www.palgrave.com/journals

Comparison of algorithms to simulate disease transmission

Xiaobei Shen[4], Zoie Shui-Yee Wong[1], Man Ho Ling[2], David Goldsman[3] and Kwok-Leung Tsui[1]*

[1]Department of Systems Engineering and Engineering Management, City University of Hong Kong, Kowloon, Hong Kong SAR, China; [2]Department of Mathematics and Information Technology, The Education University of Hong Kong, New Territories, Hong Kong SAR, China; [3]H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA; and [4]The School of Management, University of Science and Technology of China, Hefei, China

A complex model to study the spread of influenza often requires efficient algorithms to simulate disease transmission. This article studies the internal mechanisms of existing algorithms. We compare existing algorithms to simulate disease transmission in an effort to identify impact factors and put forth rules for efficient algorithm selection. Specifically, an algorithm from the infectiousness perspective is recommended when both the transmission probabilities and the fraction of infectious individuals are small, or when the transmission probabilities are large but the fraction is either sufficiently small or sufficiently large. In contrast, an algorithm from the susceptible perspective should be adopted in the case of small transmission probabilities but a large fraction of infectious individuals, or large transmission probabilities and a moderate fraction. This investigation not only helps to guide a more-efficient simulation study of disease transmission in practice but also serves as a prerequisite for the development of more-advanced simulation models.

Journal of Simulation (2016). doi:10.1057/s41273-016-0003-3

Keywords: influenza spread; simulation models; algorithms to simulate disease transmission

1. Introduction

The emergence of successive global influenza pandemics has attracted increasing attention from society and public heath officials. Since the end of 2003, the spread of the highly pathogenic avian influenza H5N1 through wild and domestic poultry in Southeast Asia has been regarded as one of the most-serious human pandemic outbreaks (Abbott and Pearson, 2004; Ferguson *et al*, 2005; Peiris *et al*, 2007) In April 2009, a new influenza A virus, H1N1, was announced by the World Health Organization (WHO); and on June 11, 2009, WHO declared that an influenza pandemic had emerged in the world (Andradóttir *et al*, 2011).

Careful study of influenza spread is needed to inform public policy with respect to the control of such pandemics and to evaluate pandemic preparedness plans. Due to the complex modelling issues posed by pandemics, simulation techniques have gained widespread use in this field. Longini *et al* (2004) compared the effectiveness of various intervention strategies based on a discrete-time, stochastic simulation model of influenza spread within a structured population. Longini's stochastic simulation model was proposed by Elveback *et al* (1976) and was also applied by Halloran *et al* (2002) for the study of smallpox. Subsequently, various investigations have been conducted based on the original or modified simulation models to analyze the effectiveness of social distancing interventions (Kelso *et al*, 2009) and employee vaccination (Lee *et al*, 2010), and to evaluate intervention strategies and quantify the potential costs and benefits of different options (Ferguson *et al*, 2001; Keeling *et al*, 2001; Riley *et al*, 2003).

In most of these simulation models, the changing status of any generic person in an outbreak of a disease follows the well-known "susceptible - exposed - infectious - recovered/removed" (SEIR) paradigm (Anderson and May, 1991). These simulation models typically use a common algorithm to depict disease transmission between individuals. Such an algorithm generally works in the following way. It is assumed that each person is either fully infectious or fully susceptible, and undergoes daily contacts with others in the household, neighborhood, community, and school/workplace. These four locations where disease transmission may occur are called "contact groups" (Longini *et al*, 2005). After a susceptible individual has gone through all potential contacts, his/her status of being infectious or not is determined based on a number of Bernoulli trials. The capacity of simulation models to handle detailed population information is essentially limited by the speed of the algorithm to simulate the disease transmission.

To develop more-accurate, efficient, and powerful simulation models of influenza spread, the inherent mechanism underlying available algorithms must be thoroughly examined. However, to the best of our knowledge, no studies have ever been conducted on this important issue. In this article, we analyze and compare existing algorithms to understand their characteristics and identify rough rules for selecting the most-

Correspondence: Kwok-Leung Tsui, Department of Systems Engineering and Engineering Management, City University of Hong Kong, Kowloon, Hong Kong SAR, China.
E-mail: kltsui@cityu.edu.hk

efficient one in applications. Specifically, we focus on an individual day and investigate the efficiency of algorithms in determining who has been infected by the end of that day. Although we are concerned about transitions from the susceptible state to the exposed state, we are primarily interested in whether or not a person is infectious. In any case, the numbers of infectious and susceptible individuals are given at the beginning of the day, and these are updated as the simulation progresses from day to day. We want to determine which algorithm carries out these updates the most quickly.

The remainder of this article is organized as follows. In Section 2, we review some existing update algorithms and determine factors that influence their efficiency. A comprehensive understanding of such existing algorithms is a prerequisite for developing new and more-efficient algorithms. Based on the identified factors, in Section 3, we conduct simulation experiments to verify the intuition derived from Section 2, and then presents the basic guidelines for real applications and further research. Finally, several remarks draw the article to its conclusion in Section 4.

2. Algorithms to simulate disease transmission

In this section, we present several individual-level algorithms to simulate influenza spread. For ease of exposition, useful notation is given in Table 1. We consider g contact groups for each person in the population. The values of the infection probabilities p_{uvk}'s are assumed to be given and are usually updated from day to day. Each person of interest, either infectious or susceptible, is assumed to have daily contact with others in his contact groups (Andradóttir *et al*, 2011; Longini *et al*, 2004, 2005). Each susceptible individual has the chance of contacting infectious individuals in multiple groups and/or at multiple times. The possibility of multiple contacts is a standard assumption in the influenza spread literature and is critical when constructing an algorithm to simulate transmission. In the following, we review some existing algorithms from two perspectives: (i) the infectiousness perspective in which the algorithm loops through all of the infectious agents each day to see how many susceptibles they will infect, and (ii) the susceptible perspective in which the algorithm loops through all of the susceptibles to see who gets infected.

Table 1 Notations

Symbol	*Meaning*
N_i	Total number of infectious persons
N_s	Total number of susceptible persons
g	Number of contact groups
u	Infectious person ($u = 1, 2, \ldots, N_i$)
v	Susceptible person ($v = 1, 2, \ldots, N_s$)
k	Contact group ($k = 1, 2, \ldots, g$)
G^i_{uk}	kth contact group of infectious person u
G^s_{vk}	kth contact group of susceptible person v
n^i_{vk}	Number of infectious persons in G^s_{vk}
n^s_{uk}	Number of susceptible persons in G^i_{uk}
p_{uvk}	Probability of v being infected by u in group k

2.1. Algorithm from the infectiousness perspective

From the infectiousness perspective, a transmission algorithm determines how many susceptible individuals will be infected by each specific infectious individual u. We discuss the algorithm proposed by Tsai *et al* (2010) in this section and refer to it as the "Tsai + Sieve" algorithm. Assume that for a specific infectious person u, a susceptible person v exists in both the groups G^i_{u1} and G^i_{u2} but in no other groups. The probability of v being infected by this u in G^i_{u1} is p_{uv1} and that in G^i_{u2} is p_{uv2}. Then, considering the multiple contacts, the probability that v is not infected in the Tsai + Sieve algorithm is $q_{uv} = (1 - p_{uv1})(1 - p_{uv2})$.

Now suppose that g contact groups are associated with each individual. When infectious individual u is in contact group G^i_{uk} ($k = 1, \ldots, g$), he will fully contact all n^s_{uk} susceptible persons. Accordingly, there is an associated vector recording the transmission probabilities for the n^s_{uk} susceptibles, $(p_{u11}, \ldots, p_{un^s_{uk}1})$. At this point, the *sieve algorithm* (Algorithm 1) is implemented in Tsai + Sieve. Let $P_{uk} = \max_{v=1,\ldots,n^s_{uk}}\{p_{uvk}\}$, where $k = 1, 2, \ldots, g$.

Algorithm 1 The sieve algorithm

```
Generate $R \sim \text{Bin}(n^s_{uk}, P_{uk})$, where Bin denotes the Binomial distribution and $R$ will serve as an upper bound for the number of possible infected persons in group $G^i_{uk}$.
Randomly select $R$ candidates from the susceptible population having size $n^s_{uk}$ in $G^i_{uk}$.
For each selected candidate $v$, generate a $\text{Bern}(p_{uvk}/P_{uk})$ random variable to determine if $v$ is infected, where Bern denotes the Bernoulli distribution.
```

The value of R depends on n^s_{uk} and P_{uk}; and the advantage of applying the sieve algorithm is highlighted when n^s_{uk} is large and P_{uk} is small, in which case one must only conduct a few Bernoulli trials. The sieve algorithm is similar to the thinning algorithm for nonhomogeneous Poisson processes (Lewis and Gerald, 1979), becoming inefficient when P_{uk} is significantly larger than the second-largest transmission probability.

Note that for purposes of efficiency, the status of any infected susceptible individuals should be updated immediately; and hence the size of the susceptible population is a dynamically decreasing variable. More specifically, the first infectious individual u in G^i_{u1} is associated with the susceptible population of size n^s_{u1}. After removing the susceptible individuals infected by the first u, a second infectious individual in that same group would encounter a smaller susceptible population. In other words, we remove newly infected individuals in order to avoid the overhead of Bernoulli trials for individuals who are already infected. In any case, the gradually decreasing size of the susceptible population contributes to the efficiency of the algorithm. We now summarize the Tsai + Sieve algorithm in Algorithm 2. Let $P = \max_{u=1,\cdots,N_i;k=1,\cdots,g}\{P_{uk}\}$. The computational complexity of the Tsai + Sieve algorithm is $\mathcal{O}(N_iN_sP)$ for a generic time period (e.g., one day).

Algorithm 2 Tsai+Sieve algorithm

```
for each infectious individual $u$ do
    for each contact group $G^i_{uk}$ do
        determine $P_{uk} = \max\{p_{uvk}\}$, $v = 1, \ldots, n^s_{uk}$
        apply the sieve algorithm based on $P_{uk}$
        update the status of the susceptible individuals in $G^i_{uk}$
    end for
    remove the susceptible individuals infected by $u$ from the susceptible population
end for
```

2.2. Algorithms from the susceptible perspective

From the susceptible perspective, we investigate whether susceptible individual v would be infected after he has come into contact with all of the infectious individuals in the associated contact groups. A well-known algorithm from this viewpoint was introduced by Longini *et al* (2004, (2005), and hence we refer to it as the "Longini algorithm".

Again, we assume that there are g contact groups. In group G^s_{vk} $(k = 1, \ldots, g)$, the probability of v being infected by any of the n^i_{vk} infected people in that group is $P_{vk} = 1 - \prod_{u=1}^{n^i_{vk}}(1 - p_{uvk})$. Then, the probability of v being infected after going through all of the contact groups that day is $P_v = 1 - \prod_{k=1}^{g}\prod_{u=1}^{n^i_{vk}}(1 - p_{uvk})$.

Assume that at the beginning of a particular day there are N_s susceptible individuals. For the entire susceptible population, we have an associated set of that day's infection probabilities $\{P_1, \ldots, P_{N_s}\}$. To determine how many of the susceptibles are infected at the end of that day, N_s corresponding Bernoulli trials are performed. The computational complexity of the Longini algorithm is $\mathcal{O}(N_iN_s)$ for a generic day.

Algorithm 3 Longini algorithm

```
for each susceptible individual $v$ do
    calculate $P_v = 1 - \prod_{k=1}^{g}\prod_{u=1}^{n^i_{vk}}(1 - p_{uvk})$
    conduct a Bern($P_v$) trial
    update the status of $v$
end for
```

Ling *et al* (2015) combine the Longini algorithm with the sieve algorithm for what might be a more-efficient simulation from the susceptible perspective. We refer to such an algorithm as the "Longini + Sieve algorithm", described in Algorithm 4. The parameter $P_A = \max_v\{P_v\}$.

The only difference between Algorithms 3 and 4 is that the latter tries to reduce the number of Bernoulli trials from N_s to some smaller value through the sieve algorithm. The efficiency of Algorithm 4 is determined by how small the value of P_A can be. If there is a susceptible individual v who contacts a great many u's, then Algorithm 4 may lose its advantage, as the corresponding P_v may be very close to 1. The computational complexity of the Longini + Sieve algorithm is $\mathcal{O}(N_iN_sP_A)$ for a generic day.

Algorithm 4 Longini+Sieve algorithm

```
for each susceptible individual $v$ do
    calculate $P_v = 1 - \prod_{k=1}^{g}\prod_{u=1}^{n^i_{vk}}(1 - p_{uvk})$
end for
determine $P_A = \max\{P_v\}$
apply the sieve algorithm based on $P_A$
update the status of the susceptible individuals
```

2.3. Analysis of algorithm efficiency

In this section we compare Algorithms 2, 3, and 4 to (i) identify the factors influencing algorithm efficiency and (ii) summarize rough rules for selecting the most-efficient

Table 2 Summary of the Tsai + Sieve, Longini, and Longini + Sieve algorithms

Algorithm	*2. Tsai + Sieve*	*3. Longini*	*4. Longini + Sieve*
Perspective	Infectiousness	Susceptible	Susceptible
Multiple contacts	Yes	Yes	Yes
Time-consuming steps	Determine P_{uk} Sieve algorithm	Calculate P_v Bernoulli trials	Calculate P_v, P_A Sieve algorithm
#Bernoulli trials	$\leq \sum_k R_k$, where $R_k \sim$ Bin (n^s_{uk}, P_{uk})	$(1-\beta)M$	$R \sim$ Bin $((1-\beta)M, P_A)$
Factors	p_{uvk}, M, β		

algorithm to simulate disease transmission. The efficiency is compared for a one-day horizon, and the sizes of the infectious and susceptible populations are fixed for all of the candidate algorithms in order to carry out apples-to-apples comparisons. Let the entire population size be M, among which the fraction of the infectious population is β, i.e., $N_i = \beta M$ and $N_s = (1-\beta)M$. The salient characteristics of the three algorithms are given in Table 2. Note that the symbol "$\leq$" in the table indicates that the presented value is the upper bound of the number of Bernoulli trials in Algorithm 2.

In Table 2, the efficiency of the Tsai + Sieve algorithm depends on the values of β and the p_{uvk}'s for a given M. For that algorithm, the outermost for loop ("for each infectious individual u") is executed βM times; so when β is small, this number is relatively small. Also, the number of Bernoulli trials is reduced by the sieve algorithm, and that reduction is especially pronounced when all of the p_{uvk}'s are small. With large p_{uvk}'s, although the sieve algorithm component becomes inefficient, the Tsai + Sieve algorithm may still be efficient, as it benefits from the reduction of the susceptible population as sampling progresses. When β is large, the algorithm loses its efficiency in the outermost for loop but gains some benefits from the small susceptible population. Again, small p_{uvk}'s endow the sieve algorithm with great effectiveness, while large p_{uvk}'s eventually lead to a significant reduction of the susceptible population. As a result, we deduce that the Tsai + Sieve algorithm is efficient when almost all of the p_{uvk}'s are small or large.

Considering the algorithms incorporating the susceptible perspective, the efficiencies of Algorithms 3 (Longini) and 4 (Longini + Sieve) are affected by the values of β and the p_{uvk}'s. A small β brings a large number of executions of the for loops in the two algorithms but requires less effort for the calculation of P_v; and a large β has the opposite effect.

3. Simulation experiments and results

In this section, we evaluate our intuition about the algorithms through simulation studies. In the following experiments, algorithm efficiency is determined by the computational time (CPU time) in seconds, and the factors include M, β, and the p_{uvk}'s. In addition, only one contact group is considered, i.e., $g = 1$.

Two scenarios are considered for the transmission probability p_{uv1}. Scenario I assumes that the p_{uv1} of each contact between an infective u and a susceptible v independently follows a uniform distribution, U (δ_L, δ_U). Scenario II concerns the case in which only a small number of susceptible individuals have large transmission probabilities. Therefore, in Scenario II, $p_{uv1} = 1 - e^{-\lambda Y}$, where Y follows a standard exponential distribution, and the expectation $E(p_{uv1}) = \lambda/(1+\lambda)$. Figure 1 plots the histograms of the generated transmission probabilities (10000 for each histogram) for different values of the parameter $\lambda = \{0.0001, 0.001, 0.01, 0.1\}$ from Scenario II. Note that the scales of the four histograms in Figure 1 are different.

3.1. *Algorithm efficiency w.r.t. fraction β*

To investigate the relationship between algorithm efficiency and the fraction of infectious individuals, we set $M = 10000$, $\delta_L = 0$, and $\delta_U = \{0.00005, 0.0005, 0.005, 0.05\}$ in Scenario I, and $\lambda = \{0.0001, 0.001, 0.01, 0.1\}$ in Scenario II. Under each scenario, we gradually increase the fraction β and calculate the corresponding computational time. The program is written in Fortran and the code can be provided on request. Table 3 presents the average CPU times and attack rates (the number of susceptible people infected with the disease divided by the total number of susceptibles) after repeating the simulation trial 500 times. The expectation of the transmission probabilities, $E(p_{uv1})$, is shown below the assignment of δ_U or λ. The symbols "T + S", "L", and "L + S" represent the Tsai + Sieve, Longini, and Longini + Sieve algorithms, respectively. The bold font indicates the smallest CPU time among all of the algorithms for a given parameter setting.

The CPU time of the Tsai + Sieve algorithm exhibits an increasing and then decreasing pattern as β increases in Table 3. The decreasing pattern appears after $\beta = 0.5$ when δ_U or λ are small. However, when the transmission probabilities are large, the decrease presents at a smaller β. This indicates that in addition to the role of β, reducing the size of the remaining susceptible population in this algorithm plays an important role w.r.t. CPU time. Specifically, a larger δ_U or λ brings a sharper reduction of the susceptible population in the outermost for loop and enables the decreasing pattern to present at a smaller β. The sieve algorithm makes a significant contribution to this algorithm because each infectious individual can interact with potentially many susceptibles and accordingly initiate a large number of Bernoulli trials; and hence the reduction of such trials improves the efficiency of the Tsai + Sieve algorithm.

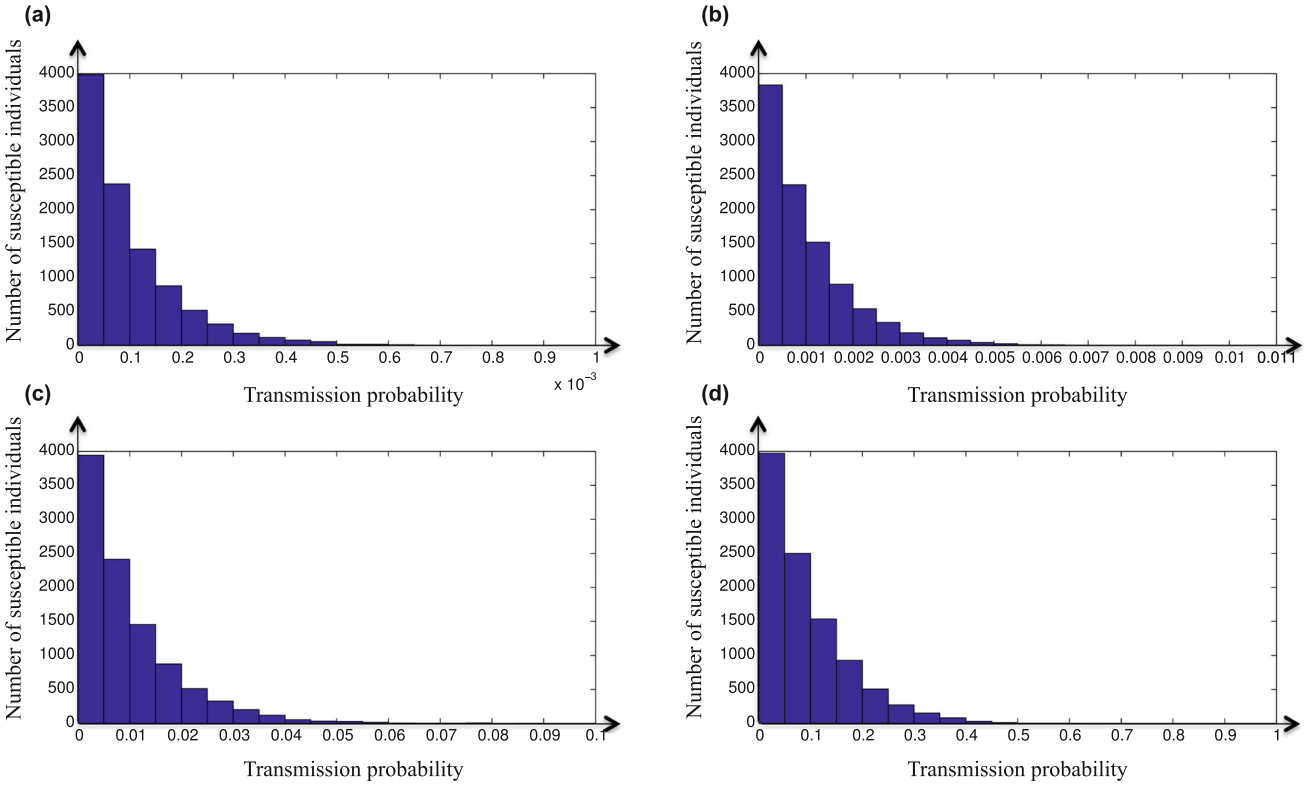

Figure 1 Histograms of the generated transmission probabilities in Scenario II with different values of λ. **a** $\lambda = 0.0001$, **b** $\lambda = 0.001$, **c** $\lambda = 0.01$, and **d** $\lambda = 0.1$.

For the Longini algorithm and the Longini + Sieve algorithm, the hill-shaped patterns of the CPU times still present as β increases. Table 3 shows that the CPU times usually achieve their maxima when $\beta = 0.5$. This can be explained as follows. Let t_m and t_B denote the times needed to perform one multiplication and one Bernoulli trial, respectively. Then the time cost of the Longini algorithm should be (very roughly) about $\beta(1-\beta)M^2t_m + (1-\beta)Mt_B$. Similarly, the time for the Longini + Sieve algorithm is approximately $\beta(1-\beta)M^2t_m + Rt_B$, where $R \sim$ Bin $((1-\beta)M, P_A)$. Both functions are concave in β and their maxima ought to be achieved for values of $\beta \geq 0.5$, which is in line with the results from the simulation study.

Note that in the Longini + Sieve algorithm, the expected time cost of Bernoulli trials is $E[Rt_B] = (1-\beta)MP_At_B \leq (1-\beta)Mt_B$, so the expected time cost of Bernoulli trials in the Longini + Sieve algorithm is a bit smaller than that in the Longini algorithm. This indicates that the sieve algorithm can save at least a few Bernoulli trials in the calculations. However, applying the sieve algorithm may also incur some costs, e.g., the set-up cost of generating the upper bound R and randomly selecting R candidates, that may overwhelm the savings in Bernoulli trials. This phenomenon is borne out in Table 3. For all cases considered in that table, Longini + Sieve is less (more) efficient than Longini for all $\beta \geq 0.01$ ($\beta < 0.01$). The intuition is that a large β indicates a large number of infectious individuals and thus leads to large values of P_A; in that case, the sieve algorithm's reduction in the number of Bernoulli trials becomes less significant.

Comparing the candidates together, we can verify from Table 3 that these algorithms are equivalent in terms of their ability to predict consistent attack rates. This means that any of the algorithms can be reliably used to depict the degree of influenza spread. With regard to efficiency comparison, we find that only the Longini and Tsai + Sieve algorithms are most efficient under typical conditions. In particular, in the case of small transmission probabilities, the Tsai + Sieve algorithm is the winner when only a small proportion of the population is infectious. As β increases in Table 3, the Longini algorithm tends to be the most efficient. In the case of large transmission probabilities, the Tsai + Sieve algorithm is the most efficient when β is either small or large, while the Longini algorithm generally outperforms the other two when β is moderate. A more-detailed investigation of the relationship between efficiency and transmission probabilities is the focus of the next section.

3.2. Algorithm efficiency w.r.t. transmission probabilities

We now conduct a simulation experiment to investigate the relationship between algorithm efficiency and transmission probabilities (or equivalently, the values of δ_U and λ). We set

Table 3 Algorithm computational time in seconds (attack rate × 100 %) w.r.t. β with given δ_U in Scenario I and λ in Scenario II

β	$\delta_U = 0.00005$			$\delta_U = 0.0005$			$\delta_U = 0.005$			$\delta_U = 0.05$		
	$E(p_{uv1}) = 2.5e{-5}$			$E(p_{uv1}) = 2.5e{-4}$			$E(p_{uv1}) = 2.5e{-3}$			$E(p_{uv1}) = 2.5e{-2}$		
	$T + S$	L	$L + S$	$T + S$	L	$L + S$	$T + S$	L	$L + S$	$T + S$	L	$L + S$
0.0001	**0.000** (0.00)	0.005 (0.00)	0.002 (0.00)	**0.000** (0.03)	0.004 (0.02)	0.002 (0.02)	**0.000** (0.26)	0.004 (0.25)	0.002 (0.25)	**0.000** (2.52)	0.004 (2.51)	0.002 (2.49)
0.0010	**0.001** (0.03)	0.005 (0.02)	0.003 (0.03)	**0.001** (0.25)	0.005 (0.25)	0.003 (0.25)	**0.001** (2.47)	0.005 (2.48)	0.003 (2.46)	**0.002** (22.3)	0.005 (22.4)	0.004 (22.3)
0.0100	**0.008** (0.25)	0.009 (0.25)	0.010 (0.25)	**0.008** (2.43)	0.010 (2.46)	0.011 (2.48)	**0.008** (22.2)	0.009 (22.2)	0.011 (22.2)	**0.008** (92.0)	0.009 (92.1)	0.013 (92.0)
0.1000	0.073 (2.48)	**0.048** (2.46)	0.072 (2.45)	0.074 (22.1)	**0.048** (22.1)	0.075 (22.1)	0.054 (91.8)	**0.047** (91.8)	0.077 (91.8)	**0.021** (100)	0.045 (100)	0.077 (100)
0.5000	0.204(11.7)	**0.121** (11.8)	0.200 (11.8)	0.164 (71.4)	**0.122** (71.3)	0.213 (71.4)	**0.093** (100)	0.126 (100)	0.207 (100)	**0.013** (100)	0.120 (100)	0.210 (100)
0.8000	0.130 (18.0)	**0.077** (18.1)	0.129 (18.1)	0.100 (86.5)	**0.078** (86.5)	0.135 (86.4)	**0.033** (100)	0.075 (100)	0.136 (100)	**0.004** (100)	0.077 (100)	0.137 (100)
β	$\lambda = 0.0001$			$\lambda = 0.001$			$\lambda = 0.01$			$\lambda = 0.1$		
	$E(p_{uv1}) = 9.999e{-5}$			$E(p_{uv1}) = 9.99e{-4}$			$E(p_{uv1}) = 9.9e{-3}$			$E(p_{uv1}) = 9.09e{-2}$		
	$T + S$	L	$L + S$	$T + S$	L	$L + S$	$T + S$	L	$L + S$	$T + S$	L	$L + S$
0.0001	**0.000** (0.01)	0.004 (0.01)	0.002 (0.01)	**0.000** (0.10)	0.004 (0.10)	0.002 (0.10)	**0.000** (0.99)	0.004 (0.99)	0.002 (0.99)	**0.001** (9.10)	0.004 (9.07)	0.004 (9.14)
0.0010	**0.001** (0.10)	0.006 (0.10)	0.003 (0.10)	**0.001** (1.00)	0.005 (1.00)	0.003 (1.00)	**0.002** (9.48)	0.005 (9.51)	0.004 (9.53)	**0.005** (61.4)	0.006 (61.4)	**0.005** (61.5)
0.0100	**0.009** (0.99)	0.010 (1.01)	0.012 (1.00)	**0.008** (9.50)	0.011 (9.51)	0.011 (9.49)	0.015 (63.0)	**0.009** (63.0)	0.013 (63.1)	0.012 (100)	**0.010** (100)	0.013 (100)
0.1000	0.080 (9.49)	**0.046** (9.49)	0.079 (9.54)	0.067 (63.2)	**0.045** (63.2)	0.077 (63.2)	0.054 (100)	**0.048** (100)	0.078 (100)	**0.012** (100)	0.046 (100)	0.074 (100)
0.5000	0.196 (39.4)	**0.122** (39.3)	0.213 (39.3)	0.143 (99.3)	**0.119** (99.3)	0.209 (99.3)	**0.031** (100)	0.123 (100)	0.217 (100)	**0.005** (100)	0.115 (100)	0.212 (100)
0.8000	0.125 (55.0)	**0.091** (55.2)	0.134 (55.1)	0.085 (100)	**0.074** (100)	0.136 (100)	**0.011** (100)	0.078 (100)	0.133 (100)	**0.002** (100)	0.075 (100)	0.135 (100)

Bold values are used to highlight the best performance among three algorithms.

Table 4 Parameters of the simulation experiment in Section. 3.2

Symbol	*Value*
β	$\{0.005, 0.01, 0.05, 0.5\}$
δ_U	$\{0.00002, 0.00004, 0.0002, 0.0004, 0.002, 0.004, 0.2, 0.4\}$
λ	$\{0.00002, 0.00004, 0.0002, 0.0004, 0.002, 0.004, 0.2, 0.4\}$
δ_L	0
M	10000
g	1

the parameters as in Table 4. The attack rate is no longer considered, as it was well-discussed in Section 3.1 .

Table 5 gives the CPU times of the three algorithms based on 10000 simulation replications. In general, the Tsai + Sieve algorithm is more sensitive to the transmission probabilities for any given β. Larger transmission probabilities result in a significant decrease in the susceptible population after each run of the outermost for loop of the algorithm. However, we mention that the advantage of the Tsai + Sieve algorithm will tend to ease up in the case of large transmission probabilities due to the weakening effect of the sieve algorithm.

The Longini model is not sensitive to the transmission probabilities at all, which is guaranteed by the nature of this algorithm. More concretely, the efficiency of the Longini algorithm is only related to (i) how many susceptible individuals are in the population and (ii) how many infectious individuals will be contacted for each susceptible person. Regarding the Longini + Sieve algorithm, although it is also robust to the transmission probabilities in Table 5, it should theoretically tend to run slower w.r.t. larger transmission probabilities under a given β—a limitation of the sieve algorithm. However, in real applications, the transmission probabilities of different contacts are usually very small (see Longini *et al*, 2004, 2005). Therefore, we only investigate the algorithm efficiency w.r.t. small transmission probabilities, and both the Longini and Longini + Sieve algorithms are robust to the small probabilities.

Next, we compare the three algorithms from the two perspectives together. For clarity, we divide the analysis into several cases according to the value of β. Note that the analysis is based on a population size of $M = 10000$.

Case 1 (β is sufficiently small) With few infectious individuals, the Tsai + Sieve algorithm is the most efficient for most transmission probabilities. However, it is difficult to define "sufficiently small", as the value may be influenced by the various experimental factors, In Scenario I, a sufficiently small β seems to be ≤ 0.01, while in Scenario II a sufficiently small value might be ≤ 0.005, though in the $\beta = 0.005$ case, the Longini algorithm wins for large λ.

Case 2 (β moderately small) For moderately small $\beta = 0.01$, Tsai + Sieve wins in Scenario I for all δ_U, and Longini generally wins in Scenario II for all λ.

Case 3 (β moderately large) The Longini algorithm becomes the best algorithm in the efficiency comparison as β increases moderately; see, for example, the cases of $\beta = 0.01$ in Scenario II or $\beta = 0.05$ for either scenario.

Case 4 (β sufficiently large) With a large number of infectious individuals, the Longini algorithm is the most efficient when the transmission probabilities are small. When the transmission probabilities are large, the Tsai + Sieve algorithm becomes the best among the three.

Table 5 Algorithm computational time in seconds w.r.t. δ_U in Scenario I and λ in Scenario II for various β

	$\beta = 0.005$			$\beta = 0.01$			$\beta = 0.05$			$\beta = 0.5$		
	$T + S$	L	$L + S$	$T + S$	L	$L + S$	$T + S$	L	$L + S$	$T + S$	L	$L + S$
δ_U												
0.00002	**0.004**	0.006	0.006	**0.009**	0.013	0.011	0.044	**0.026**	0.044	0.239	**0.124**	0.214
0.00004	**0.005**	0.006	0.006	**0.011**	0.014	0.011	0.044	**0.024**	0.044	0.232	**0.125**	0.216
0.00020	**0.005**	0.006	0.006	**0.009**	0.014	0.012	0.045	**0.025**	0.048	0.218	**0.122**	0.214
0.00040	**0.005**	0.006	0.007	**0.009**	0.013	0.012	0.045	**0.024**	0.047	0.202	**0.126**	0.211
0.00200	**0.005**	0.008	0.007	**0.010**	0.013	0.012	0.042	**0.027**	0.044	0.156	**0.122**	0.211
0.00400	**0.005**	0.007	0.008	**0.009**	0.013	0.011	0.040	**0.025**	0.046	0.131	**0.124**	0.215
0.02000	**0.005**	0.006	0.008	**0.010**	0.014	0.012	0.034	**0.026**	0.047	**0.029**	0.121	0.215
0.04000	**0.005**	0.007	0.008	**0.009**	0.013	0.011	0.031	**0.026**	0.044	**0.016**	0.121	0.213
λ												
0.00002	**0.005**	0.008	0.006	0.010	**0.009**	0.011	0.045	**0.028**	0.040	0.227	**0.124**	0.207
0.00004	**0.005**	0.007	0.007	**0.009**	**0.009**	0.012	0.045	**0.028**	0.042	0.224	**0.117**	0.198
0.00020	**0.005**	0.008	0.007	**0.009**	**0.009**	0.011	0.039	**0.029**	0.041	0.198	**0.119**	0.208
0.00040	**0.006**	0.007	0.007	**0.010**	**0.010**	0.011	0.042	**0.027**	0.047	0.174	**0.118**	0.206
0.00200	**0.004**	0.007	0.007	0.012	**0.010**	0.011	0.044	**0.029**	0.040	0.132	**0.122**	0.206
0 .00400	**0.006**	0.007	0.008	0.012	**0.010**	0.011	0.041	**0.027**	0.041	**0.068**	0.115	0.206
0.02000	0.011	**0.008**	**0.008**	0.016	**0.010**	0.012	0.035	**0.027**	0.043	**0.020**	0.118	0.208
0.04000	0.012	**0.008**	**0.008**	0.016	**0.009**	0.012	**0.023**	0.025	0.042	**0.011**	0.122	0.209

Bold values are used to highlight the best performance among three algorithms.

Table 6 CPU times in seconds of the algorithms w.r.t. population size M under Scenarios I and II

M	$(\beta, \delta_U) = (0.0005, 0.0005)$			$(\beta, \delta_U) = (0.1, 0.05)$			$(\beta, \delta_U) = (0.1, 0.0005)$		
	$T + S$	L	$L + S$	$T + S$	L	$L + S$	$T + S$	L	$L + S$
2000	**0.000**	0.001	0.001	**0.002**	0.003	0.004	0.004	**0.003**	**0.003**
4000	**0.000**	0.002	0.001	**0.008**	**0.008**	0.014	0.014	**0.010**	0.013
6000	**0.000**	0.003	0.002	**0.013**	0.018	0.027	0.031	**0.018**	0.030
8000	**0.000**	0.004	0.003	**0.018**	0.030	0.049	0.053	**0.032**	0.051
10 000	**0.001**	0.004	0.003	**0.021**	0.045	0.077	0.074	**0.048**	0.075
12 000	**0.001**	0.006	0.004	**0.027**	0.063	0.110	0.116	**0.069**	0.112
14 000	**0.001**	0.006	0.004	**0.032**	0.087	0.151	0.151	**0.090**	0.153
16 000	**0.001**	0.008	0.005	**0.037**	0.116	0.197	0.196	**0.116**	0.194
18 000	**0.002**	0.009	0.006	**0.042**	0.146	0.242	0.247	**0.141**	0.240
20 000	**0.002**	0.009	0.007	**0.046**	0.172	0.305	0.300	**0.189**	0.301
M	$(\beta, \lambda) = (0.0005, 0.0001)$			$(\beta, \lambda) = (0.05, 0.1)$			$(\beta, \lambda) = (0.02, 0.001)$		
	$T + S$	L	$L + S$	$T + S$	L	$L + S$	$T + S$	L	$L + S$
2000	**0.000**	0.001	0.001	**0.002**	**0.002**	**0.002**	**0.001**	**0.001**	**0.001**
4000	**0.000**	0.002	0.001	**0.005**	**0.005**	0.007	**0.003**	**0.003**	**0.003**
6000	**0.000**	0.002	0.002	**0.007**	0.009	0.015	0.007	**0.006**	0.007
8000	**0.000**	0.004	0.002	**0.010**	0.016	0.028	0.012	**0.010**	0.013
10 000	**0.001**	0.005	0.003	**0.013**	0.027	0.042	0.019	**0.014**	0.018
12 000	**0.001**	0.005	0.003	**0.015**	0.035	0.059	0.027	**0.020**	0.026
14 000	**0.001**	0.007	0.004	**0.019**	0.046	0.080	0.036	**0.025**	0.036
16 000	**0.001**	0.008	0.005	**0.021**	0.059	0.102	0.047	**0.031**	0.044
18 000	**0.002**	0.009	0.006	**0.024**	0.076	0.131	0.059	**0.039**	0.057
20 000	**0.002**	0.009	0.007	**0.027**	0.092	0.161	0.072	**0.046**	0.069

Bold values are used to highlight the best performance among three algorithms.

3.3. Relative efficiency w.r.t. the population size

Up to this point, all of the simulation experiments have been based on a constant population size M. To ensure the relative efficiencies of the algorithms shown in the previous experiments are not affected significantly by the values of M, we construct the following simulation experiment for validation. Three different settings of the fraction β and the transmission probabilities are chosen under each scenario: $(\beta, \delta_U) = \{(0.0005, 0.0005), (0.1, 0.05), (0.1, 0.0005)\}$ for Scenario I and $(\beta, \lambda) = \{(0.0005, 0.0001), (0.05, 0.1), (0.02, 0.001)\}$ for Scenario II. Let the population size change from 2000 to 20000 in steps of 2000. Table 6 records the CPU times based on 2000 simulation replications and shows how the algorithm efficiency changes with the value of M under the two scenarios.

Table 6 reveals that the Tsai + Sieve algorithm has the best efficiency under the first two settings for Scenarios I and II, while the Longini algorithm has the advantage in the third. When a small population is of interest, the three algorithms are almost the same in terms of efficiency. With an increase in the population size, the advantages of the Tsai + Sieve and Longini algorithms become increasingly significant. Nevertheless, the relative efficiency ranking of the three algorithms stays the same despite the value of M. Hence, our conclusions based on the simulation experiments with $M = 10000$ as discussed in Sects. 3.1 and 3.2 seem to be credible.

3.4. General guidelines for application

Through the foregoing simulation experiments, we conclude that the efficiency of any algorithm depends on both the infectious fraction β and the transmission probabilities represented by δ_U and λ in Scenarios I and II, respectively. When the transmission probabilities are smaller than, say 10^{-3}, the Tsai + Sieve algorithm should be used when β is small (e.g., $\beta \leq 0.01$), while the Longini algorithm is recommended when β is large (e.g., $\beta \geq 0.05$). However, when the transmission probabilities are large overall (e.g., larger than 0.001), the Tsai + Sieve algorithm should be selected whenever β is sufficiently small (termed β_L) or sufficiently large (termed β_U), and the Longini algorithm should be used when β is a moderate value within the interval (β_L, β_U). How we define "sufficiently small/large" depends critically on the size of the transmission probabilities. In general, larger transmission probabilities correspond to smaller β_L and β_U and a smaller range of the interval $(\beta_U - \beta_L)$.

In real applications, the transmission probabilities are usually small and an efficient algorithm for simulation of disease transmission should be selected based on the value of β. When we are interested in the early stages of a disease, the Tsai + Sieve algorithm is suggested, as a very small proportion of the population is usually infectious at that point. However, note that the number of infectious individuals will increase as the influenza spreads, and thus the Tsai + Sieve algorithm may not always be the most-efficient choice. For $\delta_U = 0.00005$, $\lambda = 0.00005$ and $\beta = \{10^{-4}, 10^{-3.5}, \ldots, 10^{-0.5}\}$, Figure 2

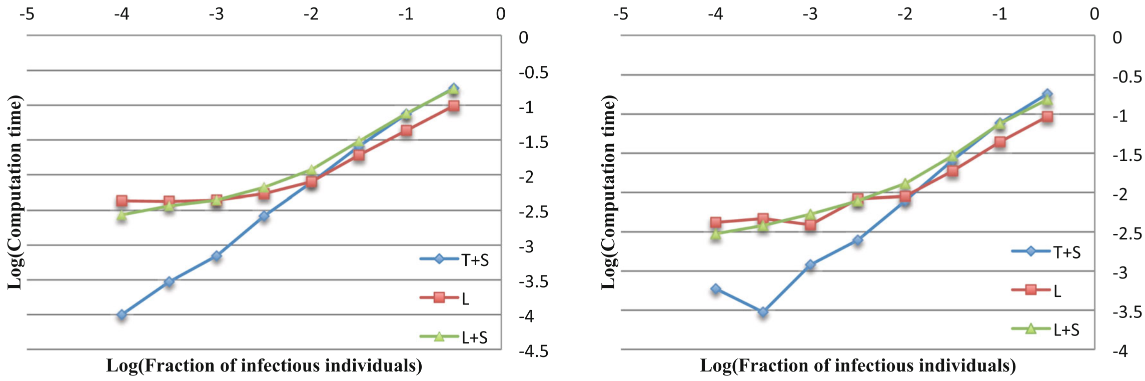

Figure 2 Relative efficiency of the algorithms w.r.t. the increasing fraction β of infectious individuals. The left figure depicts the case of Scenario I and the right figure is for Scenario II.

displays the relative efficiency of the algorithms w.r.t. the increasing fraction of infectious individuals (Scenario I on the left and Scenario II on the right). With an increase in the fraction β, the Tsai + Sieve algorithm should be replaced by the Longini algorithm when β is larger than a certain value.

4. Conclusions

In this article, we have emphasized the importance of algorithm efficiency in simulating a model of disease transmission for the eventual purpose of studying strategies for the timely control of influenza spread. To develop more-efficient algorithms and more-advanced simulation models for disease transmission, it is important to understand the characteristics of the existing algorithms and identify their advantages/disadvantages. To the best of our knowledge, no previous studies have shed light on such issues.

To guide real applications in this area, we have reviewed three alternative transmission algorithms, i.e., the Tsai + Sieve, Longini, and Longini + Sieve algorithms, from the viewpoints of either infectious or susceptible individuals, and we have distinguished some rough rules for selecting the most-efficient algorithm among the three candidates under different conditions. In addition, we have indicated the basic ideas for applying the algorithms in actual applications.

In terms of future research, one might well develop a new algorithm based on the results of this study. We have learned that the most time-consuming step in the Longini algorithm is the calculation of P_v for each susceptible individual, and the analogous bottleneck in the Tsai + Sieve algorithm is the tremendous number of Bernoulli trials. Therefore, when trying to develop a more-efficient algorithm, one may think about how to avoid these time-consuming steps. Considering the multiple contact groups for each individual in real applications, we could even optimize the existing simulation strategies according to the advantages and disadvantages of the different algorithms discussed in this article. In the following, we list some of the more-interesting findings and explain meaningful directions for further studies.

In applications, both (i) the fraction of the infectious individuals among the entire population and (ii) the overall transmission probabilities can be used as metrics for selecting algorithms in terms of their efficiency. The transmission probabilities are usually very small and in such cases no algorithm is associated with the absolute advantage in all scenarios. Therefore, we should select the most-efficient algorithm based on the value of β. In general, the Tsai + Sieve algorithm should be applied when only a few individuals are infectious. When faced with many infectious individuals, the Longini algorithm is suggested for a more-efficient simulation study.

In reality, β is a random variable instead of a constant. For example, on day 1 of an outbreak, only several infectious individuals exist in the population and thus we choose the Tsai + Sieve algorithm for the simulation. As the disease spreads, say at day 10, β may have increased to a large value, e.g., $\beta = 0.2$. At that point, the Tsai + Sieve algorithm loses its advantage and should be replaced by the Longini algorithm. Therefore, we believe that it is necessary to develop a switch strategy for the transmission algorithms based on such real-life situations. In future research, we will study the problem of when to switch the disease transmission algorithm in the simulation model.

In addition to the switch strategy, the following topics merit consideration. First, given the multiple contact groups associated with each person, a more-complicated investigation of the algorithms may be conducted to identify the additional advantages/shortages of different algorithms in such cases. Second, considering the time-consuming step in the Longini algorithm, an improved procedure of determining the probability of being infected for each susceptible individual would be a significant step towards a more-efficient transmission algorithm. Moreover, note that we only consider one contact group for each individual in the current study, and the real world is much more complicated than what we have presented here. An agent based simulation model may be built by considering the advantages/disadvantages of the algorithms incorporating multiple contact groups for each individual. It would also be interesting to study those models in the context of metropolitan cities with the scale of 1 to 10 million population.

Acknowledgments—This research was supported by the Research Grants Council Collaborative Research Fund (Ref. CityU8/CRF/12G), Theme-Based Research Scheme (Ref: T32-101/15R) and National Natural Science Foundation of China (Ref: 71420107023).

References

Abbott A and Pearson H (2004). Fear of human pandemic grows as bird flu sweeps through Asia. *Nature* **427**: 472–473.

Anderson R and May R (1991). *Infectious Disease of Humans: Dynamics and Control.* Oxford, UK: Oxford University Press.

Andradóttir S, Chiu W, Goldsman D, Lee M, Tsui K, et al. (2011). Reactive strategies for containing developing outbreaks of pandemic influenza. *BMC Public Health* **11**:S1.

Elveback L, Fox J, Ackerman E, Langworthy A, Boyd M, et al. (1976). An influenza simulation model for immunization studies. *American Journal of Epidemiology* **103**: 152–165.

Ferguson N, Cummings D, Cauchemez S, Fraser C, Riley S, et al. (2005). Strategies for containing an emerging influenza pandemic in Southeast Asia. *Nature* **437**: 209–214.

Ferguson N, Donnelly C and Anderson R (2001). Transmission intensity and impact of control policies on the foot and mouth epidemic in Great Britain. *Nature* **413**: 542–548.

Halloran M, Longini I, Nizam A and Yang Y (2002). Containing bioterrorist smallpox. *Science* **128**: 1428–1432.

Keeling M, Woolhouse M, Shaw D, Matthews L, Chase-Topping M, et al. (2001). Dynamics of the 2001 UK foot and mouth epidemic: Stochastic dispersal in a heterogeneous landscape. *Science* **294**: 813–817.

Kelso J, Milne G and Kelly H (2009). Simulation suggests that rapid activation of social distancing can arrest epidemic development due to a novel strain of influenza. *BMC Public Health* **9**: 117.

Lee B, Brown S, Cooley P, Zimmerman R, Wheaton W, et al. (2010). A computer simulation of employee vaccination to mitigate an influenza epidemic. *American Journal of Preventive Medicine* **38**: 247–257.

Lewis P and Gerald S (1979). Simulation of nonhomogeneous Poisson processes by thinning. *Naval Research Logistics Quarterly* **26**: 403–413.

Ling M, Wong S and Tsui K (2015). Efficient heterogeneous sampling for stochastic simulation with an illustration in healthcare applications. *Communications in Statistics: Simulation and Computation*. doi:10.1080/03610918.2014.977914.

Longini I, Halloran M, Nizam A and Yang Y (2004). Containing pandemic influenza with antiviral agents. *American Journal of Epidemiology* **159**: 623–633.

Longini I, Nizam A, Xu S, Ungchusak K, Hanshaoworakul W, et al. (2005). Containing pandemic influenza at the source. *Science* **309**: 1083–1087.

Peiris J, Jong M and Guan Y (2007). Avian influenza virus (H5N1): A threat to human health. *Clinical Microbiology Review* **20**: 243–267.

Riley S, Fraser C, Donnelly C, Ghani A, Abu-Raddad L, et al. (2003). Transmission dynamics of the etiological agent of SARS in Hong Kong: Impact of public health interventions. *Science* **300**: 1961–1966.

Tsai M, Chern T, Chuang J, Hsueh C, Kuo H, et al. (2010). Efficient simulation of the spatial transmission dynamics of influenza. *PLoS One* **5**(11):e13292.

Received 16 December 2015;
accepted 24 May 2016