

時系列解析手法による COVID-19 国内新規感染者数予測モデル

Shumpei Fukuda

2023.12.16

1 概要/Abstract

概要: 本論文の大義は、新型コロナウイルス感染症（以下 COVID-19）について、日本国内新規感染者数の予測モデルを作成することで、医療機関や公的機関が事前対策を実施する際の指標を提供するものである。一方で個人的関心としては、複数のモデルを作成し、比較・評価する一連のプロセスを通じて著者自身が時系列分析の流れやモデリング手法を理解する目的もある。なお、作成するモデルは「統計モデル」「状態空間モデル」「機械学習モデル」である。

Abstract: The main purpose of this study is to develop a model for predicting the number of newly infected cases of novel coronavirus infection (COVID-19) in Japan, thereby providing an indicator for medical institutions and public organizations to implement proactive measures. On the other hand, as a personal interest, the author also aims to understand the flow of time-series analysis and modeling methods through a series of processes of creating, comparing, and evaluating multiple models. The models to be created are "statistical models," "state space models," and "machine learning models".

Contents

1	概要/Abstract	1
2	記述/Description	4
2.1	データ取得元	4
2.2	データ構造: 期間と変数	4
2.3	基本統計量	4
3	モデリング/Modeling	5
3.1	前提知識 1. 定常性	5
3.2	前提知識 2. 確率過程	5
3.3	前処理	6
4	伝統的手法 / Box-Jenkins Method	11
4.1	同定	11
4.2	パラメータ推定	15
4.3	モデル診断	15
5	統計モデル/Statistical Model	17
5.1	統計モデリング概略	17
5.2	前提知識 3. 統計モデル	18
5.3	予測精度	19
5.4	ARIMA モデル	19
5.5	SARIMA モデル	21
5.6	Python による実装	22
6	状態空間モデル/State-Space Model	23
6.1	予測精度	23
6.2	状態空間モデル	23
6.3	アルゴリズム	25
6.4	カルマンフィルタ	26
6.5	Python による実装	28
6.6	特微量エンジニアリング	28
6.7	予測結果の可視化	29
7	機械学習モデル/Machine Learning Model	30
7.1	予測精度	30
7.2	Random Forest	30
7.3	アルゴリズム	31
7.4	Python による実装	31
7.5	汎化性能の確認	33
7.6	LightGBM	34

7.7	アルゴリズム	35
7.8	Python による実装	35
7.9	汎化性能の確認	37
8	おわりに/Conclusion	38

2 記述/Description

2.1 データ取得元

本論文では、新型コロナウイルス感染症（以下 COVID-19）について、日本国内新規感染者数の予測モデルを作成する。使用したデータは厚生労働省のオープンデータである。^[1]

2.2 データ構造: 期間と変数

日本国内における COVID-19 新規感染者数のデータは「国内合計」および「都道府県別」が公開されているが、そのうち「国内合計数」の日次データのみを用いた。また、モデリングの対象期間は国内累計感染者数が 10,000 人を超えた始めた「2020 年 5 月 1 日」から、感染症法上の位置付けが季節性インフルエンザと同じ「5 類」に移行される直前の「2023 年 4 月 30 日」まで全 3 年間 (1,095 日) を対象とした。ただし、後述する通り実際に成立した（一定の予測精度を確保できた）モデルはより限定的な期間を対象としている。よって、データ構造は「新規感染者数（国内合計）」と「日付（西暦年-月-日）」からなる「1095 行 × 2 列」である。なお欠損値はない。

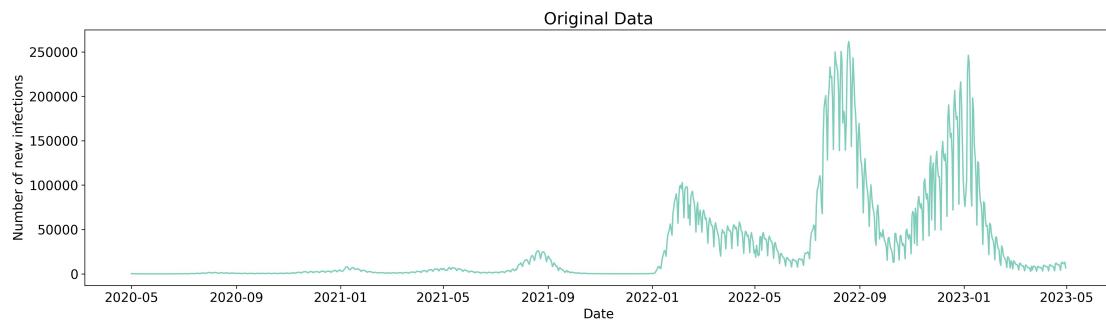


Figure1: 3 年間の新規感染者数の推移 (データ元: 厚生労働省 [1])

当該期間の国内新規感染者数の推移を可視化したのが上記グラフである。横軸に期間（日）を、縦軸に新規感染者数（人）をとる。以後、本時系列データを原系列（original）と呼ぶ。

2.3 基本統計量

基本統計量は以下の通りである（小数点第一位以下を四捨五入）。着目すべきは最大値と最小値の変動幅であろう。標準偏差（standard deviation）が約 51,465 人と期間中大きく変動していることがわかる。

	データ数	平均	標準偏差	最小値	25% 点	50% 点	75% 点	最大値
原系列	1,095	30,724	51,466	20	983	5,508	40,941	261,735

Table1: 原系列の基本統計量

3 モデリング/Modeling

これより具体的なモデリングを進める。そこで前提知識として、時系列分析において重要な概念である**定常性**(stationarity)について補足する。時系列データの分析においては、データの定常性を確認することが重要である。定常性を満たす時系列はたとえ不規則な現象でも、その構造は時間的に変化しない一定の確率モデルの実現値をみなすことができる。一言でいえば、モデリングや予測を行う上で都合が良いのである。^[2] また、定常系列を前提とする多くの統計的時系列モデルを適用する上で、必要もある。

3.1 前提知識 1. 定常性

定常性とは、次の条件を満たすことである。なお、本時系列分析では**弱定常性**(weakly stationary)または**共分散定常**を定常性と呼ぶ。ちなみに**強定常性**(strict stationarity)という性質もある。後述する MA(q) モデルは、強定常性の代表例である。^[3] 確率過程 X_t について、弱定常性の条件は次の通りである。^[2]

■1. 平均が一定

$$\mathbb{E}[X_t] = \mu \quad (\mu \text{は } t \text{ と無関係な定数}) \quad (1)$$

■2. 分散が有限

$$\text{Var}[X_t] < \infty \quad (2)$$

■3. 自己共分散が時間差のみに依存

$$\text{Cov}[X_t, X_{t-k}] = E[(X_t - \mu)(X_{t-k} - \mu)] = \gamma(k) \quad (3)$$

$$\text{※特に } k = 0 \text{ のとき, } \text{Cov}[X_t, X_t] = E[(X_t - \mu)^2] = \text{Var}(X_t) = \sigma^2 \quad (4)$$

なお、弱定常過程において自己相関は以下のようになり、自己相関も時点に依存しなくなる。

$$\text{Corr}[X_t, X_{t-k}] = \frac{\gamma(k)}{\sqrt{\gamma(0)\gamma(0)}} = \frac{\gamma(k)}{\gamma(0)} \quad (5)$$

ここで、 $\gamma(k)$ はラグ k における自己共分散、 $\gamma(0)$ は分散（ラグ 0 の自己共分散）を表す。したがって、次のように表される。^[4]

$$\rho_k = \frac{\gamma(k)}{\gamma(0)} \quad (6)$$

3.2 前提知識 2. 確率過程

前節で述べた通り、本時系列分析では定常性の仮定をおく。さらに本節では簡単に時系列分析のイメージを掴むために、**確率過程**について明示しておく。つまり「離散時間確率過程 $\{X_t\} | t \in \mathbb{Z}$ について定常性の仮定のもとでは、 X_n, X_{n-1} の同時分布と X_2, X_1 の同時分布は一致する」。ここで、確率変数 X_t は各時刻 t ($t \in \mathbb{Z}, \mathbb{Z}$ は整数) に対応している。すなわち、**離散時間確率過程**(discrete time stochastic process)を想定している。なお、厳密な定義は参考文献を参照されたい。^[2]

3.3 前処理

仮に原系列 $\{y_t\}$ が非定常であっても、ある変換を施した系列 $\{z_t\}$ は定常に見えることがある。^[2] 変換の具体例を以下に示す。

3.3.1 変換の具体例

■移動平均 移動平均では、一定期間の平均値を用いることで短期的な変動を除去し系列を平滑化できる。直感的には「原系列からトレンド・循環変動 TC_t を抽出している」といえる。^[2]

$$\widehat{TC}_t = \frac{1}{2k+1} \sum_{j=-k}^k y_{t+j} \quad (7)$$

なお、上記は単純移動平均という。より一般には各時点のデータに対してウェイト（ウインドウともいう）を用いて加重平均を求めることがある。^[5]

■対数系列 対数変換により非線形の関係を線形の形に変換し、データの分散が一様になると、誤差項がほぼ正規分布とみなせる場合がある。^[6] 後述するが、乗法モデルは対数変換を施すことで加法モデルに帰着できる。^[2] なお、原系列がゼロや非常に小さい値を含む場合は対数関数が定義されない。よって小さな定数 `const`（ここでは'1e-5') をデータに加えている。

$$\log(y_t + \text{const}) \quad (8)$$

■一次差分系列 系列が全体的に単調な変動を含む場合、差分を取って解析することが有効である ^[2]。たとえば、差分系列 $z_n = \Delta y_n = y_n - y_{n-1}$ があるとき、 y_n が直線 $y_n = a + bn$ で表せる場合、 $z_n = \Delta y_n = (a + bn) - (a + b(n-1)) = b$ となる。傾き b が一定となるため、トレンドを除去できる。^[6] なおここでは、一日前との差分を取っているため「一次差分」と呼ばれる。直観的には、移動平均とは対称的に「トレンドを除去してトレンド周りの動きに着目する」と言える。

$$\Delta y_t = y_t - y_{t-1} \quad (9)$$

代表的な例として、経済現象や物理現象で現れるランダムウォークモデル (random walk model) がある。

$$y_t = y_{t-1} + \epsilon_t \quad (10)$$

ここで ϵ_t はホワイトノイズ (White noise) を表す。ランダムウォークは非定常な過程だが、一次差分を取ることで定常な時系列に変換できる。

$$\Delta y_t = (y_{t-1} + \epsilon_t) - y_{t-1} = \epsilon_t \quad (11)$$

このように、一次差分系列 Δy_t はホワイトノイズと等しく、定常過程となる。なお、ホワイトノイズ ϵ_t は次の性質を持つ確率変数である。

平均が 0:	$\mathbb{E}[\epsilon_t] = 0$
分散が一定:	$\text{Var}[\epsilon_t] = \sigma^2$
各時点間が無相関:	$\text{Cov}(\epsilon_t, \epsilon_s) = 0 \quad t \neq s$

■季節階差系列 特定の季節性を持つデータの周期的なパターンを除去する。たとえば、月次データであれば12ヶ月前の観測値を除くことで、年周期（季節性）の変動要因を除去することができる。同様に日次データであれば、ある時点の観測値から7日前の観測値を引くことで、週単位の周期性の影響を取り除くことができる。なお後述するが、新規感染者数のミクロなトレンドを確認すると、週単位での変動に関連性が見られた。

$$\Delta y_t = y_t - y_{t-7} \quad (12)$$

■対数差分系列 以降はこれまでの変換の組み合わせである。^[7] 対数変換後に一次差分を取っている。

$$\Delta \log(y_t) = \log(y_t) - \log(y_{t-1}) \quad (13)$$

■対数季節階差系列 同様に対数系列にさらに季節性階差を取ることで、対数変換の利点（分散の安定化）と季節階差の利点（季節性の除去）を組み合わせることができる。

$$\Delta_7 \log(y_t) = \log(y_t) - \log(y_{t-7}) \quad (14)$$

なお、3年間全期間を対象とした分析では季節階差系列および対数季節階差系列に対して、欠損値が発生するためリストワイズ除去法（listwise deletion）を適用している。また、移動平均系列の最小期間は1日としているため、欠損値は発生しない（仮に1日しかなくとも、その日の値を平均値とする）。

3.3.2 各系列の可視化（マクロ）

前節で説明した変換を原系列に施した結果を可視化する。

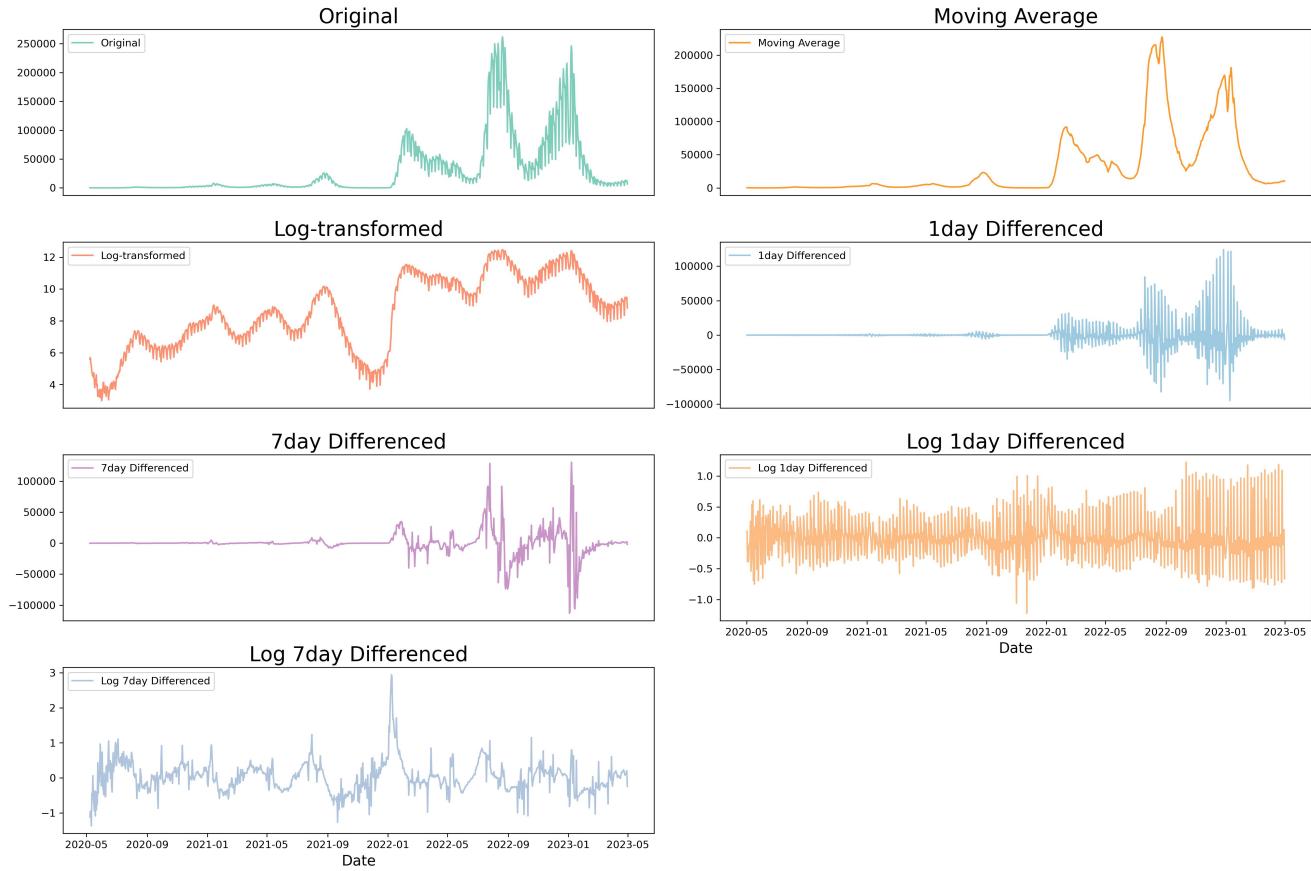


Figure2: 各系列の可視化

たとえば左上の原系列 (original) と比較すると、右下の対数一次差分系列 (Log 1day Differenced) はおおよそ平均が 0 で一定の変動を繰り返しているよう変換されたことが確認できる。

3.3.3 時系列構造分解

時系列データは次の通りその変動を分解することができる。

■加法モデル まず加法モデル (additive model) と呼ばれるモデルを示す。

$$Y_t = T_t + C_t + S_t + I_t \quad (15)$$

- | |
|---|
| Y_t : 時刻 t における観測値 |
| T_t : 傾向変動 (Trend) - 長期的な上昇/下降傾向 |
| C_t : 循環変動 (Cycle) - 周期が定まらない変動 |
| S_t : 季節変動 (Seasonal) - 季節性/特定周期変動 |
| I_t : 不規則変動 (Irregular) - 不規則かつ短期に起こる変動/ノイズ |

実際には傾向変動 T_t と循環変動 C_t とを分解する必要性は小さいため、これらを組み合わせたトレンド・循環変動 TC_t を用いて以下のように分解することが多い.[2]

$$Y_t = T_t + S_t + I_t \quad (16)$$

一部の系列について分解したものをグラフとして示す(色は,2 の図に対応している)。

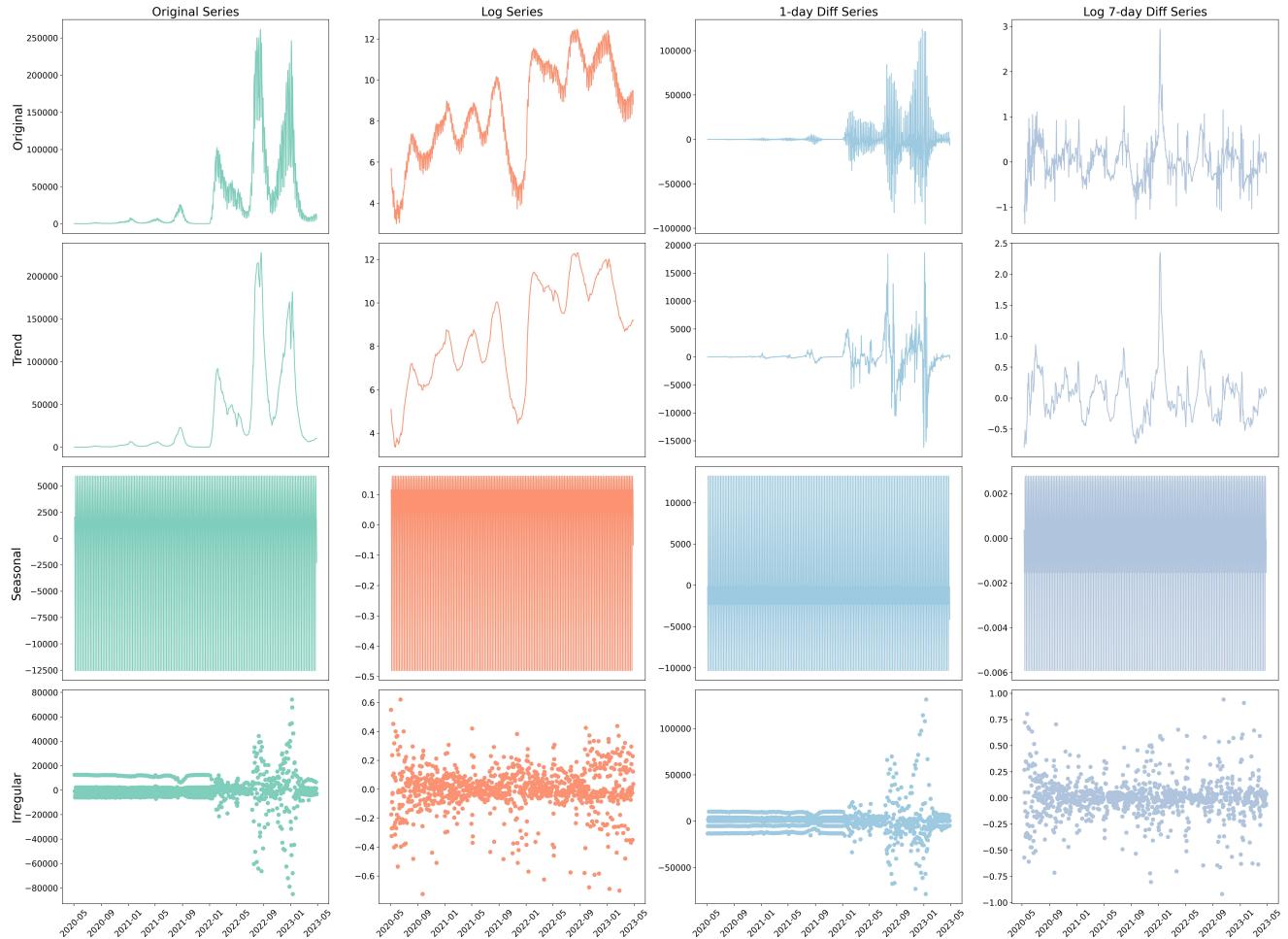


Figure3: 時系列構造分解

■乗法モデル また、**乗法モデル** (multiplicative model) と呼ばれるモデルもある（記号の定義は加法モデルと同様）。

$$Y_t = T_t \times C_t \times S_t \times I_t \quad (17)$$

なお乗法モデルは、対数変換を施すことで加法モデルに帰着する.[2]

3.3.4 季節変動の可視化(ミクロ)

ここで時系列データの周期性を確認しておく。Figure3 の図で確認した通り、原系列の季節変動 (Seasonal) はマクロ(全期間)では非常に変動周期が細かく確認しづらい。しかし、ミクロ(1ヶ月レベル)でみると、1週間単位での周期性が確認できる。詳細はモデリングの際に記すが、たとえばコロナの各波のうち、期間の近い4つの波の最初の1ヶ月について周期性を可視化すると次の通りである。

波	期間	日数	可視化
第3波	2020-12-01 ~ 2021-02-28	90日	有
第4波	2021-03-18 ~ 2021-07-11	116日	
第5波	2021-07-01 ~ 2021-09-30	92日	有
第6波	2022-01-01 ~ 2022-03-31	90日	有
第7波	2022-07-01 ~ 2022-09-30	92日	有
第8波	2022-11-30 ~ 2023-01-24	56日	

Table2: COVID-19 各波の期間

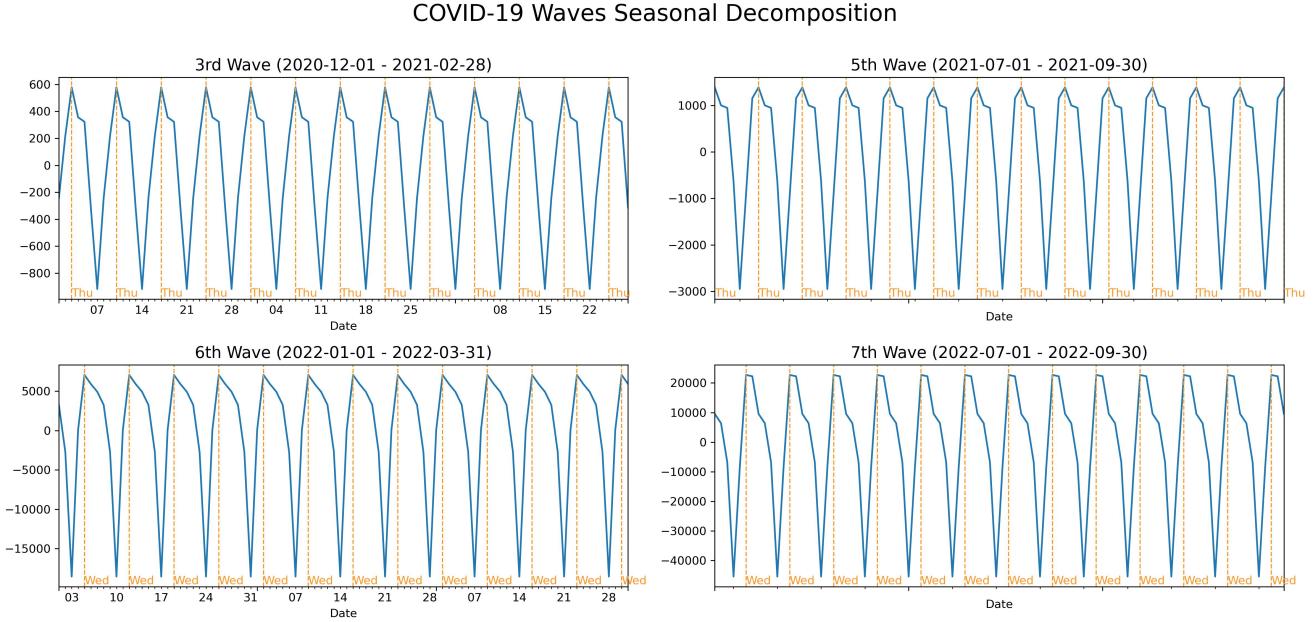


Figure4: 周期性の確認

3.3.5 周期性の解釈

まず構造分解した上で、周期性に着目している点は留意が必要である。いずれも一週間の中で火曜日あるいは水曜日に想定的な最大値をとり、その前日（火曜日）あるいは前々日（月曜日）に相対的な最低値をとっている。専門家ではないので言及はできないが、たとえば一般に病院の日曜休診が多く、検査が受けられないことが想定される。よって、「日曜日あるいは翌月曜日に新規感染者数が観測されにくい」というような曜日固有の仮説は整合する可能性がある（もちろん、潜伏期間や結果が出るまでのラグ、報告されるまでのラグなどは考慮が必要である）。一方で「第5波のデルタ株に比べて、第6波のオミクロン株（BA.1/BA.2）は潜伏期間が短縮されている」（国立感染症研究所[8]）といった点は、周期性の期間比較からは確認できない。

4 伝統的手法 / Box-Jenkins Method

これより伝統的に使われてきた手法のうち, Box and Jenkins のモデル作成手順を確認する [2]. 通称, ボックス・ジェンキンス法 (Box-Jenkins method) とは, モデルの同定, パラメータの推定, モデルの診断, といった一連の流れを経てモデリングを行うためのフレームワークである. たとえば次のような流れである.

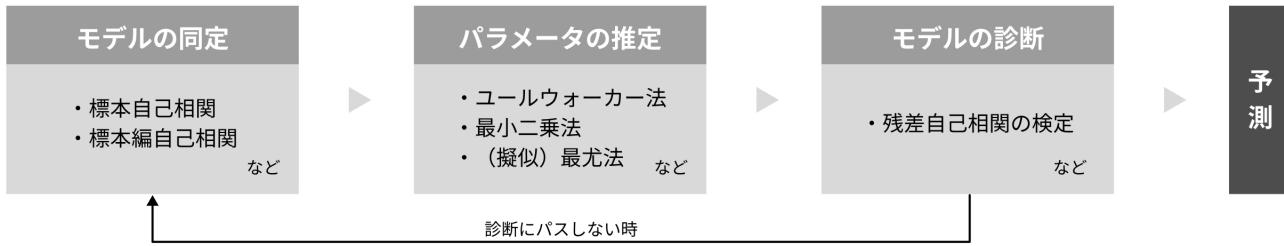


Figure5: モデリング手順 (参考 : 白石 (2022)[2])

4.1 同定

まず自己相関や偏自己相関を基に, モデルの同定 (Identification) を行う. 既述した時系列の構造分解が適切に行われていれば, 標本自己共分散関数のグラフ (コレログラム) を見ただけでもある程度, 同定できる. ただし, 同定は標本サイズが十分に大きい時, すなわち次のような漸近的性質を前提としている点に注意が必要である.[2]

$$\hat{r}(h) \xrightarrow[n \rightarrow \infty]{P} r(h) \quad (18)$$

ここで, $\hat{r}(h)$ は標本自己共分散関数つまり標本からの推定値, $r(h) = \text{Cov}[X_t, X_{t+h}]$ は理論的な自己共分散関数, h は時間差つまりラグ (lag) を意味する.

4.1.1 自己相関関数と偏自己相関関数

既述の通り, 自己相関 (ACF : Autocorrelation Function) と偏自己相関 (PACF : Partial Autocorrelation Function) を確認することでも, おおよその同定が可能である.

■自己相関関数 過去の観測との関係性の強さを把握することができる.[2]

$$\rho(h) = \frac{\text{Cov}[X_t, X_{t+h}]}{\sqrt{\text{Var}[X_t]\text{Var}[X_{t+h}]}} \quad (19)$$

とくに定常過程において次のように表すことができる.

$$\rho(h) = \frac{r(h)}{r(0)} \quad (20)$$

■偏自己相関関数 偏自己相関では中間の観測値の影響を除いて、時系列の異なる時点との相関の直接的な強さを測る。たとえば自己相関関数 $\rho(h)$ は時間差 h だけ離れた関係の強さを測っていたが、このとき X_{t+h} と X_t の間には中間の観測値 $X_{t+1}, \dots, X_{t+h-1}$ の影響が含まれる。一方、偏自己相関関数では X_{t+h} からこれら中間の影響を除き、 X_t と X_{t+h} との直接的な線形関係の強さを示す指標となる。偏自己相関係数 ϕ_{hh} を用いると、時系列データは次のような線形和の関係式で表すことができる。

$$X_{t+h} - \mu = \phi_{h1}(X_{t+h-1} - \mu) + \dots + \phi_{11}(X_t - \mu) \quad (21)$$

ここで、 μ は平均値、 $\phi_{h1}, \dots, \phi_{11}$ は偏自己相関係数を表す。データの平均値からの偏差を考慮しており、データが持つトレンドや季節性などの影響を除去した純粋な自己相関の強さを評価することが直感的にわかる。なお、具体的にはコレログラム (correlogram) を参考に次の基準を目安に判断できる。

モデル	自己相関 (ACF)	偏自己相関 (PACF)
AR(p)	減衰していく	p+1 次以降 0
MA(q)	q+1 次以降 0	減衰していく
ARMA(p,q)	減衰していく	減衰していく

Table3: ACF と PACF の特性 (参考: 沖本 (2010)[4])

これらの基準をもとに、COVID-19 新規感染者数の標本自己相関と標本偏自己相関のコレログラムを描画する。たとえば一次差分系列 (1day Diff Series) の自己相関 (ACF) が 7 日周期で残っており、ここでも 1 週間単位での変動を考慮する必要が示唆されている。

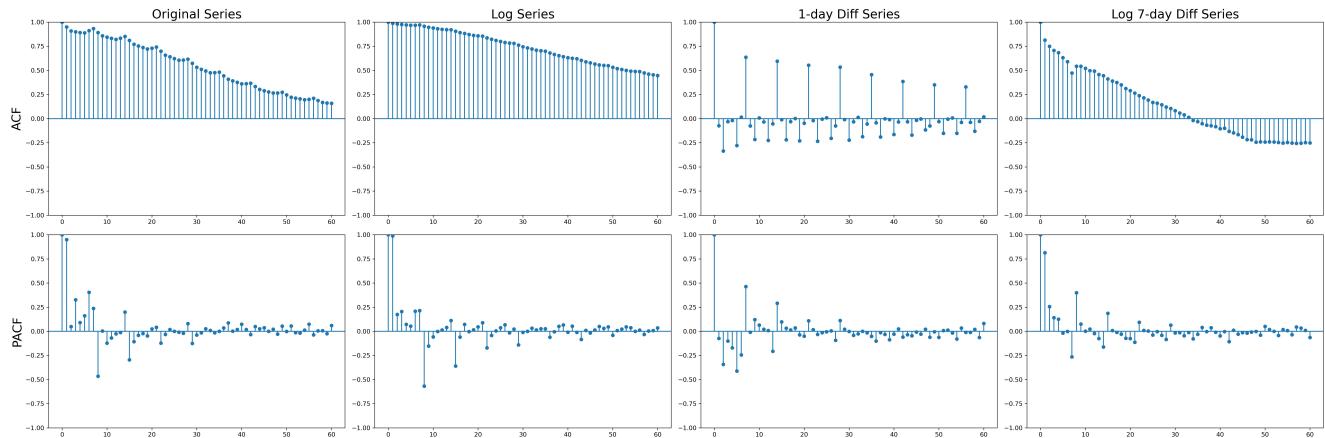


Figure6: 標本自己相関と標本偏自己相関のコレログラム

4.1.2 情報量規準

ここまで標本自己相関と標本偏自己相関を見てきたが, これだけの情報で ARMA モデルの次数を決めることは通常できない. そこで, 複数のモデルを作成し客観的な指標に従って最良のモデルを選択することが多い. 最尤法の推定結果を基に最適なモデルを選択する基準が**情報量規準** (information criterion) である.[4] 一般に, 次の通り定義される.

$$IC = -2L(\hat{\theta}) + p(T)k \quad (22)$$

ここで, $L(\hat{\theta})$ は対数尤度を最尤推定値で評価した最大対数尤度, T はモデルの推定に用いた標本サイズ, $p(T)$ は T の何らかの関数, $p(k)$ は推定に使用したパラメータの数である. 定義式右辺の第 1 項はモデルの当てはまりの良さを, 第 2 項はモデルの複雑さに対するペナルティを表す. 2 つはトレードオフの関係にある. パラメータを増やす以上に当てはまりが改善されなくなるとき, すなわち情報量規準を最小化するモデルが最適だといえる.

■AIC 本論文では以下の**赤池情報量規準** (AIC:akaike information criterion) を用いる.

$$AIC = -2L(\hat{\theta}) + 2k \quad (23)$$

これは, 情報量基準 (IC) における $p(T) = 2$ としたものである.[4]

4.1.3 単位根検定

AIC によるモデル選択により ARMA(p,q) の次数まではわかるが, ARIMA(p,d,q) のように d 次階差の同定ができない (ARMA などモデルについては次章で説明する). そこで行うのが**単位根検定** (unit root test) である.[9] ここで**単位根** (unit root) とは, 原系列が非定常過程であり, d 階差分をとった系列が定常になることがある.

4.1.4 ADF 検定

単位根検定のひとつ, ADF 検定 (Augmented Dickey-Fuller Test) は, 真のモデルを AR(p) モデルと仮定し, 以下の仮説検定を行う.[9][10]

- ・帰無仮説 $H_0: \rho = 0$ 単位根あり
- ・対立仮説 $H_1: \rho < 0$ 単位根なし

Augmented(拡張) とある通り, ADF 検定は AR(1) モデルを仮定した**DF 検定** (Dickey-Fuller Test) を拡張したものである. たとえば DF 検定の場合, 次章で紹介する AR(1) のモデル式を用いて次のように表すことができる.

$$y_t = \phi_1 y_{t-1} + \epsilon_t \quad (24)$$

ここで, y_t は時刻 t での観測値, ϕ_1 は自己回帰係数, y_{t-1} は時刻 $t-1$ での観測値, ϵ_t は誤差項で $\epsilon_t \sim i.i.d (0, \sigma^2)$, i.i.d は**独立同分布** (independent and identically distributed) を示す. $\phi_1 = 1$ ならば, y_t は誤差項 (ホワイトノイズ) の累積和で表現できる. すなわち, 前章で触れたランダムウォークであり, 単位根を持つことがわかる. さらにこれを AR(p) モデルへ一般化したのが, ADF 検定である.[9]

■検定結果 検定の結果、有意水準 1% で有意である。よって単位根がなく、差分を取らずとも定常性を持つことが確認できた。しかし、定常性の定義を考えると、定常過程はトレンドを持たない [4] はずである。Figure3 の図で確認した通り、原系列の時系列に定常性があるように見えないため KPSS 検定も行うこととした。

ADF Statistic	-3.86
p-value	0.0024
Critical Values	
1%	-3.436
5%	-2.864
10%	-2.568

Table4: ADF 検定結果

4.1.5 KPSS 検定

ADF 検定では 2 つの留意点がある。まず、誤差項が独立同一分布にしたがうという強い仮定が置かれている。そのため、Phillips-Perron により誤差項の時間への依存性や分散の不均一性を認めた **PP 検定** (Phillips-Perron test) が提案されている。[11] 詳細は沖本 (2010)[4] を確認されたい。また、ADF 検定において根が 1 に近いとき、帰無仮説が誤っているにも関わらず棄却できない第二種の過誤に陥る可能性も指摘されている。そこで、この欠陥を回避するための手法が提案された。[11] それが **KPSS 検定** (Kwiatkowski-Phillips-Schmidt-Shin Test) である。KPSS 検定では次のモデルを仮定する。

$$Y_t = \alpha + \beta t + \sum_{i=1}^t u_i + \epsilon_t \quad (25)$$

ここで、 α は定数項 (切片)、 β はトレンド項の係数、 $\sum_{i=1}^t u_i$ は時刻 i から時刻 t までの累積和を示し、 u_i はランダムウォーク (ランダムな変動) 成分、 ϵ_t はホワイトノイズ (誤差項) である。さらに次の仮説を考える。

- ・帰無仮説 $H_0: \sigma_u^2 = 0$ 定常 (トレンド定常または差分定常)
- ・対立仮説 $H_1: \sigma_u^2 \neq 0$ 単位根あり (非定常)

つまり、帰無仮説が棄却されれば、「たとえトレンドを除去しても単位根が残る」と結論づけられる。[9]

■検定結果 実際の検定結果は次の通りである。検定の結果、有意水準 1% で有意である。帰無仮説 (H_0) において「トレンド定常 (時間の経過によって平均値が変化するが、差分を取ると定常性を達成できる) や差分定常 (差分を取ることで定常過程になる)」という仮説が置かれている。しかし仮説が棄却されたいま、対立仮説 (H_1) の通り時系列には単位根が存在し、非定常過程である可能性が高い。非定常性の場合、時系列の平均や分散が時間の経過によって変化し続けることを示す。よって「単純な差分だけでは定常性を達成できない可能性が高い」ことを意味する。

KPSS Statistic	2.088
p-value	0.01
Critical Values	
10%	0.347
5%	0.463
2.5%	0.574
1%	0.739

Table5: KPSS 検定結果

4.1.6 同定のまとめ

ADF 検定/KPSS 検定が同時に有意であったことから、時系列データはトレンド定常でない可能性が高いと解釈できる。つまり、データには時間に依存するトレンドが存在するが、差分を取るなどして定常性を達成する必要性が高い。よって、トレンド除去や適切な差分を用いてデータを分析していく。

4.2 パラメータ推定

Box-Jenkins 法では、前節のモデル同定を経て、**パラメータ推定** (estimation) に移る。

4.2.1 最尤法

一般にモデルが θ をパラメータとするパラメトリックモデル $f(y) = f(y|\theta)$ の形をしているとき、対数尤度 l はパラメータ θ の関数と考えることができる。従って、以下の $\ell(\theta)$ を対数尤度関数と呼ぶ。

$$\ell(\theta) = \begin{cases} \sum_{n=1}^N \log f(y_n|\theta) & \text{独立の場合} \\ \log f(y_1, \dots, y_N|\theta) & \text{一般の場合} \end{cases} \quad (26)$$

この対数尤度関数 $\ell(\theta)$ は θ で定まるモデルの良さを評価した量である。よって $\ell(\theta)$ を最大にする θ を求めることで $f(y) = f(y|\theta)$ の最適なパラメータ値を定めることができる。このように対数尤度（あるいは尤度）を最大化することでパラメータを推定する方法を**最尤法** (Maximum likelihood method) といい、推定されたパラメータ $\hat{\theta}$ を**最尤推定値** (Maximum likelihood estimate) と呼ぶ。^[12] より一般には次のように表せる。

$$\hat{\theta}_{ML} = \arg \max_{\theta \in \Theta} L_n(\theta) = \arg \max_{\theta \in \Theta} \ell_n(\theta) \quad (27)$$

4.2.2 最小二乗法

また、真値と推定値との残差の平方和を最小にすることでパラメータを推定する方法もある。これを**最小二乗法** (least squares method) といい、推定されたパラメータを**最小二乗推定量** (Least squares estimate) と呼ぶ。

$$L(\theta) = \sum_{n=1}^N (y_n - \hat{y}_n(\theta))^2 \quad (28)$$

誤差項に正規分布を仮定したモデルの場合、最尤推定値と最小二乗推定値は一致し、解析的に解ける場合が多い。ただし、時系列分析においては、AR モデルなどの例外的な場合を除き、最尤推定値やその近似値は解析的に求めるは難しい。そこで、一般には**準ニュートン法** (quasi-Newton method) により数値最適化 (numerical optimization) のアルゴリズムを用いられる。^[12] また、次章以降では Python の **statsmodels** パッケージの利用や探索的パラメータサーチを行うため、これ以上の説明は割愛する。

4.3 モデル診断

Box-Jenkins 法では、前節のパラメータ推定を経て、**モデル診断** (diagnostic checking) に移る。

4.3.1 無相関の検定

正規性を次の通り表す。

$$\{U_t\} \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2) \quad (29)$$

観測値 $\{X_t | t \in \{1, \dots, n\}\}$ が AR(p) モデルに従うと仮定し、推定したモデルが適切である場合、モデルに仮定した正規性により、残差 $\{\hat{U}_t\}$ も（漸近的に）無相関となる。^[2] そこで、モデルの残差がランダムである（自己

相関がない) ことを確認する統計的検定方法が **Ljung-Box 検定** (Ljung-Box test)(別名かばん検定) である. ラグ m が十分に大きい場合, ある固定した自然数 m に対して, 次の統計的仮説検定を行う.

- ・帰無仮説 $H_0: \rho(1) = \dots = \rho(m) = 0$
- ・対立仮説 $H_1: \exists k \in \{1, \dots, m\} \text{ s.t. } \rho(k) \neq 0$

検定統計量 Q は次の通り定義する.

$$Q = n(n+2) \sum_{k=1}^m \frac{1}{n-k} \hat{\rho}(k)^2 \quad (30)$$

このとき, 有意水準を α とすると, 次の通り判断する.

$$Q > \chi_{m-p}^2(\alpha) \Rightarrow \text{棄却}, \quad Q \leq \chi_{m-p}^2(\alpha) \Rightarrow \text{採択} \quad (31)$$

Ljung-Box 検定は様々な次数で行い, そのすべてで有意な自己相関がないことを確認する.[9] なお,Ljung-Box 検定では, 上記の仮定のときに検定統計量 Q が $m-p$ のカイ二乗分布に収束することを利用している.[2]

4.3.2 正規性の検定

Jarque-Bera 検定 (Jarque-Bera test) は, 予測モデルの残差が正規分布に従うかどうかを検定する. 残差の分布の歪度 (Skewness) と尖度 (Kurtosis) を, 正規分布における理論上の歪度と尖度と比較して, 両者に差があるかどうかを判断するものである.[9] 統計的仮説検定は次の通りである.

- ・帰無仮説 H_0 : 残差が正規分布に従う
- ・対立仮説 H_1 : 残差が正規分布に従わない

また, 検定統計量 JB は次の通り定義する.

$$JB = \frac{n}{6} \left(S^2 + \frac{(K-3)^2}{4} \right) \quad (32)$$

ここで, n はサンプルサイズ, S は残差の歪度, K は残差の尖度を表す. なお, 本節で紹介した Ljung-Box 検定および Jarque-Bera 検定は, 後述する statsmodel のパッケージにおいても使用される.

5 統計モデル/Statistical Model

ここまで過程を踏まえ、実際の統計モデリングを行う。

5.1 統計モデリング概略

なおすでに長期(3年)、中期(1年)、短期(1ヶ月)の各期間で予測モデルを作成したが予測精度が著しく低かった。その理由として波ごとに新規感染者数の増減傾向が異なること[13]が考えられたため、期間を特定の波に[14][15]限定してここからモデリングを行っていく(なお疫学的な期間の定義により厚生労働省の資料によっても数日ずれがある)。

波	期間	日数	モデリング対象
第3波	2020-12-01 ~ 2021-02-28	90日	
第4波	2021-03-18 ~ 2021-07-11	116日	
第5波	2021-07-01 ~ 2021-09-30	92日	
第6波	2022-01-01 ~ 2022-03-31	90日	○
第7波	2022-07-01 ~ 2022-09-30	92日	
第8波	2022-11-30 ~ 2023-01-24	56日	

Table6: COVID-19 各波の期間 (参考: 厚生労働省 [14][15])

5.1.1 モデリングの対象期間

本レポートではこれらの期間の中でも第6波を対象としてモデルを作成していく。具体的には次の通り約2ヶ月半のデータをもとに、続く2週間の新規感染者数を予測するモデルである。特に検証データの前半は、学習データの傾向通り減少傾向だが、後半は一転して増加傾向が見られる。この変曲点をうまく再現できるかどうかが大きな鍵となる。

データセット	期間
学習データ	2022-01-01 ~ 2022-03-14
検証データ	2022-03-15 ~ 2022-03-31

Table7: 学習データと検証データの期間

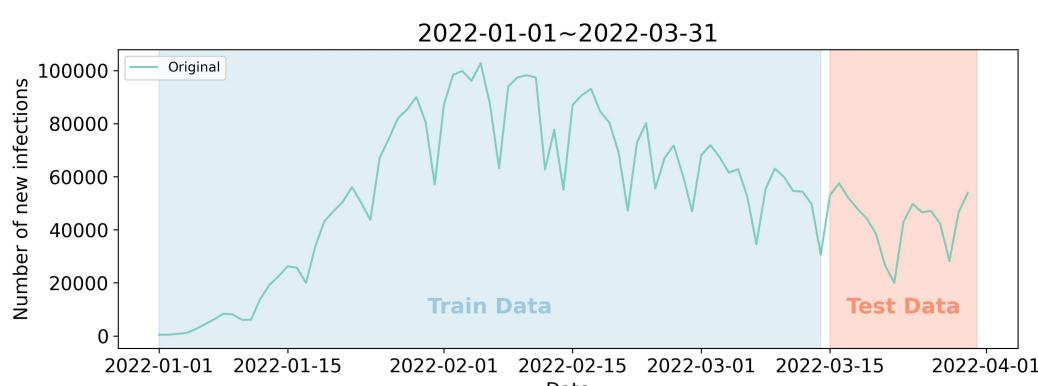


Figure7: 第6波における学習データと検証データの期間

5.2 前提知識 3. 統計モデル

前提知識として、時系列分析で使用される統計モデルの基本的な要素を紹介しておく。

5.2.1 AR(p): 自己回帰モデル

AR 過程 (Autoregressive process) の定義は「過程が自身の過去に回帰された形で表現される過程」である。一般化された AR(p) は p 期間までの過去の値に回帰する。^[4] 具体的には次の式で表される。

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \epsilon_t \quad (33)$$

ここで、 y_t は時刻 t の時系列値、 $\phi_1, \phi_2, \dots, \phi_p$ は自己回帰係数、 c は定数項、 ϵ_t はホワイトノイズの誤差項を表す。なお AR(p) モデルが定常であるための条件は、以下に記す AR 特性方程式のすべての解（複素解を含む）[10] が単位円外にあることである。本レポートではこれ以上深入りしないが、AR 特性方程式は次のように表される。^[4]

$$1 - \phi_1 z - \phi_2 z^2 - \cdots - \phi_p z^p = 0 \quad (34)$$

5.2.2 MA(q): 移動平均モデル

MA 過程 (Moving Average process) の定義は「ホワイトノイズの線形和で表すことができる過程」である。一般化された MA(q) は「現時点および過去の q 次点までのホワイトノイズの線形和で表すことができる過程」である。^[4] 具体的には次の式で表される。

$$y_t = \mu + \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \cdots + \theta_q \epsilon_{t-q} \quad (35)$$

$$(36)$$

ここで、 y_t は時刻 t の時系列値、 $\theta_1, \theta_2, \dots, \theta_q$ は移動平均係数、 μ は平均、 ϵ_t はホワイトノイズの誤差項を表す。なお、AR(p) モデルとは異なり、MA(q) モデルは常に定常である。

5.2.3 ARMA(p,q): 自己回帰移動平均モデル

ARMA 過程 (Autoregressive Moving Average process) の定義は「自己回帰項と移動平均項を両方含む過程」である。AR 部分は過去の値が現在の値に与える影響を、MA 部分は過去の誤差が現在の値に与える影響をモデル化する。なお、既述の通り AR 過程が常に定常となるわけではないので、AR を含む ARMA 過程も同様に常に定常というわけではない。ただし、次の論理から AR(自己回帰項) が定常であれば、ARMA が定常であると帰着できる。

- 1. 一般に、定常過程に定常過程を加えても定常過程となる
- 2. AR 過程と MA 過程の和は ARMA 過程となる
- 3. MA 過程は AR(∞) で書くことができる

上記 3 は反転可能性 (invertible) と呼ばれ、その条件は MA 特性方程式で確認できる。^[4] なお ARMA(p,q) は具体的には次の式で表される。

$$Y_t = \phi_1 Y_{t-1} + \phi_2 Y_{t-2} + \cdots + \phi_p Y_{t-p} + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \cdots + \theta_q \epsilon_{t-q} + \epsilon_t \quad (37)$$

ここで、 Y_t は時刻 t の時系列値、 $\phi_1, \phi_2, \dots, \phi_p$ は自己回帰係数、 $\theta_1, \theta_2, \dots, \theta_q$ は移動平均係数、 ϵ_t はホワイトノイズの誤差項を表す。なお、その他の統計モデルは次節以降で紹介する。

5.2.4 モデリングの方針

次節より、これまで説明してきた ARMA モデルをベースライン (精度の評価基準となるモデル) として、事前の同定で最低限必要とが考えられた差分系列を考慮した ARIMA モデルおよび季節階差を考慮した SARIMA モデルへと段階的に近づけていき、精度の変化を確認する。

5.3 予測精度

さきに予測結果を示す。実際のモデリングによる予測精度は次の通りである。なお、ここで **RMSE** は二乗平均平方根誤差 (Root Mean Squared Error)、すなわち平均二乗誤差 (Mean Squared Error) の平方根である。また、**MAE** は平均絶対誤差 (Mean Absolute Error)、**MAPE** は平均絶対パーセンテージ誤差 (Mean Absolute Percentage Error) である。これらは時系列分析に限らず一般的な評価指標であるため、説明は割愛する。なお、RMSE と MAE については小数点第 1 位以下を四捨五入する。

指標	ARMA	ARIMA	SARIMA
RMSE	10,897	10,330	8,516
MAE	8,696	8,083	7,026
MAPE	24.02	23.62	20.21

Table8: モデルの精度比較

5.3.1 パラメータ推定

また、パラメータについてはグリッドサーチで最適なパラメータを検証している。

モデル	パラメータ	AIC	Ljung-Box(p 値)	Jarque-Bera(p 値)
ARIMA	$(p,d,q) = (2,1,4)$	1565	0.77	0.24
SARIMA	$(p,d,q)(P,D,Q)s = (4,2,0)(1,1,0)14$	1247	0.36	0.42

Table9: 最適なパラメータ,AIC 値, 検定結果

5.4 ARIMA モデル

非定常過程に対して ARMA モデルを適用することはできない。^[9] そこで、非定常データに対して d 回の差分を取ることでデータを定常に変換するのが **ARIMA モデル**、すなわち自己回帰和分移動平均モデル (autoregressive integrated moving average model) である。その後、自己回帰 (AR) 特性と移動平均 (MA) 特性を組み合わせたのが ARIMA(p,d,q) モデルである。ここでラグ演算子 (Backward shift) を用いると $B y_t = y_{t-1}$ と書ける。さらに 2 点ずらす場合、 $B^2 y_t = y_{t-2}$ と累乗表記で表現できる。よって ARIMA モデルは次の式で表される。^[9]

$$(1 - \sum_{i=1}^p \phi_i B^i) \Delta^d y_t = (1 + \sum_{j=1}^q \theta_j B^j) \epsilon_t \quad (38)$$

ここで, y_t は時刻 t の時系列値, $\phi_1, \phi_2, \dots, \phi_p$ は自己回帰係数, $\theta_1, \theta_2, \dots, \theta_q$ は移動平均係数, B はバックソフト演算子, Δ^d は d 階差分表記, ϵ_t はホワイトノイズの誤差項, c は定数項を表す. これをよりシンプルに表記すると, 次の通りである.[9]

$$\phi(B)\Delta^d y_t = \theta(B)\epsilon_t \quad (39)$$

5.4.1 予測結果の可視化

ARIMA モデルによる予測結果は次の通りである.

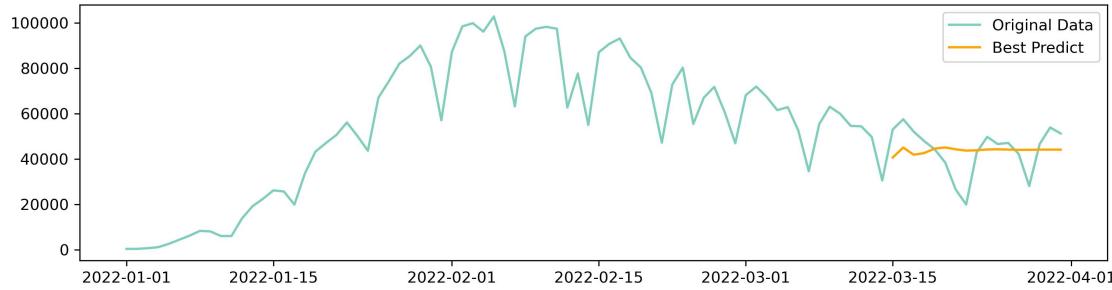


Figure8: ARIMA モデル全期間

予測期間の拡大図を以下に示す.

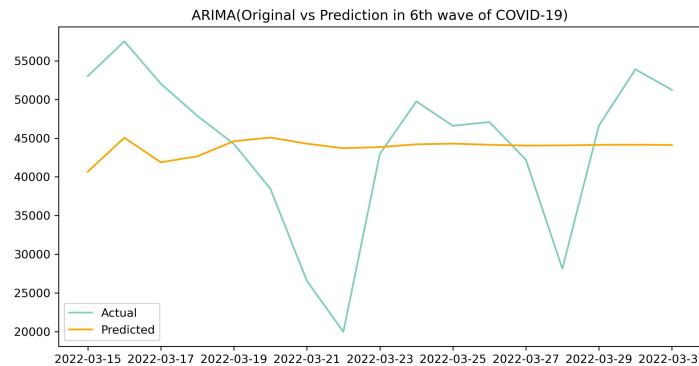


Figure9: ARIMA モデル予測期間

5.4.2 結果の解釈

区間内の平均をとったような、直線に近い予測になっている. 誤差関数上でこそ SARIMA モデルと比べて精度は大きく変わらないが、実際は区間内の変動が全く表現できており、総じて精度の低いモデルといえる.

5.5 SARIMA モデル

ARIMA モデルに季節性成分を加えたものが **SARIMA モデル**, すなわち**季節性自己回帰和分移動平均モデル** (seasonal autoregressive integrated moving average model) である. 季節的なパターンを持つ時系列データに有効である. このモデルは通常, $\text{SARIMA}(p,d,q)(P,D,Q)[s]$ と表記され, 季節性成分の周期を s で表す. SARIMA モデルは一般的に次のように表される.

$$(1 - \sum_{i=1}^p \phi_i B^i)(1 - \sum_{I=1}^P \Phi_I B^{sI}) \Delta^d \Delta_s^D y_t = (1 + \sum_{j=1}^q \theta_j B^j)(1 + \sum_{J=1}^Q \Theta_J B^{sJ}) \epsilon_t \quad (40)$$

ここで, y_t は時刻 t の時系列値, $\phi_1, \phi_2, \dots, \phi_p$ は非季節性自己回帰係数, $\Phi_1, \Phi_2, \dots, \Phi_P$ は季節性自己回帰係数, $\theta_1, \theta_2, \dots, \theta_q$ は非季節性移動平均係数, $\Theta_1, \Theta_2, \dots, \Theta_Q$ は季節性移動平均係数, B はバックシフト演算子, Δ^d は非季節性の d 階差分表記, Δ_s^D は季節性の D 階差分表記, ϵ_t はホワイトノイズの誤差項, c は定数項を表す. なお, よりシンプルに表記すると次の通り表すことができる (次節に補足あり).[9]

$$\phi(B)\Phi(B)\Delta^d\Delta_s^D y_t = \theta(B)\Theta(B)\epsilon_t \quad (41)$$

5.5.1 予測結果の可視化

予測結果は次の通りである.

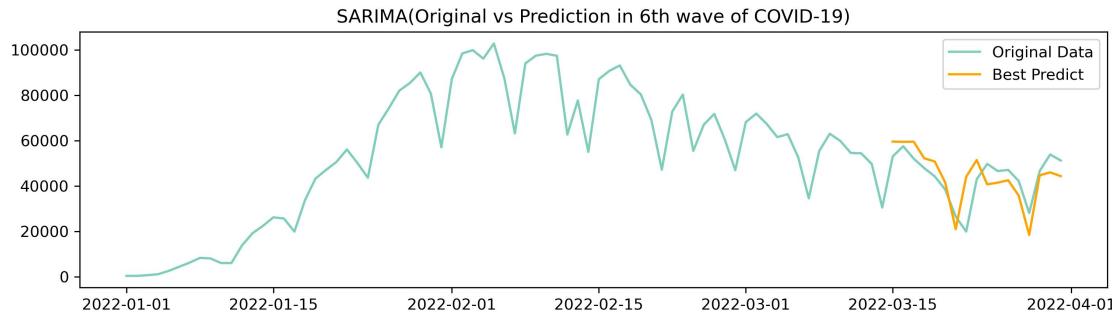


Figure10: SARIMA モデル全期間

予測期間の拡大図を以下に示す.

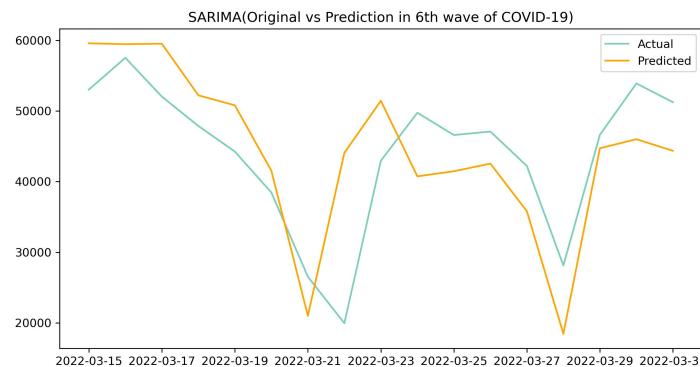


Figure11: SARIMA モデル予測期間

5.5.2 結果の解釈

ARIMA モデルと比べると周期変動をうまく捉えたモデルとなっている。トレンドに引っ張られており、予測期間後半の上昇傾向への転換は表現できていないが、当てはまりは大きく改善した。

5.6 Python による実装

Python での実装に際しては ARMA,ARIMA,SARIMA では,statsmodels.tsa.arima モジュールの **ARIMA** クラスを用いた.[16] なお、主なハイパーパラメータは次の通りである。

パラメータ	説明	デフォルト値
order	モデルの (p,d,q) オーダー	(0, 0, 0)
seasonal_order	季節成分の (P,D,Q,s) オーダー	(0, 0, 0, 0)
exog	外生変数	None
trend	決定論的トレンドのパラメータ	None

Table10: ARIMA Basic Parameters(参考: 公式サイト [16])

5.6.1 補足

本章の終わりに、補足説明をしておく。特に SARIMA モデルは非季節性と季節性の要素が同時に表現されているため、具体的なイメージを持ちたい。

■バックシフト演算子 バックシフト演算子 B は、現時点の値から過去の任意の時点の値を表現した演算子である。時系列データを操作する物理的な演算子というよりは、モデル表現を簡潔にし、理解しやすくするための数理的表現方法だと位置付けられる.[9]

■季節性自己回帰係数 SARIMA モデルの季節性自己回帰部分における係数 Φ (大文字の P) は、特定の季節性周期 (たとえば年次、四半期、月次、週次など) における過去の値の影響を現在の値に反映させる。

■季節性移動平均係数 SARIMA モデルの季節性移動平均部分における係数 Θ (大文字の Q) は、過去の季節性ホワイトノイズ (季節性成分に特有のランダムな変動) である。つまり、季節性パターンを除去した後に残るランダムなノイズの影響を現在の値に反映させる。

■季節性差分 SARIMA モデルにおける季節性差分 (大文字の D) は、時系列データに季節性のパターン (s) が存在する場合に使用される (例えば、毎年同じ月に顕著な変動が見られる等)。「季節性の次数 (s) を固定して、SARIMA(p,d,q)(P,1,Q)[s] とすれば、季節階差をとったデータに対する ARIMA モデルとみなせる。[9] あくまでも s は周期性パターンの表現であり、 D は何回差分を取るかの「回数」を意味するといえる。なお、「日次データの場合、SARIMA でなく ARIMAX モデルにより外生変数として処理することでうまくいく場合もある」など、SARIMA モデルの使い分けや直感的理解は馬場 (2018)[9] を参照されたい。

6 状態空間モデル/State-Space Model

本章では状態空間モデル (state-space model) による予測を行う.

6.1 予測精度

さきに予測結果を示す. 実際のモデリングによる予測精度は次の通りである.

指標	ARMA	ARIMA	SARIMA	StateSpace
RMSE	10,897	10,330	8,516	8,448
MAE	8,696	8,083	7,026	4,777
MAPE	24.02	23.62	20.21	15.83

Table11: モデルの精度比較

6.2 状態空間モデル

状態空間モデルは、「状態」と「観測」の2つの方程式で表現する. 非定常過程でも扱えるため、汎用性が高い.[2] y_n を ℓ 変量の時系列とする時、次のように時系列を表現するモデルを、状態空間モデルと呼ぶ.[6] なお総じて、状態空間モデルは、時系列分析で用いる様々なモデルを統一的に扱うことができる。たとえば、AR, MA, ARMA など線形モデルのほとんどが状態空間モデル(次式(42)のシステムモデル)に対応づけられる.[2]

6.2.1 システムモデル

システムモデルは、状態方程式によって記述される。なお、 \mathbf{x}_t は状態 (state) と呼ばれる。

$$\mathbf{x}_t = \mathbf{F}_t \mathbf{x}_{t-1} + \mathbf{G}_t \mathbf{v}_t \quad (42)$$

\mathbf{x}_t : 時点 t において直接観測できない k 次元確率ベクトル
 \mathbf{F}_t : 行列 ($k \times k$)
 \mathbf{G}_t : 行列 ($k \times m$)
 \mathbf{v}_t : 状態ノイズベクトル ($m \times 1$)

6.2.2 観測モデル

観測モデルは、観測方程式によって記述される。

$$\mathbf{y}_t = \mathbf{H}_t \mathbf{x}_t + \mathbf{w}_t \quad (43)$$

\mathbf{y}_t : 時刻 t において直接観測できる ℓ 次元観測ベクトル
 \mathbf{H}_t : 行列 ($\ell \times k$)
 \mathbf{w}_t : 観測ノイズベクトル ($\ell \times 1$)

特に, \mathbf{F}_t と \mathbf{H}_t が線形であり, \mathbf{v}_t と \mathbf{w}_t が正規分布に従うとき, 線形ガウス状態空間モデルと呼ぶ. 線形ガウス状態空間モデルでは, 差分を取る必要がないなどメリットはあるが, 正規分布を仮定しているため対数変換を施すなどが適宜必要である.[9] また, \mathbf{v}_t , \mathbf{w}_t を次の通り定義する。

$$\mathbf{v}_t \sim \mathcal{N}(0, \mathbf{Q}_t) \quad (44)$$

$$\mathbf{w}_t \sim \mathcal{N}(0, \mathbf{R}_t) \quad (45)$$

ここで \mathbf{Q}_t と \mathbf{R}_t はそれぞれ状態ノイズと観測ノイズの共分散行列 $(m \times m), (\ell \times \ell)$ を示す。[2][12]

6.2.3 モデルの関係性イメージ図

ここで簡単に2つのモデルの関係性を図示する. なお時系列の観測値ベクトル $\mathbf{y}_j = \{y_1, \dots, y_t\}$ はある時点 t までの観測値を行列として表現したものである.

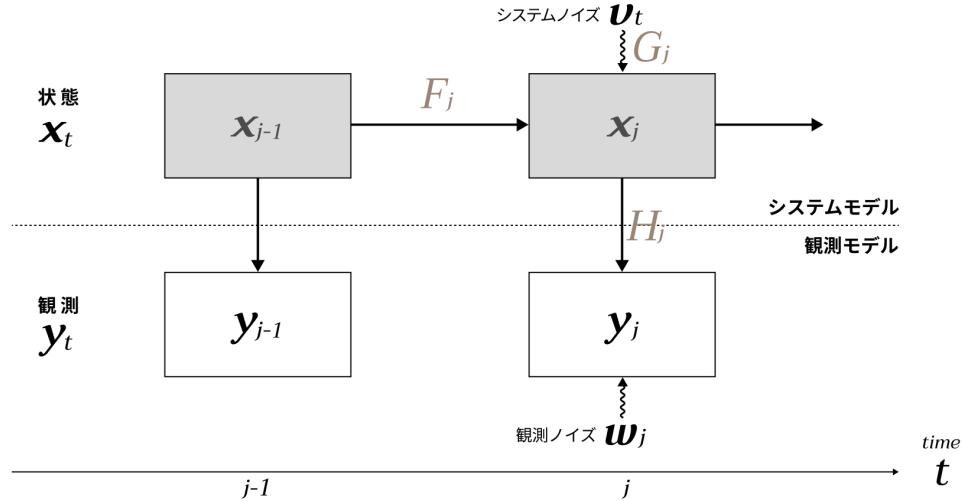


Figure12: 状態空間モデルイメージ (参考: 白石 (2022)[2])

6.2.4 構造と解釈

まず私たちが実際に観測できるのは, Figure(12) にある \mathbf{y}_t である. また, \mathbf{y}_t は過去の \mathbf{y}_{t-1} でなく, システムモデル \mathbf{x}_t と誤差 (ノイズ) である観測ノイズ \mathbf{w}_t に定められる. つまり時系列構造としては式 (42) の \mathbf{x}_t に, 式 (43) の \mathbf{x}_t を代入することで, 間接的に時系列構造が保たれる.[2] 別の表現をするならば, 行列 \mathbf{F}_t は過去を未来へ写す線形写像 (変換)[17] であり, 動的に変化する時間 t の関数 [12] でもある. また, \mathbf{H}_t は潜在 (内部) 状態から観測可能な状態への変換である. 状態空間モデルはシステムの不確実性やランダムな変動を取り入れながら, 確率的な推移を記述しているといえる.

6.2.5 状態

過度に複雑でなく, 適切に予測できるモデルは, 外挿にも役立つような本質的な変動を抽出し, 一次的でランダムな変動を分離して表現していると言える. たとえば物理的なシステムでは, 過去の情報は現在を経由して将来に伝えられる. したがって, 過去の情報のうち将来の動きに関連する部分は何らかの形で現時点に集約されていると考えることができる. この集約された情報が「状態」だといえる.

6.2.6 状態推定

時系列の観測値ベクトル $\mathbf{y}_j = \{y_1, \dots, y_t\}$ と状態空間モデルが与えられたとき, 状態 x_t を推定する問題は, 「状態推定」と呼ばれる. 観測区間 $[1, j]$ と, 推定する状態の時点 t との大小関係により [2], 以下のように 3 つの場合に分けられる. 時系列の予測, 補間, パラメータ推定, 成分分解などにおいて状態推定を利用して統一的に解決できる.[12]

$$t < j : \text{予測}, \quad t = j : \text{フィルタ}, \quad t > j : \text{平滑化}$$

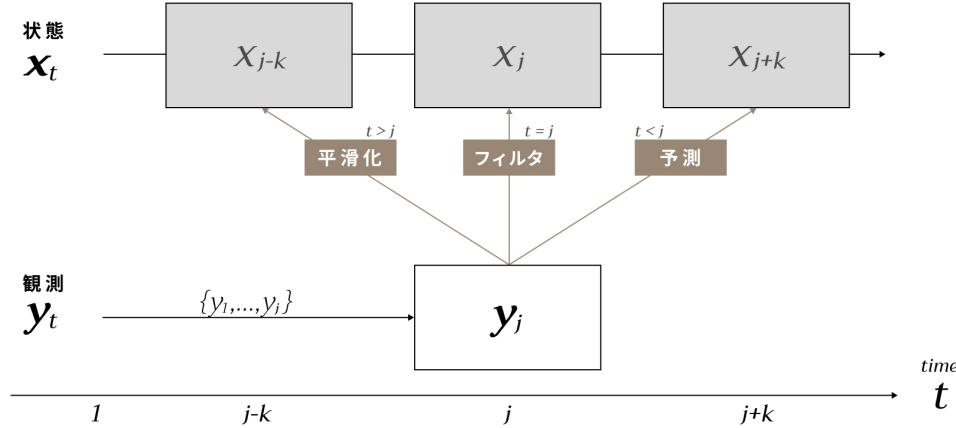


Figure13: 状態推定の 3 類型 (参考: 北川 (2017)[12])

6.3 アルゴリズム

最終的にこの状態を推定する問題は, 観測値ベクトル \mathbf{y}_t が与えられたときの, 状態 (スカラ) x_t の条件付き分布:

$$p(x_t | \mathbf{y}_j) \quad (46)$$

を求める問題に帰着する. このとき, 予測分布 $p(x_t | \mathbf{y}_{t-1})$ とフィルタ分布 $p(x_t | \mathbf{y}_t)$ は状態の定義から, 以下が成り立つ.

$$p(x_t | x_{t-1}, \mathbf{y}_{t-1}) \quad (47)$$

$$= p(x_t | x_{t-1}), p(y_t | x_t, \mathbf{y}_{t-1}) \quad (48)$$

$$= p(y_t | x_t) \quad (49)$$

よって, 以下の式により逐次的に計算できる.[12]

6.3.1 一期先予測/prediction

$$p(x_t | \mathbf{y}_{t-1}) = \int p(x_t | x_{t-1}) p(x_{t-1} | \mathbf{y}_{t-1}) dx_{t-1} \quad (50)$$

6.3.2 フィルタ/filtering

$$p(x_t | \mathbf{y}_t) = \frac{p(y_t | x_t) p(x_t | \mathbf{y}_{t-1})}{p(y_t | \mathbf{y}_{t-1})} \quad (51)$$

なお分母は, $p(y_t|\mathbf{y}_{t-1}) = \int p(y_t|x_t)p(x_t|\mathbf{y}_{t-1}) dx_t$ である. (52)

6.3.3 平滑化/smoothing

ここでは固定区間平滑化を示す.

$$p(x_t|\mathbf{y}_T) = p(x_t|\mathbf{y}_t) \int \frac{p(x_{t+1}|\mathbf{y}_T)p(x_{t+1}|x_t)}{p(x_{t+1}|\mathbf{y}_n)} dx_{t+1} \quad (53)$$

平滑化の数式は, 以下の要素から構成されていることがわかる.

- | |
|---|
| <ol style="list-style-type: none"> 1. 状態 x_t のフィルタ分布 : $p(x_t \mathbf{y}_t)$ 2. 状態 x_{t+1} の一期先予測分布 : $p(x_{t+1} \mathbf{y}_t)$ 3. 状態 x_{t+1} の平滑化分布 : $p(x_{t+1} \mathbf{y}_T)$ |
|---|

なお, データの終端では, 以下が成り立つ.

$$\text{平滑化分布: } p(x_t|\mathbf{y}_T) = \text{フィルタ分布: } p(x_t|\mathbf{y}_t) \quad (t = T) \quad (54)$$

つまり, 逐次フィルタにより予測分布とフィルタ分布をデータの終端まですべて求めておくと, 時間軸と逆方向に平滑化分布を計算できる.[12]

6.4 カルマンフィルタ

状態空間モデルでは, 逐次的な計算アルゴリズムによって, 状態 \mathbf{x}_t の条件付き周辺分布を効率的に計算できる. このアルゴリズムをカルマンフィルタ (Kalman filter) という.[2] 観測値 \mathbf{y}_j が与えられたうえでの状態 \mathbf{x}_t の条件付き期待値と条件付き分散共分散行列を次の通り表す.

$$x_{t|j} = \mathbb{E}[\mathbf{x}_t|\mathbf{y}_j] \quad (55)$$

$$\mathbb{V}[\mathbf{x}_t|\mathbf{y}_j] = \mathbb{E}[(\mathbf{x}_t - x_{t|j})(\mathbf{x}_t - x_{t|j})^\top] \quad (56)$$

ここで, ノイズ (44)(45) は正規分布に従うため, 状態 \mathbf{x}_t の条件付き周辺分布の推定は, $\mathbf{x}_{t|j}$, $\mathbb{V}_{t|j}$ の推定問題となる.[2] よってカルマンフィルタは, 観測値 \mathbf{y}_j , 初期値 $x_{j|j}$ および $\mathbb{V}_{j|j}$ を入力として, $x_{t|j}$, $\mathbb{V}_{t|j}$ を出力するアルゴリズムであり, 次の**1期先予測とフィルタ**の組み合わせを交互に繰り返すことで逐次求めることができる.[2]

■1期先予測:

$$\mathbf{x}_{t|t-1} = \mathbf{F}_t \mathbf{x}_{t-1|t-1} \quad (57)$$

$$\mathbb{V}_{t|t-1} = \mathbf{F}_t \mathbb{V}_{t-1|t-1} \mathbf{F}_t^\top + \mathbf{G}_t \mathbf{Q}_t \mathbf{G}_t^\top \quad (58)$$

- | | |
|---|---|
| $\mathbf{x}_{t t-1}$: 時点 t での状態の予測値
$\mathbb{V}_{t t-1}$: 時点 t での予測の不確実性
\mathbf{G}_t : プロセスノイズ行列 | $\mathbf{x}_{t-1 t-1}$: 時点 $t-1$ での状態の推定値
\mathbf{F}_t : 状態遷移行列
\mathbf{Q}_t : プロセスノイズの共分散行列 |
|---|---|

$\mathbb{V}_{t|t-1}$ は次の式より導ける.

$$\begin{aligned}
\mathbb{V}_{t|t-1} &= \mathbb{E}[(\mathbf{x}_t - \mathbf{x}_{t|t-1})(\mathbf{x}_t - \mathbf{x}_{t|t-1})^\top] \\
&= \mathbb{E}[(\mathbf{F}_t \mathbf{x}_{t-1} + \mathbf{G}_t \mathbf{v}_t - \mathbf{F}_t \mathbf{x}_{t-1|t-1})(\mathbf{F}_t \mathbf{x}_{t-1} + \mathbf{G}_t \mathbf{v}_t - \mathbf{F}_t \mathbf{x}_{t-1|t-1})^\top] \\
&= \mathbf{F}_t \mathbb{E}[\mathbf{x}_{t-1} \mathbf{x}_{t-1}^\top] \mathbf{F}_t^\top + \mathbf{G}_t \mathbb{E}[\mathbf{v}_t \mathbf{v}_t^\top] \mathbf{G}_t^\top \\
&\quad - \mathbf{F}_t \mathbb{E}[\mathbf{x}_{t-1} \mathbf{x}_{t-1|t-1}^\top] \mathbf{F}_t^\top - \mathbf{F}_t \mathbb{E}[\mathbf{x}_{t-1|t-1} \mathbf{x}_{t-1}^\top] \mathbf{F}_t^\top \\
&\quad + \mathbf{F}_t \mathbb{E}[\mathbf{x}_{t-1|t-1} \mathbf{x}_{t-1|t-1}^\top] \mathbf{F}_t^\top \\
&\quad + \mathbf{G}_t \mathbb{E}[\mathbf{v}_t (\mathbf{x}_{t-1} - \mathbf{x}_{t-1|t-1})^\top] \mathbf{F}_t^\top \\
&\quad + \mathbf{F}_t \mathbb{E}[(\mathbf{x}_{t-1} - \mathbf{x}_{t-1|t-1}) \mathbf{v}_t^\top] \mathbf{G}_t^\top \\
&= \mathbf{F}_t \mathbb{V}_{t-1|t-1} \mathbf{F}_t^\top + \mathbf{G}_t \mathbf{Q}_t \mathbf{G}_t^\top
\end{aligned}$$

導出に際して、基本的な線形代数のルールおよび定義式 (44) より下記が関連している.

- ・状態方程式は $\mathbf{x}_t = \mathbf{F}_t \mathbf{x}_{t-1} + \mathbf{G}_t \mathbf{v}_t$
- ・状態ベクトル \mathbf{x}_t の予測誤差は $\mathbf{x}_t - \mathbb{E}[\mathbf{x}_t] = \mathbf{x}_t - \mathbf{x}_{t|t-1} = \mathbf{x}_t - \mathbf{F}_t \mathbf{x}_{t-1|t-1}$
- ・ $\mathbb{E}[\mathbf{v}_t (\mathbf{x}_{t-1} - \mathbf{x}_{t-1|t-1})^\top] = \mathbb{E}[\mathbf{v}_t] \mathbb{E}[(\mathbf{x}_{t-1} - \mathbf{x}_{t-1|t-1})^\top] = \mathbf{0} \quad \because \mathbb{E}[\mathbf{v}_t] = \mathbf{0}$
- ・ $\mathbb{E}[\mathbf{v}_t \mathbf{v}_t^\top] = \mathbf{Q}_t$
- ・仮定より \mathbf{v}_t と \mathbf{x}_{t-1} , あるいは \mathbf{v}_t と $\mathbf{x}_{t-1|t-1}$ は独立であり, 積の期待値は $\mathbf{0}$

■フィルタリング (更新):

$$\mathbf{x}_{t|t} = \mathbf{x}_{t|t-1} + \mathbf{K}_t (\mathbf{y}_t - \mathbf{H}_t \mathbf{x}_{t|t-1}) \quad (59)$$

$$\mathbb{V}_{t|t} = (\mathbf{I} - \mathbf{K}_t \mathbf{H}_t) \mathbb{V}_{t|t-1} \quad (60)$$

$$\mathbf{K}_t = \mathbb{V}_{t|t-1} \mathbf{H}_t^\top (\mathbf{H}_t \mathbb{V}_{t|t-1} \mathbf{H}_t^\top + \mathbf{R}_t)^{-1} \quad (61)$$

\mathbf{K}_t : カルマンゲイン	\mathbf{H}_t : 観測行列
\mathbf{y}_t : 時点 t の観測値	\mathbf{R}_t : 観測ノイズの共分散行列
$\mathbf{x}_{t t}$: 更新後の時点 t の状態の推定値	$\mathbb{V}_{t t}$: 更新後の時点 t の不確実性
\mathbf{I} : 単位行列	

特に \mathbf{K}_t はカルマンゲイン (Kalman gain) という. 観測データをどの程度, 状態推定値に反映させるかの重みとして機能する. 1期先予測とフィルタを繰り返すことで逐次的に任意の時点の状態の条件付き分布を推定できる.

6.5 Python による実装

Python での実装に際して状態空間モデルでは, `textbfstatsmodels.tsa.statespace` モジュールの **SARIMAX** クラス [18] を用いた. なおアルゴリズム上では, 推定に状態空間モデルを使用している. また, 季節性効果 (seasonal effects), トレンド多項式 (trend polynomials), 外生回帰変数 (exogenous regressors) を指定できる. なお, 主なハイパーパラメータは次の通りである.

パラメータ	説明	デフォルト値
exog	外生変数の配列	None
order	モデルの (p,d,q) オーダー	(1, 0, 0)
seasonal_order	季節成分の (P,D,Q,s) オーダー	(0, 0, 0, 0)
trend	決定論的トレンドのパラメータ	None
measurement_error	観測誤差の仮定有無	False
mle_regression	最尤法による外生変数の回帰係数の推定	True

Table12: SARIMAX の主なパラメータ (参考:statsmodels developers[18])

6.6 特徴量エンジニアリング

状態空間モデルでは, 時系列データの潜在的なダイナミクスを捉えることが重要だと考えられる. そこで状態空間モデルを記述する上で, 原系列と対数変換系列の両方に対して, 過去の観測値が現在に与える影響を表す「ラグ特徴量」と, データのトレンドや季節性を表す「変化率特徴量」の生成を行った. これらの特徴量を使用することで, モデルはデータの複雑なパターンをより詳細に学習して, 予測精度が向上をめざす. 具体的には次のような特徴量を追加した.

■原系列に対するラグ特徴量 原系列 y_t に対するラグ特徴量で, 過去の時点の値を表す.

$$\text{Lag}_1 y_t = y_{t-1} \quad (62)$$

$$\text{Lag}_7 y_t = y_{t-7} \quad (63)$$

■原系列に対する前日からの変化率特徴量 原系列の前日からの変化率で, 連続する観測値の差の割合を表す.

$$\text{ChangeRate } y_t = \frac{y_t - y_{t-1}}{y_{t-1}} \quad (64)$$

■対数系列に対するラグ特徴量 対数変換した系列 $\log(y_t)$ に対するラグ特徴量で, 同様に過去の時点の値を表す.

$$\text{LogLag}_1 y_t = \log(y_{t-1}) \quad (65)$$

$$\text{LogLag}_7 y_t = \log(y_{t-7}) \quad (66)$$

■対数系列に対する前日からの変化率特徴量 対数変換した系列の前日からの変化率で、対数値の差の割合を表す。

$$\text{LogChangeRate } y_t = \frac{\log(y_t) - \log(y_{t-1})}{\log(y_{t-1})} \quad (67)$$

6.7 予測結果の可視化

最後に状態空間モデルでの予測結果を可視化する。

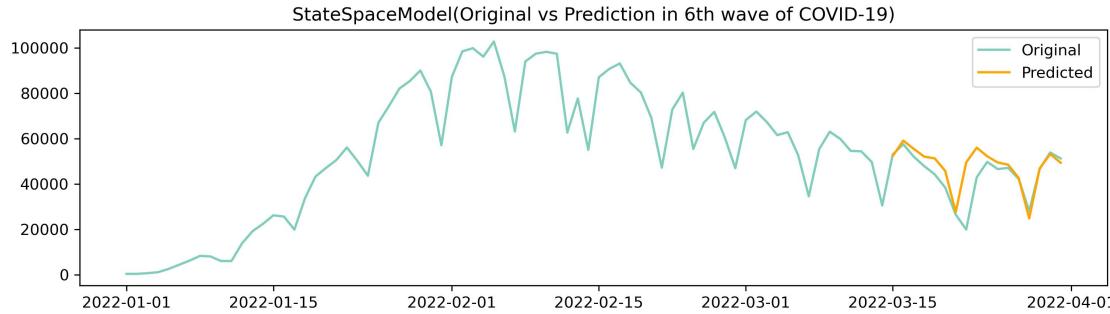


Figure14: 状態空間モデル 1

予測期間の拡大図を以下に示す。

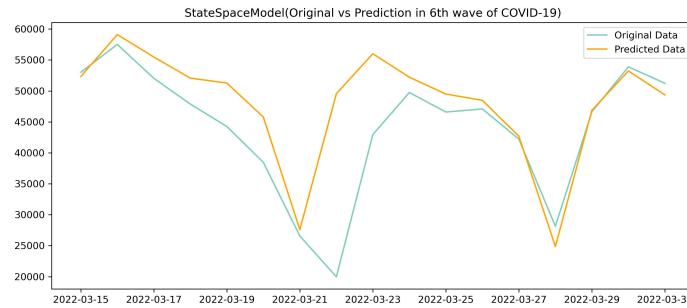


Figure15: 状態空間モデル 2

6.7.1 結果の解釈

多少の誤差はあるが、うまく周期性をとらえた予測ができている。グラフ形状から推察するに、直前 2 週間の影響を強く反映している。これまで同様、減少傾向を予測していることに変わりはないが、その減少幅は非常に穏やかでほぼ横ばいを示している。

7 機械学習モデル/Machine Learning Model

本章では機械学習モデルによる予測を行う。具体的には **Random Forest(RF)**, **Light GBM(LGBM)** を用いた。

7.1 予測精度

さきに予測結果を示す。実際のモデリングによる予測精度は次の通りである。なお上記 2 種類の機械学習モデルについては、第 6 波で作成したモデルを用いて、それぞれに第 7 波でのデータも予測し、汎化性能を検証している。詳しくは後述する。

指標	ARMA	ARIMA	SARIMA	SS	RF	RF7th	LGBM	LGBM7th
RMSE	10,897	10,330	8,516	8,448	2,797	1,684	2,461	2,704
MAE	8,696	8,083	7,026	4,777	1,629	1,469	1,886	2,140
MAPE	24.02	23.62	20.21	15.83	5.78	3.21	5.01	4.81

Table13: モデル別の精度比較

7.2 Random Forest

Random Forest(RF) は多数の回帰木を組み合わせることで、個々の回帰木の過学習を抑制しながらモデルを構築できる。複数の木による平均化と、多様な回帰木による木同士の相関低減により、バギングを改良し分散低減効果が高まるためである。特徴は次の通りである。^{[19][20]}

■**アンサンブル学習** : 多数の弱学習器 (weak learner) での合議 (committee) により、各木の予測結果を平均化する (決定木による分類タスクの場合は多数決)。

■**バギング (Bootstrap Aggregating)** : 各回帰木はデータセットからランダムに選ばれたブートストラップ標本を使用して訓練する。木を十分深くすれば、バイアスは相対的に低くなる。

■**推定値** : 生成された木は同一の分布に従うため、ブートストラップ標本の平均値は、個々の木の期待値と等しい。なお、後述するブースティングを用いた手法は、バイアスを除去するように木を成長させため、同一の分布とならない。

■**特徴量選択** : 各分岐 (node) で分割を決定する際に、ランダムに選ばれた一部の特徴量のみ考慮する。

7.3 アルゴリズム

Random Forest のアルゴリズムは次の通りである.[19][20]

1. $m = 1$ から M 個までの回帰木 T に対して以下を繰り返す
 - (a) N 個の d 次元の学習データからのブートストラップ標本 Z_m を取り出す
 - (b) 以下の a~c において Z_m を学習データとしてノード t を分割し回帰木 T_m を成長させる
 - i. d 個の特徴量から d' 個の特徴を選択する ($d' = \lfloor \sqrt{d} \rfloor$ 推奨)
 - ii. d' 個の特徴から最適な変数と分割点を選ぶ
 - iii. ノード t を $\text{left}(t)$ と $\text{right}(t)$ に分岐する
2. ランダムフォレスト $\{T_m\}_{m=1}^M$ を出力する
3. 入力データ x に対して $\hat{f}_{(x)}^M = \frac{1}{M} \sum_{m=1}^M T_m(x)$

なお、分類問題の場合、ステップ 3 において、 m 番目の木の識別結果を、 $y_m(x) \in \{\hat{C}_1, \dots, \hat{C}_K\}$ とする。また、 $\{T_m\}_{m=1}^M$ の識別結果を $\hat{C}_i = \arg \max_j |\hat{C}_j|$ とする。 $|\hat{C}_j|$ はクラス C_j と判断した木の数である。

7.4 Python による実装

モデリングに際しては、scikit-learn モジュールの `sklearn.ensemble.RandomForestRegressor` クラスを用いた。なお、基本的なハイパーパラメータは次の通りである.[21] また以下の `min_samples_leaf` は、回帰においてモデルを滑らかにする効果がある。

パラメータ	説明	デフォルト値
<code>n_estimators</code>	木の本数	100
<code>max_features</code>	変数のサブセット数	1.0
<code>criterion</code>	分割時の基準となる関数	<code>squared_error</code>
<code>max_depth</code>	木の最大深さ	<code>None</code>
<code>min_samples_split</code>	内部ノードを分割に必要な最小サンプル数	2
<code>min_samples_leaf</code>	葉ノードに必要な最小サンプル数	1
<code>min_impurity_decrease</code>	分割時に不純度を減少させる最小値	0.0
<code>bootstrap</code>	ブートストラップサンプルの使用	True

Table14: Random Forest の主なパラメータ (参考:scikit-learn developers[21])

とくに重要なパラメータは、`n_estimators` と `max_features` である。`n_estimators` は森の木の本数であり、大きいほど精度は良くなる反面、計算時間が長くなる。また、ある本数を超えると、改善されなくなる。`max_features` は、ノードを分割する際に考慮する変数のサブセットのサイズである。小さいほど分散は小さくなる反面、バイアスが大きくなる。

7.4.1 パラメータチューニング

学習データは5分割してクロスバリデーションを行い、パラメータをグリッドサーチで検証した。最終的なパラメータは次の通りである。なお、今回は他の期間で汎化性能を確認しているが、たとえば `min_sample_leaf` は各葉について最低1つしかデータをもたないため、過学習の恐れがある。よって、データやモデルの特性に応じて適宜検証が必要である。なお、`min_samples_split=2` と `max_depth=None` を組み合わせることで木を完全に展開すると、良い結果が得られるなど推奨値は公式ドキュメント [22] を参照している。

7.4.2 予測結果の可視化

Random Forest での予測結果は次の通りである。

パラメータ	値	推奨値
<code>n_estimators</code>	200	できるだけ大きく
<code>max_features</code>	1.0	1.0 or 0.3
<code>max_depth</code>	8	None かつ
<code>min_samples_leaf</code>	1	2

Table15: モデルのハイパーパラメータ

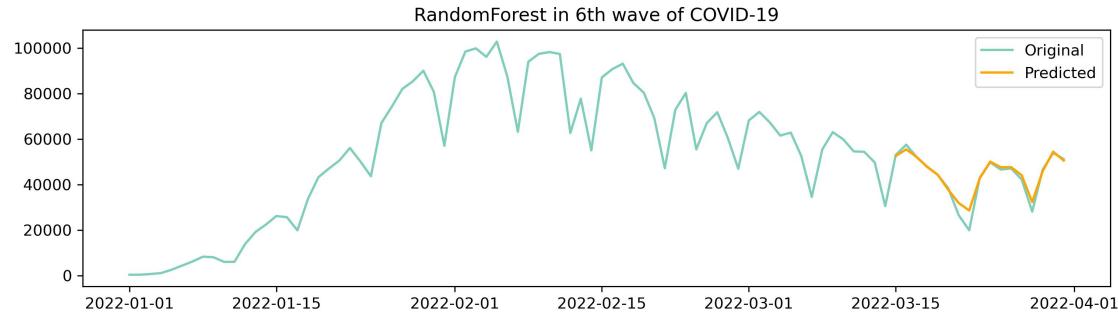


Figure16: Random Forest 全期間

予測期間の拡大図を以下に示す。

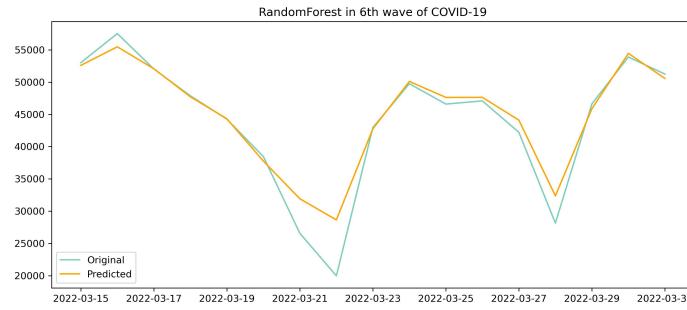


Figure17: Random Forest 予測期間

7.4.3 結果の解釈

Random Forest(RF) の予測結果では、MAPE 約 5% の誤差に落ちていた。外れ値や極端なばらつきがなく、直感的にもよく予測できている。また、これまでのモデルの中で最も高い予測精度を実現できた。なお、本章の最後に同じ構造の特徴量を用いて第7波の新規感染者数も予測し、Random Forest の汎用性も確認している。

7.4.4 変数の相対的重要性

Random Forest はその内部がわかりにくいが、各決定木における変数の影響は確認できる。なお、多くの場合、入力予測変数が均一に応答と関連することはほとんどなく、ほんの一部の変数が応答に決定的な影響を与えている。^[20] そこで、ここでは予測変数の相対的重要性を評価する。具体的には、次のような式で表される。

$$\mathcal{I}_\ell^2(T) = \sum_{t=1}^{J-1} \hat{i}_t^2 I(v(t) = \ell) \quad (68)$$

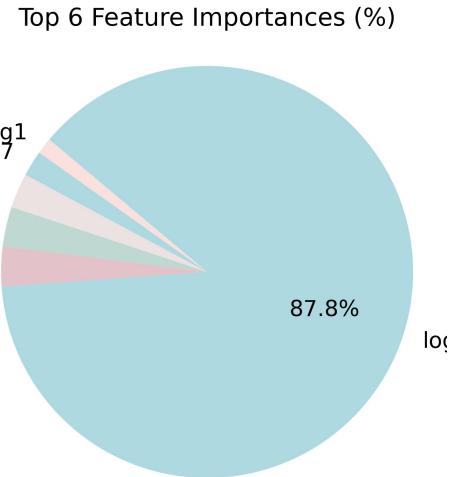
まず、決定木 T に着目している。ノード t において特定の変数 ℓ が分割に使用された際の予測誤差の減少量の 2 乗 \hat{i}_t^2 の総和をとったものである（分類では不純度を用いる）。この重要度尺度は、単純に木全体で平均化して加法的木展開に一般化できる。^[20] 具体的には次式で表すことができる。

$$\mathcal{I}_\ell^2 = \frac{1}{M} \sum_{m=1}^M \mathcal{I}_\ell^2(T_m) \quad (69)$$

考え方としては各特徴量が各決定木の分割においてどれだけモデルの予測精度を向上させたかという 2 乗誤差の減少量の平均で評価する。なお、後述する勾配ブースティングも同じ方法で評価できる。^[20]

7.4.5 相対的重要性の可視化

比率を可視化すると次の通りである。影響があった変数は 6 つのみであり、相対的な比率を比較した結果、対数変換系列の影響が約 9 割と大きな比重を占めている。



7.5 汎化性能の確認

最後に、これまで第 6 波で使用してきた特徴量を用いて、最も予測精度の高かった Random Forest が過学習に陥らないかを確認する。ここでは前述の第 6 波予測モデルと全く同一のモデルを使用して、第 7 波の最後の 2 週間にに対して新規感染者数を予測した。

データセット	期間
検証データ期間 (第 7 波の総期間)	2022-09-15 ~ 2022-09-30 2022-07-01 ~ 2022-09-30

Table16: 第 7 波の期間

7.5.1 予測結果の可視化

Random Forest での予測結果は次の通りである。

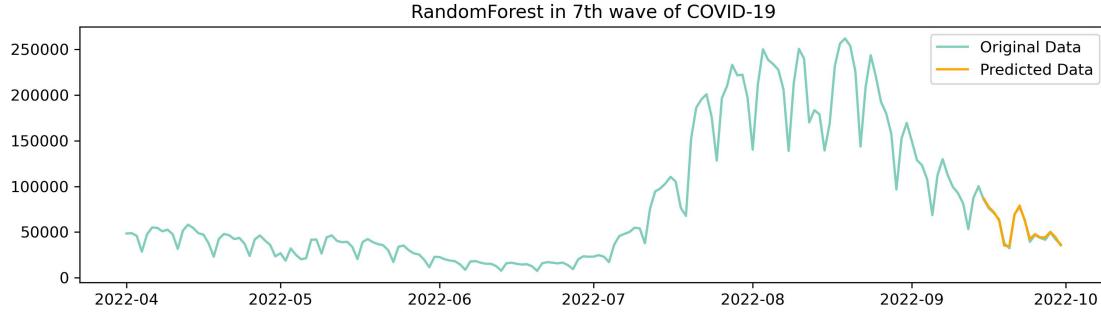


Figure18: Random Forest 全期間 (第 7 波)

予測期間の拡大図を以下に示す。

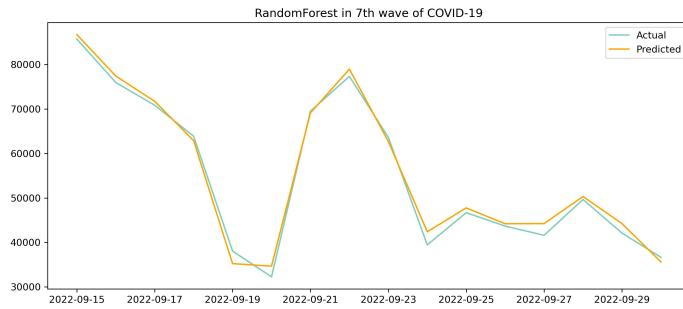


Figure19: Random Forest 予測期間 (第 7 波)

7.5.2 結果の解釈

Random Forest(RF7th) の予測結果では、MAPE3% 台の誤差と第 6 波よりも高い精度で予測できた。学習した期間よりも減少トレンド自体は強いにも関わらずうまく予測できている。

7.6 LightGBM

LightGBM は、木構造の学習アルゴリズムをベースに使用する勾配ブースティング木 (gradient tree boosting) のモジュールである。学習速度の高速化や精度向上、メモリ使用量削減、並列処理など、多くの利点がある。^[23] ここからは LightGBM のアルゴリズムの根幹である、勾配ブースティングに注目する。

7.7 アルゴリズム

勾配ブースティング木のアルゴリズムは次の通りである.[20]

1. $f_0(x) = \operatorname{argmin}_{\gamma} \sum_{i=1}^N L(y_i, \gamma)$ のように初期化する
2. $m=1$ から M に対して, 以下を行う.
 - (a) $i = 1, 2, \dots, N$ に対して次を計算する

$$r_{im} = - \left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)} \right]_{f=f_{m-1}}$$
 - (b) 終端領域 R_{jm} ($j = 1, 2, \dots, j_m$) を与える回帰木を目標 r_{im} に適合させる
 - (c) $j = 1, 2, \dots, j_m$ に対して次の計算をする

$$\gamma_{jm} = \operatorname{argmin}_{\gamma} \sum_{x_i \in R_{jm}} L(y_i, f_{m-1}(x_i) + \gamma)$$
 - (d) $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{j_m} \gamma_{jm} I(x \in R_{jm})$
3. $\hat{f}_M(x)$ を出力する

7.7.1 アルゴリズムの解釈

やや難解だが, 紐解くとおおよそ次のように意味していると考えられる. まずここでは, 実測値と予測値との差に対する損失規準 $L(y, f(x))$ を用いている. アルゴリズムの 1 では, 平均値のような定数 γ を用いて, $L(y, f_0(x = \gamma))$ と初期値としている(つまり, 終端頂点を 1 つだけ持つ分割のない木である [20]). その後 2 では, イテレーション $m = 1$ から M において得られたデータ x_i に対して現在のモデル f_{m-1} から残差 r を計算して, 負の勾配とする. そして, すべての領域 R_{jm} において新たな損失関数 $L(y_i, f_{m-1}(x_i)) + \gamma$ を最小にする γ_{jm} を求め, 次のイテレーション m における予測値 $f_m(x)$ を更新する. 最後に, 3 ではすべてのイテレーション M を通した予測値 $\hat{f}_M(x)$ を出力する.

7.8 Python による実装

モデリングに際しては, LightGBM モジュールの `lightgbm.LGBMRegressor` クラスを用いた. なお, 基本的なハイパーパラメータは次の通りである.

パラメータ	説明	デフォルト値
boosting_type	ブースティングの種類	gbdt
num_leaves	木の葉の数	31
max_depth	木の最大深さ	-1
learning_rate	学習率	0.1
n_estimators	木の数	100
min_data_in_leaf	葉に必要な最小サンプル数	20
reg_alpha	L1 正則化の係数	0.0
reg_lambda	L2 正則化の係数	0.0

Table17: LightGBM の主なパラメータ (参考:Microsoft[23])

特に重要なのは次の通りである. まず `num_leaves` は, 小さくすることでモデルの複雑さを制御する反面, 予測精度が低下する. `min_data_in_leaf` では過学習を防ぐことができる. `max_depth` では, 木の深さを明示的

に制限できる。その他、計算速度向上、精度向上、過学習抑制のための調整方法は公式ドキュメントにも記載されている。^[24]

7.8.1 パラメータチューニング

学習データは5つに分割してクロスバリデーションを行い、パラメータをグリッドサーチにより探索的に検証した。最終的なパラメータは次の通りである。こちらもRandom Forestの時と同様に、データやモデルの特性に応じて適宜過学習がないか検証が必要である。

パラメータ	値
num_leaves	10
learning_rate	0.005
n_estimators	1,000
max_depth	3
min_samples_leaf	1

Table18: モデルのハイパーパラメータ

7.8.2 予測結果の可視化

LightGBMでの予測結果は次の通りである。

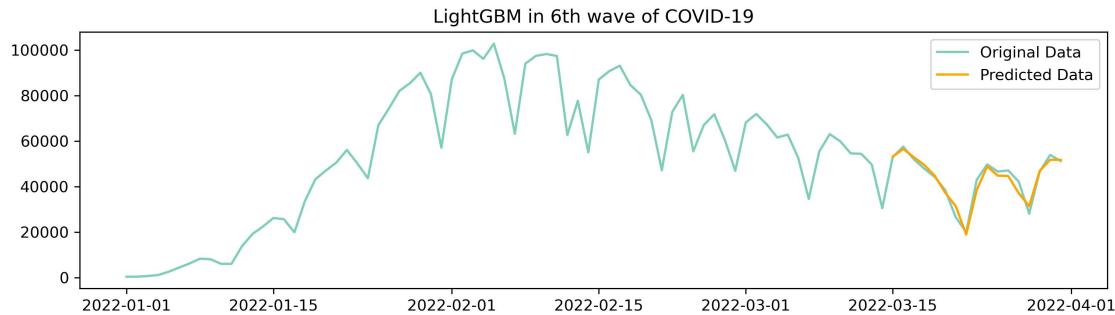


Figure20: LightGBM 全期間

予測期間の拡大図を以下に示す。

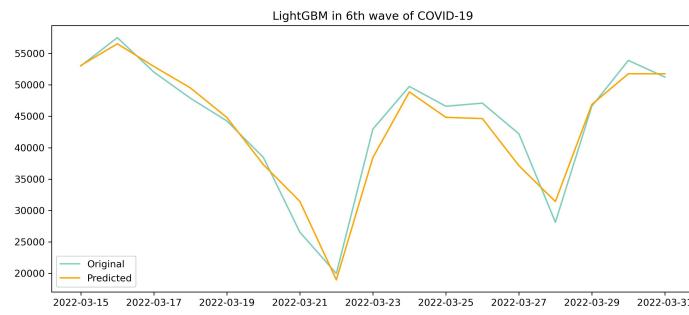


Figure21: LightGBM 予測期間

7.8.3 結果の解釈

予測期間中の傾向、周期性、最大値と最小値を的確に捉えており、的確なモデルが構成できた。

7.9 汎化性能の確認

最後に、これまで第6波で使用してきた特徴量を用いて、最も予測精度の高かったLightGBMが過学習に陥らないかを確認する。ここでは前述の第6波予測モデルと全く同一のモデルを使用して、第7波の最後の約2週間にに対して新規感染者数を予測した。

データセット	期間
検証データ期間 (第7波の総期間)	2022-09-15 ~ 2022-09-30 2022-07-01 ~ 2022-09-30

Table19: 第7波の期間

7.9.1 予測結果の可視化

LightGBMでの予測結果は次の通りである。

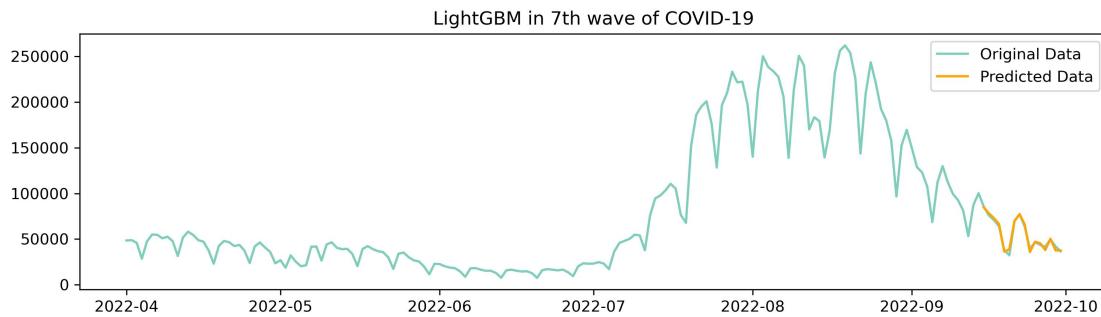


Figure22: LightGBM 全期間 (第7波)

予測期間の拡大図を以下に示す。

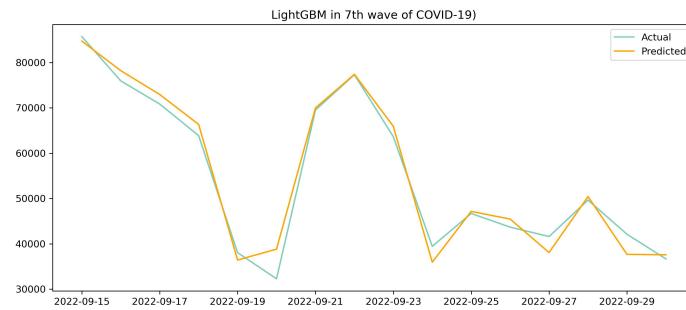


Figure23: LightGBM 予測期間 (第7波)

7.9.2 結果の解釈

LightGBM(LGBM7th)の予測結果では、MAPE4%台の誤差となり、若干ではあるが第6波よりも高い精度で予測できた。Random Forestと同様に、学習した期間よりも減少トレンド自体は強いにも関わらずうまく予測できている。

8 おわりに/Conclusion

これまで確認してきた通り、新規感染者数の日次データを用いた予測では、期間を適切に定めて機械学習による予測モデルを作成することで、誤差 5% 前後の精度と汎化性能を両立できることが確認できた。最後に個人的な感想を述べる。今回は、時系列分析を段階的に作成しモデルを比較することで、体系的な学びが得られた。また、状態空間モデルの理解や LATEX でのペーパー作成、そして機械学習モデルの精度と汎化性能の両立など、初めて挑戦したことばかりで全工程に 1 ヶ月を要したが、得られたものは大きい。なにより、文献を参照しながら事実を示す過程と、自己の解釈を示す過程を通じて、ロジックを積み上げる感覚を養うことができた。

気になる点があればご指摘をいただけすると幸いです。

References

- [1] 厚生労働省. オープンデータ. URL: <https://www.mhlw.go.jp/stf/covid-19/open-data.html>.
- [2] 白石博. 時系列データ解析. 森北出版, 2022.
- [3] P.J. Brockwell and R.A. Davis. *Time Series: Theory and Methods*. Springer, 2006.
- [4] 沖本竜義. 経済・ファイナンスデータの計量時系列分析. 朝倉書店, 2010.
- [5] 日本統計学会, ed. 増訂版 日本統計学会公式認定 統計検定 1級対応 統計学. 増訂版. 日本統計学会, 2023.
- [6] 北川源四郎. 時系列解析入門. 岩波書店.
- [7] 北川源四郎. 時系列データ解析. 2020. URL: http://www.mi.u-tokyo.ac.jp/consortium2/pdf/4-4_literacy_level_note.pdf.
- [8] 国立感染症研究所. 新型コロナウイルス感染症の直近の感染状況等(2022年7月27日現在). URL: <https://www.niid.go.jp/niid/ja/2019-ncov/11345-covid19-ab92th.html>.
- [9] 馬場真哉. 時系列分析と状態空間モデルの基礎:RとStanで学ぶ理論と実装. プレアデス出版.
- [10] 日本統計学会, ed. 日本統計学会公式認定 統計検定準1級対応 統計学実践ワークブック. 日本統計学会, 2020.
- [11] 林直嗣. “金融時系列の構造変化と単位根検定”. In: 経営志林 37.3 (2000), pp. 13–32.
- [12] 北川源四郎. “状態空間モデル”. In: 岩波データサイエンス Vol.6. Ed. by 岩波データサイエンス刊行委員会. 岩波書店, 2017, pp. 5–31.
- [13] 厚生労働省. 新型コロナウイルス感染症のこれまでの疫学と今後想定される伝播動態. URL: <https://www.mhlw.go.jp/content/10900000/001088930.pdf>.
- [14] 厚生労働省. 第3波、第5波、第6波、第7波の比較(まとめ). URL: <https://www.mhlw.go.jp/content/10900000/001010896.pdf>.
- [15] 厚生労働省. オミクロン株による第8波における死者数の増加に関する考察. URL: <https://www.mhlw.go.jp/content/10900000/001062650.pdf>.
- [16] statsmodels-developers. *statsmodels.tsa.arima.model.ARIMA*. URL: <https://www.statsmodels.org/stable/generated/statsmodels.tsa.arima.model.ARIMA.html>.
- [17] 田中久稔. 計量経済学のための数学. 日本評論社, 2019.
- [18] statsmodels-developers. *statsmodels.tsa.statespace.sarimax.SARIMAX*. URL: <https://www.statsmodels.org/stable/generated/statsmodels.tsa.statespace.sarimax.SARIMAX.html>.
- [19] 平井有三. はじめてのパターン認識. 森北出版, 2012.
- [20] Trevor Hastie et al. 統計的学習の基礎—データマイニング・推論・予測—. 共立出版, 2014.
- [21] scikit-learn developers. *Scikit-learn: RandomForestRegressor*. URL: <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html>.
- [22] scikit-learn developers. *Scikit-learn:1.11. Ensembles: Gradient boosting, random forests, bagging, voting, stacking: parameter tuning guidelines*. URL: <https://scikit-learn.org/stable/modules/ensemble.html>.
- [23] Microsoft Corporation. *LightGBM: Parameters*. URL: <https://lightgbm.readthedocs.io/en/stable/Parameters.html>.
- [24] Microsoft Corporation. *LightGBM: Parameters-Tuning*. URL: <https://lightgbm.readthedocs.io/en/stable/Parameters-Tuning.html>.