AN APPROXIMATE ANALYSIS OF TANDEM QUEUES WITH GENERAL BLOCKING NODES

YANG WOO SHIN¹, DONG OK KIM², AND DUG HEE MOON³

¹DEPARTMENT OF STATISTICS, CHANGWON NATIONAL UNIVERSITY, CHANGWON, 51140, REPUBLIC OF KOREA
Email address: †ywshin@changwon.ac.kr

²INSTITUTE OF INDUSTRIAL TECHNOLOGY, CHANGWON NATIONAL UNIVERSITY, CHANGWON, 51140, REPUBLIC OF KOREA

³DEPARTMENT OF INDUSTRIAL AND SYSTEMS ENGINEERING, CHANGWON NATIONAL UNIVERSITY, CHANGWON, 51140, REPUBLIC OF KOREA

ABSTRACT. A tandem queue that consists of nodes with buffers of finite capacity and general blocking scheme is considered. The service time distribution of each node is exponential whose rate depends on the state of the node. The blocking scheme at a node may be different from that of other nodes. An approximation method for the system based on decomposition method is presented. The effectiveness of the method is investigated numerically.

1. INTRODUCTION

Tandem queue is a queueing network in which service nodes are linked along a single flow path one after another and customers arrive from outside at the first node and are processed at the nodes in sequence, and leave the system from the last node. Tandem queues with finite capacity nodes have been widely used for performance modeling of computer systems, telecommunication networks and manufacturing systems [1, 2, 3]. The limited buffer capacity leads to the blocking phenomenon. When a node reaches its maximum capacity, the flow of customers from the upstream node into the downstream node is stopped, and the blocking phenomenon arises. Various blocking mechanisms in queueing networks with finite capacity nodes have been introduced in the literature to represent distinct behaviors of real systems [4]. The blocking type mostly used in modelling manufacturing systems is blocking after service (BAS) scheme, or sometimes called manufacturing blocking, in which if the buffer of the destination node is full upon a service completion at a node, the server is forced to stop its service, and the customer is held at the node where it has recently completed its service until the destination

Received November 25 2021; Revised March 7 2022; Accepted in revised form March 8 2022; Published online March 25 2022.

2010 Mathematics Subject Classification. 60K25, 68M20.
Key words and phrases. tandem queue, general blocking, decomposition.
† Corresponding author. This paper was supported by Changwon National University in 2021-2022.
can accommodate it. Under the blocking before service (BBS) scheme, sometimes called communication blocking, the server at each node checks the state of the destination node before starting a service and if there is an available space at the destination node, then the server starts its service, otherwise, the server is blocked and does not start its service. Two subcategories of BBS scheme distinguish whether the server can be used as a buffer when the node is blocked, called BBS-SO (server occupied) or not, called BBS-SNO (server not occupied). Another type of blocking scheme is kanban blocking under which the customers blocked upon a service completion share the buffer space of the node along with the other customers that are either waiting for service or being served, and the server continues processing customers in the node unless the server is not blocked. Cheng and Yao [5] develop a general blocking (GB) scheme by introducing parameters for the upper limits on the number of customers waiting in buffer and being in service, the number of blocked customers, and the capacity of the node. The GB scheme includes ordinary manufacturing, communication, and kanban blocking as special cases by specifying the values of the parameters in different ways. The structural properties in tandem queues with GB scheme such as the line reversibility and the effects of system parameters, buffer size, variability of service time and the control parameters of the blocking scheme to the system performance are presented in [5, 6, 7]. In this paper, we focus on the quantitative (numerical) method for the analysis of the tandem queues with exponential service time under GB scheme.

Models with finite buffers and exponential service times can be represented by finite state Markov chains. However, a numerical solution of the associated Markov chain is seriously limited by the complexity of state space and computational time that grow exponentially as the number of nodes increases. Hence, approximate analytical methods and simulation have been used for numerical analysis of the system as alternatives of exact solution.

Many approximation methods for queueing networks with blocking have been proposed in literature both for open and closed models and surveys of some methods have been presented [8, 9, 1, 2]. One of the most common method among the approximation techniques is decomposition method in which the original long line is decomposed into subsystem that are mathematically tractable, and the performance of original system is approximated by that of the subsystems. For more about decomposition method for tandem queues, see e.g. [10, 11, 12, 13]. Most of the works cited above pertain queueing systems under BAS blocking strategy.

The objective of this paper is to present an approximate analysis for the tandem queue with finite buffer under general blocking scheme. The processing time at each node is exponentially distributed and the service rate depends both on the number of customers that are waiting for service in queue or being served and the number of blocked customers. This system includes the tandem queue with multiple servers and general blocking at each node as a special case by specifying the service rates. Our approach is based on decomposition method. The contribution of this paper is to present an approximation method for a very general model in blocking mechanism sense. The method is very effective in accuracy and computation time.

The paper is organized as follows. In Section 2 we describe the model in detail. Some preliminaries and subsystem are presented in Sections 3 and 4, respectively. Approximate formulae for the parameters of subsystems are presented in Section 5 and an algorithm for
calculating performance measures is given in Section 6. Application of the result to the system with multiple servers is described in Section 7. The results of the approximation method are compared numerically with simulation and existing methods in Section 8. Finally, some concluding remarks are given in Section 9.

2. Model and Assumptions

We consider a tandem queueing network $L$ that consists of $N$ nodes $W_i$ with finite capacity $c_i < \infty$, $i = 1, 2, \cdots, N$. The customers arrive from outside according to a Poisson process whose rate depends on the state of the first node and customers at $W_i$ are processed (served) according to an exponential distribution whose rate depends on the state of $W_i$. The blocked customer is the one that has completed service at a node, but cannot be sent to the next node due to a limited buffer capacity of the downstream node. The blocked customers may continue to share the buffer space of the node along with the other customers that are either waiting for service or being served upon. We classify the customers at node $W_i$ into two types, blocked customers (BC) and active customers (AC) that are waiting for service or being in service at node $W_i$.

Blocking scheme. The blocking process of each node is controlled by three parameters $(a_i, b_i, c_i)$, where $a_i$ and $b_i$ are the upper limits on the number of active customers and blocked customers at $W_i$, respectively, and $c_i$ is the capacity of $W_i$. It is natural to assume that $1 \leq a_i \leq c_i$, $0 \leq b_i \leq c_i$, $a_i + b_i \geq c_i$. We assume that the last node is never blocked and $b_N = 0$.

The features of the node with $b_i > 0$ are different from those of the node with $b_i = 0$. In case of $b_i = 0$, $W_i$ cannot hold any blocked customers and the node follows the BBS rule. If $b_i > 0$, then the behavior (blocking or joining to the next node) of the customers in $W_i$ is determined after a service completion. We describe the behaviors of two types of nodes separately.

(i) The case of $b_i > 0$. Upon a service completion at $W_i$, if there are no places available for active customers in $W_{i+1}$, that is, the number of active customers at $W_{i+1}$ is $a_{i+1}$ or total number of customers at $W_{i+1}$ is $c_{i+1}$, then the customer just completed its service is blocked and is stocked at $W_i$. If the number of blocked customers at $W_i$ reaches $b_i$ upon a service completion, then the service process at $W_i$ is forced to stop. Even the service process is stopped, the node $W_i$ can hold active customers arriving from the upstream node $W_{i-1}$ if there is an available space for active customers at $W_i$.

(ii) The case of $b_i = 0$. In this case, if the number of active customers in $W_{i+1}$ reaches $a_{i+1}$ or total number of customers in $W_{i+1}$ reaches $c_{i+1}$, the service process at $W_i$ is forced to stop until there is an available space for active customers in $W_{i+1}$. The idle server can accept an active customer even it is stopped its service, that is, $W_i$ follows BBS-SO blocking scheme. The node $W_i$ starts new service upon there is a place available for active customers at $W_{i+1}$. Hereafter BBS means BBS-SO blocking. Define the state of service process at $W_i$ at time $t$ by

$$M_i(t) = \begin{cases} 0^*, & \text{service process at } W_i \text{ is stopped}, \\ 0, & \text{otherwise}. \end{cases}$$
Service time. The distribution of service time at node $W_i$ is exponential and the service rate may depend on the state of the node $W_i$. Denote the service rate by $\mu_i(x, y)$, where $x$ is the number of AC and $y$ is the number of BC for $b_i > 0$, and for $b_i = 0$, $y$ is the server state at $W_i$. It can be seen from the assumption that $\mu_i(x, b_i) = 0$ for $b_i > 0$ and $\mu_i(x, 0^*) = 0$ for $b_i = 0$, and $\mu_i(0, y) = 0$.

Arrival process from source node. Customers arrive to the first node $W_1$ from the source node (or outside) $W_0$ according to a Poisson process whose rate depends on the state of $W_1$. The node $W_0$ is assumed to be never starved. Denote the arrival rate to the first node $W_1$ by $\lambda_1(x, y)$, where $x$ is the number of AC, and for $b_1 > 0$, $y$ is the number of BC at $W_1$ and for $b_1 = 0$, $y \in \{0, 0^*\}$ is the server state of $W_1$.

3. Stochastic processes and transition rates

3.1. Stochastic processes. Let $X_i^a(t)$ and $Y_i(t)$ be the number of active customers and blocked customers, respectively, in $W_i$ at time $t$, and $X_i(t) = X_i^a(t) + Y_i(t)$, $Z_i(t) = X_i(t) + Y_i(t)$.

For describing the behavior of the node $W_i$, we use two dimensional stochastic processes

$$
V_i(t) = (Z_i(t), X_i(t)),
$$

$$
W_i(t) = \begin{cases} 
(Z_i(t), X_i(t)), & b_i > 0, \\
(X_i(t), Y_i(t)), & b_i = 0.
\end{cases}
$$

Let $\xi_i = a_i + b_{i-1}$ and $\kappa_i = c_i + b_{i-1}$, $i = 1, 2, \cdots, N$. The state space of $Z_i(t)$ is $Z_i = \{0, 1, \cdots, \kappa_i\}$. Once $Z_i(t) = n$ is given, it can be seen from $0 \leq X_i(t) \leq \xi_i$ and $0 \leq Y_i(t) \leq b_i$ that $l_i(n) \leq X_i(t) \leq u_i(n)$, where

$$
l_i(n) = \max(n - b_i, 0), \quad u_i(n) = \min(n, \xi_i).
$$

Note that if $Y_i(t) = y$ is given, then $0 \leq X_i(t) \leq x_i^*(y)$, where

$$
x_i^*(y) = \min(\xi_i, \kappa_i - y).
$$

The state space $\mathcal{V}_i$ of $V_i(t)$ and the space $\mathcal{W}_i$ of $W_i(t)$ are as follow:

$$
\mathcal{V}_i = \{(n, x) : l_i(n) \leq x \leq u_i(n), 0 \leq n \leq \kappa_i\},
$$

$$
\mathcal{W}_i = \begin{cases} 
\{(x, y) : 0 \leq x \leq x_i^*(y), 0 \leq y \leq b_i\}, & b_i > 0, \\
\{(x, 0) : 0 \leq x \leq \xi_i\}, & b_i = 0.
\end{cases}
$$

Note that for given $V_i(t) = (n, x)$, the maximal value $x_{i-1}(n, x)$ of $X_{i-1}(t)$ and the state $y_{i-1}(n, x)$ of $Y_{i-1}(t)$ for $b_{i-1} > 0$ are determined by

$$
y_{i-1}(n, x) = \max(0, n - c_i, x - a_i),
$$

$$
x_{i-1}(n, x) = \max(\xi_{i-1}, \kappa_{i-1} - y_{i-1}(n, x)).
$$

and the state of $M_{i-1}(t)$ for $b_{i-1} = 0$ is

$$
M_{i-1}(t) = \begin{cases} 
0, & n < c_i \text{ and } x < a_i, \\
0^*, & n = c_i \text{ or } x = a_i.
\end{cases}
$$
Let
\[ D_{i-1} = \{(n, x) \in V_i : l_i(n) \leq x \leq \min(n, a_i - 1), 0 \leq n \leq c_i - 1\} \]
and for \(0 \leq y \leq b_{i-1}\),
\[ B_{i-1}(y) = \{(c_i + y, x) \in V_i : l_i(c_i + y) \leq x \leq a_i + y\} \]
\[ \cup \{(n, a_i + y) \in V_i : a_i + y \leq n < c_i + y\}. \]

Noting that \(D_{i-1}\) is the set of states of \(V_i(t)\) on which there is an available space for active customers in \(W_i\), and \(B_{i-1}(0)\) is the set on which there are no places available for active customers in \(W_i\) and \(Y_{i-1}(t) = 0\), it can be seen that for \(b_{i-1} > 0\),
\[ \{Y_{i-1}(t) = 0\} = \{V_i(t) \in D_{i-1} \cup B_{i-1}(0)\}, \]
\[ \{Y_{i-1}(t) = y\} = \{V_i(t) \in B_{i-1}(y)\}, 1 \leq y \leq b_{i-1}, \]
and for \(b_{i-1} = 0\),
\[ \{M_{i-1}(t) = 0\} = \{V_i(t) \in D_{i-1}\}, \]
\[ \{M_{i-1}(t) = 0^*\} = \begin{cases} \{V_i(t) \in B_{i-1}(0)\}, & b_i > 0, \\ \{W_i(t) \in \{(c_i, 0), (c_i, 0^*)\}\}, & b_i = 0. \end{cases} \]

We introduce notation for later use. For given \(V_i(t) = (n, x)\), we write the state of \(Y_{i-1}(t)\) for \(b_{i-1} > 0\) and \(M_{i-1}(t)\) for \(b_{i-1} = 0\) by a unified form
\[ y_{i-1}^t(n, x) = \begin{cases} y_{i-1}(n, x), & b_{i-1} > 0, \\ 0, & (n, x) \in D_{i-1}, b_{i-1} = 0, \\ 0^*, & (n, x) \in B_{i-1}(0), b_{i-1} = 0 \end{cases} \]
and \(y_{i-1}^t(x + 0^*, x)\) means \(y_{i-1}^t(x, x)\).

3.2. **Transition rates of** \(W_i(t)\). Given \(W_i(t) = (x, y) \in W_i\), the state transitions of \(W_i(t)\) are occurred by a service completion at \(W_{i-1}\) or \(W_i\), a departure of blocked customers from \(W_i\) for \(b_i > 0\), and a departure from \(W_{i+1}\) for \(b_i = 0\) and \(y = 0^*\). Now we derive the transition rates of \(W_i(t)\) for each case described above.

(i) **The rate** \(\lambda_i(x, y)\) **from** \((x, y)\) **to** \((x + 1, y)\). The transition from \((x, y)\) to \((x + 1, y)\) is occurred by a service completion at \(W_{i-1}\) and the rate is for \(2 \leq i \leq N - 1\),
\[ \lambda_i(x, y) = \sum_{j=1}^{h} P(X_{i-1}(t) = j | W_i(t) = (x, y)) \mu_{i-1}(j, k), \]
where \(h = x_{i-1}(x + y, x)\) and \(k = y_{i-1}^*(x + y, x)\).

(ii) **The rate** \(\beta_i(x, y)\) **from** \((x, y)\) **to** \((x - 1, y + 1)\) **for** \(b_i > 0\). Given \(W_i(t) = (x, 0)\), the customer being served at \(W_i\) is blocked to enter the next node \(W_{i+1}\) upon a service completion if there are no places available for active customers. Thus for \(1 \leq x \leq \xi_i\),
\[ \beta_i(x, 0) = P(V_{i+1}(t) \in B_i(0) | W_i(t) = (x, 0)) \mu_i(x, 0). \]
If \(1 \leq y \leq b_i - 1\), the customer is blocked upon it service completion and hence
\[ \beta_i(x, y) = \mu_i(x, y), 1 \leq x \leq x_i^*, 1 \leq y \leq b_i - 1. \]
(iii) The rate \( \delta_i(x, 0) \) from \( W_i(t) = (x, 0) \) to \( (x - 1, 0) \) for \( b_i > 0 \). It can be seen from (3.1) that

\[
\delta_i(x, 0) = P(V_{i+1}(t) \in D_i | W_i(t) = (x, 0)) \mu_i(x, 0) = \mu_i(x, 0) - \beta_i(x, 0), \quad 1 \leq x \leq \xi_i.
\]

(iv) The rate \( \beta_i^0(x) \) from working state \( (x, 0) \) to blocking state \( (x - 1, 0^*) \) in the node with \( b_i = 0 \). Let

\[
D_i^0 = \{(c_{i+1} - 1, x) \in V_{i+1} : l_{i+1}(c_{i+1} - 1) \leq x \leq a_{i+1} - 1\}
\]

\[
\cup \{(n, a_{i+1} - 1) : a_{i+1} - 1 \leq n < c_{i+1} - 1\}.
\]

If \( V_{i+1}(t) \in D_i^0 \), then there is only one place available for active customers in \( W_{i+1} \) at time \( t \). The transition of \( W_i(t) \) from \( (x, 0) \) to \( (x - 1, 0^*) \) occurs if a service at \( W_i \) is completed on \( D_i^0 \). Thus the rate from \( (x, 0) \) to \( (x - 1, 0^*) \) is for \( 1 \leq x \leq \xi_i \)

\[
\beta_i^0(x) = P(V_{i+1}(t) \in D_i^0 | W_i(t) = (x, 0)) \mu_i(x, 0)
\]

(v) The rate \( \delta_i^0(x) \) from \( (x, 0) \) to \( (x - 1, 0) \) in the node with \( b_i = 0 \). In case of \( b_i = 0 \), customers at \( W_i \) join \( W_{i+1} \) upon a service completion and the resulting state of \( M_i(t) \) is one of the two states \( 0 \) or \( 0^* \). It can be seen from (3.2) that

\[
\delta_i^0(x) = \mu_i(x, 0) - \beta_i^0(x), \quad 1 \leq x \leq \xi_i.
\]

(vi) The rate \( \delta_i(x, y) \) from \( W_i(t) = (x, y) \) to \( (x, y - 1) \), \( y \geq 1 \). Let

\[
\tilde{B}_i(y) = \{(n', x') \in B_i(y) : y_i(n' - 1, x') = y - 1, n' > x'\}
\]

\[
\cup \{(a_{i+1} + y, a_{i+1} + y)\}.
\]

If \( V_{i+1}(t) \in \tilde{B}_i(y) \) with \( y \geq 1 \) and a departure from the node \( W_{i+1} \) occurs, then the resulting state of \( V_{i+1}(t) \) is in \( \tilde{B}_i(y - 1) \). If \( W_i(t) = (x, y) \) with \( y \geq 1 \), a departure from \( W_i \) is occurred by a departure from \( W_{i+1} \) on \( \tilde{B}_i(y) \) and hence for \( y \geq 1 \)

\[
\delta_i(x, y) = \sum_{(n', x') \in \tilde{B}_i(y)} P(V_{i+1}(t) = (n', x') | W_i(t) = (x, y)) \delta_{i+1}^*(x', n' - x'),
\]

where

\[
\delta_{i+1}^*(x', n' - x') = \begin{cases} \delta_{i+1}^1(x', n' - x'), & b_{i+1} > 0, \\ \delta_i^0(x'), & b_{i+1} = 0 \end{cases}
\]

with \( \delta_N(x, 0) = \mu_N(x, 0) \) and \( \delta_N(x, y) = 0, y \geq 1 \). Noting that for \( b_{i+1} = 0 \), \( \tilde{B}_i(y) = \{(c_{i+1} + y, c_{i+1} + y)\} \), it can be seen that for \( b_{i+1} = 0 \)

\[
\delta_i(x, y) = \mu_{i+1}(c_{i+1} + y, 0), \quad 1 \leq y \leq b_i.
\]

(vii) The rate \( \alpha_i(x) \) from blocking state \( (x, 0^*) \) to working state \( (x, 0) \) in the node with \( b_i = 0 \). The transition of \( W_i(t) \) from \( (x, 0^*) \) to \( (x, 0) \) is occurred by one of two types of departures from \( W_{i+1} \), a departure of an active customer on \( V_{i+1}(t) = (a_{i+1}, a_{i+1}) \) and a
departure of a blocked customer on \( V_{i+1}(t) = (c_{i+1}, x') \) with \( x' < a_{i+1} \). It can be seen that for \( b_{i+1} > 0 \),

\[
\alpha_i(n) = P(V_{i+1}(t) = (a_{i+1}, a_{i+1})|W_i(t) = (n, 0^*))\delta_{i+1}(a_{i+1}, a_{i+1}) + \sum_{x = l_{i+1}(c_{i+1})} P(V_{i+1}(t) = (c_{i+1}, x)|W_i(t) = (n, 0^*)\delta_{i+1}(x, c_{i+1} - x)
\]

and for \( b_{i+1} = 0 \),

\[
\alpha_i(n) = \mu_{i+1}(c_{i+1}, 0).
\]

4. Subsystems

We decompose the system into \( N - 1 \) subsystems \( L_i, i = 2, 3, \cdots, N \) for an approximate analysis using decomposition method. The subsystem \( L_i \) is a tandem queue that consists of two nodes, say \( W^{u}_{i-1} \) and \( W^{d}_i \) which are pseudo nodes that correspond to \( W_{i-1} \) and \( W_i \), respectively. The node \( W^{u}_{i-1} \) has a buffer of size \( g^{u}_{i-1} = g_{i-1} + b_{i-2} \), and the capacity of \( W^{u}_{i-1} \) is \( \kappa^{u}_{i-1} = \kappa_{i-1} \), the capacity for active customers is \( a^{u}_{i-1} = \xi_{i-1} \), the capacity for blocked customers is \( b^{u}_{i-1} = b_{i-1} \). Customers arrive to the first node \( W^{u}_{i-1} \) according to a Poisson process with rate \( \lambda_{i-1}(j, k) \) and the service rate is \( \mu_{i-1}(j, k) \), when \( W_{i-1}(t) = (j, k) \). The parameters for \( W^{d}_i \) are the same as \( W_i \). The throughput of \( L \) is approximated with that of \( L_N \). Since \( W_i \) and \( W_N \) have different features from that of \( W_i, 2 \leq i \leq N - 1 \), we describe the subsystems \( L_i \) \((2 \leq i \leq N - 1)\) and \( L_N \) separately.

4.1. The subsystem \( L_i \) with \( b_i > 0, 2 \leq i \leq N - 1 \). For describing the subsystem \( L_i \) with \( b_i > 0 \), define the stochastic processes \( \Psi_i(t) = (Z_i(t), X_i(t), X_{i-1}(t)) \), \( 2 \leq i \leq N - 1 \). The state space \( S_i \) of \( \Psi_i \) is \( S_i = \bigcup_{n=0}^{u_i} S_i(n) \), where

\[
S_i(n) = \{(n, x, j) : 0 \leq j \leq x_{i-1}(n, x), l_i(n) \leq x \leq u_i(n)\}, \ 2 \leq i \leq N - 1,
\]

It can be easily seen that the number of states in \( S_i(n) \) is

\[
s_i(n) = \sum_{x = l_i(n)} u_i(n) - (x_{i-1}(n, x) + 1), \ 2 \leq i \leq N - 1.
\]

The stochastic process \( \Psi_i = \{\Psi_i(t), t \geq 0\} \) forms a Markov chain on the state space \( S_i \) with generator of the form

\[
Q_i = \begin{pmatrix} B_i(0) & A_i(0) & 0 & \cdots & 0 \\ C_i(1) & B_i(1) & A_i(1) & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ C_i(\kappa_{i-1}) & 0 & 0 & \cdots & B_i(\kappa_{i-1}) \\ C_i(\kappa_{i}) & 0 & 0 & \cdots & A_i(\kappa_{i-1}) \\ 0 & B_i(\kappa_{i}) & C_i(\kappa_{i}) & \cdots & 0 \\ 0 & 0 & B_i(\kappa_{i}) & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & B_i(\kappa_{i}) \\ 0 & 0 & 0 & \cdots & B_i(\kappa_{i}) \\ \end{pmatrix} - \Delta_i,
\]

where \( \Delta_i \) is the diagonal matrix that makes \( Q_i e = 0 \) and \( e \) is the column vector of appropriate size whose components are all 1. The block matrix \( B_i^{(n)} \) is the square matrix of size \( s_i(n) \)
whose diagonals entries are all 0 and their off-diagonal components correspond to the transition rates in $S_i(n)$ without changing the level $Z_i(t) = n$. The components of the block matrices $A_{i}^{(n)}$ and $C_{i}^{(n)}$ are the transition rates of $\Psi_i$ from the states of $S_i(n)$ to the states of $S_i(n + 1)$ and $S_i(n - 1)$, respectively.

In this subsection, for given $(Z_i(t), X_i(t)) = (n, x)$, denote the maximal number of active customers at $W_{i-1}$ by $h = x_{i-1}(n, x)$ for the simplicity of notation unless confusion.

The matrices $A_{i}^{(n)}$. For each $(n, x), (n + 1, x') \in V_i$, denote by $A_{i}^{(n)}[x, x']$ the block matrix component of size $(h + 1) \times (x_{i-1}(n + 1, x') + 1)$ whose $(j, j')$-entry corresponds to the transition rate from $(n, x, j) \in S_i(n)$ to $(n + 1, x', j') \in S_i(n + 1)$. It can be easily seen that $A_{i}^{(n)}[x, x'] = 0$ for $x' \neq x + 1$ and that for $i_1(n) \leq x \leq u_1(n), i_1(n + 1) \leq x + 1 \leq u_1(n + 1)$, the $(j, j')$-component of $A_{i}^{(n)}[x, x + 1], 0 \leq j \leq h, 0 \leq j' \leq x_{i-1}(n + 1, x + 1)$ is

$$A_{i}^{(n)}[x, x + 1] = \mu_{i-1}(j, y_{i-1}^a(n, x))1(j' = j - 1),$$

where $1(A) = 1$ if $A$ is true, otherwise 0.

The matrices $B_{i}^{(n)}$. For $(n, x), (n + 1, x') \in V_i$, let $B_{i}^{(n)}[x, x']$ the block matrix component of size $(x_{i-1}(n + 1, x) + 1) \times (x_{i-1}(n, x') + 1)$ whose $(j, j')$-entry corresponds to the transition rate from $(n, x, j) \in S_i(n)$ to $(n, x', j') \in S_i(n)$. The transition of $\Psi_i$ without changing the level occurs by a service completion at $W_{i-2}$ and blocking of a customer just completed its service at $W_i$.

(i) Service completion at $W_{i-2}$. Note that if a service is completed at the node $W_{i-2}$, then the transition of $\Psi_i$ from $(n, x, j)$ to $(n, x, j + 1)$ occurs, and given $W_{i-1}(t) = (j, k)$, the service at $W_{i-2}$ does not depend on the state $(n, x)$ of $V_i(t)$. Thus the transition rate from $(n, x, j + 1)$ to $(n, x, j + 1)$ is $\lambda_{i-1}(j, k)$ and the upper diagonal entries of matrix $B_{i}^{(n)}[x, x]$ is $\lambda_{i-1}(j, k)$ and others are all zero, that is, the $(j, j')$-component of $B_{i}^{(n)}[x, x]$ is for $0 \leq j, j' \leq h$,

$$B_{i}^{(n)}[x, x] = \lambda_{i-1}(j, y_{i-1}^a(n, x))1(j' = j + 1).$$

(ii) Occurrence of blocking at $W_i$. If a customer at $W_i$ is blocked upon a service completion on the state $\Psi_i(t) = (n, x, j)$, the resulting state of $\Psi_i(t)$ immediately after an occurrence of blocking is $(n, x - 1, j)$. Thus the $(j, j')$-component of $B_{i}^{(n)}[x, x - 1]$ is for $0 \leq j \leq h$ and $0 \leq j' \leq x_{i-1}(n, x - 1)$,

$$B_{i}^{(n)}[x, x - 1] = \beta_i(x, n - x)1(j' = j).$$

The matrices $C_{i}^{(n)}$. Each component of $C_{i}^{(n)}$ corresponds to the transition rate from a state in $S_i(n)$ to a state in $S_i(n - 1)$ which is occurred by a departure of an active customer or a blocked customer from $W_i$. A departure of an active customer results in the state transition from $(n, n, j)$ to $(n - 1, n - 1, j)$ for $n \geq 1$ and a departure of a blocked customers results in the state transition from $(n, x, j)$ to $(n - 1, x, j)$ for $0 \leq x < n.$
For each \((n, x), (n - 1, x') \in \mathcal{V}_i\), denote by \(C^{(n)}_i[x, x']\) the block matrix component of size \((h + 1) \times (x_i - 1(n - 1, x') + 1)\) whose \((j, j')\)-entry corresponds to the transition rate from \((n, x, j) \in \mathcal{S}_i(n)\) and \((n - 1, x', j') \in \mathcal{S}_i(n - 1)\). It can be easily seen that \(C^{(n)}_i[x, x'] = 0\) except for \(x' = x\) with \(x < n\) and \(x' = n - 1\) with \(x = n\).

For given \(V_i(t) = (n, x)\), let

\[
h' = \begin{cases} x_i - 1(n - 1, x), & x < n, \\ x_i - 1(n - 1, n - 1), & x = n. \end{cases}
\]

It can be easily seen that \(h \leq h'\). Let \(C^{u}_{i-1}(n, x)\) be the \((h + 1) \times (h' + 1)\) matrix whose \((j, j')\)-component is the transition probability of \(X_{i-1}(t)\) from \(j\) to \(j'\) given that a departure from \(W_i\) is occurred on \(\Psi_i(t) = (n, x, j)\). Then it can be easily seen that

\[
\left[ C^{u}_{i-1}(n, x) \right]_{jj'} = 1(j' = j), \quad 0 \leq j \leq h, \quad 0 \leq j' \leq h'
\]

and for \(l_i(n) \leq x \leq u_i(n), l_i(n - 1) \leq x' \leq u_i(n - 1),\)

\[
C^{(n)}_i[x, x'] = \begin{cases} C^{u}_{i-1}(n, x) \delta_i(x, x - n) \mathbf{1}(x' = x), & x < n, \\ C^{u}_{i-1}(n, n) \delta_i(0) \mathbf{1}(x' = n - 1), & x = n, \\ 0, & \text{otherwise}. \end{cases}
\]

### 4.2 The subsystem \(L_i\) with \(b_i = 0\)

The subsystem \(L_i\), \(2 \leq i \leq N - 1\). For describing the subsystem \(L_i\) with \(b_i = 0\), define the stochastic processes \(\Psi_i^0(t) = (X_i(t), M_i(t), X_{i-1}(t))\). The state space \(S^0_i\) of \(\Psi_i^0\) is \(\{\Psi_i^0(t), t \geq 0\}\) is \(S^0_i = \cup_{n=0}^{\kappa_i} S^0_i(n), 2 \leq i \leq N - 1\), where

\[
S^0_i(n) = \{(n, 0, j), (n, 0^*, j) : 0 \leq j \leq x_{i-1}(n, n)\}
\]

and the number of states in \(S^0_i(n)\) is

\[
s^0_i(n) = 2(x_{i-1}(n, n) + 1), \quad 2 \leq i \leq N - 1
\]

and \(y^*_i(n, n)\) becomes

\[
y^*_i(n, n) = \begin{cases} \max(0, n - c_i), & b_{i-1} > 0, \\ 0, & n < c_i, b_{i-1} = 0, \\ 0^*, & n = c_i, b_{i-1} = 0. \end{cases}
\]

The generator of \(\Psi_i^0\) is the same form as (4.1) with

\[
A^{(n)}_i = I_2 \otimes A^{u}_{i-1}(n), \quad 0 \leq n \leq \kappa_i - 1,
\]

\[
B^{(n)}_i = \begin{pmatrix} (n, 0) & (n, 0^*) \\ (n, 0^*) & \alpha_i(n) I_{n+1} & B^{(n)}_i[0] \end{pmatrix}, \quad 0 \leq n \leq \kappa_i,
\]

\[
C^{(n)}_i = \begin{pmatrix} (n, 0) & C^{u}_{i-1}(n, n) \delta^0_i(n) & C^{u}_{i-1}(n, n) \delta^0_i(n) \\ (n, 0^*) & 0 & 0 \end{pmatrix}, \quad 1 \leq n \leq \kappa_i,
\]
where $I_n$ is the identity matrix of size $n$, $h = x_{i-1}(n, n)$, $h' = x_{i-1}(n+1, n+1)$, and $A_{i-1}(n)$ is the $(h + 1) \times (h' + 1)$ matrix whose $(j, j')$-component is, for $0 \leq j < h$, $0 \leq j' < h'$,

$$[A_{i-1}(n)]_{jj'} = \mu_{i-1}(j, y^*_{i-1}(n, n))1(j' = j - 1)$$

and $B^{(n)}_i[0]$ is the square matrices of size $h + 1$ whose $(j, j')$-component is, for $0 \leq j, j' \leq h$,

$$[B^{(n)}_i[0]]_{jj'} = \lambda_{i-1}(j, y^*_{i-1}(n, n))1(j' = j + 1).$$

The subsystem $L_N$. Since $M_N$ is never blocked, $Y_N(t) = 0$ and hence $Z_N(t) = X_N(t)$, $V_N(t) = (X_N(t), X_N(t))$, $W_N(t) = (X_N(t), 0)$ and $\kappa_N = \xi_N$. In the later, denote the $Z_N(t)$, $V_N(t)$ and $W_N(t)$ by $X_N(t)$. Thus $\Psi_N(t) = (X_N(t), X_{N-1}(t))$ and the state space of $\Psi_N$ is

$$\mathcal{S}_N(n) = \{(n, j) : 0 \leq j \leq x_{N-1}(n, n)\}.$$ 

The number of states in $\mathcal{S}_N(n)$ is $\kappa_N(n) = x_{N-1}(n, n) + 1$, $0 \leq n \leq \kappa_N$. The $(j, j')$ component of the matrices $A_N^{(n)}$, $B_N^{(n)}$ and $C_N^{(n)}$ are as follows:

$$[A_N^{(n)}]_{jj'} = \mu_N(j, y^*_{N-1}(n, n))1(j' = j - 1),$$

$$[B_N^{(n)}]_{jj'} = \lambda_N(j, y^*_{N-1}(n, n))1(j' = j + 1),$$

$$[C_N^{(n)}]_{jj'} = \mu_N(n, 0)1(j' = j).$$

5. Approximation of the parameters and performance measures

Now we assume that the system is in stationary state and let

$$\pi_i(n, x, j) = \lim_{t \to \infty} P(\Psi_i(t) = (n, x, j)),$$

and $\pi_i = (\pi_i(n), n = 0, 1, \ldots, \kappa_i)$ with $\pi_i(n) = (\pi_i(n, x, j), (n, x, j) \in \mathcal{S}_i(n))$. The limiting distribution $\pi_i(n, s, j)$, $s = 0, 0^*$ of $\Psi_i(t)$ and $\pi_i$ are defined similarly.

Performance measures. Once the stationary distribution $\pi_i$ of $\Psi_i$ is obtained, the performance can be obtained as follows:

- Throughput :

$$\Theta = \left(\sum_{n=1}^{\kappa_i} \pi_N(n)e\right)\mu_N(n, 0)$$

- Mean number of customers in $W_i$ :

$$\mathbb{E}[W_i] = \sum_{n=1}^{\kappa_i} \sum_{x=L_i(n)} x_{i-1}(n, x) \sum_{j=0}^{\min(n, c_i)} \pi_i(n, x, j)$$
• Mean number of active customers in $W_i$:

$$\mathbb{E}[W_i^a] = \sum_{n=1}^{\kappa_i} u_i(n) \sum_{x=\l_i(n)} \min(x, a_i) \pi_i(n, x, j)$$

• Mean number of blocked customers in $W_i$:

$$\mathbb{E}[W_i^b] = \sum_{n=1}^{\kappa_i} u_i(n) \sum_{x=\l_i(n)} \min(n - x, b_i) \pi_i(n, x, j).$$

**Approximation of $\lambda_i(x, y)$**. The marginal distribution $p_i^d(x, y) = P(W_i(t) = (x, y))$ is

$$p_i^d(x, y) = \sum_{j=0}^{x_i-1(x+y,x)} \pi_i(x + y, y, j), (x, y) \in W_i,$$

where $\pi_i(x + 0^*, 0^*, j) = \pi_i(x, 0^*, j)$ for $y = 0^*$ in the node with $b_i = 0$. Then $\lambda_i(x, y)$ is approximated by the formula, for $(x, y) \in W_i$

$$\lambda_i(x, y) = \sum_{j=1}^{h} P(X_{i-1}(t) = j|W_i(t) = (x, y))\mu_{i-1}(j, k)$$

$$= \frac{1}{p_i^d(x, y)} \sum_{j=1}^{h} \pi_i(x, y, j)\mu_{i-1}(j, k),$$

where $h = x_{i-1}(x + y, x)$ and $k = y_{i-1}(x + y, x)$.

**Approximation of $\beta_i-1(x, y)$, $\delta_i-1(x, y)$ for $b_i-1 > 0$.** We consider two cases of $b_i > 0$ and $b_i = 0$ separately.

**Case (i) $b_i > 0$.** The marginal distribution $p_i^{n}(j, k) = P(W_i^n(t) = (j, k))$ is given by,

$$0 \leq j \leq x_i^n(k)$$

$$p_i^{n}(j, 0) = \sum_{(n,x) \in B_{i-1}(0) \cup D_{i-1}} \pi_i(n, x, j) = \sum_{n=0}^{c_i} \sum_{x=\l_i(n)} \pi_i(n, x, j),$$

$$p_i^{n}(j, k) = \sum_{(n,x) \in B_{i-1}(k)} \pi_i(n, x, j)$$

$$= \sum_{x=\l_i(c_i+k)} \pi_i(c_i + k, x, j) + \sum_{n=a_i+k}^{c_i+k-1} \pi_i(n, a_i + k, j), 1 \leq k \leq b_i-1.$$
Thus for $1 \leq j \leq \xi_{i-1}$,

$$
\beta_{i-1}(j, 0) = P([V_i(t) \in B_{i-1}(0)]|W_{i-1}(t) = (j, 0))\mu_{i-1}(j, 0) = \frac{1}{p^n_{i-1}(j, 0)} \left( \sum_{x = l_i(c_i)}^{a_i} \pi_i(c_i, x, j) + \sum_{n = a_i}^{c_i-1} \pi_i(n, a_i, j) \right)\mu_{i-1}(j, 0),
$$

$$
\delta_{i-1}(j, 0) = \mu_{i-1}(j, 0) - \beta_{i-1}(j, 0)
$$

and for $0 \leq j \leq x^{*}_{i-1}(k), 1 \leq k \leq b_{i-1}$,

$$
\beta_{i-1}(j, k) = \mu_{i-1}(j, k), \quad 0 \leq j \leq x^{*}_{i-1}(k), 1 \leq k \leq b_{i-1},
$$

$$
\delta_{i-1}(j, k) = \sum_{(n, x) \in \mathcal{B}_{i-1}(k)} P(V_i(t) = (n, x)|W_{i-1}^n(t) = (j, k))\delta_i(x, n - x)
$$

$$
= \frac{1}{p^n_{i-1}(j, k)} \sum_{(n, x) \in \mathcal{B}_{i-1}(k)} \pi_i(n, x, j)\delta_i(x, n - x)
$$

with $\delta_N(x, 0) = \mu_N(x, 0)$ and $\delta_N(x, y) = 0, y \geq 1$.

Case (ii) $b_i = 0$. In case of $b_i = 0$ and $b_{i-1} > 0$, the marginal distribution $p^n_{i-1}(j, k) = P(W_{i-1}^n(t) = (j, k))$ is given by

$$
p^n_{i-1}(j, 0) = \sum_{n=0}^{c_i} (\pi_i(n, 0, j) + \pi_i(n, 0^*, j)),
$$

$$
p^n_{i-1}(j, k) = \pi_i(c_i + k, 0, j) + \pi_i(c_i + k, 0^*, j), 1 \leq k \leq b_{i-1}.
$$

The $\beta_{i-1}(j, k)$ and $\delta_{i-1}(j, k)$ are approximated as follows:

$$
\beta_{i-1}(j, k) = \mu_{i-1}(j, k), \quad 1 \leq j \leq x^{*}_{i-1}(k), 1 \leq k \leq b_{i-1} - 1,
$$

$$
\beta_{i-1}(j, 0) = P(X_i(t) = c_i|W_{i-1}^n(t) = (j, 0))\mu_{i-1}(j, 0)
$$

$$
= \frac{\pi_i(c_i, 0, j) + \pi_i(c_i, 0^*, j)}{p^n_{i-1}(j, 0)}\mu_{i-1}(j, 0), \quad 1 \leq j \leq \xi_{i-1},
$$

$$
\delta_{i-1}(j, 0) = \mu_{i-1}(j, 0) - \beta_{i-1}(j, 0), \quad 1 \leq j \leq \xi_{i-1},
$$

$$
\delta_{i-1}(j, k) = P(W_i(t) = (c_i + k, 0)|W_{i-1}^n(t) = (j, k))\mu_i(c_i + k, 0)
$$

$$
= \frac{\pi_i(c_i + k, 0, j)}{p^n_{i-1}(j, k)}\mu_i(c_i + k, 0), \quad 0 \leq j \leq x^{*}_{i-1}(k), 1 \leq k \leq b_{i-1}.
$$

Approximation of $\alpha_{i-1}(j), \beta_{i-1}^0(j)$ and $\delta_{i-1}^0(j)$ for $b_{i-1} = 0$. 
Case (i) $b_i > 0$. In case of $b_{i-1} = 0$ and $b_i > 0$, the marginal distribution $p^u_{i-1}(j, y) = P(W^u_{i-1}(t) = (j, y))$ is given by
\begin{align*}
p^u_{i-1}(j, 0^*) &= \sum_{x = l_i(c_i)}^{a_i-1} \pi_i(c_i, x, j) + \sum_{n = a_i}^{c_i-1} \pi_i(n, a_i, j), \\
p^u_{i-1}(j, 0) &= \sum_{n=0}^{c_i-1 \min(n, a_i-1)} \sum_{x=l_i(n)}^{\infty} \pi_i(n, x, j).
\end{align*}

The formulae for approximation of $\alpha_{i-1}(j)$, $0 \leq j \leq \xi_{i-1}$ and $\beta_{i-1}^0(j)$, $\delta_{i-1}(j)$, $1 \leq j \leq \xi_{i-1}$ are given as follows:
\begin{align*}
\alpha_{i-1}(j) &= P(V_i(t) = (a_i, a_i)|W^u_{i-1}(t) = (j, 0^*)) \delta_i(a_i, 0) \\
&\quad + \sum_{x = l_i(c_i)}^{a_i-1} P(V_i(t) = (c_i, x)|W_i(t) = (j, 0^*)) \delta_i(x, c_i - x) \\
&= \frac{1}{p^u_{i-1}(j, 0^*)} \left( \sum_{x = l_i(c_i)}^{a_i-1} \pi_i(a_i, a_i, j) \delta_i(a_i, 0) + \sum_{x = l_i(c_i)}^{a_i-1} \pi_i(c_i, x, j) \delta_i(x, c_i - x) \right), \\
\beta_{i-1}^0(j) &= \frac{1}{p^u_{i-1}(j, 0)} \left( \sum_{x = l_i(c_i-1)}^{a_i-1} \pi_i(c_i - 1, x, j) + \sum_{n = a_i-1}^{c_i-2} \pi_i(n, a_i - 1, j) \right) \mu_{i-1}(j, 0), \\
\delta_{i-1}^0(j) &= \mu_{i-1}(j, 0) - \beta_{i-1}^0(j).
\end{align*}

Case (ii) $b_i = 0$. In case of $b_i = 0$ and $b_{i-1} = 0$, the marginal distribution of $W^u_{i-1}(t)$ is
\begin{align*}
p^u_{i-1}(j, 0^*) &= \pi_i(c_i, 0, j) + \pi_i(c_i, 0^*, j), \\
p^u_{i-1}(j, 0) &= \sum_{n=0}^{c_i-1} \pi_i(n, 0, j) + \pi_i(n, 0^*, j)).
\end{align*}

The formulae for approximation of $\alpha_{i-1}(j)$, $\beta_{i-1}^0(j)$ and $\delta_{i-1}^0(j)$ are given as follows:
\begin{align*}
\alpha_{i-1}(j) &= \frac{\pi_i(c_i, 0, j)}{p^u_{i-1}(j, 0^*)} \mu_{i}(c_i, 0), \quad 0 \leq j \leq \xi_{i-1}, \\
\beta_{i-1}^0(j) &= \frac{\pi_i(c_i - 1, 0, j) + \pi_i(c_i - 1, 0^*, j)}{p^u_{i-1}(j, 0)} \mu_{i-1}(j, 0), \\
\delta_{i-1}^0(j) &= \mu_{i-1}(j, 0) - \beta_{i-1}^0(j), \quad 1 \leq j \leq \xi_{i-1}.
\end{align*}

6. Algorithm

The parameters for the components of $Q_i$ are calculated by the following iterative algorithm.
0. Initial setting:

(1) Initial assumption: Initially assuming that there are sufficient number of active customers in $W_{i-2}$, and the customers arrive to the upstream station $W_{i-1}^\mu$ in the subsystem $L_i$, according to Poisson process with rate $\lambda_{i-1}(j, k) = \mu_{i-2}(M, y)$, where $M$ is a sufficiently large number, for example, $M \geq \max_{1 \leq i \leq N} \xi_i$ and $y = y_{i-2}(j + k, j)$. For example, if $W_{i-2}$ has $m_{i-2}$ identical servers in parallel whose service time is exponential with rate $\mu_{i-2}$, then

$$\lambda_{i-1}(j, k) = (m_{i-2} - m_i^b(y))\mu_{i-2}, \quad 3 \leq i \leq N,$$

where $y = y_{i-2}(j + k, j)$ and

$$m_i^b(y) = \begin{cases} \max(y - b_i^*, 0), & y < b_i \text{ or } y = 0 \text{ with } b_i = 0, \\ m_i, & y = b_i > 0 \text{ or } y = 0^*. \end{cases}$$

Note that $\lambda_{i}(j, k)$ is determined by the assumption of arrival process.

(2) Construct the matrices $A_i^{(n)}, 2 \leq i \leq N$. Note that the matrices $A_i^{(n)}$ do not contain any unknown parameters and they are not necessary to be updated in iteration step.

(3) Calculate $\pi_N$ of $Q_N$ and throughput $\Theta^{(0)}$, and calculate $\beta_{N-1}(x, y)$ and $\delta_{N-1}(x, y)$ for $b_{N-1} > 0$ and $\alpha_{N-1}(x)$, $\beta_{N-1}^0(x)$ and $\delta_{N-1}^0(x)$ for $b_{N-1} = 0$. Note that the matrices $C_i^{(n)}$ are not necessary to be updated in the iteration step.

1. Backward step.: For $i = N - 1, N - 2, \cdots, 2$,

   (1) update the matrices $B_i^{(n)}$ and $C_i^{(n)}$ using $\beta_i(x, y)$ and $\delta_i(x, y)$ for $b_{i-1} > 0$ ($\alpha_i(x)$, $\beta_i^0(x)$ and $\delta_i^0(x)$ for $b_{i-1} = 0$) calculated in the previous step and calculate $\pi_i$ of $Q_i$, and

   (2) if $i \geq 3$, then update $\beta_{i-1}(x, y)$ and $\delta_{i-1}(x, y)$ for $b_{i-1} > 0$ and $\alpha_{i-1}(x)$, $\beta_{i-1}^0(x)$ and $\delta_{i-1}^0(x)$ for $b_{i-1} = 0$ using the formulae in Section 5.

   (3) If $i = 2$, then update $\lambda_2(x, y)$ using the stationary distribution $\pi_2$ of $Q_2$, and go to the forward step.

2. Forward step.: For $i = 3, \cdots, N - 1$,

   (1) update the matrices $B_i^{(n)}$ using $\lambda_{i-1}(x, y)$ calculated in the previous step and calculate $\pi_i$ of $Q_i$, and

   (2) update $\lambda_i(x, y)$ using the formulae in Section 5.

3. Tolerance check.: In the last subsystem $L_N$, calculate the throughput and check the stopping criterion as follows:

   (1) Update $B_N^{(n)}$ and $\pi_N$ of $Q_N$.

   (2) Calculate throughput $\Theta^{(m)}$.

   (3) Check the tolerance (stopping criterion)

$$TOL = |\Theta^{(m)} - \Theta^{(m-1)}| < \epsilon,$$

where $\Theta^{(m)}$ is the throughput obtained in the $m$th iteration and $\epsilon > 0$ is the predetermined tolerance. If the stopping criterion is satisfied, then stop the iteration, otherwise
update $\beta_{N-1}(x, y)$ and $\delta_{N-1}(x, y)$ for $b_{N-1} > 0$ \((\alpha_{N-1}(x), \beta_{N-1}^0(x))\) and $\delta_{N-1}^0(x)$ for $b_{N-1} = 0$), and repeat the backward and forward iterations until the stopping criterion is satisfied.

**Remark.** One can start the iteration with initial guess of the departure rates under the assumption that $W_i^d$ in $L_i$ is never blocked. In this case, the iteration step is performed by the procedure that the arrival rates are updated in forward step, and then departure rates are updated in backward step in the first iteration, and repeat this procedure until the stopping criterion is satisfied.

**Complexity of algorithm.** The stationary distribution $\pi_i$ of $Q_i$ can be calculated using well-known matrix geometric method (e.g. Shin [14]). Within the iterative algorithm, solving a subsystem is time consuming. To solve subsystem $L_i$ using the algorithm, we must invert $K_i = \kappa_i + 1$ matrices with the maximum size $s_i^* = \max_{0 \leq n \leq K_i} s_i(n)$. Therefore the complexity of one iteration becomes $O(\sum_{i=1}^{N} K_i(s_i^*)^3)$.

The number of iterations required is difficult to predict because it depends on the tolerance $\epsilon$ and the length of the line and system parameters. For example, as shown from numerical experiments in Section 8, the number of iterations increases with the line length. Although the convergence of the iteration scheme is not proven analytically, extensive numerical experiments indicate the convergence of the iteration.

7. **TANDEM QUEUES WITH MULTIPLE SERVERS UNDER GENERAL BLOCKING SCHEME**

We apply the method to the system with multiple exponential servers and several blocking scheme.

Consider a tandem queue in which the node $W_i$ consists of service station $M_i$ and a buffer $G_i$ of capacity $g_i$, as depicted in Fig.1. The service station $M_i$ has $m_i$ identical servers in parallel and the service time of each server at $M_i$ is of exponential with rate $\mu_i$. The capacity of the node $W_i$ is $c_i = g_i + m_i$. Let $a_i^*$ be the size of buffer space in $G_i$ for active customers and let $b_i^* (\leq b_i)$ be the size of buffer space in $G_i$ for blocked customers. The node $W_i$ can contain $a_i = a_i^* + m_i$ active customers. Note that $0 \leq a_i^* \leq g_i$, $0 \leq b_i^* \leq g_i$ and $a_i^* + b_i^* \geq g_i$ and the maximal number of blocked servers is $b_i^* = b_i - b_i^* (\leq m_i)$. Assuming that the last node $M_N$ is never blocked, $b_N = 0$ and $a_N = c_N$.

The source node $W_0$ behaves like a system with $m_0$ servers and a virtual buffer of size $b_0$ for blocked customers to enter the first node $W_1$. We assume that $W_0$ is never starved and each server in $W_0$ starts service immediately after a service completion unless the server is blocked. The service time of each server is exponential with rate $\mu_0$. The arrival rate to $W_1$ is

$$\lambda_1(x, y) = (m_0 - y_0^b(x, y))\mu_0,$$

where $y_0^b(x, y)$ is the number of blocked servers in $W_0$. If $W_0$ is a BBS node, then the customers arrive according to an ordinary Poisson process with constant rate $m_0\mu_0$ and blocked customers are lost, that is,

$$\lambda_1(x, y) = \begin{cases} m_0\mu_0, & x < a_1, x + y < c_1, \\ 0, & \text{otherwise}. \end{cases}$$
Figure 1. Tandem queueing network with general blocking

Figure 2. Blocking mechanism for \( b_i > 0 \)

Case (i) BAS node \((b_i > 0)\). In this case, the blocked customer is stocked at the buffer \( G_i \) if there is a place available for blocked customers, otherwise it is stayed at the server just service completed and the server is blocked (Figure 2).

The server blocked at \( M_i \) is forced to stop its service until there is a place available for active customers at \( W_{i+1} \). If the number of blocked customers at \( W_i \) reaches \( b_i \) upon a service completion, then all the servers at \( M_i \) are blocked and forced to stop their service. We coin this type of blocking rule the generalized blocking after service (GBAS) rule. This blocking scheme contains many BAS blocking schemes as a special cases:

1. The blocking scheme with \( b_i = b_i^* + m_i \) is an ordinary GB scheme in the system with multiple servers.
2. If \( b_i^* = 0 \) and \( b_i = m_i \), then the blocking mechanism is the ordinary BAS rule in [12].
3. If \( b_i = b_i^* > 0 \), then the blocked customers are stocked only at the buffer. All the servers are blocked upon the blocking level reaches \( b_i \), however, the idle server can accept an active customer even it is stopped its service. The blocking state of all the servers changes to working states if the level of blocked customers downs to the below of \( b_i \).
When $0 < b^*_i < m_i$, then all the servers at $M_i$ are forced to stop their service upon the number of the blocked customers at $M_i$ reaches $b_i$, however the idle server can accept an active customer even it is stopped its service. The servers that are not blocked start new service whose length is exponential distributed random variable with rate $\mu_i$ upon the level of blocked customers becomes less than $b_i$.

Given $W_i(t) = (x, y)$, the number of blocked servers $m^b_i(y)$ at $M_i$ is

$$m^b_i(y) = \begin{cases} \max(y - b^*_i, 0), & y < b_i, \\ m_i, & y = b_i \end{cases}$$

and hence the service rate from $M_i$ is given by

$$\mu_i(x, y) = \min(x, m_i - m^b_i(y)) \mu_i.$$  

Case (ii) BBS node ($b_i = 0$). In this case, the service rate of $M_i$ depends on the number $x$ of AC and the server state $s$ of $M_i(t)$ and is given by

$$\mu_i(x, s) = \begin{cases} \min(x, m_i) \mu_i, & s = 0, \\ 0, & s = 0^* \end{cases}.$$  

8. Numerical Results

To investigate the accuracy of the method proposed in this paper, the method is applied to the tandem queue with multiple servers and the results are compared with simulations. The simulation models for the systems in the tables were developed using ARENA [15]. The simulation run time was set to 100,000 unit times, including a warm-up time of 10,000 unit times. Ten replications were conducted for each case and the half length of the 95\% confidence interval (c.i.) was obtained. A tolerance $\epsilon = 10^{-5}$ is used for stopping criterion in (6.1). To validate the simulation program, the simulation are compared with the exact one for throughput of the system with $N = 2$ in Table 1. The table shows that simulation can be used as an alternative of exact analysis.

The current approximation for the ordinary BAS system with $N = 5$, $N = 11$ and $b^*_i = 0$, $b_i = m_i$ is compared with the method (SM) of Shin and Moon [12] in Tables 2. The simulation

<table>
<thead>
<tr>
<th>$m_i$</th>
<th>$g_i$</th>
<th>$(b^<em>_0, b^</em>_1, b^*_2)$</th>
<th>$(b_0, b_1, b_2)$</th>
<th>Exact</th>
<th>Sim (c.i.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>(0,0,0)</td>
<td>(0,0,0)</td>
<td>0.7359</td>
<td>0.7357(±0.0017)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0,0,0)</td>
<td>(0,1,1)</td>
<td>0.7543</td>
<td>0.7542(±0.0012)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0,2,2)</td>
<td>(0,3,3)</td>
<td>0.7677</td>
<td>0.7675(±0.0014)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(3,2,2)</td>
<td>(3,3,3)</td>
<td>0.8266</td>
<td>0.8257(±0.0012)</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>(0,0,0)</td>
<td>(0,0,0)</td>
<td>0.7744</td>
<td>0.7798(±0.0011)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0,0,0)</td>
<td>(0,3,3)</td>
<td>0.7966</td>
<td>0.8009(±0.0008)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0,0,0)</td>
<td>(3,3,3)</td>
<td>0.8360</td>
<td>0.8363(±0.0009)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(3,3,3)</td>
<td>(3,5,5)</td>
<td>0.8495</td>
<td>0.8520(±0.0009)</td>
</tr>
</tbody>
</table>
results (Sim) and the numerical results for SM in Tables 3 are from [12]. The measure of the deviation of approximation (App) from the simulation (Sim) is calculated by the formula \( D(\%) = ((\text{App} - \text{Sim})/\text{Sim}) \times 100 \). Table 3 shows that the accuracy of current method is similar to that of SM.

We consider a tandem queue that consists of \( N \) nodes and each node has multiple servers and follows a general blocking scheme. Customers arrive from outside \((W_0)\) according to a Poisson process and the blocked customers entering into the node \( W_1 \) are lost, that is, \( m_0 = 1, b_0 = 0, \) and \( \mu_0 = 1.0 \). We assume that \( a_i = m_i + g_i \) and \( a_i^* = g_i \). In the following, the \( N \)-dimensional vectors \( \mathbf{m} = (m_1, \ldots, m_N) \) and \( \mathbf{g} = (g_1, \ldots, g_N) \) mean the number of servers and the buffer size in the system, respectively. For example, the vector \( \mathbf{m} = (1, 3, 2, 4) \) means that \( m_1 = 1, m_2 = 3, m_3 = 2, m_4 = 4 \). Similarly, denote the upper limits of customers in buffer and in the system by the \( N \)-dimensional vectors \( \mathbf{b}^* = (b_1^*, \ldots, b_{N-1}^*, 0) \) and \( \mathbf{b} = (b_1, \ldots, b_{N-1}, 0) \), respectively. If the number \( m_i \) of servers at node \( W_i \), \( i = 1, 2, \ldots, N \) are the same as a constant \( k \), we write \( m_i = k \) instead of using vector \( \mathbf{m} \). Similarly, if \( g_i, 1 \leq i \leq N \) are the same as a constant \( j \), then we write \( j \) in stead of vector \( \mathbf{g} \).

**Table 2.** Throughput for the system of length \( N = 5 \) with \( g_i = 3 \)

<table>
<thead>
<tr>
<th>( m_i )</th>
<th>( (3,3,3,3) )</th>
<th>( (3,1,3,1) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( b_i^* )</td>
<td>( b_i )</td>
<td>Sim</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0.727</td>
</tr>
<tr>
<td>( m_i )</td>
<td>( m_i )</td>
<td>0.764</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>0.764</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>0.777</td>
</tr>
<tr>
<td>( m_i )</td>
<td>( c_i )</td>
<td>0.786</td>
</tr>
</tbody>
</table>

**Table 3.** Throughput for tandem queues under ordinary BAS blocking

<table>
<thead>
<tr>
<th>( N )</th>
<th>( m_i )</th>
<th>( \mu_i )</th>
<th>( g_i )</th>
<th>Sim (CI)</th>
<th>SM (D(%)</th>
<th>App (D(%))</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>3</td>
<td>( \frac{1}{m_1} )</td>
<td>0</td>
<td>0.643 (±0.002)</td>
<td>0.634 (-1.3)</td>
<td>0.637 (-0.8)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0.776 (±0.003)</td>
<td>0.777 (0.1)</td>
<td>0.775 (0.0)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td>0.819 (±0.004)</td>
<td>0.820 (0.1)</td>
<td>0.819 (0.0)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.0</td>
<td>1.930 (±0.003)</td>
<td>1.903 (-1.4)</td>
<td>1.912 (-0.9)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>2.330 (±0.004)</td>
<td>2.330 (0.0)</td>
<td>2.326 (-0.2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td>2.461 (±0.005)</td>
<td>2.461 (0.0)</td>
<td>2.458 (-0.1)</td>
</tr>
<tr>
<td>11</td>
<td>( m_{2i} = 4, )</td>
<td>( \frac{1}{m_i} )</td>
<td>0</td>
<td>0.557 (±0.002)</td>
<td>0.545 (-1.7)</td>
<td>0.543 (-2.6)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0.734 (±0.001)</td>
<td>0.737 (0.4)</td>
<td>0.730 (-0.4)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td>0.788 (±0.003)</td>
<td>0.790 (0.2)</td>
<td>0.786 (-0.3)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.0</td>
<td>0.726 (±0.003)</td>
<td>0.725 (-0.2)</td>
<td>0.721 (-0.7)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>0.853 (±0.002)</td>
<td>0.863 (1.2)</td>
<td>0.854 (-0.1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td>0.888 (±0.002)</td>
<td>0.894 (0.7)</td>
<td>0.889 (-0.1)</td>
</tr>
</tbody>
</table>
The throughput for the system of length \( N = 5 \) with \( g_i = 3 \) are listed in Table 2. The throughput and the mean number \( E[W] = \sum_{i=1}^{N} E[W_i] \) of customers in the system are presented in Table 5 for the systems of length \( N = 10 \) with parameters in Table 4. The mean number \( E[W_i] \) of customers in node \( W_i \) are presented in Table 6. In Table 2, Table 5 and Table 6, the half length of 95\% confidence interval of simulation results for throughput and for the mean number of customers are less than 0.002 and 0.3, respectively, and confidence intervals are omitted. The table shows that the approximation works well for the throughput and mean number of customers.

### Table 4. Scenarios for the system of length \( N = 10 \) in Table 5 and Table 6

<table>
<thead>
<tr>
<th>Cases</th>
<th>Type</th>
<th>( m_i )</th>
<th>( g_i )</th>
<th>( b_i^* )</th>
<th>( b_i )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BBS</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>BAS</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>BAS</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>Kanban</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>Mixed</td>
<td>1</td>
<td>3</td>
<td>( b_i^* )</td>
<td>( b_i )</td>
</tr>
<tr>
<td>6</td>
<td>BBS</td>
<td>( m )</td>
<td>( g )</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>BAS</td>
<td>( m )</td>
<td>( g )</td>
<td>( g_0 )</td>
<td>( g_0 )</td>
</tr>
<tr>
<td>8</td>
<td>Kanban</td>
<td>( m )</td>
<td>( g )</td>
<td>( g_0 )</td>
<td>5</td>
</tr>
<tr>
<td>9</td>
<td>Mixed</td>
<td>( m )</td>
<td>( g )</td>
<td>( b_i^* )</td>
<td>( b_i )</td>
</tr>
<tr>
<td>10</td>
<td>Mixed</td>
<td>( m )</td>
<td>( g )</td>
<td>( b_i^* )</td>
<td>( b_i )</td>
</tr>
</tbody>
</table>

### Table 5. Throughput and mean number \( E[W] \) of customers in the system with \( N = 10 \)

<table>
<thead>
<tr>
<th>Cases</th>
<th>( \mu_i )</th>
<th>Throughput</th>
<th>( E[W] )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( \frac{1}{m_i} )</td>
<td>( \frac{1}{m_i} )</td>
<td>( \frac{1}{m_i} )</td>
</tr>
<tr>
<td>1</td>
<td>1.0</td>
<td>0.641</td>
<td>0.637</td>
</tr>
<tr>
<td>2</td>
<td>1.0</td>
<td>0.714</td>
<td>0.710</td>
</tr>
<tr>
<td>3</td>
<td>1.0</td>
<td>0.730</td>
<td>0.724</td>
</tr>
<tr>
<td>4</td>
<td>1.0</td>
<td>0.737</td>
<td>0.731</td>
</tr>
<tr>
<td>5</td>
<td>1.0</td>
<td>0.703</td>
<td>0.697</td>
</tr>
<tr>
<td>6</td>
<td>( \frac{1}{m_i} )</td>
<td>0.662</td>
<td>0.645</td>
</tr>
<tr>
<td>7</td>
<td>( \frac{1}{m_i} )</td>
<td>0.716</td>
<td>0.710</td>
</tr>
<tr>
<td>8</td>
<td>( \frac{1}{m_i} )</td>
<td>0.890</td>
<td>0.895</td>
</tr>
<tr>
<td>9</td>
<td>( \frac{1}{m_i} )</td>
<td>0.721</td>
<td>0.720</td>
</tr>
<tr>
<td>10</td>
<td>( \frac{1}{m_i} )</td>
<td>0.704</td>
<td>0.699</td>
</tr>
</tbody>
</table>

The table shows that the approximation works well for the throughput and mean number of customers.
Run time. The current algorithm was performed on a laptop computer at 2.80GHz 16.0 GB RAM using Mathematica® [16] for the system with $a_i = b_i = c_i$, $a^*_i = b^*_i = g_i$ and $\mu_i = \frac{1}{m_i}$. The stopping criterion $\epsilon = 10^{-5}$ was used. The number of iterations (NI) and run time (CPU) in seconds are listed in Table 7. The behavior of the run time and the number of iteration of the algorithm as a function of the buffer size is depicted in Fig. 3. The table and figure show that the run time of the algorithm increases with the line length, buffer size and the number of servers at each node, and it depends significantly on the buffer size and the number of servers. The number of iterations is more sensitive to the length $N$ of the line than the buffer size and the number of servers.

Table 6. Mean number $E[W_i]$ of customers at node $W_i$ in the system with $N = 10, \mu_i = \frac{1}{m_i}$

<table>
<thead>
<tr>
<th>Node</th>
<th>Case 1 (BBS) App (D(%))</th>
<th>Case 4 (Kanban) App (D(%))</th>
<th>Case 9 (Mixed) App (D(%))</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.68 (0.2) 2.36 (0.3)</td>
<td>3.55 (0.1)</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2.42 (0.2) 2.67 (0.2)</td>
<td>3.25 (1.0)</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2.28 (0.4) 2.72 (-0.6)</td>
<td>3.91 (-1.1)</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>2.16 (0.5) 2.71 (-1.2)</td>
<td>3.04 (-6.3)</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>2.06 (0.5) 2.66 (-1.9)</td>
<td>3.44 (-2.7)</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1.96 (0.7) 2.60 (-2.2)</td>
<td>3.72 (-2.1)</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>1.86 (0.9) 2.53 (-2.1)</td>
<td>2.87 (1.3)</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>1.75 (1.2) 2.42 (-1.8)</td>
<td>2.71 (0.8)</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>1.59 (1.1) 2.25 (-1.3)</td>
<td>2.03 (0.1)</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>1.32 (1.1) 1.90 (-1.3)</td>
<td>2.23 (0.5)</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>20.09 (0.5) 25.12 (-1.2)</td>
<td>30.74 (-1.4)</td>
<td></td>
</tr>
</tbody>
</table>

Table 7. CPU time for Kanban system in seconds ($\mu_i = 1.0/m_i$)

<table>
<thead>
<tr>
<th>$N$</th>
<th>$g_i$</th>
<th>$m_i = 1$ NI CPU</th>
<th>$m_i = 3$ NI CPU</th>
<th>$m_i = 5$ NI CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>1</td>
<td>8 0.2 7 1.4 7 4.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>7</td>
<td>1.4 7 4.9 7 13.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>7</td>
<td>4.9 7 13.8 7 32.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>13.6 7 31.8 7 69.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>15</td>
<td>1.0 14 6.1 14 22.9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>13</td>
<td>5.6 13 20.5 13 60.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>12</td>
<td>19.6 12 56.1 12 133.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>12</td>
<td>55.8 12 131.4 12 284.4</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
9. CONCLUDING REMARKS

An approximation method for tandem queues with finite buffers and exponential service times with state dependent service rates under general blocking scheme was presented. The model considered in this paper is very general and it contains the system with multi-server node and many classical blocking scheme such as ordinary manufacturing blocking, communication blocking, and kanban blocking as special cases. Extensive numerical experiments show that the current method is very effective in the sense of accuracy of approximation and computation time even for the system that consists of nodes with different blocking schemes.

REFERENCES

PREDICTING KOREAN FRUIT PRICES USING LSTM ALGORITHM

TAE-SU PARK1, JONGHAE KEUM2, HOISUB KIM2, YOUNG ROCK KIM2,3, AND YOUNGHO MIN3†

1 DEPARTMENT OF MATHEMATICS, HANKUK UNIVERSITY OF FOREIGN STUDIES, YONGIN, 17035, REPUBLIC OF KOREA
Email address: tessinu51@gmail.com

2 SCHOOL OF MATHEMATICS, KOREA INSTITUTE FOR ADVANCED STUDY, 85 HOEGIRO DONGDAEMUN-GU, SEOUL, 02455, REPUBLIC OF KOREA
Email address: jhkeum@kias.re.kr, hskimndial@naver.com

3 DEPARTMENT OF MATHEMATICS EDUCATION, HANKUK UNIVERSITY OF FOREIGN STUDIES, SEOUL, 02450, REPUBLIC OF KOREA
Email address: rocky777@hufs.ac.kr, †pumpzxc@gmail.com

ABSTRACT. In this paper, we provide predictive models for the market price of fruits, and analyze the performance of each fruit price predictive model. The data used to create the predictive models are fruit price data, weather data, and Korea composite stock price index (KOSPI) data. We collect these data through Open-API for 10 years period from year 2011 to year 2020. Six types of fruit price predictive models are constructed using the LSTM algorithm, a special form of deep learning RNN algorithm, and the performance is measured using the root mean square error. For each model, the data from year 2011 to year 2018 are trained to predict the fruit price in year 2019, and the data from year 2011 to year 2019 are trained to predict the fruit price in year 2020. By comparing the fruit price predictive models of year 2019 and those models of year 2020, the model with excellent efficiency is identified and the best model to provide the service is selected. The model we made will be available in other countries and regions as well.

1. INTRODUCTION

Fruits are the oldest food of mankind and many countries around the world are conducting research on the supply and demand of fruits to analyze the market price of fruits [1, 2, 3, 4]. One of the factors that greatly influences fruit supply and price is the weather, which is measured by indicators such as temperature, precipitation and wind speed. If the appropriate temperature and precipitation are not adjusted for each fruit, problems such as lowering of sugar content or damage to the fruit occur, which adversely affects the wholesale price of fruit [5]. The uncertainty of the weather makes fruit suppliers feel anxious. Therefore, various countermeasures such as high tunnels, revenue insurance, and weather insurance have been
proposed to solve this problem [6]. Several countries have implemented insurance policies for crops, including fruits, to stabilize fruit prices and protect fruit suppliers. An example of a nationally implemented fruit supply and demand policy is the US Supplemental Nutrition Assistance Program (SNAP). As part of the consumption policy to improve the people’s healthy eating habits and increase fruit consumption, it is supporting subsidies for fruit consumption by the low-income class. It was based on a study of the relationship between government subsidies and fruit consumption reported to the US Department of Agriculture (USDA) [7]. Also, for fruit price stabilization, not only the supply and demand of domestically produced fruits but also the prices of imported fruits are considered. Fruits imported indiscriminately at very low prices adversely affect the stabilization of fruit prices in the region, and excessively high tariffs also prevent the provision of fresh fruits at good prices to consumers. Rickard, BJ, & Lei, L. simulated the reduction of global tariffs and elimination of sanitary and phytosanitary (SPS) in the apple and orange markets to analyze the impact of tariff and non-tariff barriers on consumers and suppliers in the international market and suggested strategies for stabilizing fruit prices [8]. In this way, the price of fruit is determined by many variables related to supply and demand. The stable fruit price guarantees economic benefits for suppliers and provides healthy food to consumers.

Stabilization of fruit prices is a very important issue from a regional and international perspective, and various studies are being conducted for this purpose. However, there are not many studies that apply deep learning to fruit or agricultural and fishery products data. In this paper, we pay attention to the prediction of fruit prices using deep learning to help stabilize fruit prices. We can suggest several policies through accurate fruit price prediction. If fruit prices are predicted to be higher than those in previous years, the government can stabilize fruit prices by increasing imports of overseas substitute fruits and implementing policies that guarantee the profits of suppliers. Conversely, if fruit prices are expected to be low, policies to support consumers’ fruit consumption can be planned [9].

As an example of the introduction of deep learning on fruit prices in Korea, there is a study on a fruit price predictive model using artificial intelligence by Im, J. M., Kim, W. Y., Byoun, W. J., & Shin, S. J. [10]. They predicted fruit price using time series data based on LSTM (Long Short Term Memory) algorithm among RNN (Recurrent Neural Network) algorithms. In their paper, the attempt to analyze fruit prices through deep learning was good, but the data used for training the deep learning algorithm were only short-term weather data, and the prediction target value was only the price of apples in a specific period. There is the other study by Shin, S., Lee, M., & Song, S. K. who studied agricultural product price prediction using LSTM network [11]. In their paper, 108 features for training was used to predict the price of agricultural products and root mean square error (RMSE) was used as a performance measurement tool. They obtained the root mean square error ranging from 0.065 to 0.121 for city/agricultural products.

The goal of our paper is to present predictive models for the market price of fruits using the LSTM algorithm, and to provide highly efficient predictive models with a small number of features by analyzing the performance of each fruit price predictive model.
This paper consists of 8 sections and proceeds in the following order. In Section 2, we explain RNN and LSTM, which are machine learning techniques used in this study. Section 3 contains a description of the workflow and an overview of the progress of this study. In Section 4, we collect the data through the Open-API, and describe the collected data. In Section 5, we explain the process of preprocessing the data obtained in Section 3 and the merging of data used for training. We construct predictive models for fruit prices in Section 6. In Section 7, we summarizes the learning process, compares and analyzes the performance of the models. Finally, we discuss the conclusions of this study in Section 8.

2. Methodology

Unlike general programming, machine learning refers to programming that allows a program to learn and develop by itself by implanting a neural network that mimics the structure of a human neuron into an algorithm. When the number of hidden layers of the neural network used in the machine learning increases, we call this deep learning. The core idea of deep learning is to find the weights and biases of the neural network that minimizes the loss function through backpropagation method [12]. In deep learning, the number of weights and biases increases exponentially as the neural network grows deeper.

The fruit prices, weather, and Korea composite stock price index data that we are going to cover in this paper are time series data, and which generally have very long sequence. Therefore we need to use long-length deep neural networks for handling time series data. However, it takes a lot of memory and time to store and update all the weights and biases to train these deep neural networks. This problem was solved by RNN using weights and biases again. In 1993, Schmidhuber, J. stated that RNN is suitable for handling data requiring more than 1000 subsequent layers [13]. For this reason, the RNN is used when dealing with time series data in various research fields, and our models are also created using LSTM that is a kind of RNN algorithm. In this section, we introduce the basic RNN algorithm and the LSTM algorithm to explain our models.

![Figure 1. Basic feedforward neural networks and recurrent neural networks](image)
2.1. **Recurrent Neural Network (RNN)**. The RNN algorithm is one of artificial neural network and has a recurrent characteristic of applying the weights obtained through learning. Fig. 1 is a diagram comparing the basic feed-forward neural network and the recurrent neural network, in short, RNN. In the figure, the arrow direction means the network flow. $X$ indicates the input layer, $h$ is the hidden layer, and $Y$ is the output layer. In the basic feed-forward neural network, input data are transmitted from the input layer to the hidden layer and then from the hidden layer to the output layer. However, in the RNN, the hidden layer has a structure that receives information from the hidden layer of the previous time step as well as information from the input layer. This flow of information is called a loop or recurrent edge, and the name RNN is derived from this.

![Diagram showing RNN structure](image)

**Figure 2.** Recurrent neural network

Figure 2 shows the unfolded recurrent neural network structure. The detailed method of calculating the outputs of the hidden layer and the output layer with the input data in the $t$-step is as follows. Let $W_{xh}$, $W_{hh}$, $W_{hy}$ be the weight matrix between the input $X_t$ and the hidden layer, the weight matrix for the recursive edge, and the weight matrix between the hidden layer and the output layer, respectively. The new input data $X_t$ and the past learning results $h_{t-1}$ are multiplied by the corresponding weights and added, and then the bias $b_{ht}$ at the $t$-step is added. It is expressed as a linear combination

$$Z_t = W_{xh}X_t + W_{hh}h_{t-1} + b_{ht}. \quad (2.1)$$

By putting this input value $Z_t$ into the hyperbolic tangent activation function, the output $h_t$ at $t$-step can be calculated. If the weight matrix is defined as $W_h = [W_{xh} : W_{hh}]$, then this process can be simply expressed as follows.
\[ h_t = \tanh \left( W_h \left[ X_t \right] + b_h \right), \quad (2.2) \]

\[ y_t = W_{hy} h_t + b_y, \]

where \( b_y \) is the bias for output layer. Now, the output \( h_t \) is transferred to the next hidden layer, multiplied by the weight again, added to the input value \( W_{xh} X_{t+1} \) of the next layer, and then the bias \( b_h \) is added, repeating the structure. Here, the weights \( W_{xh}, W_{hh}, W_{hy} \) are used again. Because of this structural characteristic, the order of the data has a great influence on the learning result.

Although RNN is good at handling sequential data, there are some problems when we deal with very long sequences. In backpropagation phase, we calculate the gradient of the loss function at time step \( t \) as follows:

\[
\frac{\partial L_t}{\partial W_{hh}} = \frac{\partial L_t}{\partial y_t} \times \frac{\partial y_t}{\partial h_t} \times \left( \sum_{k=1}^{t} \frac{\partial h_t}{\partial h_k} \times \frac{\partial h_k}{\partial W_{hh}} \right),
\]

where \( \frac{\partial h_t}{\partial h_k} = \prod_{i=k+1}^{t} \frac{\partial h_i}{\partial h_{i-1}} \) which is \((t - k)\) multiplicative terms. By the Eq. (2.2), we can obtain

\[
\frac{\partial h_i}{\partial h_{i-1}} = \frac{\partial \tanh(Z_i)}{\partial Z_i} \frac{\partial Z_i}{\partial h_{i-1}}.
\]

In the Eq. (2.1), \( Z_i \) consists of the input term \( X_i \), the previous hidden layer output \( (h_{i-1}) \) and bias term \( (b_h) \). Therefore, only the weight matrix \( W_{hh} \) remains in \( \frac{\partial Z_i}{\partial h_{i-1}} \). Since the derivative of the \( \tanh \) activation function is positive and smaller than 1, therefore if \( t \) is very large and \( \| W_{hh} \| < 1 \), then vanishing gradient problem is occurred where \( \| \cdot \| \) is any matrix norm [14]. Also if \( t \) is very large and norms of \( W_{hh} \) are large enough to overpower the smaller derivative of \( \tanh \), the exploding gradient problem is occurred [15]. In next subsection, we introduce LSTM as the solution to these problems.

### 2.2. LSTM (Long Short Term Memory)

Hochreiter and Schmidhuber proposed LSTM, a modification of the RNN, to solve the vanishing gradient problem caused by prolonged learning [16]. LSTM has an internal structure called a memory cell, and the memory cell contains a recurrent edge that maintains an appropriate weight \( \| W \| = 1 \). The output of this recurrent edge is called a cell state.

The flow of information in the memory cell is controlled through several gates. In general, there are three types of gates in an LSTM cell. These are the forget gate, the input gate, and the output gate. The forget gate \( f_t \) adjusts the output of the hidden unit at the \( t - 1 \) time step \( (h_{t-1}) \) and the input at the \( t \) time step \( (X_t) \) to determine which information to pass through and which information to suppress [17]. \( f_t \) is calculated as follows:

\[ f_t = \sigma(W_{xf} X_t + W_{hf} h_{t-1} + b_f), \]
where $\sigma$ is the sigmoid function. The input gate ($i_t$) and the input node ($g_t$) update the cell state,

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i)$$
$$g_t = \sigma(W_{xg}x_t + W_{hg}h_{t-1} + b_g).$$

The cell state at time step $t$ ($C_t$) is updated by element-wise adding information received from the input gate and input node.

$$C_t = (C_{t-1} \otimes f_t) \oplus (i_t \otimes g_t)$$

where the symbol $\otimes$ means element-wise multiplication and the symbol $\oplus$ means element-wise addition, respectively. This network is designed to obtain the cell state of the next time step without directly multiplying the cell state of the previous time step with any weight. This structure of LSTM solves the vanishing gradient problem by avoiding the problem of multiplying weights by superposition. Using this cell state ($C_t$), the output of the hidden unit at the time step $t$ is calculated as follows:

$$h_t = o_t \otimes \tanh(C_t),$$

where $o_t$ is the output gate defined by $o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o)$. Figure 3 shows the detailed structure of the LSTM.

---

3. FRUIT PRICE PREDICTION WORKFLOW

We will explain the workflow that summarizes the process of this study in Fig. 4. A brief description of each part is as follows.

First, data are collected through Open-API (Open Application Programming Interface). The collected data are fruit price data, weather data, and comprehensive stock index data. The
collected fruit price data contain prices of a variety of fruits including our target fruits (apples, pears, persimmons, bananas, and oranges) by 5 different regions in Korea. This raw data include quality and daily price of each fruit. Weather data have 15 features which are average temperature, minimum temperature, maximum temperature, daily precipitation, maximum instantaneous wind speed, maximum wind speed, average wind speed, average dew point temperature, minimum relative humidity, average relative humidity, average vapor pressure, hot water time, total solar time, total solar radiation and average surface temperature. KOSPI data and KOSDAQ data are collected as comprehensive stock index data. Data collection will be described in detail in Section 4.

Next, we will preprocess the collected data. Fruit price data consist of data based on the trading day. Preprocessing of weather data is performed through multiple linear regression analysis. Composite stock index data are extracted as KOSPI data on the same day as the fruit trading day, and preprocessing is performed. The preprocessing content will be described in detail in Section 5.

Based on the preprocessed data, six types of data sets are constructed according to the independent variables used for each model. The details of the independent variables used in each model will be described in detail in Section 6.

For each model, we will train the data from year 2011 to year 2018 to predict the fruit price in year 2019 and compare it with the actual fruit price. Similarly, we will train on data from year 2011 to year 2019 to predict fruit prices in year 2020. We will use the root mean square error to measure performance and select the best model. This will be described in detail in Section 7.
4. Data Collection and Virtualization

In this study, fruit price data, weather data related to fruit prices, and comprehensive stock index data are collected. The collected data are daily data from January 2011 to December 2020 in Korea. All data are collected using Open-API. Fruit price data are collected from Korea Agro-Fisheries & Food Trade Corporation, in short KAMIS (www.kamis.or.kr) Open-API and weather data in public data portal (www.data.go.kr) are collected. Also, we could collect data through the Open-API and pandas-datareader library for the comprehensive stock index. High price, low price, opening price, closing price, trading volume in KOSPI index, and revised stock information are standardized and collected by date through the Python library.

4.1. Fruit price data. Fruit price data are collected via Open-API and are provided by KAMIS. The ‘wholesale/retail price information by category’ API is selected among the open APIs provided. We request the data from KAMIS using the request URL and request parameter, and we receive data as the response field.

<table>
<thead>
<tr>
<th>Request Parameter</th>
<th>Value</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>p_cert_key</td>
<td>string</td>
<td>certification key</td>
</tr>
<tr>
<td>p_cert_id</td>
<td>string</td>
<td>requester id</td>
</tr>
<tr>
<td>p Returntype</td>
<td>string</td>
<td>Return Type</td>
</tr>
<tr>
<td>(json:Json data form, xml: XML data form)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>p_product_cls_code</td>
<td>string</td>
<td>division</td>
</tr>
<tr>
<td>( 01:retail, 02:wholesale, default: 02 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>p_item_category_code</td>
<td>string</td>
<td>class code</td>
</tr>
<tr>
<td>(100:food crops, 200:vegetables, 300: special crop, 400: fruits, 500: livestock, 600: seafood, default: 100)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>p_country_code</td>
<td>string</td>
<td>retail price select area</td>
</tr>
<tr>
<td>p_regday</td>
<td>string</td>
<td>Date: yyyy-mm-dd (default: latest survey date)</td>
</tr>
<tr>
<td>p_convert_kg_yn</td>
<td>string</td>
<td>Whether to convert in kg unit (Y: 1 kg unit display, N: information survey unit display, ex: rice 20 kg)</td>
</tr>
</tbody>
</table>
The request parameter is a variable requested together with the request URL when we request data from the server. Table 1 shows the data explaining the configuration of the request variable. Table 2 describes an extract of some of the output results.

### Table 2. Response elements

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>condition</td>
<td>string</td>
<td>request message</td>
</tr>
<tr>
<td>item_name</td>
<td>string</td>
<td>Item name</td>
</tr>
<tr>
<td>itemcode</td>
<td>string</td>
<td>Item code</td>
</tr>
<tr>
<td>dpr1</td>
<td>string</td>
<td>View Date Price</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### 4.2. Weather data and composite stock index data.

Weather data and comprehensive stock index data are collected using Open-API provided by the public data portal (www.data.go.kr). In the case of weather data, the ‘terrestrial (synoptic, ASOS) daily data inquiry service’ is selected among the Open-APIs provided through ‘Daily Weather Data Inquiry’, and this is a service that inquires the daily weather data observed with the synoptic meteorological observation equipment. We requested corresponding data through a request URL and a request parameter to receive the corresponding data, and we received data as an output result field (response field). Table 3 has the contents extracted from some of the request variables. The following is the request url. http://apis.data.go.kr/1360000/AsosDalyInfoService/getWthrDataList

### Table 3. Weather request parameters

<table>
<thead>
<tr>
<th>Item name</th>
<th>Item size</th>
<th>Category</th>
<th>sample data</th>
<th>Item Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serviceKey</td>
<td>100</td>
<td>1</td>
<td>certification key (URL Encode)</td>
<td>Public data portal certification key</td>
</tr>
<tr>
<td>numOfRows</td>
<td>4</td>
<td>0</td>
<td>10</td>
<td>number of results per page Default: 10</td>
</tr>
<tr>
<td>pageNo</td>
<td>4</td>
<td>0</td>
<td>1</td>
<td>page number Default: 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4 below shows some of the output values.

GDP (Gross Domestic Product) is a representative indicator that shows the overall situation of a country’s economy at a glance, but in the case of GDP, it is difficult to apply to this study because the indicator GDP is calculated on a quarterly basis. Therefore, instead of GDP, the stock index is an indicator that can determine the trend of the stock market. Indices of the Korean stock market include the KOSPI (Korean version of the US Dow Jones) and the KOSDAQ (Korean version of the US NASDAQ). To collect these data, the pandas.datareader library is
used, and the closing price information by date of the stock market trend is preprocessed and used in this study. The collected data are shown in Table 5, and the information about the columns constituting the collected data are in the order of date, high price, low price, opening price, closing price, trading volume, and adjusted close.

**Table 5. KOSPI response elements**

<table>
<thead>
<tr>
<th>Date</th>
<th>High</th>
<th>Low</th>
<th>Open</th>
<th>Close</th>
<th>Volume</th>
<th>Adj Close</th>
</tr>
</thead>
<tbody>
<tr>
<td>2010-01-04</td>
<td>1696.14</td>
<td>1681.71</td>
<td>1681.71</td>
<td>1696.14</td>
<td>296500</td>
<td>1696.14</td>
</tr>
<tr>
<td>2010-01-05</td>
<td>1702.39</td>
<td>1686.45</td>
<td>1701.62</td>
<td>1690.62</td>
<td>408900</td>
<td>1690.62</td>
</tr>
<tr>
<td>2010-01-06</td>
<td>1706.89</td>
<td>1696.1</td>
<td>1697.88</td>
<td>1705.32</td>
<td>426000</td>
<td>1705.32</td>
</tr>
<tr>
<td>2010-01-07</td>
<td>1707.9</td>
<td>1683.45</td>
<td>1702.92</td>
<td>1683.45</td>
<td>462400</td>
<td>1683.45</td>
</tr>
<tr>
<td>2010-01-08</td>
<td>1695.26</td>
<td>1668.84</td>
<td>1694.06</td>
<td>1695.26</td>
<td>380000</td>
<td>1695.26</td>
</tr>
<tr>
<td>2020-12-23</td>
<td>2769.08</td>
<td>2716.28</td>
<td>2737.74</td>
<td>2759.82</td>
<td>1121300</td>
<td>2759.82</td>
</tr>
<tr>
<td>2020-12-24</td>
<td>2812.16</td>
<td>2762.6</td>
<td>2762.6</td>
<td>2806.86</td>
<td>1030900</td>
<td>2806.86</td>
</tr>
<tr>
<td>2020-12-28</td>
<td>2834.59</td>
<td>2799.56</td>
<td>2820.95</td>
<td>2808.6</td>
<td>1006200</td>
<td>2808.6</td>
</tr>
<tr>
<td>2020-12-29</td>
<td>2823.44</td>
<td>2792.06</td>
<td>2810.55</td>
<td>2820.51</td>
<td>1046800</td>
<td>2820.51</td>
</tr>
<tr>
<td>2020-12-30</td>
<td>2887.21</td>
<td>2809.35</td>
<td>2820.36</td>
<td>2873.47</td>
<td>1074000</td>
<td>2873.47</td>
</tr>
</tbody>
</table>

4.3. **Data collection and virtualization results.** The collected data is in XML format and preprocessing of the data is required to utilize it. Since the Open-API output result is in XML format, data virtualization is realized by using the Pandas Python Library to standardize it in a table format. Afterwards, we will remove empty and unnecessary data in preprocessing phase.

5. **Data preprocessing**

5.1. **Fruit price data preprocessing.** Fruit price data of KAMIS exist by region (Seoul, Busan, Daegu, Daejeon, Gwangju). The data to be targeted here would be fruit price data in Seoul, and fruit price data in the rest of regions are used as an independent variable (feature) to train. From the collected regional fruit sales data, high quality fruit products are selected
and the learning data for this fruit is extracted. For example, after selecting an apple (Fuji) in Seoul, the transaction date and transaction price are extracted at Pandas data table. After that, apple (Tsugaru), apple (Hongro), pear (Wonhang), pear (Shingo), banana, and orange data are extracted by same way for other fruits in Seoul. After completing the data extraction for the Seoul area, the price and transaction date are extracted in the same way as the method of extracting the price in the Seoul area while changing the region.

Because the sales period differs for each fruit, there is no full time-series fruit data on an annual basis. For time series data learning, we preprocess the data by adding information on the week and day of the week to the fruit data in this study. First, we construct data using the transaction price of the day before the holiday on days when no sales were made, such as holidays, and we put 0 won for a period with no transactions for more than 5 days. A year is counted as 52 weeks, and values up to 52 are assigned to the week, and values from 0 to 4 are assigned from Monday to Friday when there is a transaction to preprocess the time series data. Table 6 below shows some of the preprocessed apple (Fuji) data.

### Table 6. Part of apple (Fuji) data

<table>
<thead>
<tr>
<th>date</th>
<th>day of week</th>
<th>week of year</th>
<th>Pusan</th>
<th>Daejeon</th>
<th>Daegu</th>
<th>Gwangju</th>
<th>Seoul</th>
</tr>
</thead>
<tbody>
<tr>
<td>2011-01-03</td>
<td>0</td>
<td>1</td>
<td>6000</td>
<td>6000</td>
<td>6000</td>
<td>6000</td>
<td>6000</td>
</tr>
<tr>
<td>2011-01-04</td>
<td>1</td>
<td>1</td>
<td>6000</td>
<td>6333</td>
<td>6000</td>
<td>6000</td>
<td>6000</td>
</tr>
<tr>
<td>2011-01-05</td>
<td>2</td>
<td>1</td>
<td>6000</td>
<td>6333</td>
<td>6000</td>
<td>6000</td>
<td>6000</td>
</tr>
<tr>
<td>2011-01-06</td>
<td>3</td>
<td>1</td>
<td>6000</td>
<td>6333</td>
<td>6000</td>
<td>6000</td>
<td>6000</td>
</tr>
<tr>
<td>2011-01-07</td>
<td>4</td>
<td>1</td>
<td>6000</td>
<td>6333</td>
<td>6000</td>
<td>6000</td>
<td>6000</td>
</tr>
<tr>
<td>2011-01-10</td>
<td>0</td>
<td>2</td>
<td>6000</td>
<td>6333</td>
<td>6000</td>
<td>6000</td>
<td>6000</td>
</tr>
<tr>
<td>2011-01-11</td>
<td>1</td>
<td>2</td>
<td>6000</td>
<td>6333</td>
<td>6000</td>
<td>6000</td>
<td>6000</td>
</tr>
<tr>
<td>2011-01-12</td>
<td>2</td>
<td>2</td>
<td>6000</td>
<td>6333</td>
<td>6000</td>
<td>6000</td>
<td>6000</td>
</tr>
<tr>
<td>2011-01-13</td>
<td>3</td>
<td>2</td>
<td>6000</td>
<td>6333</td>
<td>6000</td>
<td>6000</td>
<td>6000</td>
</tr>
<tr>
<td>2011-01-14</td>
<td>4</td>
<td>2</td>
<td>6000</td>
<td>6333</td>
<td>6000</td>
<td>6000</td>
<td>6000</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

5.2. **Weather data preprocessing.** We could extract 15 columns to be used for deep learning from the weather data table standardized through Open-API. The data consist of weather data from January 2011 to December 2020. The 15 extracted columns are average temperature, minimum temperature, maximum temperature, sum of rainfall, maximum instantaneous wind speed, maximum wind speed, average wind speed, average temperature of dew point, minimum relative humidity, average relative humidity, average pressure of vapor, sunshine duration, sum of sunshine hour, sum of solar radiation and average temperature on surface. Information and meaning for each column is presented in table 7.

5.3. **Multiple linear regression analysis.** The performance of the model is not guaranteed by using all 15 columns from weather data. Some features may not be very helpful for training.
With this possibility in mind, we extract features that are highly relevant to the target values, in other words, fruit prices in Seoul. Results for these extractions will be analyzed in section 7.

We perform feature extraction through multiple linear regression analysis. Multiple linear regression analysis is an analysis method that verifies the effect of two or more continuous independent variables on continuous dependent variables. The concept and analysis method are the same as simple regression analysis, only the number of independent variables is different. For multiple linear regression analysis, preprocessed data of apples, pears, persimmons, bananas, and oranges are prepared. Thereafter, multiple linear regression analysis is performed between each fruit prices and the weather data. We have 5 weather features that are commonly significant, that is, Significance F is less than 0.05.

<table>
<thead>
<tr>
<th>Item name</th>
<th>Item Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>average temperature</td>
<td>daily average temperature</td>
</tr>
<tr>
<td>minimum temperature</td>
<td>daily minimum temperature</td>
</tr>
<tr>
<td>maximum temperature</td>
<td>daily maximum temperature</td>
</tr>
<tr>
<td>sum of rainfall</td>
<td>daily precipitation</td>
</tr>
<tr>
<td>maximum instantaneous wind speed</td>
<td>daily maximum instantaneous wind speed</td>
</tr>
<tr>
<td>maximum wind speed</td>
<td>daily maximum wind speed</td>
</tr>
<tr>
<td>average wind speed</td>
<td>daily average wind speed</td>
</tr>
<tr>
<td>average temperature of dew point</td>
<td>average temperature when water vapor condenses</td>
</tr>
<tr>
<td>minimum relative humidity</td>
<td>daily minimum relative humidity</td>
</tr>
<tr>
<td>average relative humidity</td>
<td>daily average relative humidity</td>
</tr>
<tr>
<td>average pressure of vapor</td>
<td>the pressure due to vaporization of a solid or liquid</td>
</tr>
<tr>
<td>sunshine duration</td>
<td>the length of time between when the sun’s center rises to the horizon and sets again on the horizon</td>
</tr>
<tr>
<td>sum of sunshine hour</td>
<td>the amount of time direct sunlight hits the Earth’s surface</td>
</tr>
<tr>
<td>sum of solar radiation</td>
<td>the radiant energy of the sun reaching the Earth’s surface</td>
</tr>
<tr>
<td>average temperature on surface</td>
<td>the temperature of the air near the surface of the earth</td>
</tr>
</tbody>
</table>

Table 8. Multiple linear regression analysis of weather data for apple (Fuji)

<table>
<thead>
<tr>
<th>Degree of Freedom</th>
<th>Sum of Squares</th>
<th>Mean of Squares</th>
<th>F</th>
<th>Significance F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regression</td>
<td>15</td>
<td>527591627</td>
<td>35172775.13</td>
<td>33.66018</td>
</tr>
<tr>
<td>Residual</td>
<td>2081</td>
<td>2174514214</td>
<td>1044937.152</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>2096</td>
<td>2702105841</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The apple data used to multiple linear regression analysis are daily data of 2,096 cases for a total of 10 years from year 2011 to year 2020. Table 8 is the first part of the contents of multiple linear regression analysis based on 95% reliability of apple data. Significance F is an indicator that verifies whether it is statistically valid. As the value of F is large and the value of Significance F is less than 0.05, a more significant value can be obtained. In the case of the above apple, Significance F is less than 0.05, so it can be said to be significant.
Table 9. Multiple linear regression analysis of apple data

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Item name</th>
<th>Coefficients</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y-intercept</td>
<td>Y-intercept</td>
<td>2474.547</td>
<td>0.009537</td>
</tr>
<tr>
<td>avgTa</td>
<td>average temperature</td>
<td>7.736941</td>
<td>0.88927</td>
</tr>
<tr>
<td>minTa</td>
<td>minimum temperature</td>
<td>-19.4318</td>
<td>0.439966</td>
</tr>
<tr>
<td>maxTa</td>
<td>maximum temperature</td>
<td>-19.0545</td>
<td>0.481196</td>
</tr>
<tr>
<td>sumRn</td>
<td>sum of rainfall</td>
<td>4.43451</td>
<td>0.017636</td>
</tr>
<tr>
<td>maxInsWs</td>
<td>maximum instantaneous wind speed</td>
<td>-215.362</td>
<td>1.52E-23</td>
</tr>
<tr>
<td>maxWs</td>
<td>maximum wind speed</td>
<td>359.9087</td>
<td>4.4E-16</td>
</tr>
<tr>
<td>avgWs</td>
<td>average wind speed</td>
<td>215.7026</td>
<td>5.55E-05</td>
</tr>
<tr>
<td>avgTd</td>
<td>average temperature of dew point</td>
<td>16.24501</td>
<td>0.68126</td>
</tr>
<tr>
<td>minRhm</td>
<td>minimum relative humidity</td>
<td>0.85888</td>
<td>0.827902</td>
</tr>
<tr>
<td>avgRhm</td>
<td>average relative humidity</td>
<td>-9.55187</td>
<td>0.365955</td>
</tr>
<tr>
<td>avgPv</td>
<td>average pressure of vapor</td>
<td>57.14733</td>
<td>5.74E-08</td>
</tr>
<tr>
<td>ssDur</td>
<td>sunshine duration</td>
<td>197.0722</td>
<td>5.34E-09</td>
</tr>
<tr>
<td>sumSsHr</td>
<td>sum of sunshine hour</td>
<td>-86.9069</td>
<td>2.46E-09</td>
</tr>
<tr>
<td>sumGsr</td>
<td>sum of solar radiation</td>
<td>59.33607</td>
<td>7.58E-09</td>
</tr>
<tr>
<td>avgTs</td>
<td>average temperature on surface</td>
<td>-42.0829</td>
<td>0.001735</td>
</tr>
</tbody>
</table>

Table 9 is the second part of the multiple linear regression analysis of apple data. The Y values are as follows.

\[
Y \text{ value} = 2474.55 + 7.74\text{avg Ta} - 19.43\text{min Ta} - 19.05\text{max Ta} \\
+ 4.43\text{sum Rn} - 215.36\text{max Ins Ws} + 359.91\text{max Ws} + 215.70\text{avg Ws} \\
+ 16.25\text{avg Td} + 0.86\text{min Rhm} - 9.55\text{avg Rhm} + 57.15\text{avg Pv} \\
+ 197.07\text{ss Dur} - 86.91\text{sum SsHr} + 59.34\text{sum Gsr} - 42.08\text{avg Ts}.
\]

The p-value is an index that verifies whether it is statistically valid, and a value of 0.05 or less based on a 95% confidence level can be considered as a significant value. Through multiple linear regression analysis of the remaining types of apples, pears, persimmons, bananas, and oranges, five factors including maximum instantaneous wind speed, average vapor pressure, heating time, total solar time, and total solar radiation, which are weather features commonly related to all fruit prices are selected.

5.4. **Comprehensive stock index data preprocessing.** We collect the comprehensive stock index data from year 2011 to year 2020. The data consist of year, KOSPI index, and KOSDAQ index columns, and it is composed of daily data. Since KOSPI data are judged to be suitable for domestic market analysis than KOSDAQ, so we use KOSPI data as the Korean stock market index. In order to analyze the fruit price, the KOSPI data are preprocessed to be the same day as the day on which the fruit was traded.
5.5. Data merge preprocessing. We combine all the preprocessed data to create a database. Based on the fruit data, it goes through the process of merging the weather data and the KOSPI index data on the day that the fruit was traded. Then we can sort all the data by date and each column means a feature. This database contains 45 columns of price in 5 cities of 9 fruits, 2 columns of numeric data for dates and weeks, 15 columns of weather data, and one column of KOSPI data. We extracted five features from 15 weather data using multiple linear regression analysis in Section 5.3. We will use this database as training data and test data for learning various models.

6. Build and train predictive models

LSTM is mainly used in deep learning in three modes: one-to-many, many-to-one, and many-to-many. One-to-many sequence problems are sequence problems in which a single input value from \( t \)-step gives a vector of multiple time-steps. This is generally used for image captioning [18]. Many-to-one sequence problems take a vector of multiple time-steps as input and return a single output. Here, this output value is used for prediction value. We typically use many-to-one LSTM network for sentiment analysis or text classification, as well as for stock price prediction [19]. Many-to-many sequence problems take a vector of multiple time steps as input and returns a vector of multiple time steps as output. Here, the input time-steps and the output time-steps may be the same or different depending on the problem. It can be used for machine translation and video classification [20]. In this study, we train our algorithm using a many-to-one LSTM.

We construct six types of models to predict fruit prices. First of all, six models commonly include fruit price data in Seoul and numeric data for dates and weeks defined in Section 5.1. The fruit prices data in Seoul, which is included in common in all models, are used not only for a training feature, but also as a prediction target. For example, suppose we have an input sequence of length \( s \) whose time step starts at \( t \)-step and ends at \((t + s - 1)\)-step. If we take this sequence that includes fruit price data in Seoul as input data, we get one prediction value as output. The weights are updated by comparing the output of this predicted value with the \((i + s)\)th fruit price data in Seoul, which is the actual value. That is, the fruit price in Seoul on the next day of the sequence is used as a prediction target.

The six models are constructed using three common data columns and the merged database from Section 5.5. We use 15 weather data obtained through Section 5.2 for Model 1, and use the data of 5 weather features obtained through multiple linear regression analysis of Section 5.3 for Model 2. Model 3 is a model in which KOSPI data is added to the training data of Model 2 above. Model 4 contain regional fruit price data in addition to the training data of Model 2. Model 5 is constructed by adding the KOSPI index and regional price data to Model 2. The last model is a model using all weather data, KOSPI index, and fruit price data by region.

The number of features used for training is 18 for Model 1, 8 for Model 2, 9 for Model 3, 12 for Model 4, 13 for Model 5, and 23 for Model 6, and predicts the price of fruits using the LSTM algorithm. The model consists of relatively fewer variables than the 108 features used in the paper [11]. Table 10 below summarizes the contents of the manufactured model.
7. ANALYZING THE PERFORMANCE OF PREDICTIVE MODELS

7.1. How to measure model performance. We measure the error between the actual fruit price and the predicted fruit price obtained through the model to evaluate the performance of the model. The root mean square error (RMSE), which is widely used in deep learning field, is used as the error measurement method [11, 21]. The RMSE equation is as follows:

\[
RMSE = \sqrt{\frac{1}{T} \sum_{t=1}^{T} (Y_{t}^p - Y_{t}^a)^2},
\]

where \(Y_{t}^p\) is a predicted value, \(Y_{t}^a\) is an actual value, and \(T\) is the number of observations.

7.2. Training process of predictive models. In preprocessing step, data are normalized in the range (0,1) using min_max scaling and reshaped according to the training feature. Let \(S = \{x_1, \ldots, x_N, y\}\) be the database set of training features and a target where each \(x_i\) and \(y\) is a vector with \(m\) elements for \(i = 1, 2, \ldots, N\). Here, \(y\) is the target vector and the number of elements \(m\) is the length of time series data for daily. Then we can choose the elements of \(S\) to construct the feature matrix \(M_k\), but each \(M_k\) must contain the target vector \(y\). If \(M_k\) takes \(n\) elements in \(S\), then \(M_k\) becomes an \(m\)-by-\(n\) matrix.

Let \(R_k\) be the many-to-one LSTM network models for \(k = 1, \ldots, 6\). To train the models \(R_k\), we need to prepare the input data for training according to the length of the input sequence. If the length of the input sequence is \(s\), it is possible to extract input data \(M_{kj}\) having \(s\)-length rows from the feature matrix \(M_k\) for \(j = 1, \ldots, (m-s)\). Since we do not have \((m+1)\)-th row of feature matrix to update the weight, \((m-s+1)\)-th input data cannot be used for training. Now, we train the models \(R_k\) using all \(M_{kj}\).
The trained model $R_k$ predicts the target price for next day using the $s$-length sequential data. For example, if we put the $s$-length sequential data which do not have target value in trained model $R_k$, then $R_k$ returns a predicted scalar value. Let $y_p$ be the vector of actual values for a period we want to predict. To obtain $d$ predicted values for the period, we need to $d$ sequences. Using this sequences, we estimate the predicted values with the trained model $R_k$ and measure the RMSE between the predicted value and the target value $y_p$ for model accuracy.

The following is a pseudocode for training part of algorithm.

Let $S = \{x_1, x_2, ..., x_N, y\}$ and $M_k = \{x_1, x_2, ..., x_{m_k}\}$. Here, $M_k$ is the subset of $S$ and the elements/vectors of $M_k$ consist of training feature for $R_k$, for $k = 1, \ldots, 6$.

---

1: Set LSTM units, hyperparameters and optimizer to define LSTM Network
2: Normalize the dataset into values from 0 to 1 using min_max_scaling
3: Select feature set and organize dataset
4: for i <- 1 to #R_k do
5: for n_epochs and batch_size do
6: Train the models
7: end for
8: end for
9: for i <- 1 to #R_k do
10: Run Predictions
11: Calculate the loss function using root_mean_square_error
12: Compare prediction and real price
13: end for

---

We train the models while changing the values of the hyperparameters and measure the RMSE of the models to find the hyperparameters most suitable for features. 80% of the prepared data are used as training data, and 20% of those is used as test data of the training model. In this study, the six types of models classified by training features are divided into two versions according to the period of the training data. One version of models is trained with data from year 2011 to year 2018, and the other version of models is trained with data from year 2011 to year 2019. Models trained with data from year 2011 to year 2018 are used to predict fruit prices in year 2019, and models trained with data from year 2011 to year 2019 are used to predict fruit prices in year 2020, respectively. The training is conducted a total of 2916 times (sequence length(3) \times output size of hidden layer(3) \times LSTM layers(3) \times year(2) \times fruit variety(9) \times types of models(6)). Here sequence length varies 5, 10, 20, and output size of hidden layer is varies 1, 3, 5, and the numbers of LSTM layers are 2, 4, 6. Table 11 below is part of the table recording the training performance.

7.3. **Learning model analysis.** In this subsection, we find the optimal hyperparameter among the experimental results in Section 7.2. The hyperparameters to be adjusted are the sequence
### Table 11. Part of RMSE according to hyperparameter

<table>
<thead>
<tr>
<th>sequence length</th>
<th>output size of hidden dimension</th>
<th>the number of LSTM layers</th>
<th>year</th>
<th>fruit</th>
<th>Model 1</th>
<th>Model 2</th>
<th>Model 3</th>
<th>Model 4</th>
<th>Model 5</th>
<th>Model 6</th>
<th>average RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1</td>
<td>2</td>
<td>2020</td>
<td>apple(Fuji)</td>
<td>0.063</td>
<td>0.069</td>
<td>0.066</td>
<td>0.121</td>
<td>0.066</td>
<td>0.068</td>
<td>0.075</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>2</td>
<td>2020</td>
<td>apple(Tsugaru)</td>
<td>0.061</td>
<td>0.061</td>
<td>0.106</td>
<td>0.106</td>
<td>0.080</td>
<td>0.190</td>
<td>0.093</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>2</td>
<td>2020</td>
<td>apple(Hongjan)</td>
<td>0.071</td>
<td>0.068</td>
<td>0.113</td>
<td>0.088</td>
<td>0.089</td>
<td>0.077</td>
<td>0.077</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>2</td>
<td>2020</td>
<td>banana</td>
<td>0.018</td>
<td>0.018</td>
<td>0.018</td>
<td>0.018</td>
<td>0.018</td>
<td>0.018</td>
<td>0.022</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>2</td>
<td>2020</td>
<td>orange(Navel USA)</td>
<td>0.057</td>
<td>0.056</td>
<td>0.051</td>
<td>0.139</td>
<td>0.065</td>
<td>0.060</td>
<td>0.072</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>2</td>
<td>2020</td>
<td>orange(Valencia USA)</td>
<td>0.067</td>
<td>0.139</td>
<td>0.155</td>
<td>0.158</td>
<td>0.131</td>
<td>0.143</td>
<td>0.131</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>2</td>
<td>2020</td>
<td>pear(Shirgogi)</td>
<td>0.080</td>
<td>0.081</td>
<td>0.106</td>
<td>0.066</td>
<td>0.062</td>
<td>0.077</td>
<td>0.077</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>2</td>
<td>2020</td>
<td>pear(Weihuang)</td>
<td>0.011</td>
<td>0.039</td>
<td>0.047</td>
<td>0.138</td>
<td>0.073</td>
<td>0.097</td>
<td>0.086</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>2</td>
<td>2020</td>
<td>persimmon</td>
<td>0.014</td>
<td>0.013</td>
<td>0.154</td>
<td>0.074</td>
<td>0.076</td>
<td>0.100</td>
<td>0.100</td>
</tr>
</tbody>
</table>

length, output size of hidden layer, and the number of LSTM layers. For convenience of notation, we describe the hyperparameters as \([S : \cdot ; H : \cdot ; L : \cdot]\), where \(S\), \(H\), and \(L\) mean sequence length, output size of hidden layer, and the numbers of LSTM layers, respectively. We calculate the average RMSE of 54 experiments (fruit variety(9) \(\times\) type of models(6)) performed with the same hyperparameters in a total of 2916 experiments. The results are separately organized by year and shown in Fig. 5. The blue bar on the graph represents year 2019 and the orange bar represents year 2020. The table 12 encloses the detailed values of the average RMSE.

![Figure 5. Average RMSE graph by hyperparameters](image)

In Fig. 5 and Table 12, the hyperparameter with the smallest average RMSE in year 2019 is \([S : 5; H : 1; L : 2]\), whose value is 0.061820, and the smallest average RMSE in year
2020 is \([S : 20; H : 1; L : 2]\) and its value is 0.079671. However, the average value of the two years for the hyperparameter \([S : 10; H : 1; L : 2]\) is 0.071096, which is smaller than 0.071625 and 0.071287 for the average value of \([S : 5; H : 1; L : 2]\) and \([S : 20; H : 1; L : 2]\), respectively. Therefore, hyperparameter \([S : 10; H : 1; L : 2]\) is the optimal hyperparameter in our experiments over year 2019 and 2020.

### 7.4. RMSE by model

In this subsection, we evaluate the performance of models with hyperparameters \([S : 10; H : 1; L : 2]\). The performance of each model is measured using the average RMSE of 9 fruits. When the average RMSE is small, we evaluate that the performance of the model is excellent. The result is shown in the Fig. 6.

Among the experimental results, Model 1 has the best performance, that is to mean that the average RMSE of Model 1 is the smallest over two years. In addition, there are noticeable performance differences between the average RMSE in year 2019 and 2020 for models except Model 1 and Model 5. Due to this performance difference, in the experiment in Section 7.3 the average RMSE in year 2020 was measured to be greater than the average RMSE in year 2019 for all hyperparameters. Here, we focus on the Model 1 and Model 5, which have small performance differences between year 2019 and 2020. The features used to train Model 1 except for common features (Weekly, Daily data, and fruit price in Seoul) are 15 weather data, and the total number of features is 18. On the other hand, the features used to train Model 5 consist of 5 weather data obtained through multiple linear regression analysis, fruit price data by region, and KOSPI, and the total number of features is 13. In terms of performance, Model 1 is the best, but Model 5 is also efficient enough. We compare the actual value and the predicted value of fruit price through Model 5.

Figures 7 and 8 show the comparison of fruit prices in 2019 and the comparison of fruit prices in 2020, respectively. The blue line on the graph is the real price and the orange line is the predicted price. The point at which the fruit prices are zero means that there is no trade in

### Table 12. Measured average RMSE value by hyperparameters

<table>
<thead>
<tr>
<th>hyperparameters</th>
<th>2019</th>
<th>2020</th>
<th>hyperparameters</th>
<th>2019</th>
<th>2020</th>
</tr>
</thead>
<tbody>
<tr>
<td>([S : 5; H : 1; L : 2])</td>
<td>0.062</td>
<td>0.081</td>
<td>([S : 10; H : 3; L : 6])</td>
<td>0.086</td>
<td>0.105</td>
</tr>
<tr>
<td>([S : 5; H : 1; L : 2])</td>
<td>0.072</td>
<td>0.092</td>
<td>([S : 10; H : 5; L : 2])</td>
<td>0.072</td>
<td>0.097</td>
</tr>
<tr>
<td>([S : 5; H : 1; L : 6])</td>
<td>0.078</td>
<td>0.094</td>
<td>([S : 10; H : 5; L : 4])</td>
<td>0.085</td>
<td>0.101</td>
</tr>
<tr>
<td>([S : 5; H : 3; L : 2])</td>
<td>0.071</td>
<td>0.099</td>
<td>([S : 10; H : 5; L : 6])</td>
<td>0.087</td>
<td>0.100</td>
</tr>
<tr>
<td>([S : 5; H : 3; L : 4])</td>
<td>0.080</td>
<td>0.104</td>
<td>([S : 20; H : 1; L : 2])</td>
<td>0.063</td>
<td>0.080</td>
</tr>
<tr>
<td>([S : 5; H : 3; L : 6])</td>
<td>0.078</td>
<td>0.101</td>
<td>([S : 20; H : 1; L : 4])</td>
<td>0.076</td>
<td>0.096</td>
</tr>
<tr>
<td>([S : 5; H : 5; L : 2])</td>
<td>0.071</td>
<td>0.097</td>
<td>([S : 20; H : 1; L : 6])</td>
<td>0.081</td>
<td>0.109</td>
</tr>
<tr>
<td>([S : 5; H : 5; L : 4])</td>
<td>0.080</td>
<td>0.098</td>
<td>([S : 20; H : 3; L : 2])</td>
<td>0.071</td>
<td>0.092</td>
</tr>
<tr>
<td>([S : 5; H : 5; L : 6])</td>
<td>0.076</td>
<td>0.097</td>
<td>([S : 20; H : 3; L : 4])</td>
<td>0.082</td>
<td>0.100</td>
</tr>
<tr>
<td>([S : 10; H : 1; L : 2])</td>
<td>0.062</td>
<td>0.080</td>
<td>([S : 20; H : 3; L : 6])</td>
<td>0.089</td>
<td>0.112</td>
</tr>
<tr>
<td>([S : 10; H : 1; L : 4])</td>
<td>0.075</td>
<td>0.094</td>
<td>([S : 20; H : 5; L : 2])</td>
<td>0.075</td>
<td>0.094</td>
</tr>
<tr>
<td>([S : 10; H : 1; L : 6])</td>
<td>0.083</td>
<td>0.096</td>
<td>([S : 20; H : 5; L : 4])</td>
<td>0.088</td>
<td>0.104</td>
</tr>
<tr>
<td>([S : 10; H : 3; L : 2])</td>
<td>0.072</td>
<td>0.099</td>
<td>([S : 20; H : 5; L : 6])</td>
<td>0.092</td>
<td>0.112</td>
</tr>
<tr>
<td>([S : 10; H : 3; L : 4])</td>
<td>0.086</td>
<td>0.105</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
the market. The results show that prediction for orange (Valencia USA) in both year 2019 and 2020 does not perform well compared to other fruits. Especially for year 2020, the predictive model works well for trading day of orange (Valencia USA), but the problem is that it does not predict the duration of the trade at all.

7.5. **External factors of fruit price predictions.** In the subsection 7.4, we can see that our prediction of sales period with predictive model does not match the actual sales period of orange (Valencia USA). Therefore, this subsection will explain the reason. To do this, we look at orange (Valencia USA) sales data from year 2011 to year 2020. In the Fig. 9, the x-axis represents the sale date of the orange (Valencia USA) and the y-axis represents the price. The red line in the graph is data for 2020. In other years, orange (Valencia USA) traded from June to November on average, whereas in 2020 only traded for one month, from the third week of June to the second week of July.

We could find the reason for the low import volume and short import period of orange (Valencia USA) in 2020 in the report on Agricultural and livestock export and import trends by Korea Rural Economic Institute (KREI). According to this report, the import volume of oranges from January to September 2020 in Korea decreased compared to the previous year, due to sluggish consumption according to COVID-19 and an increase in demand for substitute fruits such as Korean mandarin [22, 23]. In particular, it is reported that the Valencia USA, one of the orange varieties selected for our model’s performance test, was traded only in one
month, from the third week of June to the second week of July. By the result, we can see that the price of oranges in 2020 was significantly affected by unexpected external factors such as COVID-19, so the flow of fruit prices was significantly different from previous 9 years.

To investigate the effect of the 2020 orange (Valencia USA) data on the model evaluation in practice, we measure the model’s performance by subtracting the 2020 orange (Valencia USA) for all hyperparameters used in Section 7.3. The orange bar in Fig. 10 represents the average RMSE in 2020, and the green bar represents the average RMSE in 2020 excluding the orange (Valencia USA). This graph shows that the RMSE for the green bar is smaller than the RMSE for the orange bar for all hyperparameters, which means that the orange price data for 2020 raises the average RMSE. Since the orange (Valencia USA) price data for 2020 is data under special circumstances, as mentioned above, we determine that the orange (Valencia USA) price for 2020 is not appropriate to evaluate our models.

7.6. RMSE analysis excluding orange (Valencia USA). In this subsection, we evaluate the predictive model using eight fruits except orange (Valencia USA). As in the evaluation method of Section 7.4, we measures the average RMSE of fruits, excluding the RMSE of oranges (Valencia USA) in both 2019 and 2020. The blue and orange bars in Fig. 11 represent the average RMSE for all fruits in 2019 and 2020, respectively, as shown in Fig. 6 in Section 7.4. Also, the gray and yellow bars in Fig. 11 represent the average RMSE for fruits excluding
oranges (Valencia USA) in 2019 and 2020, respectively. In the 2019 predictive model, the average of the difference between the blue bars and the gray bars is 0.0018. This means that the average RMSE is reduced by 0.0018 by excluding orange (Valencia USA). Also, in the 2020 predictive models, if orange (Valencia USA) is excluded from orange bars, the average RMSE decreases by 0.0065, which is about 3.58 times the result of 2019. This shows that, as analyzed in Section 7.5, the orange (Valencia USA) price in 2020 was formed under special circumstances and it is more difficult to predict compared to the orange (Valencia USA) price in other years.

Now, we consider the models with excellent performance among the six models. The RMSE of 2020 (yellow bar in Fig. 11) is used for the evaluation because it used the most data for training, and we assumes that the smaller the RMSE, the better the performance. Models 1, 2, 5, and 6 have the average RMSE under 0.070 in 2020, yellow bars in Fig. 11.

First, Model 6 uses the most features. However, since the performance of Model 1 using sub-features of the features used in Model 6 is better than that of Model 6, Model 6 is excluded from the good predictive model. The average RMSEs of predictive Model 1, 2, and 5 are 0.061, 0.059, and 0.060, respectively. Also, the number of features used for training are 18, 8, and 13, respectively. The features used for training Model 2 are obtained through multiple linear regression analysis of the features used in Model 1 with the target. Through this performance
comparison of Model 1 and Model 2, it is revealed that multiple linear regression analysis can exclude features that do not significantly improve performance. Therefore, in terms of average RMSE, Model 2 shows the best performance with the fewest number of features. However, the difference in the average RMSE of the three models is not noticeably large. Therefore, we want to check the distribution of RMSE by model.

Figure 12 is box plots showing the RMSE distribution of the models. Box plots are used for the purpose of quickly checking the range and median of a data set using a picture when it is difficult to visually check a lot of data. Here, the boxes mean the distribution of the models according to the fruits, and box plots show that the smaller the vertical size of the box, the more uniform performance of the model. Models with uniform performance are analyzed to be stable. The vertical sizes of the boxes of Model 1, 2, and 5 are 0.010, 0.013, and 0.008, respectively. Therefore, we evaluate that Model 5 shows the most uniform performance regardless of the fruits. Also, Model 2 has the largest performance deviation among the three models.

In conclusion, Model 1 has the largest average RMSE among the three models, and the performance deviation is intermediate between Model 2 and Model 5. In addition, the number of features used for training Model 1 is the largest. Model 2 use the smallest number of features. It has the smallest average RMSE, and has a large performance deviation. At last, Model 5
shows the most uniform performance although the average RMSE is greater than Model 2. Therefore, we evaluate Model 5 as the best model for application to various fruits.

8. CONCLUSION AND DISCUSSION

This paper provides a fruit price prediction algorithm using deep learning. LSTM network is employed as deep learning methods to create predictive models of time series data. As for the training features, KOSPI data and regional fruit price data are added to the weather data that have been used in other papers [10, 11]. Here, we set the weather data as a feature that affects the supply of fruit, and the KOSPI and regional fruit price data are set as the feature that affects the demand for fruit. We find the hyperparameters of the LSTM network suitable for fruit price prediction and evaluate six models composed of the prepared training features. The characteristic of our models is that the number of features is small (the model with the most features uses 23 features, and the model with the fewest features uses 8 features). During evaluation, we find an unsuitable fruit to evaluate our model, and analyze this fruit from the data. As a result, we find out that the price of the fruit was greatly affected by the influence of
unexpected external factors such as COVID-19. So, we measure the performance of the models except for that fruit in order to evaluate the models except under special circumstances.

As comparing the average RMSE of Model 1 and Model 2, it is revealed that feature selection through multiple linear regression analysis in the preprocessing step do not increase the average RMSE of the model. Therefore, it is more advantageous to use Model 2 which has fewer features than to use Model 1 when we predict the fruit price with LSTM. Among the remaining models, the models with the smallest average RMSE are Model 2 and Model 5, which means that Model 2 and Model 5 have the best performance. We adds features related to the demand for fruit to the features of Model 2 in Model 5. Since Model 5 and Model 2 have similar average RMSE, then, in terms of performance, this result seems that there is no need to include the feature of demand for fruit in model. However, as a result of comparing the performance distribution of the models, Model 5 has more uniform performance than Model 2. These results show that adding features related to fruit demand increases the stability of the model. Therefore, when we use our predictive model to predict the prices of other fruits that we have not tested, we judge that using Model 5 will give more stable results.

In conclusion, this paper demonstrates that training the LSTM network using the features of supply and demand of fruits is superior in performance and stability. In addition, we show that feature selection through multiple linear regression analysis reduces the number of features used in the model while maintaining the model’s performance. Also we show that weather

![Figure 11. Average RMSE by model excluding orange (Valencia USA)](image)
data, regional fruit price data, and KOSPI data used in Model 5 are the features optimized for the fruit price predictive model.

For future work, we will use many-to-many LSTM algorithm to get weekly or long-term predictions. Also, we can replace the LSTM algorithm with a bidirectional LSTM [19] or a transformer attention mechanism to get better performance.

ACKNOWLEDGMENTS

Tae-Su Park and Young Rock Kim were supported by Hankuk University of Foreign Studies Research Fund of 2021. JongHae Keum and Hoisub Kim were supported by KIAS Individual Grant (MG008512) at Korea Institute for Advanced Study. Young Rock Kim and Youngho Min were supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2021R1A2C1011467).

REFERENCES


[8] Rickard, B. J., & Lei, L., How important are tariffs and nontariff barriers in international markets for fresh fruit?, Agricultural Economics, 42 (2011), 19–32.


[22] FTA Implementation Support Center, Agricultural and livestock export and import trend Vol. 8, No. 3, Korea Rural Economic Institute, (2020).

[23] FTA Implementation Support Center, Agricultural and livestock export and import trend Vol. 8, No. 4, Korea Rural Economic Institute, (2020).
RELIABILITY OF NUMERICAL SOLUTIONS OF THE G-EULER PROCESS

DONG WON YU

DEPARTMENT OF MATHEMATICS, CHUNG-ANG UNIVERSITY, SEOUL 06974, REPUBLIC OF KOREA
Email address: dwyu@cau.ac.kr

ABSTRACT. The G-Euler process has been proposed to overcome the difficulties of the calculation of the exponential function of the Jacobian. It is an explicit method that uses the exponential function of the scalar skew-symmetric matrix. We define the moving shapes of true solutions and the moving shapes of numerical solutions. It is discussed whether the moving shape of the numerical solution matches the moving shape of the true solution. The match rates of these two kinds of moving shapes are sequentially calculated by the G-Euler process without using the true solution. It is shown that the closer the minimum match rate is to 100%, the more closely the numerical solutions follow the true solutions to the end. The minimum match rate indicates the reliability of the numerical solution calculated by the G-Euler process. The graphs of the Lorenz system in Perko [1] are different from those drawn by the G-Euler process. By the way, there is no basis for claiming that the Perko’s graphs are reliable.

1. INTRODUCTION

Generalized Runge-Kutta processes were proposed by Lawson [2] in 1967. They have been rediscovered many times and are commonly known as an integrating factor methods. These numerical methods use the exponential function of the Jacobian or an approximation to it [3].

The G-Euler process has been proposed in [4, 5] to overcome the difficulty of the calculation of the exponential function of the Jacobian. It is summarized as follows:

We consider the well-posed nonlinear autonomous system

\[ x'(t) = f(x(t)) = Jx(t) + g(x(t)), \quad x(0) = x_0 \in \mathbb{R}^m, \quad t \in [0, T], \]

(1.1)

where \( f(0) = 0 \) and \( J = \frac{\partial}{\partial x} f(0) \). Let \( P \) be the eigenvector matrix of \( J \). Then, the matrix \( J \) is transformed into the canonical form [7] as follows:

\[ P^{-1}JP = \hat{S} + \hat{N}, \]

where \( \hat{S} \) is semisimple, \( \hat{N} \) is nilpotent and \( \hat{S}\hat{N} = \hat{N}\hat{S} \).
The scalar skew-symmetric matrix $S$ is extracted from $J$:

$$S \equiv \alpha I + \frac{1}{2} (\hat{S} - \hat{S}^T),$$

where $\alpha$ denotes the maximum of real parts of eigenvalues of $J$.

Let us apply the linear transformation $x(t) = Py(t)$ to (1.1):

$$y'(t) = Sy(t) + u(y(t)), \quad y(0) = y_0 = P^{-1}x_0, \quad t \in [0, T],$$

(1.2)

where

$$u(y) = P^{-1}g(Py) + (\hat{S} + \hat{N} - S)y.$$

The Eq. (1.2) is called the $S$-transformed system.

We apply the Lawson’s transformation $y(t) = e^{tS}z(t)$ to (1.2). Then, we have

$$z'(t) = e^{-tS}u(e^{tS}z(t)), \quad z(0) = y(0) = P^{-1}x_0, \quad t \in [0, T].$$

(1.3)

Apply the Euler method to (1.3) and the reverse transformations $z_n = e^{-tnS}y_n$ to its approximate solution. Then, the $G$-Euler process

$$y_{n+1} = e^{hS} \left\{ y_n + h u(y_n) \right\}, \quad x_{n+1} = Py_{n+1}$$

(1.4)

is obtained [4, 5].

Since the exponential function $e^{hS}$ of the scalar skew-symmetric matrix $S$ can be exactly computed by intrinsic functions such as $\exp$, $\cos$ and $\sin$ [5], the $G$-Euler process is an explicit method. If the high order Runge-Kutta methods are applied to (1.3), then the high order $G$-Runge-Kutta processes are obtained [4, 5], and they are superior to the $G$-Euler process.

The $G$-Euler process is a convergent explicit scheme and in Yu [4] it has been compared with RKSUITE developed by Brankin, Gladwel and Shampine through several nonlinear systems. Yu [5] could’t show the merits of the $G$-Euler process. In this paper, we will show that the numerical solutions computed by the $G$-Euler process follow the true solutions to the end, and its graphs of Lorenz system will be compared with the graphs in Perko [1].

In Section 2, the moving shape of the true solution and the moving shape of the numerical solution are defined. And these two kinds of moving shapes are classified into three types, respectively. It is discussed whether the moving shape of the numerical solution matches the moving shape of the true solution.

We arrive the conclusion that the match or the mismatch is determined by the $G$-Euler process without using the true solutions.

In Section 3, the match rates are sequentially calculated by the $G$-Euler process. The true solution of Example 1 is known. It is shown that if the minimum match rate is near 100%, then the numerical solutions follow the true solutions to the end. This is also confirmed at Case 2-1 in Example 2 which is Lorenz system.

In Case 2-2 of Example 2, the projections obtained by the classical Runge-Kutta forth-order method are different from the projections obtained by the $G$-Euler process. These Runge-Kutta’s projections are similar to the projections in Perko [1]. However, there is no basis for claiming that the Perko’s projections are reliable. On the other hand, if the minimum match
rate is near 100\%, then the graphs obtained by the G-Euler process are reliable more than the graphs in Perko [1].

2. THE MOVING SHAPE OF TRUE SOLUTIONS AND THE MOVING SHAPE OF NUMERICAL SOLUTIONS

Let us consider the problem (1.1) in the X-space and the problem (1.2) in the Y-space. In the Y-space, we use the Euclidean inner product $\langle \cdot, \cdot \rangle$, the corresponding vector norm $|\cdot|$ and the related matrix norm $|\cdot|_P$.

In the X-space, we use the transformed inner product $\langle \cdot, \cdot \rangle_P$ [6] defined by

$$<\zeta_1, \zeta_2>_P = <P^{-1}\zeta_1, P^{-1}\zeta_2> \cdot$$

Then, the corresponding vector norm $|\cdot|_P$ and the related matrix norm $|\cdot|_P$ are respectively given by

$$|\zeta|_P = |P^{-1}\zeta| \quad \text{and} \quad |A|_P = |P^{-1}AP|.$$ 

2.1. Definitions. Let us consider only two adjacent true solutions $x(t)$ and $\bar{x}(t)$ of (1.1). Their initial values are $x_0$ and $\bar{x}_0$, respectively. Since their corresponding true solutions $y(t)$ and $\bar{y}(t)$ of the S-transformed system (1.2) are related by $y(t) = P^{-1}x(t)$ and $\bar{y}(t) = P^{-1}\bar{x}(t)$, it follows that

$$y_0 = P^{-1}x_0, \quad \bar{y}_0 = P^{-1}\bar{x}_0 \quad \text{and} \quad |x(t) - \bar{x}(t)|_P = |y(t) - \bar{y}(t)|.$$

Let’s assume that we know the true solutions $y(t)$ and $\bar{y}(t)$ first. And we define the functions $\nu : [0, T] \to \mathbb{R}$ and $L : [0, T] \to \mathbb{R}$ differently from in Yu [5] as follows:

$$\nu(t) \equiv \frac{\langle y(t) - \bar{y}(t), y(t) - \bar{y}(t) \rangle}{|y(t) - \bar{y}(t)|^2}, \quad (2.1)$$

$$L(t) \equiv \frac{|u(y(t)) - u(\bar{y}(t))|}{|y(t) - \bar{y}(t)|}. \quad (2.2)$$

Since the problem (1.1) is well-posed, the true solutions $y(t)$, $\bar{y}(t)$ and the functions $\nu(t)$, $L(t)$ are continuous.

Let us introduce the function

$$\phi(t) \equiv |y(t) - \bar{y}(t)|^2.$$ 

The derivative of $\phi$ is given by

$$\phi'(t) = 2 \langle y'(t) - \bar{y}'(t), y(t) - \bar{y}(t) \rangle$$

$$= 2 \langle S\{y(t) + u(y(t))\} - \{S\bar{y}(t) + u(\bar{y}(t))\}, y(t) - \bar{y}(t) \rangle$$

$$= 2\{<S(y(t) - \bar{y}(t)), y(t) - \bar{y}(t)> + <u(y(t)) - u(\bar{y}(t)), y(t) - \bar{y}(t)>\}$$

and it follows that $\phi$ satisfies the differential equation

$$\phi'(t) = 2(\alpha + \nu(t))\phi(t), \quad t \in [0, T], \quad \phi(0) = |y_0 - \bar{y}_0|^2.$$
Multiplication of both sides with
\[ \eta(t) = e^{-\int_0^t (\alpha + \nu(\tau)) \, d\tau} \]
leads to the equality
\[ \frac{d}{dt}(\phi(t)\eta(t)) = 0. \]
This implies that
\[
\left| \frac{y(t_{n+1}) - y(t_n)}{y(t_n) - y(t_m)} \right| = e^{\int_{t_n}^{t_{n+1}} (\alpha + \nu(\tau)) \, d\tau} \equiv \kappa(t_n). \tag{2.3}
\]

Let’s consider a fixed step \( t_n \). Since \( \nu(t) \) is a continuous function, there exists a maximum real number \( h_{0n} \), so that the followings hold for all \( t \in (t_n, t_n + h_{0n}) \).
\[
\begin{align*}
\alpha + \nu(t_n) < 0 & \implies \alpha + \nu(t) < 0, \tag{2.4} \\
\alpha + \nu(t_n) > 0 & \implies \alpha + \nu(t) > 0, \tag{2.5} \\
\alpha + \nu(t_n) = 0 & \implies \begin{cases} 
\alpha + \nu(t) < 0, \\
\alpha + \nu(t) = 0, \\
\alpha + \nu(t) > 0.
\end{cases} \tag{2.6}
\end{align*}
\]
Here, \( h_{0n} \) is determined according to the step \( t_n \) and the graph of \( \alpha + \nu(t) \). From now on, \( h_{0n} \) will be written as \( h_{0n} \). The number \( h_{0n} \) is the maximum step size at the step \( t_n \).

We determine the step \( t_{n+1} \) as follows:
\[ t_{n+1} \equiv t_n + h \quad \text{and} \quad h < h_{0n}. \tag{2.7} \]
Then, the followings are derived from (2.3), (2.4), (2.5) and (2.6)
\[
\begin{align*}
\alpha + \nu(t_n) < 0 & \implies \kappa(t_n) - 1 < 0, \tag{2.8} \\
\alpha + \nu(t_n) > 0 & \implies \kappa(t_n) - 1 > 0, \tag{2.9} \\
\alpha + \nu(t_n) = 0 & \implies \kappa(t_n) - 1 \leq 0. \tag{2.10}
\end{align*}
\]
From (2.3), (2.8), (2.9) and (2.10), the moving shape of true solutions \( y(t_n) \) and \( \overline{y}(t_n) \) is classified by the sign of \( \alpha + \nu(t_n) \) (see p.17 in [6]).

**Definition 2.1.** If
\[ h < h_{0n} \quad \text{and} \quad \alpha + \nu(t_n) < 0 \quad \{ > 0, = 0, \text{ resp.} \}, \]
then two true solutions \( y(t_n) \) and \( \overline{y}(t_n) \) behave contractive { expansive, parallel, resp. } along increasing \( t \in (t_n, t_{n+1}) \). In this case, we will simply say that the moving shape of the true solution \( y(t) \) is contractive (expansive, parallel, resp.) at the step \( t_n \).
In the G-Euler process, the following is derived from (1.4), (2.1), (2.2) and $|e^{hS}| = e^{\alpha h}$. Where, the step size $h$ is determined by (2.7).

$$|y_{n+1} - \bar{y}_{n+1}|^2 = |e^{hS}\left\{y_n + h\mathbf{u}(y_n)\right\} - e^{hS}\left\{\bar{y}_n + h\mathbf{u}(\bar{y}_n)\right\}|^2$$

$$= e^{2\alpha h}|\left\{y_n - \bar{y}_n\right\} + h\left\{\mathbf{u}(y_n) - \mathbf{u}(\bar{y}_n)\right\}|^2$$

$$= e^{2\alpha h}\{1 + 2\nu_n(h)h + L_n(h)^2h^2\}|y_n - \bar{y}_n|^2,$$

where $\nu_n(h)$ and $L_n(h)$ are approximations of $\nu(t_n)$ and $L(t_n)$, and given by

$$\nu_n(h) \equiv \frac{\mathbf{u}(y_n) - \mathbf{u}(\bar{y}_n)}{|y_n - \bar{y}_n|},$$  

(2.11)

$$L_n(h) \equiv \frac{|\mathbf{u}(y_n) - \mathbf{u}(\bar{y}_n)|}{|y_n - \bar{y}_n|}.  

(2.12)$$

Hence, we have

$$\frac{|y_{n+1} - \bar{y}_{n+1}|}{|y_n - \bar{y}_n|} = e^{\alpha h}\sqrt{1 + 2\nu_n(h)h + L_n(h)^2h^2} \equiv \kappa_n(h).  

(2.13)$$

From (2.13), the moving shape of numerical solutions $y_n$ and $\bar{y}_n$ is classified by the sign of $\kappa_n(h) - 1$.

**Definition 2.2.** If

$$h < h_{0n} \quad \text{and} \quad \kappa_n(h) - 1 < 0 \quad \{ > 0, = 0, \text{ resp.} \},$$

then two numerical solutions $y_n$ and $\bar{y}_n$ behave contractive \{ expansive, parallel, resp. \} along increasing $t \in (t_n, t_{n+1})$. In this case, we will simply say that the moving shape of the numerical solution $y_n$ is contractive (expansive, parallel, resp.) at the step $t_n$.

From Definition 2.1 and Definition 2.2, we have

**Definition 2.3.** If for $h < h_{0n}$,

$$(\alpha + \nu(t_n))(\kappa_n(h) - 1) > 0 \quad \text{(2.14)}$$

or

$$\alpha + \nu(t_n) = 0 \quad \text{and} \quad \kappa_n(h) - 1 = 0,$$

then the moving shape of the numerical solution $y_n(h)$ matches the moving shape of the true solution $y(t_n)$ at the step $t_n$.

Here, the moving shape of the true solution $y(t_n)$ and the moving shape of the numerical solution $y_n(h)$ will be collectively called the two kinds of moving shapes at $t_n$.  

In the G-Euler process, the following is derived from (1.4), (2.1), (2.2) and $|e^{hS}| = e^{\alpha h}$. Where, the step size $h$ is determined by (2.7).
2.2. The G-Euler process. We execute duplicately the G-Euler process with adjacent initial values $y_0$ and $\gamma_0$. Then, $y_n$, $\gamma_n$, $\nu_n(h)$, $L_n(h)$ and $\kappa_n(h)$ are calculated by (1.4), (2.11), (2.12) and (2.13).

In order to check the ability of the G-Euler process, we define two functions as follows:

$$
\psi(h) \equiv 1 + 2\nu_n(h)h + L_n(h)^2h^2 \quad \text{and} \quad \zeta(h) \equiv e^{-2\alpha h}.
$$

Then, $\kappa_n(h)$ can be written as

$$
\kappa_n(h) = \sqrt{\zeta(h)^{-1}\psi(h)}.
$$

Since $\psi(0) = \zeta(0) = 1$, $d\psi(h)/dh = 2\nu_n(h) + 2L_n(h)^2h$ and $d\zeta(h)/dh = -2\alpha e^{-2\alpha h}$, two functions $\psi(h)$ and $\zeta(h)$ intersect at the point $(0, 1)$ and the slopes of their tangents at the point $(0, 1)$ are

$$
\frac{d}{dh}\psi(0) = 2\nu_n(h) \quad \text{and} \quad \frac{d}{dh}\zeta(0) = -2\alpha.
$$

For the following Case (1), (2) and (3), let us take a look for the graphs of $\psi(h)$ and $\zeta(h)$ near the intersection point $(0, 1)$. Figure 1 is a sample graph of $\psi$ and $\zeta$ for Case (1).

![Figure 1](image)

(1) If $\alpha + \nu_n(h) < 0$, then $\frac{d}{dh}\psi(0) < \frac{d}{dh}\zeta(0)$. Therefore, as shown in Fig. 1, there is a real number $h_0$ such that for all $h \in (0, h_0)$,

$$
\psi(h) < \zeta(h) \quad \text{i.e.} \quad \kappa_n(h) = \sqrt{\zeta(h)^{-1}\psi(h)} < 1.
$$

Taken together, if $\alpha + \nu_n(h) < 0$ and $h < h_0$, then $\kappa_n(h) - 1 < 0$.

(2) If $\alpha + \nu_n(h) > 0$, then $\frac{d}{dh}\psi(0) > \frac{d}{dh}\zeta(0)$. Hence, there is a real number $h_0$ such that for all $h \in (0, h_0)$,

$$
\psi(h) > \zeta(h) \quad \text{i.e.} \quad \kappa_n(h) = \sqrt{\zeta(h)^{-1}\psi(h)} > 1.
$$

Taken together, if $\alpha + \nu_n(h) > 0$ and $h < h_0$, then $\kappa_n(h) - 1 > 0$. 
According to Case (1) and Case (2), we found that the G-Euler process compares \( h \) and \( h_0 \), and determines the sign of \((\alpha + \nu_n(h))(\kappa_n(h) - 1)\) as follows:

\[
\begin{align*}
    h < h_0 & \implies (\alpha + \nu_n(h))(\kappa_n(h) - 1) > 0 \quad (2.15) \\
    h > h_0 & \implies (\alpha + \nu_n(h))(\kappa_n(h) - 1) < 0. \quad (2.16)
\end{align*}
\]

Here, \( h_0 \) is regarded as an approximation of \( h_{0n} \) defined in (2.4), (2.5) and (2.6). \( h_0 \) and \( h_{0n} \) are virtual values that are not calculated. However, the inequality (2.15) or (2.16) is automatically determined when running the G-Euler process.

(3) If \( \alpha + \nu_n(h) = 0 \), then \( \frac{d}{dt}\psi(0) = \frac{d}{dt}\zeta(0) \). Therefore, \( h_0 = 0 \).

From Case (1), Case (2) and Case (3), the following is derived:

\[
\alpha + \nu_n(h) \rightarrow 0 \implies h_{0n} \rightarrow 0. \quad (2.17)
\]

2.3. Match and Mismatch. So far we have considered only one fixed step \( t_n \). From now on, we will discuss whether the two kinds of moving shapes match or mismatch at \( N \) steps defined as

\[ t_n = nh, \quad n = 1, 2, 3, \ldots, N. \]

Here, \( h \) is a given fixed step size and \( N \) is a largest natural number such as \( hN \leq T \).

Let us apply duplicately the G-Euler process (1.4) to the S-transformed system (1.2) with adjacent initial values \( y_0 = y(0) \) and \( \overline{y}_0 = \overline{y}(0) \). For convenience, let \( \overline{y}_0 \) be

\[ \overline{y}_0 = \theta y_0 \quad (\theta \neq 1). \]

Then the G-Euler process calculates sequentially

\[ y_n, \quad \overline{y}_n, \quad \nu_n(h), \quad \kappa_n(h), \quad n = 1, 2, 3, \ldots, N. \]

Since the G-Euler process is a convergent scheme, the graph of \( \alpha + \nu_n(h) \) approaches the graph of \( \alpha + \nu(t) \) when the step size \( h \) decreases. See Fig. 2 (a).

![Figure 2](image-url)

**Figure 2.** The signs of \( \alpha + \nu(t_n) \) and \( \alpha + \nu_n(h) \) are the same for all steps in the non-zero region.
Let us consider the points $z_i$ and $t_{zi}$ such as
\[
\alpha + \nu(z_i) = 0 \quad \text{and} \quad \alpha + \nu_{zi}(h) = 0.
\]
Here, $t_{zi}$ is the value of $t$ corresponding to $t = z_i$, and $\nu_{zi}(h)$ represents an approximation of $\nu(t_{zi})$.

The points $z_i$ and $t_{zi}$ are called the zero points, where $z_i$ and $t_{zi}$ may not be steps. An interval $[t_{zi}, z_i]$ or $[z_i, t_{zi}]$ is called the zero region, and the part except zero regions is called non-zero region. See Fig. 2. Since the G-Euler process is a convergent scheme, as the step size decreases, all zero regions decrease.

As you can see from Fig. 2 (b), the signs of $\alpha + \nu(t_n)$ and the signs of $\alpha + \nu_n(h)$ equal at all steps $t_n$ in the non-zero region and do not equal at all steps $t_n$ in the zero region. So the sign of $(\alpha + \nu(t_n))(\kappa(t_n) - 1)$ in (2.14) is equal to the sign of $(\alpha + \nu_n(h))(\kappa_n(h) - 1)$ in (2.15) or (2.16) at all $t_n$ of the non-zero region.

In non-zero region, according to the big or small relationship between $h$ and $h_0$, the match or the mismatch is determined by the G-Euler process without using the true solution.

The zero regions are around the zero points $t_{zi}$, and the following holds from (2.17).
\[
t_n \rightarrow t_{zi} \quad \implies \quad \alpha + \nu_n(h) \rightarrow 0 \quad \implies \quad h_0 \rightarrow 0.
\]

When passing through a zero point $t_{zi}$, we consider the following steps and their maximum step sizes.
\[
... < t_{n-2} < t_{n-1} < t_n \leq t_{zi} < t_{n+1} < t_{n+2} < ... \\
... \quad h_{0n-2}, \quad h_{0n-1}, \quad h_{0n}, \quad h_{0n+1}, \quad h_{0n+2}, \quad ... 
\]

Then, some of the steps belong to the zero region and some of these maximum step sizes are less than the fixed step size $h$. This is where the mismatch arises.

Even in zero region and around of zero points, the match or the mismatch is determined by the G-Euler process without using the true solution.

In this way, the G-Euler process determines matches or mismatches in turn from $n = 1$ to $n = N$.

2.4. Application. Table 1 shows the signs of $\alpha + \nu_n(h)$ and $\kappa_n(h) - 1$ at the steps ..., $t_{n-2}$, $t_{n-1}$, $t_n$, $t_{n+1}$, $t_{n+2}$, ..., which is passing through the zero point $t_{zi}$ as the above. Type A, Type B, Type C, ... stand for the various types of mismatches arised at passing through the zero point. Type A, Type B, Type C, ... have 0, 1, 2, ... mismatches, respectively. In some cases, there may be more mismatches.

Let us look at the results of Case 2-1 of Example 2 in the next section. Here, $h = 0.0001$ and $0 \leq t \leq 3$.

Figure 3 is the graph of $\alpha + \nu_n(0.0001)$. In Fig. 3, there are 17 zero points $t_{zi}$ and they are roughly as follows: $t_{zi} = 0.1196(A), 0.3598(A), 0.4372(A), 0.5013(A), 0.8258(C), 1.0277(B), 1.1174(A), 1.1888(B), 1.4457(A), 1.7294(B), 1.8180(A), 1.8964(A), 2.1822(A), 2.3938(A), 2.4867(A), 2.5707(B)$ and $2.8425(B)$. 
There are 17 zero points $t_z$. (Type A: 11, Type B: 5, Type C: 1)

**Figure 3.** Graph of $\alpha + \nu_n(h)$ of Case 2-1 in Example 2 for $0 \leq t \leq 3$ with $h = 0.0001$.

Of these 17 zero points, 11 are Type A, 5 are Type B and 1 is Type C. There are seven mismatches. The mismatches may occur when passing the zero point $t_z$, but passing the zero point $t_z$ does not necessarily result in a mismatch.

In this example, the time $T$ is finite, the number of zero points is finite, and the number of mismatches around each zero point is finite. Therefore, the number of all mismatches is finite and they are dispersed. When the step size decreases, the number of mismatches does not

<table>
<thead>
<tr>
<th>Type</th>
<th>Time</th>
<th>$\alpha + \nu_k(h)$</th>
<th>$\kappa_k(h) - 1$</th>
<th>$\alpha + \nu_k(h)$</th>
<th>$\kappa_k(h) - 1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>$t_n$</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>A</td>
<td>$t_{n+1}$</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>B</td>
<td>$t_n$</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>B</td>
<td>$t_{n+1}$</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>B</td>
<td>$t_{n+2}$</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>B</td>
<td>$t_{n+1}$</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>B</td>
<td>$t_{n+2}$</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>C</td>
<td>$t_{n-1}$</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>C</td>
<td>$t_n$</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>C</td>
<td>$t_{n+1}$</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>C</td>
<td>$t_{n+2}$</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>C</td>
<td>$t_{n+1}$</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>C</td>
<td>$t_{n+2}$</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>
change significantly (see Table 2). And the projections on the \((x_1, x_3)\)-plane of solution curves approach a constant one, i.e. a true solution (see Fig. 8 (a), (b), (c) and (d)). That is, 

\[
\text{The numerical solutions obtained by the G-Euler process follow the true solutions to the end.}
\]

In order to confirm this fact, we will discuss the match rate in the next section.

3. **The Match Rate of Moving Shapes**

The G-Euler process calculates the followings in turn from \(n = 1\) to \(n = N\).

\[
y_n, \quad y_n, \quad \nu_n(h), \quad L_n(h), \quad \kappa_n(h).
\]

Let \(n\) be fixed and let \(\mathbb{K}(t_n, h)\) be the number of \(k\) such that for all \(1 \leq k \leq n\)

\[
(\alpha + \nu_k(h))(\kappa_k(h) - 1) > 0
\]

or

\[
\alpha + \nu_k(h) = 0 \quad \text{and} \quad \kappa_k(h) - 1 = 0.
\]

Then, \(\mathbb{K}(t_n, h)\) is the number of times that two kinds of moving shapes match from \(k = 1\) to \(k = n\). In this case, \(\mathbb{K}(t_1, h), \mathbb{K}(t_2, h), \mathbb{K}(t_3, h), \ldots\) are calculated sequentially by the G-Euler process up to \(\mathbb{K}(t_N, h)\).

We introduce the following definitions.

**Definition 3.1.** The match rate \(R(t_n, h)\) at \(t_n\) is defined by

\[
R(t_n, h) = \frac{\mathbb{K}(t_n, h)}{n} \times 100\%.
\]

The minimum match rate \(R_{\text{min}}(h)\) is defined by

\[
R_{\text{min}}(h) = \min_{n=1}^{N} R(t_n, h).
\]

The match rates are also calculated by the G-Euler process without using true solutions.

In order to survey the role of the match rate, we will apply the G-Euler process to the two examples. The true solution of Example 1 is known. \(RE_{\text{max}}(h)\) and \(RE_{\text{min}}(h)\) represent the maximum relative error and the minimum relative error, respectively. These relative errors are measured by the Euclidean norm in the Y-space. The true solution of Example 2 is unknown.

The program is compiled using Visual FORTRAN and it is executed by a personal computer.

**Table 2.** The number of zero points, the number of mismatches and the minimum match rates for Case 2-1 of Example 2 of the next section. Here, \(0 \leq t \leq 3\).

<table>
<thead>
<tr>
<th>(h)</th>
<th>Zero points</th>
<th>A-Type</th>
<th>B-Type</th>
<th>C-Type</th>
<th>Mismatch</th>
<th>(R_{\text{min}}(h))</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>19</td>
<td>13</td>
<td>5</td>
<td>1</td>
<td>7</td>
<td>97.0827 %</td>
</tr>
<tr>
<td>0.001</td>
<td>17</td>
<td>9</td>
<td>8</td>
<td>0</td>
<td>8</td>
<td>99.1667 %</td>
</tr>
<tr>
<td>0.0001</td>
<td>17</td>
<td>11</td>
<td>5</td>
<td>1</td>
<td>7</td>
<td>99.9664 %</td>
</tr>
<tr>
<td>0.00001</td>
<td>17</td>
<td>10</td>
<td>7</td>
<td>0</td>
<td>7</td>
<td>99.9916 %</td>
</tr>
</tbody>
</table>
Example 1. Consider the nonlinear autonomous system (cf. Yu [4]).
\[
\begin{align*}
x_1' &= -2x_1 - (2 + \epsilon)x_2 + \epsilon x_3 + (x_1 + x_2)(x_1 + x_2 - x_3), \\
x_2' &= 4x_1 + (4 + \epsilon)x_2 - (2 + \epsilon)x_3 - (x_1 + x_2)(x_1 + x_2 - x_3), \\
x_3' &= 4x_1 + 4x_2 - 2x_3.
\end{align*}
\] (3.1)
Here, \(x_1(0) = 2 + \frac{1}{16 + \epsilon^2}, \ x_2(0) = -3 - \frac{1}{16 + \epsilon^2}, \ x_3(0) = -2\) and its true solution is
\[
\begin{align*}
x_1(t) &= \cos(2t) + \sin(2t) + \exp(\epsilon t) + \frac{1}{16 + \epsilon^2}(\cos(4t) - 4\sin(4t)), \\
x_2(t) &= -2\cos(2t) - \exp(\epsilon t) - \frac{1}{16 + \epsilon^2}(\cos(4t) - 4\sin(4t)), \\
x_3(t) &= -2\cos(2t).
\end{align*}
\]

At the equilibrium point \(0\), Jacobian matrix \(J = \frac{\partial \mathbf{f}(0)}{\partial \mathbf{x}}\), its eigenvector matrix \(P\) and its eigenvalues \(\lambda\) are given as follows:
\[
J = \begin{pmatrix}
-2 & -2 - \epsilon & \epsilon \\
4 & 4 + \epsilon & -2 - \epsilon \\
4 & 4 & -2
\end{pmatrix}; \quad P = \begin{pmatrix}
1 & 0 & 1 \\
-1 & 1 & -1 \\
-1 & 1 & 0
\end{pmatrix}; \quad \lambda = \pm 2i \text{ and } \epsilon.
\]

Using the linear transformation \(\mathbf{x} = \mathbf{P}\mathbf{y}\), the problem (3.1) in the X-space becomes the following problem in the Y-space.
\[
\frac{d}{dt} \begin{pmatrix}
y_1 \\
y_2 \\
y_3
\end{pmatrix} = \begin{pmatrix}
0 & -2 & 0 \\
2 & 0 & 0 \\
0 & 0 & \epsilon
\end{pmatrix} \begin{pmatrix}
y_1 \\
y_2 \\
y_3
\end{pmatrix} + \begin{pmatrix}
0 \\
0 \\
0
\end{pmatrix} \begin{pmatrix}
y_1(0) \\
y_2(0) \\
y_3(0)
\end{pmatrix} = \begin{pmatrix}
1 \\
-1 \\
1 + \frac{1}{16 + \epsilon^2}
\end{pmatrix} \begin{pmatrix}
y_1 \\
y_2 \\
y_3
\end{pmatrix}.
\]

Since the maximum of the real parts of eigenvalues is \(\alpha = \max\{0, \epsilon\}\), the S-transformed system (1.2) is given by
\[
\frac{d}{dt} \begin{pmatrix}
y_1 \\
y_2 \\
y_3
\end{pmatrix} = \begin{pmatrix}
\alpha & -2 & 0 \\
2 & \alpha & 0 \\
0 & 0 & \alpha
\end{pmatrix} \begin{pmatrix}
y_1 \\
y_2 \\
y_3
\end{pmatrix} + \begin{pmatrix}
-\alpha y_1 \\
-\alpha y_2 \\
(\epsilon - \alpha)y_3 + y_1y_2
\end{pmatrix}.
\]

Case 1-1: \(\epsilon = 1\ (> 0), \ \theta = 1.5\ and \ T = 150\).
The graphs of \(\alpha + \nu_n(0.0001)\) and \(\Re(t_n, 0.0001)\) are given in Fig. 4.

(1) There is only one zero point \(t_{zi}\) (see Fig. 4 (a)).

(2) The minimum match rate is near 100\% (\(\Re_{\min}(0.0001) = 99.9686\%\)).

(3) The match rates converge to 100\% (see Fig. 4 (b)).

For the G-Euler process, Table 3 shows that if the step size decreases, then the minimum relative errors approach 0 and the minimum match rates approach 100\%. Figure 5 (a), (b), (c)
and (d) show that when the step size \( h \) approaches 0, the numerical solutions obtained by the G-Euler process approach the true solution. That is, *If the minimum match rate is near 100\%, then the numerical solutions follow the true solutions to the end.*
Table 3. Maximum relative errors, minimum relative errors and minimum match rates for Case 1-1. Here, $\epsilon = 1.0, \vartheta = 1.5$ and $T = 150$.

<table>
<thead>
<tr>
<th>$h$</th>
<th>$RE_{\text{max}}(h)$</th>
<th>$RE_{\text{min}}(h)$</th>
<th>$RE_{\text{max}}(h)$</th>
<th>$RE_{\text{min}}(h)$</th>
<th>$R_{\text{min}}(h)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.0</td>
<td>0.38709E+110</td>
<td>0.57422E+00</td>
<td>0.11565E+65</td>
<td>0.14032E+01</td>
<td>0.0000 %</td>
</tr>
<tr>
<td>1.0</td>
<td>0.19612E+77</td>
<td>0.48414E+00</td>
<td>0.95531E+00</td>
<td>0.94111E+00</td>
<td>0.0000 %</td>
</tr>
<tr>
<td>0.5</td>
<td>infinity</td>
<td>0.38021E+00</td>
<td>0.37476E+00</td>
<td>0.27459E+00</td>
<td>100.0000 %</td>
</tr>
<tr>
<td>0.1</td>
<td>infinity</td>
<td>0.16035E-01</td>
<td>0.70661E-01</td>
<td>0.62362E-02</td>
<td>100.0000 %</td>
</tr>
<tr>
<td>0.01</td>
<td>infinity</td>
<td>0.16010E-03</td>
<td>0.70660E-02</td>
<td>0.50227E-04</td>
<td>96.9697 %</td>
</tr>
<tr>
<td>0.001</td>
<td>infinity</td>
<td>0.16011E-05</td>
<td>0.70650E-03</td>
<td>0.49139E-06</td>
<td>100.0000 %</td>
</tr>
<tr>
<td>0.0001</td>
<td>infinity</td>
<td>0.16011E-07</td>
<td>0.70649E-04</td>
<td>0.49032E-08</td>
<td>99.9686 %</td>
</tr>
<tr>
<td>0.00001</td>
<td>infinity</td>
<td>0.16011E-09</td>
<td>0.70649E-05</td>
<td>0.49022E-10</td>
<td>99.9969 %</td>
</tr>
<tr>
<td>0.000001</td>
<td>infinity</td>
<td>0.16011E-11</td>
<td>0.70650E-06</td>
<td>0.49039E-12</td>
<td>99.9997 %</td>
</tr>
</tbody>
</table>

In Table 3, $R_{\text{min}}(0.5), R_{\text{min}}(0.1)$ and $R_{\text{min}}(0.001)$ are all 100%. If $h_1 > h_2$ and $R_{\text{min}}(h_1) = 100\%$, then $R_{\text{min}}(h_2)$ is 100% or near 100%. $R_{\text{min}}(h) = 100\%$ means that the numerical solutions and the true solutions have the same moving shapes at all steps, not that the numerical solutions are exact.

The numerical solution curves obtained by the Euler method and the classical Runge-Kutta fourth-order method do not follow the true solution curve.
Case 1-2: \( \epsilon = -0.5 \) \((< 0)\), \( \theta = -0.8 \) and \( T = 1000 \).

The graphs of \( \alpha + \nu_n(0.001) \) and \( \mathbb{R}(t_n, 0.001) \) are given in Fig. 6.

1. There are many zero points \( t_{zi} \) (see Fig. 6 (a)).
2. The minimum match rate is near 100% \( (\mathbb{R}_{min}(0.001) = 99.9286\%) \).
3. The match rates do not converge to 100%, but are near 100% (see Fig. 6 (b)).

For the G-Euler process, Table 4 shows that if the step size decreases, then the minimum relative errors approach 0 and the minimum match rates approach 100%.

If the minimum match rate is near 100%, then the match rates converge to 100% (see Fig. 4 (b)) or approach 100% (see Fig. 6 (b)). This means that the numerical solutions obtained by the G-Euler process approach the true solutions as the step size approaches zero (see Fig. 5 and Fig. 7 (b)). We will express as

If the minimum match rate is near 100%, then the numerical solutions follow the true solutions to the end.

### Table 4. Maximum relative errors, minimum relative errors and minimum match rates for Case 1-2.

<table>
<thead>
<tr>
<th>( h )</th>
<th>( RE_{max}(h) )</th>
<th>( RE_{min}(h) )</th>
<th>( \mathbb{R}_{max}(h) )</th>
<th>( \mathbb{R}_{min}(h) )</th>
<th>( \mathbb{R}_{min}(h) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.0</td>
<td>infinity</td>
<td>0.52361E+01</td>
<td>0.14909E+01</td>
<td>0.14300E-02</td>
<td>72.1429%</td>
</tr>
<tr>
<td>1.0</td>
<td>infinity</td>
<td>0.17526E+01</td>
<td>0.76269E+00</td>
<td>0.17684E-02</td>
<td>63.7000%</td>
</tr>
<tr>
<td>0.1</td>
<td>infinity</td>
<td>0.16924E-01</td>
<td>0.41646E-01</td>
<td>0.70137E-06</td>
<td>93.6666%</td>
</tr>
<tr>
<td>0.01</td>
<td>0.39624E+17</td>
<td>0.29131E-03</td>
<td>0.95187E-01</td>
<td>0.82868E-06</td>
<td>99.3669%</td>
</tr>
<tr>
<td>0.001</td>
<td>0.11195E+02</td>
<td>0.25957E-04</td>
<td>0.10042E00</td>
<td>0.63728E-07</td>
<td>99.9286%</td>
</tr>
<tr>
<td>0.0001</td>
<td>0.24806E+00</td>
<td>0.26082E-05</td>
<td>0.10094E00</td>
<td>0.31212E-07</td>
<td>99.9937%</td>
</tr>
<tr>
<td>0.00001</td>
<td>0.10100E+00</td>
<td>0.26099E-06</td>
<td>0.10099E00</td>
<td>0.24496E-07</td>
<td>99.9993%</td>
</tr>
<tr>
<td>0.000001</td>
<td>0.10100E+00</td>
<td>0.26101E-07</td>
<td>0.10100E00</td>
<td>0.35884E-08</td>
<td>99.9999%</td>
</tr>
</tbody>
</table>
Table 5. Minimum match rates $R_{\text{min}}(h)$ for Case 2-1. Here, $\mu = 470/19$, $\theta = 0.8$ and $0 \leq t \leq 100$.

<table>
<thead>
<tr>
<th>$h$</th>
<th>$R_{\text{min}}(h)$</th>
<th>$h$</th>
<th>$R_{\text{min}}(h)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>97.0827 %</td>
<td>0.0001</td>
<td>99.9664 %</td>
</tr>
<tr>
<td>0.005</td>
<td>98.6545 %</td>
<td>0.00005</td>
<td>99.9835 %</td>
</tr>
<tr>
<td>0.001</td>
<td>99.1667 %</td>
<td>0.00001</td>
<td>99.9916 %</td>
</tr>
<tr>
<td>0.0005</td>
<td>99.5833 %</td>
<td>0.000001</td>
<td>99.9992 %</td>
</tr>
</tbody>
</table>

Figure 7 (a) shows that the numerical solution curve obtained by the Euler method does not follow the true solution curve. However, when the step size is reduced, the numerical solution follows the true solution.

Example 2. Consider the Lorenz system (see p. 335 in Perko [1]):

\[
\frac{dx_1}{dt} = 10(x_2 - x_1), \quad x_1(0) = 1,
\]

\[
\frac{dx_2}{dt} = \mu x_1 - x_2 - x_1x_3, \quad x_2(0) = -3,
\]

\[
\frac{dx_3}{dt} = x_1x_2 - \frac{8}{3}x_3, \quad x_3(0) = 5
\]

depending on the parameter $\mu$ with $\mu > 0$. These equations are symmetric under the transformation $(x_1, x_2, x_3) \to (-x_1, -x_2, x_3)$.

At the equilibrium point $0$, we find the Jacobian matrix $J = \frac{\partial f}{\partial x}(0)$, its eigenvector matrix $P$, its eigenvalues $\lambda$ and the scalar skew-symmetric matrix $S$ as follows:

\[
J = \begin{pmatrix}
-10 & 10 & 0 \\
\mu & -1 & 0 \\
0 & 0 & -8/3
\end{pmatrix}; \quad P = \begin{pmatrix} 10 & 10 & 0 \\
9+\delta & 9-\delta & 0 \\
0 & 0 & 1
\end{pmatrix};
\]

\[
\lambda = -\frac{11}{2} \pm \frac{\sqrt{81 + 40\mu}}{2}, \quad -\frac{8}{3}; \quad S = \alpha I,
\]

where $\delta = \sqrt{81 + 40\mu}$ and $\alpha = -\frac{11}{2} + \frac{\delta}{2}$ is the maximum of real parts.

Using the linear transformation $x = Py$, the Lorenz system turns into the S-transformed system (1.2):

\[
\frac{dy_1}{dt} = \alpha y_1 + u(y_2),
\]

where

\[
u\left(\begin{array}{c}
y_1 \\
y_2 \\
y_3
\end{array}\right) = \left(\begin{array}{c}
-\frac{10}{\delta}(y_1 + y_2)y_3 \\
-\delta y_2 + \frac{10}{\delta}(y_1 + y_2)y_3 \\
\frac{3-\delta}{\delta}y_3 + 5(y_1 + y_2)(9+\delta)y_1 + (9-\delta)y_2
\end{array}\right).
\]
Case 2-1: $\mu = 470/19$, $\theta = 0.8$ and $T = 100$.

The minimum match rates for each step size are given in Table 5. Figure 8 shows the projections on the $(x_1, x_3)$-plane of solution curves computed by the G-Euler process with step sizes $h = 0.01$, 0.001, 0.0001 and 0.00001. Here, $\mu = 470/19$, $\theta = 0.8$ and $0 \leq t \leq 10$.

When $h$ approaches 0, $R_{\text{min}}(h)$ approaches 100% and the projections obtained by the G-Euler process approach a fixed one (see Fig. 8). This fixed one can only be a true projection. In other words, if the minimum match rates approach 100%, then the numerical solutions approach the true solution.

The projections obtained by the Euler method and the classical Runge-Kutta forth-order method are different from the projections in Fig. 8.

Case 2-2: $h = 0.0001$, $\theta = 0.5$ and $45 \leq t \leq 50$.

Let us consider the four cases (a) $\mu = 130$, (b) $\mu = 147.5$, (c) $\mu = 160$ and (d) $\mu = 216.2$.

Figure 9 shows projections of solution curves on $(x_1, x_3)$-plane computed by the G-Euler process with the step size $h = 0.0001$. Their minimum match rates are $R_{\text{min}}(0.0001) =$ (a) 99.8673%, (b) 99.8127%, (c) 99.6743%, (d) 99.6689% and for $h = 0.00001$, $R_{\text{min}}(0.00001) =$ (a) 99.9867%, (b) 99.9813%, (c) 99.9782%, (d) 99.9502%, i.e., near 100%.
Hence, their numerical solutions follow their true solutions to the end, respectively. That is, Fig. 9 (a), (b), (c) and (d) are reliable.

We can see that the minimum match rate indicates the reliability of the numerical solution calculated by the G-Euler process.

Figure 10 shows projections of solution curves on \((x_1, x_3)\)-plane computed by the classical Runge-Kutta forth-order method. These projections are similar to the projections in Perko [1]. But they are different from the projections in Fig. 9 obtained by the G-Euler process.

There is no basis for claiming that Runge-Kutta’s projections and Perko’s projections are reliable. On the other hand, if the minimum match rate is near 100%, then the numerical solution is reliable.

4. Conclusion

The G-Euler process is an explicit and convergence scheme. The present work shows whether the moving shapes of numerical solutions computed by the G-Euler process match the moving shapes of true solutions. The match rates are calculated without using true solutions and the minimum match rate indicates the reliability of the numerical solution.

We got to know that the closer the minimum match rate is 100%, the more closely the numerical solutions follow the true solutions to the end. That is, if the minimum match rate is near 100%, then the numerical solution is reliable.

After all, we can trust the graphs obtained by the G-Euler process more than the graphs in Perko [1].
Figure 10. Projections of solution curves on \((x_1, x_3)\)-plane computed by the classical Runge-Kutta forth-order method with \(h = 0.0001\). Here, \(\theta = 0.5\) and \(45 \leq t \leq 50\).

References


ANALYSIS OF NON-INTEGER ORDER THERMOELASTIC TEMPERATURE DISTRIBUTION AND THERMAL DEFLECTION OF THIN HOLLOW CIRCULAR DISK UNDER THE AXI-SYMMETRIC HEAT SUPPLY

SATISH G. KHAVELE$^{1}$ AND KISHOR R. GAIKWAD$^{1†}$

$^{1}$PG DEPARTMENT OF MATHEMATICS, NES, SCIENCE COLLEGE, NANDED, M.S., INDIA-431601.

Email address: khavalesatish8@gmail.com, $^{†}$drkr.gaikwad@yahoo.in

ABSTRACT. Analysis of non-integer order thermoelastic temperature distribution and its thermal deflection of thin hollow circular disk under the axi-symmetric heat supply is investigated. Initially, the disk is kept at zero temperature. For $t > 0$ the parametric surfaces are thermally insulated and axi-symmetric heat supply on the thickness of the disk. The governing heat conduction equation has been solved by integral transform technique, including Mittag-Leffler function. The results have been computed numerically and illustrated graphically with the help of PTC-Mathcad.

1. INTRODUCTION


The thermoelastic analysis and its deformation of a thin hollow circular disk subject to a partially distributed and axisymmetric heat supply on the upper surface studied in [11]. Studied the time-fractional heat conduction problem in a thin hollow circular disk and its thermal deflection in [14]. Analyzed the transient thermoelastic temperture distribution of a thin circular plate and its thermal deflection under uniform heat generation in [15]. discussed the...
transient thermoelastic stress analysis for thin circular plate due to uniform internal heat generation in [16]. Introduced the generalized theory of magneto-thermo-viscoelastic spherical cavity problem under fractional order derivative using the state space approach in [17]. Recently, many thermoelastic problems have been discussed [18, 19, 20, 21, 22].

2. PROBLEM FORMULATION

We consider a thin, hollow circular disk of thickness \( h \) occupying space \( D : a \leq r \leq b, -h/2 \leq z \leq h/2 \) initially the disk is kept at zero temperature. For \( t > 0 \) the parametric surfaces are thermally insulated and axi-symmetric heat supply on the thickness of the disk.

A mathematical model is prepared considering analysis of non-integer order thermoelastic temperature distribution and it’s deflection of thin hollow circular disk under the axi-symmetric heat supply using Caputo type time fractional heat conduction equation of order \( \alpha \).

The definition of Caputo type fractional derivative given by [23]

\[
D^\alpha f(t) = \begin{cases} 
\frac{1}{\Gamma(n-\alpha)} \int_0^t (t-\tau)^{n-\alpha-1} f^{n}(\tau) \, d\tau, & n-1 < \alpha < n; \\
\frac{d^n f(t)}{dt^n}, & n=1.
\end{cases}
\]

For finding the Laplace transform, the Caputo derivative requires information of the initial values of the function \( f(t) \) and its integer derivative of the order \( k = 1, 2, \ldots, n-1 \).

\[
L\{D^\alpha f(t); s\} = s^\alpha F(s) - \sum_{k=0}^{n-1} s^{\alpha-k-1} f^{(k)}(0), \quad n-1 < \alpha < n
\]

Also, the definition of Riemann-Liouville fractional derivative given by [23]

\[
aD^\alpha_t = \left(\frac{d}{dt}\right)^n \int_a^t (t-\tau)^{n-\alpha} f(\tau) d\tau, \quad n-1 < \alpha < n.
\]

The temperature of the hollow circular disk \( T(r, z, t) \) at time \( t \) satisfying the time fractional differential equation,

\[
\frac{\partial^2 T}{\partial r^2} + \frac{1}{r} \frac{\partial T}{\partial r} + \frac{\partial^2 T}{\partial z^2} + \frac{g(r, z, t)}{k_t} = \frac{1}{k} \frac{\partial^\alpha T}{\partial t^\alpha},
\]

with the boundary conditions,

\[
kD^{1-\alpha}_{RL} \frac{\partial T}{\partial r} = 0, \quad \text{at } r = a, \text{ for } t > 0,
\]

\[
kD^{1-\alpha}_{RL} \frac{\partial T}{\partial r} = 0, \quad \text{at } r = b, \text{ for } t > 0,
\]

\[
kD^{1-\alpha}_{RL} \frac{\partial T}{\partial z} = Q_0 f(t, r), \quad \text{at } z = h/2, \text{ for } t > 0,
\]

\[
kD^{1-\alpha}_{RL} \frac{\partial T}{\partial z} = 0, \quad \text{at } z = -h/2, \text{ for } t > 0,
\]
where, $k$, $k_t$ are the thermal diffusivity and thermal conductivity, $D_{RL}^\alpha T(r, \varphi, z, t)$ for $\alpha > 0$ is the Riemann-Liouville fractional integral $I^\alpha T(r, \varphi, z, t)$ and initial conditions,

$$T = 0, \quad \text{at } t = 0, 0 < \alpha \leq 2,$$

$$\frac{\partial T}{\partial t} = 0, \quad \text{at } t = 0, 1 < \alpha \leq 2.$$  \hspace{1cm} (2.6)

The differential equation satisfied the deflection function $w(r, t)$ defined in [10] as

$$\nabla^2 \nabla^2 w = -\frac{1}{(1-\nu)}D \nabla^2 M_T,$$  \hspace{1cm} (2.8)

where

$$\nabla^2 = \frac{\partial^2}{\partial r^2} + \frac{1}{r} \frac{\partial}{\partial r},$$  \hspace{1cm} (2.9)

and $M_T$ is the thermal moment of the disk, $\nu$ is the Poisson’s ratio of the disk material, $D$ is the flexural rigidity of the disk denoted by

$$D = \frac{E h^3}{12(1-\nu^2)}.$$ \hspace{1cm} (2.10)

The term $M_T$ is defined as

$$M_T = a_t E \int_{-h/2}^{h/2} z T(r, z, t) dz,$$ \hspace{1cm} (2.11)

$a_t$ and $E$ are the coefficients of the linear thermal expansion and the Young modulus respectively.

For the plane deformation, the boundary conditions are given as:

$$\frac{\partial w}{\partial r} = 0 \quad \text{at } r = a \text{ and } r = b.$$ \hspace{1cm} (2.12)

Initially, $T = w = 0$, at $t = 0$.

Equations (2.1) to (2.12) constitute the mathematical formulation of the problem under consideration.

3. Solution of the Heat Conduction Problem

To obtain the expression for temperature function $T(r, z, t)$; Firstly we apply the finite Fourier transform and its inverse transform over the variable $z$ in the range $h/2 \leq z \leq -h/2$. Secondly we apply finite Hankel transform and its inverse transform over the variable $r$ in the range $a \leq r \leq b$ defined in [24] to Eq. (2.1) and also using the boundary and initial condition. Finally, we apply Laplace and their inverse, one obtains the expressions of the temperature $T(r, z, t)$ as:

$$T(r, z, t) = Q_0 \sum_{p=1}^{\infty} \sum_{m=1}^{\infty} K(\eta_p, z)K_0(\beta_m, r) \frac{1}{k(\beta_m^2 + \eta_p^2)} [t^{\alpha-1} E_{\alpha,\alpha}(-k(\beta_m^2 + \eta_p^2)t^\alpha)].b_{mp}.$$ \hspace{1cm} (3.1)
Now, hence the solution (4.1) satisfies the Eq. (2.8).

where

\[ b_{mp} = \left\{ \begin{array}{l}
\int_{-h/2}^{h/2} \int_{r'=a}^{b} K(\eta_p, z') r' K_0(\beta_m, r'). f(r', t') \, dr' \, dz' + \int_{t'=0}^{t} \left[ 1 - E_\alpha (-k(\beta_m^2 + \eta_p^2) t'^\alpha) \right] \\
\times \left[ \frac{k}{k_t} \int_{-h/2}^{h/2} \int_{r'=a}^{b} K(\eta_p, z') r' K_0(\beta_m, r'). g(r', z', t') \, dr' \, dz' \right] \, dt' \end{array} \right. \]  

\]

4. Determination of thermal deflection

Assume the solution of (2.8) satisfying conditions (2.12) as

\[ w(r, t) = \sum_{m=1}^{\infty} C_m(t) \left[ \frac{J_0(\beta_m r)}{J_0(\beta_m a)} - \frac{Y_0(\beta_m r)}{Y_0(\beta_m a)} \right], \]  

(4.1)

where \( \beta_1, \beta_2, \beta_3, \ldots \) are the positive root of transcendental equation

\[ \frac{J_0'(\beta a)}{J_0'(\beta b)} - \frac{Y_0'(\beta a)}{Y_0'(\beta b)} = 0. \]

It can be easily shown that

\[ \frac{\partial w}{\partial r} = \sum_{m=1}^{\infty} C_m(t) \left[ \frac{J_0(\beta_m r)}{J_0(\beta_m b)} - \frac{Y_0(\beta_m r)}{Y_0(\beta_m b)} \right] \]  

hence the solution (4.1) satisfies the Eq. (2.8).

Now,

\[ \nabla^2 \nabla^2 w = \left( \frac{\partial^2 T}{\partial r^2} + \frac{1}{r} \frac{\partial T}{\partial r} \right)^2 \sum_{m=1}^{\infty} C_m(t) \left[ \frac{J_0(\beta_m r)}{J_0(\beta_m b)} - \frac{Y_0(\beta_m r)}{Y_0(\beta_m b)} \right], \]  

(4.2)

Using the well-known result

\[ \left( \frac{\partial^2 T}{\partial r^2} + \frac{1}{r} \frac{\partial T}{\partial r} \right) J_0(\beta_m r) = -\beta_m^2 J_0(\beta_m r), \]

\[ \left( \frac{\partial^2 T}{\partial r^2} + \frac{1}{r} \frac{\partial T}{\partial r} \right) Y_0(\beta_m r) = -\beta_m^2 Y_0(\beta_m r), \]

in Eq. (4.2), one obtains

\[ \nabla^2 \nabla^2 w = \sum_{m=1}^{\infty} C_m(t) \beta_m^4 \left[ \frac{J_0(\beta_m r)}{J_0(\beta_m b)} - \frac{Y_0(\beta_m r)}{Y_0(\beta_m b)} \right]. \]

Substituting Eq. (3.1) in Eq. (2.11), one obtains

\[ M_T = -\sqrt{\frac{2}{\pi}} 2\alpha_t E h \sum_{p=1}^{\infty} \sum_{m=1}^{\infty} \left[ \cos(\eta_p h/2) + 1 \right] K_0(\beta_m, r') \left[ \frac{\beta_m^2 + \eta_p^2}{k(\beta_m^2 + \eta_p^2)} t'^{\alpha-1} E_\alpha (-k(\beta_m^2 + \eta_p^2) t'^\alpha) \right] b_{mp}, \]  

(4.3)
Finally, substituting Eq. (4.6) in Eq. (4.1), one obtains the expression for the thermal deflection $w(r, t)$ as

$$w(r, t) = -\sqrt{2 \pi} 2 \alpha E h \sum_{p=1}^{\infty} \sum_{m=1}^{\infty} \frac{[\cos(\eta_p h/2) + 1]}{\eta_p^2} \frac{1}{1 - \frac{J_0^2(\beta_m b)}{J_0^2(\beta_m a)}} \frac{1}{\beta_m^2} \frac{1}{k(\beta_m^2 + \eta_p^2)} [t^{\alpha-1} E_{\alpha,\alpha}(-k(\beta_m^2 + \eta_p^2) t^\alpha)] \eta_p, b_{mp}.$$
where $r$ is the radius measured in meter and $A > 0$.

The first five positive root of the transcendental equation \( \frac{J_0'(\beta a)}{J_0'\beta a} - \frac{Y_0'(\beta b)}{Y_0'(\beta b)} = 0 \) as defined in [24] are $\beta_1 = 3.1965$, $\beta_2 = 6.3123$, $\beta_3 = 9.4445$, $\beta_4 = 12.5812$, $\beta_5 = 15.7199$.

The cooper material was chosen for purpose of numerical calculation for a thin circular hollow disk. The numerical calculation and graphs has been carried out with help of computational mathematical software PTC Mathcad Prime-3.1.

### Table 1: Material constants

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\rho$</td>
<td>2707 kg/m$^3$</td>
</tr>
<tr>
<td>$k$</td>
<td>84.18 m$^2$/s</td>
</tr>
<tr>
<td>$k_t$</td>
<td>386 W/(m. K)</td>
</tr>
<tr>
<td>$\mu$</td>
<td>26.67</td>
</tr>
<tr>
<td>$\alpha_t$</td>
<td>$16.5 \times 10^{-6}$ 1/K</td>
</tr>
<tr>
<td>$E$</td>
<td>70 GPa</td>
</tr>
<tr>
<td>$\nu$</td>
<td>0.35</td>
</tr>
<tr>
<td>$h$</td>
<td>0.1 m</td>
</tr>
<tr>
<td>$a$</td>
<td>1 m</td>
</tr>
<tr>
<td>$b$</td>
<td>2 m</td>
</tr>
<tr>
<td>$c_p$</td>
<td>896 J/kg.K</td>
</tr>
</tbody>
</table>

In this paper, we have obtained solutions to time fractional heat conduction equation of thin hollow circular disk and its thermal deflection with the help of Caputo time fractional derivative. It is assumed that, for $t > 0$ the heat is generated within the solid at rate $t = 200$ W/m$^3$. The numerical calculation is carried out according to the values of parameters $\alpha$ reflecting the characteristic features of the solution for various orders of the time-fractional derivative. Their distinguishing values of the parameter $\alpha$ are considered, $0 < \alpha < 1$, $\alpha = 1$ and $1 < \alpha \leq 2$ depicting weak, normal and strong conductivity. Figure 1 and Figure 2 shows the variation of temperature and thermal deflection in radial distance $r$ at instants $\alpha = 0.50$ for time parameter $t = 0.25, 0.50, 0.75, 1$. Figure 1, indicate the variation of temperature in radial direction for the different time parameter. Due to the internal heat generation at a constant rate, initially the temperature is maximum at the inner circular edge ($r = 1$) and it decreases with increase the time, becomes zero towards the outer circular edge ($r = 2$). Figure 2, shows the variation of thermal deflection in radial direction for the different time parameter $t = 0.25, 0.50, 0.75, 1$. It is observed that the thermal deflection is maximum at the inner surface of the disk and it will be zero on the boundary surface. Figure 3 and Figure 4 indicates the variation of temperature and thermal deflection in radial direction at instants $t = 0.50$ for the different values of fractional order parameter $\alpha = 0.25, 0.50, 0.75, 1$.

Figure 3, shows the variation of temperature in radial direction for the parameter $\alpha = 0.25, 0.50, 0.75, 1$. It is clear that the temperature is maximum at upper surface of solid and decreases to zero at $r = 1.8$ then follows the uniform pattern. Figure 4, shows the variation of thermal deflection in the radial direction for the different values of fractional order parameters $\alpha = 0.25, 0.50, 0.75, 1$. It is observe that the due to the internal heat generation the thermal deflection is maximum at $r = 1.3$ and it will be zero on the boundary surface.

### 6. Conclusions

We investigate the temperature and thermal deflection in a thin, hollow circular disk in a theory of thermoelasticity based on fractional heat conduction with the Caputo time-fractional derivative of order $0 < \alpha < 2$. The present method is based on the direct method, using the
finite Hankel transform, the generalized finite Fourier transform and Laplace transform. The numerical results show the significant influence of the order of time derivative on the temperature and thermal deflection with radial coordinate. The fractional order parameter $\alpha$ within the range $0 < \alpha < 1$ and $1 < \alpha < 2$ represent the weak and strong conductivity, while $\alpha = 1$ represents the normal conductivity. The results presented here will be useful for researchers.
working in material sciences also useful in studying the thermal characteristics of various bodies in real-life engineering problems, mathematical biology by considering the time fractional derivative in the field equations.

ACKNOWLEDGMENTS

The authors are grateful thanks to Chhatrapati Shahu Maharaj Research, Training and Human Development Institute (SARTHI) for awarding the Chief Minister Special Research Fellowship - 2019 (CMSRF - 2019). ORCID ID: 0000-0003-3551-301X

REFERENCES


