_death_1_bx = day_by_day_death %>% filter(wave == "alpha",
                                                    cum_bx != 0) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(bx_death_count, cum_bx, date_from_start)
```

```
index_max_1_bx = which.max(day_by_day_death_1_bx$bx_death_count)
dat1_bx = cbind(day_by_day_death_1_bx$cum_bx, day_by_day_death_1_bx$date_from_start)

start_1_bx = initial_guess(index_max_1_bx, dat1_bx, tol, lower = 1/100000000000)

theta_1_bx = search_theta_2(start_1_bx, tol, dat1_bx, 1000000)

predict_growth_1_bx = growth_f(theta_1_bx, dat1_bx[,2])

list_bx_theta <- list()
list_bx <- list()

death_1_bx <-
  day_by_day_death_1_bx %>% mutate(prediction = predict_growth_1_bx) %>%
  ggplot(aes(x = date_from_start, y = prediction)) +
  geom_line() +
  geom_line(aes(y = prediction), col = 2)+
  geom_line(aes(y = cum_bx), lty = 2)+
  labs(y = "BX") +
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank(),
        axis.text.y = element_text(size = uni_size),
        axis.ticks.y=element_line(size = tick_size))

list_bx_theta[[1]] = theta_1_bx
list_bx[[1]] = death_1_bx
```

```r
#beta
day_by_day_death_2_bx = day_by_day_death %>% filter(wave == "beta",
                                                    cum_bx != 0) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(bx_death_count, cum_bx, date_from_start)


index_max_2_bx = which.max(day_by_day_death_2_bx$bx_death_count)
dat2_bx = cbind(day_by_day_death_2_bx$cum_bx, day_by_day_death_2_bx$date_from_start)


start_2_bx = initial_guess(index_max_2_bx, dat2_bx, tol)


theta_2_bx = search_theta_2(start_2_bx, tol, dat2_bx, 1000000)


predict_growth_2_bx = growth_f(theta_2_bx, dat2_bx[,2])


death_2_bx <-
  day_by_day_death_2_bx %>% mutate(prediction = predict_growth_2_bx) %>%
  ggplot(aes(x = date_from_start, y = prediction)) +
  geom_line() +
  geom_line(aes(y = prediction), col = 2)+
  geom_line(aes(y = cum_bx), lty = 2) +
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank(),
        axis.title.y=element_blank(),
        axis.text.y = element_text(size = uni_size),
        axis.ticks.y=element_line(size = tick_size))
```

```
list_bx_theta[[2]] = theta_2_bx
list_bx[[2]] = death_2_bx
```

```
#delta
day_by_day_death_3_bx = day_by_day_death %>% filter(wave == "delta",
                                                    cum_bx != 0) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(bx_death_count, cum_bx, date_from_start)

index_max_3_bx = which.max(day_by_day_death_3_bx$bx_death_count)
dat3_bx = cbind(day_by_day_death_3_bx$cum_bx, day_by_day_death_3_bx$date_from_start)

start_3_bx = initial_guess(index_max_3_bx, dat3_bx, tol, lower=0.1, upper=1.5)

theta_3_bx = search_theta_2(start_3_bx, tol, dat3_bx, 1000000)

predict_growth_3_bx = growth_f(theta_3_bx, dat3_bx[,2])

death_3_bx <-
  day_by_day_death_3_bx %>% mutate(prediction = predict_growth_3_bx) %>%
  ggplot(aes(x = date_from_start, y = prediction)) +
  geom_line() +
  geom_line(aes(y = prediction), col = 2)+
  geom_line(aes(y = cum_bx), lty = 2)+
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank(),
```

```
        axis.title.y=element_blank(),
        axis.text.y = element_text(size = uni_size),
        axis.ticks.y=element_line(size = tick_size))


list_bx_theta[[3]] = theta_3_bx
list_bx[[3]] = death_3_bx
```

```
#omicron
day_by_day_death_4_bx = day_by_day_death %>% filter(wave == "omicron",
                                                    cum_bx != 0) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(bx_death_count, cum_bx, date_from_start)


index_max_4_bx = which.max(day_by_day_death_4_bx$bx_death_count)
dat4_bx = cbind(day_by_day_death_4_bx$cum_bx, day_by_day_death_4_bx$date_from_start)


start_4_bx = initial_guess(index_max_4_bx, dat4_bx, tol)


theta_4_bx = search_theta_2(start_4_bx, tol, dat4_bx, 1000000)


predict_growth_4_bx = growth_f(theta_4_bx, dat4_bx[,2])


death_4_bx <-
  day_by_day_death_4_bx %>% mutate(prediction = predict_growth_4_bx) %>%
  ggplot(aes(x = date_from_start, y = prediction)) +
  geom_line() +
  geom_line(aes(y = prediction), col = 2)+
```

```
  geom_line(aes(y = cum_bx), lty = 2)+
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank(),
        axis.title.y=element_blank(),
        axis.text.y = element_text(size = uni_size),
        axis.ticks.y=element_line(size = tick_size))


list_bx_theta[[4]] = theta_4_bx
list_bx[[4]] = death_4_bx
```

Brooklyn

```
#alpha
day_by_day_death_1_bk = day_by_day_death %>% filter(wave == "alpha",
                                                    cum_bk != 0) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(bk_death_count, cum_bk, date_from_start)


index_max_1_bk = which.max(day_by_day_death_1_bk$bk_death_count)
dat1_bk = cbind(day_by_day_death_1_bk$cum_bk, day_by_day_death_1_bk$date_from_start)


start_1_bk = initial_guess(index_max_1_bk, dat1_bk, tol, 0.001, 2)


theta_1_bk = search_theta_2(start_1_bk, tol, dat1_bk, 1000000)


predict_growth_1_bk = growth_f(theta_1_bk, dat1_bk[,2])
```

```
list_bk_theta <- list()
list_bk <- list()


death_1_bk <-
  day_by_day_death_1_bk %>% mutate(prediction = predict_growth_1_bk) %>%
  ggplot(aes(x = date_from_start, y = prediction)) +
  geom_line() +
  geom_line(aes(y = prediction), col = 2)+
  geom_line(aes(y = cum_bk), lty = 2) +
  labs(y = "BK")+
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank(),
        axis.text.y = element_text(size = uni_size),
        axis.ticks.y=element_line(size = tick_size))


list_bk_theta[[1]] = theta_1_bk
list_bk[[1]] = death_1_bk
```

```
#beta
day_by_day_death_2_bk = day_by_day_death %>% filter(wave == "beta",
                                                    cum_bk != 0) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(bk_death_count, cum_bk, date_from_start)


index_max_2_bk = which.max(day_by_day_death_2_bk$bk_death_count)
dat2_bk = cbind(day_by_day_death_2_bk$cum_bk, day_by_day_death_2_bk$date_from_start)
```

```
start_2_bk = initial_guess(index_max_2_bk, dat2_bk, tol, 0.000001, 1.5)


theta_2_bk = search_theta_2(start_2_bk, tol, dat2_bk, 1000000)


predict_growth_2_bk = growth_f(theta_2_bk, dat2_bk[,2])


death_2_bk <-
  day_by_day_death_2_bk %>% mutate(prediction = predict_growth_2_bk) %>%
  ggplot(aes(x = date_from_start, y = prediction)) +
  geom_line() +
  geom_line(aes(y = prediction), col = 2)+
  geom_line(aes(y = cum_bk), lty = 2)+
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank(),
        axis.title.y=element_blank(),
        axis.text.y = element_text(size = uni_size),
        axis.ticks.y=element_line(size = tick_size))


list_bk_theta[[2]] = theta_2_bk
list_bk[[2]] = death_2_bk
```

```
#delta
day_by_day_death_3_bk = day_by_day_death %>% filter(wave == "delta",
                                                    cum_bk != 0) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(bk_death_count, cum_bk, date_from_start)
```

```
index_max_3_bk = which.max(day_by_day_death_3_bk$bk_death_count)
dat3_bk = cbind(day_by_day_death_3_bk$cum_bk, day_by_day_death_3_bk$date_from_start)

start_3_bk = initial_guess(index_max_3_bk, dat3_bk, tol, 0.00001, 1)

theta_3_bk = search_theta_2(start_3_bk, tol, dat3_bk, 1000000)

predict_growth_3_bk = growth_f(theta_3_bk, dat3_bk[,2])

death_3_bk <-
  day_by_day_death_3_bk %>% mutate(prediction = predict_growth_3_bk) %>%
  ggplot(aes(x = date_from_start, y = prediction)) +
  geom_line() +
  geom_line(aes(y = prediction), col = 2)+
  geom_line(aes(y = cum_bk), lty = 2)+
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank(),
        axis.title.y=element_blank(),
        axis.text.y = element_text(size = uni_size),
        axis.ticks.y=element_line(size = tick_size))

list_bk_theta[[3]] = theta_3_bk
list_bk[[3]] = death_3_bk
```

```r
#omicron
day_by_day_death_4_bk = day_by_day_death %>% filter(wave == "omicron",
                                                    cum_bk != 0) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(bk_death_count, cum_bk, date_from_start)

index_max_4_bk = which.max(day_by_day_death_4_bk$bk_death_count)
dat4_bk = cbind(day_by_day_death_4_bk$cum_bk, day_by_day_death_4_bk$date_from_start)

start_4_bk = initial_guess(index_max_4_bk, dat4_bk, tol, 0.000001, 0.8)

theta_4_bk = search_theta_2(start_4_bk, tol, dat4_bk, 1000000)

predict_growth_4_bk = growth_f(theta_4_bk, dat4_bk[,2])

death_4_bk <-
  day_by_day_death_4_bk %>% mutate(prediction = predict_growth_4_bk) %>%
  ggplot(aes(x = date_from_start, y = prediction)) +
  geom_line() +
  geom_line(aes(y = prediction), col = 2)+
  geom_line(aes(y = cum_bk), lty = 2)+
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank(),
        axis.title.y=element_blank(),
        axis.text.y = element_text(size = uni_size),
        axis.ticks.y=element_line(size = tick_size))
```

```
list_bk_theta[[4]] = theta_4_bk
list_bk[[4]] = death_4_bk
```

###Manhattan

```
#alpha
day_by_day_death_1_mn = day_by_day_death %>% filter(wave == "alpha",
                                                    cum_mn != 0) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(mn_death_count, cum_mn, date_from_start)

index_max_1_mn = which.max(day_by_day_death_1_mn$mn_death_count)
dat1_mn = cbind(day_by_day_death_1_mn$cum_mn, day_by_day_death_1_mn$date_from_start)

start_1_mn = initial_guess(index_max_1_mn, dat1_mn, tol)

theta_1_mn = search_theta_2(start_1_mn, tol, dat1_mn, 1000000)

predict_growth_1_mn = growth_f(theta_1_mn, dat1_mn[,2])

list_mn_theta <- list()
list_mn <- list()

death_1_mn <-
  day_by_day_death_1_mn %>% mutate(prediction = predict_growth_1_mn) %>%
  ggplot(aes(x = date_from_start, y = prediction)) +
  geom_line() +
```

```
  geom_line(aes(y = prediction), col = 2)+
  geom_line(aes(y = cum_mn), lty = 2)+
  labs(y="MN")+
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank(),
        axis.text.y = element_text(size = uni_size),
        axis.ticks.y=element_line(size = tick_size))

list_mn_theta[[1]] = theta_1_mn
list_mn[[1]] = death_1_mn
```

```
#beta
day_by_day_death_2_mn = day_by_day_death %>% filter(wave == "beta",
                                                    cum_mn != 0) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(mn_death_count, cum_mn, date_from_start)

index_max_2_mn = which.max(day_by_day_death_2_mn$mn_death_count)
dat2_mn = cbind(day_by_day_death_2_mn$cum_mn, day_by_day_death_2_mn$date_from_start)

start_2_mn = initial_guess(index_max_2_mn, dat2_mn, tol)

theta_2_mn = search_theta_2(start_2_mn, tol, dat2_mn, 1000000)

predict_growth_2_mn = growth_f(theta_2_mn, dat2_mn[,2])
```

```
death_2_mn <-
  day_by_day_death_2_mn %>% mutate(prediction = predict_growth_2_mn) %>%
  ggplot(aes(x = date_from_start, y = prediction)) +
  geom_line() +
  geom_line(aes(y = prediction), col = 2)+
  geom_line(aes(y = cum_mn), lty = 2)+
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank(),
        axis.title.y=element_blank(),
        axis.text.y = element_text(size = uni_size),
        axis.ticks.y=element_line(size = tick_size))


list_mn_theta[[2]] = theta_2_mn
list_mn[[2]] = death_2_mn
```

```
#delta
day_by_day_death_3_mn = day_by_day_death %>% filter(wave == "delta",
                                                    cum_mn != 0) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(mn_death_count, cum_mn, date_from_start)


index_max_3_mn = which.max(day_by_day_death_3_mn$mn_death_count)
dat3_mn = cbind(day_by_day_death_3_mn$cum_mn, day_by_day_death_3_mn$date_from_start)


start_3_mn = initial_guess(index_max_3_mn, dat3_mn, tol)
```

```
theta_3_mn = search_theta_2(start_3_mn, tol, dat3_mn, 1000000)


predict_growth_3_mn = growth_f(theta_3_mn, dat3_mn[,2])


death_3_mn <-
  day_by_day_death_3_mn %>% mutate(prediction = predict_growth_3_mn) %>%
  ggplot(aes(x = date_from_start, y = prediction)) +
  geom_line() +
  geom_line(aes(y = prediction), col = 2)+
  geom_line(aes(y = cum_mn), lty = 2)+
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank(),
        axis.title.y=element_blank(),
        axis.text.y = element_text(size = uni_size),
        axis.ticks.y=element_line(size = tick_size))


list_mn_theta[[3]] = theta_3_mn
list_mn[[3]] = death_3_mn
```

```
#omicron
day_by_day_death_4_mn = day_by_day_death %>% filter(wave == "omicron",
                                                    cum_mn != 0) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(mn_death_count, cum_mn, date_from_start)


index_max_4_mn = which.max(day_by_day_death_4_mn$mn_death_count)
```

```r
dat4_mn = cbind(day_by_day_death_4_mn$cum_mn, day_by_day_death_4_mn$date_from_start)

start_4_mn = initial_guess(index_max_4_mn, dat4_mn, tol)

theta_4_mn = search_theta_2(start_4_mn, tol, dat4_mn, 1000000)

predict_growth_4_mn = growth_f(theta_4_mn, dat4_mn[,2])

death_4_mn <-
  day_by_day_death_4_mn %>% mutate(prediction = predict_growth_4_mn) %>%
  ggplot(aes(x = date_from_start, y = prediction)) +
  geom_line() +
  geom_line(aes(y = prediction), col = 2)+
  geom_line(aes(y = cum_mn), lty = 2)+
  theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank(),
        axis.title.y=element_blank(),
        axis.text.y = element_text(size = uni_size),
        axis.ticks.y=element_line(size = tick_size))



list_mn_theta[[4]] = theta_4_mn
list_mn[[4]] = death_4_mn
```

Queens

```
death_qn = day_by_day_death %>% filter(cum_qn != 0) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(qn_death_count, cum_qn, date_from_start)


death_max_qn = death_qn %>%
  group_by(wave) %>%
  summarise(max_death = max(qn_death_count),
            dfs = max(date_from_start),
    index = which.max(qn_death_count)) %>%
  mutate(index_2 = c(index[1], index[2]+dfs[1], index[3]+dfs[1]+dfs[2], index[4]+dfs[1]+
         tt_index = c(dfs[1], dfs[1]+dfs[2], dfs[1]+dfs[2]+dfs[3], dfs[1]+dfs[2]+dfs[3]+


dfs = death_max_qn$dfs
dat_ls = vector(mode = "list", length = 4)


dat_ls[[1]] = cbind.data.frame(death_qn[1:144,]$cum_qn, 1:dfs[1])
dat_ls[[2]] = cbind.data.frame(death_qn[145:471,]$cum_qn, 1:dfs[2])
dat_ls[[3]] = cbind.data.frame(death_qn[472:620,]$cum_qn, 1:dfs[3])
dat_ls[[4]] = cbind.data.frame(death_qn[621:714,]$cum_qn, 1:dfs[4])
for(i in 1:4){
colnames(dat_ls[[i]])[1] = "cum_death"
colnames(dat_ls[[i]])[2] = "day_from_start"
}
start_qn = vector(mode = "list", length = 4)
theta_qn =vector(mode = "list", length = 4)
```

```
predict_death_qn = vector(mode = "list", length = 4)
plt_qn = vector(mode = "list", length = 4)

for(j in 1:4){
start_qn[[j]] = initial_guess(death_max_qn$index[j], dat_ls[[j]], tol, lower = 0.00001,

theta_qn[[j]] = search_theta_2(start_qn[[j]], tol, dat_ls[[j]], 1000000)

predict_death_qn[[j]] = growth_f(theta_qn[[j]], dat_ls[[j]][,2])

if(j == 1){
  plt_qn[[j]] = as_tibble(dat_ls[[j]]) %>% mutate(prediction = predict_death_qn[[j]]) %>
  ggplot(aes(x = day_from_start, y = prediction)) +
  geom_line() +
  geom_line(aes(y = prediction), col = 2)+
  geom_line(aes(y = cum_death), lty = 2) +
  labs(y= "QN")+
    theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank(),
        axis.text.y = element_text(size = uni_size),
        axis.ticks.y=element_line(size = tick_size))
} else{
    plt_qn[[j]] = as_tibble(dat_ls[[j]]) %>% mutate(prediction = predict_death_qn[[j]])
  ggplot(aes(x = day_from_start, y = prediction)) +
  geom_line(col = 2)+
  geom_line(aes(y = cum_death), lty = 2)+
```

```
      theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank(),
        axis.title.y=element_blank(),
        axis.text.y = element_text(size = uni_size),
        axis.ticks.y=element_line(size = tick_size))
}
}
```

Staten Island

```
death_si = day_by_day_death %>% filter(cum_si != 0) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(si_death_count, cum_si, date_from_start)


death_max_si = death_si %>%
  group_by(wave) %>%
  summarise(max_death = max(si_death_count),
            dfs = max(date_from_start),
    index = which.max(si_death_count)) %>%
  mutate(index_2 = c(index[1], index[2]+dfs[1], index[3]+dfs[1]+dfs[2], index[4]+dfs[1]+
         tt_index = c(dfs[1], dfs[1]+dfs[2], dfs[1]+dfs[2]+dfs[3], dfs[1]+dfs[2]+dfs[3]+


dfs_ls_si = death_max_si$dfs
dat_ls_si = vector(mode = "list", length = 4)
dat_ls_si[[1]] = cbind.data.frame(death_si[1:136,]$cum_si, 1:dfs_ls_si[1])
dat_ls_si[[2]] = cbind.data.frame(death_si[137:452,]$cum_si, 1:dfs_ls_si[2])
```

```
dat_ls_si[[3]] = cbind.data.frame(death_si[453:600,]$cum_si, 1:dfs_ls_si[3])
dat_ls_si[[4]] = cbind.data.frame(death_si[601:692,]$cum_si, 1:dfs_ls_si[4])

for(i in 1:4){
colnames(dat_ls_si[[i]])[1] = "cum_death"
colnames(dat_ls_si[[i]])[2] = "day_from_start"
}
start_si = vector(mode = "list", length = 4)
theta_si = vector(mode = "list", length = 4)
predict_death_si = vector(mode = "list", length = 4)
plt_si = vector(mode = "list", length = 4)

for(j in 1:4){
start_si[[j]] = initial_guess(death_max_si$index[j], dat_ls_si[[j]], tol, lower =1/10000

theta_si[[j]] = search_theta_2(start_si[[j]], tol, dat_ls_si[[j]], 1000000)

predict_death_si[[j]] = growth_f(theta_si[[j]], dat_ls_si[[j]][,2])

if(j == 1){
plt_si[[j]] = as_tibble(dat_ls_si[[j]]) %>% mutate(prediction = predict_death_si[[j]]) %
  ggplot(aes(x = day_from_start, y = prediction)) +
  geom_line(col = 2)+
  geom_line(aes(y = cum_death), lty = 2) +
  labs(y = "SI")+
  theme(axis.title.x=element_text(size = uni_size+1),,
        axis.text.x=element_text(size = uni_size-1),,
```

```
        axis.ticks.x=element_line(size = tick_size),,
        axis.text.y = element_text(size = uni_size),
        axis.ticks.y=element_line(size = tick_size))
} else{
  plt_si[[j]] = as_tibble(dat_ls_si[[j]]) %>% mutate(prediction = predict_death_si[[j]])
  ggplot(aes(x = day_from_start, y = prediction)) +
  geom_line(col = 2)+
  geom_line(aes(y = cum_death), lty = 2)+
    theme(axis.title.x=element_text(size = uni_size+1),,
        axis.text.x=element_text(size = uni_size-1),,
        axis.ticks.x=element_line(size = tick_size),
        axis.title.y=element_blank(),
        axis.text.y = element_text(size = uni_size),
        axis.ticks.y=element_line(size = tick_size))
}
}
```

Death Prediction Plot

```
plt_lst = c(plt_nyc, list_bx, list_bk, list_mn, plt_qn, plt_si)
do.call("grid.arrange", c(plt_lst, ncol=4,  top = "Fig.5 Fitted Curves vs Actual Pattern
```

Estimated Parameters

```
list_theta = c(list_bx_theta, list_bk_theta,list_mn_theta, theta_qn, theta_si)
theta_tibble = lapply(list_theta, as_tibble)
theta_dat = bind_rows(theta_tibble) %>% mutate(
  Wave = rep(c("alpha", "beta", "delta", "omicron"), each = 5, times = 5),
```

```
  Borough = rep(c("Bronx", "Brooklyn", "Manhattan", "Queens", "Staten_Island"), each = 2
  Estimates = rep(c("a","k","d","t0","maxit"), times = 20)
) %>%
  filter(Estimates != "maxit") %>%
  mutate(value = ifelse(Estimates == "t0" | Estimates == "a", round(value), value)) %>%
  mutate(value = ifelse(Estimates == "k", round(value,3), value))

labels_1 = c("Borough", rep(c("a", "k", "d", "t0"), 4))

rbind(death_theta, theta_dat) %>%
  unite(cb_col, c("Wave", "Estimates")) %>%
  pivot_wider(
    names_from = cb_col,
    values_from = value
  ) %>%
  kbl(format = "latex",align = "c",booktabs = TRUE, col.names = labels_1,
      digits = c(0, rep(c(0,3,5,0),4)),linesep = c("\\hline", "", "", "",
""), centering = T, caption = "Estimates of Parameters for Cumulative Death Counts") %>%
  row_spec(0,bold=TRUE) %>%
  add_header_above(c(" " = 1, "alpha" = 4, "beta" = 4,"delta" = 4,"omicron" = 4), bold =
  column_spec(1, bold = TRUE, border_right = TRUE, color = "black") %>%
  kable_styling(latex_options = "scale_down")


fatality_rate_dat = as_tibble(data.frame(
  a_case = c(48438,60578,27379,66060,14119, 103621,177569,83873,168444,50936, 21088,6584
  a_death = c(3817,5473,2399,5877,871, 1694,3187,1358,2810,796, 209,488,161,436,276, 830
```

```
  Borough = rep(c("Bronx", "Brooklyn", "Manhattan", "Queens", "Staten Island"), times =
  waves = rep(c("alpha", "beta", "delta", "omicron"), each = 5))) %>%
  mutate(max_fatality_rate = round(a_death/a_case, 4)) %>%
  select(Borough, waves, max_fatality_rate) %>%
  pivot_wider(names_from = waves,
              values_from = max_fatality_rate)

kable(fatality_rate_dat, caption = "Maximum Case Fatality Rate", booktabs = TRUE) %>%
  kable_paper() %>%
    add_header_above(c(" ", "Case Fatality Rate" = 4))
```

Hospitalized

```
day_by_day_hospitalized = day_by_day %>% select(date_of_interest, wave, hospitalized_cou
                      bx_hospitalized_count, bk_hospitalized_count, mn_hospitalized_coun
                      qn_hospitalized_count, si_hospitalized_count)

day_by_day_hospitalized = day_by_day_hospitalized %>%
  group_by(wave) %>%
  mutate(cum_hospitalized = cumsum(hospitalized_count),
         cum_bx_hospitalized = cumsum(bx_hospitalized_count),
         cum_bk_hospitalized = cumsum(bk_hospitalized_count),
         cum_mn_hospitalized = cumsum(mn_hospitalized_count),
         cum_qn_hospitalized = cumsum(qn_hospitalized_count),
         cum_si_hospitalized = cumsum(si_hospitalized_count))

day_by_day_hospitalized %>%
```

```
  pivot_longer(cum_hospitalized:cum_si_hospitalized, names_to = "Borough", names_prefix
               values_to = "cumulative_count", ) %>%
  mutate(Borough = str_remove(Borough, "_hospitalized"),
         Borough = factor(Borough, levels = c("hospitalized", "bx", "bk", "mn", "qn", "si
  ggplot(aes(x = date_of_interest, y = cumulative_count)) +
  geom_point(cex = 0.05) +
  facet_grid(Borough~wave,
             scales = "free")+
  labs(title = "Fig.6 Cumulative Hospitalization Count in Each Borough")
```

NYC

```
cum_hospitalized_prediction =
  day_by_day_hospitalized %>%
  filter(cum_hospitalized != 0) %>%
  group_by(wave) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(wave, hospitalized_count, cum_hospitalized, date_from_start, date_of_interest)
  nest(data = c("hospitalized_count", "cum_hospitalized", "date_from_start", "date_of_in
  ungroup() %>%
  mutate(
    index_max = map_dbl(.x = data, ~which.max(.x$hospitalized_count)),
    dat = map(.x = data, ~cbind(.x$cum_hospitalized, .x$date_from_start)),
    start = map2(.x = index_max, .y = dat, ~initial_guess(.x, .y, tol, lower = 1, upper
    theta = map2(.x = start, .y = dat, ~search_theta_2(.x, tol, .y, 1000000)),
    prediction_hospitalized = map2(.x = theta, .y = dat, ~growth_f(.x, .y[,2]))) %>%
  select(wave, data, prediction_hospitalized, theta)
```

```
prediction_hospitalized = cum_hospitalized_prediction %>% unnest(cols = c(data, predicti
  select(date_of_interest, prediction_hospitalized)


hospitalized_theta = cum_hospitalized_prediction %>% select(wave, theta) %>%
  mutate(a = map_dbl(.x = theta, ~.x[1]),
         k = map_dbl(.x = theta, ~.x[2]),
         d = map_dbl(.x = theta, ~.x[3]),
         t0 = map_dbl(.x = theta, ~.x[4]),
         r = map_dbl(.x = theta, ~.x[5]),
         boro = "NYC") %>%
  select(-theta)


day_by_day_hospitalized_prediction_1 = day_by_day_hospitalized %>%
  left_join(prediction_hospitalized, by = "date_of_interest") %>%
  mutate(prediction_hospitalized = ifelse(is.na(prediction_hospitalized), 0, prediction_
```

Bronx

```
cum_hospitalized_bx_prediction =
  day_by_day_hospitalized %>%
  filter(cum_bx_hospitalized != 0) %>%
  group_by(wave) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(wave, bx_hospitalized_count, cum_bx_hospitalized, date_from_start, date_of_inte
  nest(data = c("bx_hospitalized_count", "cum_bx_hospitalized", "date_from_start", "date
  ungroup() %>%
  mutate(
```

```
    index_max = map_dbl(.x = data, ~which.max(.x$bx_hospitalized_count)),
    dat = map(.x = data, ~cbind(.x$cum_bx_hospitalized, .x$date_from_start)),
    start = map2(.x = index_max, .y = dat, ~initial_guess(.x, .y, tol)),
    theta = map2(.x = start, .y = dat, ~search_theta_2(.x, tol, .y, 1000000)),
    prediction_bx = map2(.x = theta, .y = dat, ~growth_f(.x, .y[,2])))

prediction_bx = cum_hospitalized_bx_prediction %>%
  unnest(cols = c(data, prediction_bx)) %>%
  select(date_of_interest, prediction_bx)

hospitalized_theta_bx = cum_hospitalized_bx_prediction %>% select(wave, theta) %>%
  mutate(a = map_dbl(.x = theta, ~.x[1]),
         k = map_dbl(.x = theta, ~.x[2]),
         d = map_dbl(.x = theta, ~.x[3]),
         t0 = map_dbl(.x = theta, ~.x[4]),
         r = map_dbl(.x = theta, ~.x[5]),
         boro = "Bronx") %>%
  select(-theta)

day_by_day_hospitalized_prediction_2 = day_by_day_hospitalized_prediction_1 %>%
  left_join(prediction_bx, by = "date_of_interest") %>%
  mutate(prediction_bx = ifelse(is.na(prediction_bx), 0, prediction_bx))
```

Brooklyn

```
cum_hospitalized_bk_prediction =
  day_by_day_hospitalized %>%
```

```
  filter(cum_bk_hospitalized != 0) %>%
  group_by(wave) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(wave, bk_hospitalized_count, cum_bk_hospitalized, date_from_start, date_of_inte
  nest(data = c("bk_hospitalized_count", "cum_bk_hospitalized", "date_from_start", "date
  ungroup() %>%
  mutate(
    index_max = map_dbl(.x = data, ~which.max(.x$bk_hospitalized_count)),
    dat = map(.x = data, ~cbind(.x$cum_bk_hospitalized, .x$date_from_start)),
    start = map2(.x = index_max, .y = dat, ~initial_guess(.x, .y, tol, lower = 1/1000000
    theta = map2(.x = start, .y = dat, ~search_theta_2(.x, tol, .y, 1000000)),
    prediction_bk = map2(.x = theta, .y = dat, ~growth_f(.x, .y[,2])))

prediction_bk = cum_hospitalized_bk_prediction %>%
  unnest(cols = c(data, prediction_bk)) %>%
  select(date_of_interest, prediction_bk)

hospitalized_theta_bk = cum_hospitalized_bk_prediction %>% select(wave, theta) %>%
  mutate(a = map_dbl(.x = theta, ~.x[1]),
         k = map_dbl(.x = theta, ~.x[2]),
         d = map_dbl(.x = theta, ~.x[3]),
         t0 = map_dbl(.x = theta, ~.x[4]),
         r = map_dbl(.x = theta, ~.x[5]),
         boro = "Brooklyn") %>%
  select(-theta)

day_by_day_hospitalized_prediction_3 = day_by_day_hospitalized_prediction_2 %>%
```

```
  left_join(prediction_bk, by = "date_of_interest") %>%
  mutate(prediction_bk = ifelse(is.na(prediction_bk), 0, prediction_bk))
```

Manhattan

```
cum_hospitalized_mn_prediction =
  day_by_day_hospitalized %>%
  filter(cum_mn_hospitalized != 0) %>%
  group_by(wave) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(wave, mn_hospitalized_count, cum_mn_hospitalized, date_from_start, date_of_inte
  nest(data = c("mn_hospitalized_count", "cum_mn_hospitalized", "date_from_start", "date
  ungroup() %>%
  mutate(
    index_max = map_dbl(.x = data, ~which.max(.x$mn_hospitalized_count)),
    dat = map(.x = data, ~cbind(.x$cum_mn_hospitalized, .x$date_from_start)),
    start = map2(.x = index_max, .y = dat, ~initial_guess(.x, .y, tol, lower = 1/1000000
    theta = map2(.x = start, .y = dat, ~search_theta_2(.x, tol, .y, 1000000)),
    prediction_mn = map2(.x = theta, .y = dat, ~growth_f(.x, .y[,2])))

prediction_mn = cum_hospitalized_mn_prediction %>%
  unnest(cols = c(data, prediction_mn)) %>%
  select(date_of_interest, prediction_mn)

hospitalized_theta_mn = cum_hospitalized_mn_prediction %>% select(wave, theta) %>%
  mutate(a = map_dbl(.x = theta, ~.x[1]),
         k = map_dbl(.x = theta, ~.x[2]),
```

```
          d = map_dbl(.x = theta, ~.x[3]),
          t0 = map_dbl(.x = theta, ~.x[4]),
          r = map_dbl(.x = theta, ~.x[5]),
          boro = "Manhattan") %>%
  select(-theta)


day_by_day_hospitalized_prediction_4 = day_by_day_hospitalized_prediction_3 %>%
  left_join(prediction_mn, by = "date_of_interest") %>%
  mutate(prediction_mn = ifelse(is.na(prediction_mn), 0, prediction_mn))
```

Queens

```
cum_hospitalized_qn_prediction =
  day_by_day_hospitalized %>%
  filter(cum_qn_hospitalized != 0) %>%
  group_by(wave) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(wave, qn_hospitalized_count, cum_qn_hospitalized, date_from_start, date_of_inte
  nest(data = c("qn_hospitalized_count", "cum_qn_hospitalized", "date_from_start", "date
  ungroup() %>%
  mutate(
    index_max = map_dbl(.x = data, ~which.max(.x$qn_hospitalized_count)),
    dat = map(.x = data, ~cbind(.x$cum_qn_hospitalized, .x$date_from_start)),
    start = map2(.x = index_max, .y = dat, ~initial_guess(.x, .y, tol)),
    theta = map2(.x = start, .y = dat, ~search_theta_2(.x, tol, .y, 1000000)),
    prediction_qn = map2(.x = theta, .y = dat, ~growth_f(.x, .y[,2])))
```

```
prediction_qn = cum_hospitalized_qn_prediction %>%
  unnest(cols = c(data, prediction_qn)) %>%
  select(date_of_interest, prediction_qn)


hospitalized_theta_qn = cum_hospitalized_qn_prediction %>% select(wave, theta) %>%
  mutate(a = map_dbl(.x = theta, ~.x[1]),
         k = map_dbl(.x = theta, ~.x[2]),
         d = map_dbl(.x = theta, ~.x[3]),
         t0 = map_dbl(.x = theta, ~.x[4]),
         r = map_dbl(.x = theta, ~.x[5]),
         boro = "Queens") %>%
  select(-theta)


day_by_day_hospitalized_prediction_5 = day_by_day_hospitalized_prediction_4 %>%
  left_join(prediction_qn, by = "date_of_interest") %>%
  mutate(prediction_qn = ifelse(is.na(prediction_qn), 0, prediction_qn))
```

Staten Island

```
cum_hospitalized_si_prediction =
  day_by_day_hospitalized %>%
  filter(cum_si_hospitalized != 0) %>%
  group_by(wave) %>%
  mutate(date_from_start = as.numeric(date_of_interest-min(date_of_interest))+1) %>%
  select(wave, si_hospitalized_count, cum_si_hospitalized, date_from_start, date_of_inte
  nest(data = c("si_hospitalized_count", "cum_si_hospitalized", "date_from_start", "date
  ungroup() %>%
```

```
  mutate(
    index_max = map_dbl(.x = data, ~which.max(.x$si_hospitalized_count)),
    dat = map(.x = data, ~cbind(.x$cum_si_hospitalized, .x$date_from_start)),
    start = map2(.x = index_max, .y = dat, ~initial_guess(.x, .y, tol, lower = 1/1000000
    theta = map2(.x = start, .y = dat, ~search_theta_2(.x, tol, .y, 1000000)),
    prediction_si = map2(.x = theta, .y = dat, ~growth_f(.x, .y[,2])))

prediction_si = cum_hospitalized_si_prediction %>%
  unnest(cols = c(data, prediction_si)) %>%
  select(date_of_interest, prediction_si)

hospitalized_theta_si = cum_hospitalized_si_prediction %>% select(wave, theta) %>%
  mutate(a = map_dbl(.x = theta, ~.x[1]),
         k = map_dbl(.x = theta, ~.x[2]),
         d = map_dbl(.x = theta, ~.x[3]),
         t0 = map_dbl(.x = theta, ~.x[4]),
         r = map_dbl(.x = theta, ~.x[5]),
         boro = "Staten Island") %>%
  select(-theta)

day_by_day_hospitalized_prediction_6 = day_by_day_hospitalized_prediction_5 %>%
  left_join(prediction_si, by = "date_of_interest") %>%
  mutate(prediction_si = ifelse(is.na(prediction_si), 0, prediction_si))
```

Results

```
prediction = day_by_day_hospitalized_prediction_6 %>%
  select(date_of_interest, prediction_hospitalized:prediction_si) %>%
  pivot_longer(prediction_hospitalized:prediction_si, names_to = "boro", names_prefix =
               values_to = "prediction") %>%
  mutate(boro = recode(boro, hospitalized = "NYC", bx = "Bronx", bk = "Brooklyn", mn = "
         boro = factor(boro, levels = c("NYC", "Bronx", "Brooklyn", "Manhattan", "Queens

observation = day_by_day_hospitalized_prediction_6 %>%
  select(date_of_interest, cum_hospitalized:cum_si_hospitalized) %>%
  pivot_longer(cum_hospitalized:cum_si_hospitalized, names_to = "boro", names_prefix = "
               values_to = "observation") %>%
  mutate(boro = recode(boro, hospitalized = "NYC", bx_hospitalized = "Bronx", bk_hospita
         boro = factor(boro, levels = c("NYC", "Bronx", "Brooklyn", "Manhattan", "Queens

prediction %>% left_join(observation, by = c("date_of_interest","boro", "wave")) %>%
  ggplot(aes(x = date_of_interest, y = prediction, col = NULL)) +
  geom_line(col = 2, lwd = 0.5)+
  geom_line(aes(y = observation),lty = 2)+
  facet_grid(boro~wave,
             scales = "free") +
  scale_colour_manual(name = NULL,
values = c( "observation" = "black", "prediction" = 2))+
  labs(title = "Fig.7 Fitted Curves vs Actual Patterns")

labels_1 = c("Borough", rep(c("a", "k", "d", "t0"), 4))
```

```
rbind(hospitalized_theta, hospitalized_theta_bx, hospitalized_theta_bk, hospitalized_the
  as_tibble() %>% relocate(boro) %>%
  mutate(d = as.character(round(d, digits = 3))) %>%
  select(-r) %>%
  pivot_wider(values_from = a:t0, names_from = wave) %>%
  relocate(boro, ends_with("alpha"), ends_with("beta"), ends_with("delta"), ends_with("o
  kbl(format = "latex",align = "c",booktabs = TRUE, col.names = labels_1,
      digits = c(0, rep(c(0,3,5,0),4)),linesep = c("\\hline", "", "", "",
""), centering = T,caption = "Estimates of Parameters for Cumulative Hospitalization") %
  row_spec(0,bold=TRUE) %>%
  add_header_above(c(" " = 1, "alpha" = 4, "beta" = 4,"delta" = 4,"omicron" = 4), bold =
  column_spec(1, bold = TRUE, border_right = TRUE, color = "black") %>%
  kable_styling(latex_options = "scale_down")

Borough = c("alpha", "beta", "delta", "omicron")
NYC = c("0.2382", "0.0902", "0.0615", "0.0323")
Bronx = c("0.2303", "0.1045", "0.0977", "0.0500")
Brooklyn = c("0.2475", "0.0945","0.0558", "0.0329")
Manhattan = c("0.2681", "0.0843","0.0456", "0.0271")
Queens = c("0.2471", "0.0905","0.0708", "0.0275")
Staten_Island = c("0.1625", "0.0731","0.0592", "0.0275")

as.data.frame(rbind(NYC,Bronx, Brooklyn, Manhattan, Queens, Staten_Island))%>%
  knitr::kable(col.names = Borough,booktabs = TRUE,centering = T,
               caption = "Maximum Case Hospitalization Rate") %>%
  kable_paper()
```