Zhang, Nellie

**Working Paper**

# Simulating intraday transactions in the Canadian retail batch system

Bank of Canada Staff Working Paper, No. 2023-1

**Provided in Cooperation with:**
Bank of Canada, Ottawa

*Suggested Citation:* Zhang, Nellie (2023) : Simulating intraday transactions in the Canadian retail batch system, Bank of Canada Staff Working Paper, No. 2023-1, Bank of Canada, Ottawa, https://doi.org/10.34989/swp-2023-1

This Version is available at:
https://hdl.handle.net/10419/297386

BANK OF CANADA
BANQUE DU CANADA

**Staff Working Paper/Document de travail du personnel—2023-1**

Last updated: January 4, 2023

# Simulating Intraday Transactions in the Canadian Retail Batch System

by Nellie (Yinan) Zhang

Banking and Payments Department
Bank of Canada
nzhang@bankofcanada.ca



**DOI:** https://doi.org/10.34989/swp-2023-1 | **ISSN** 1701-9397

# Acknowledgements

For their comments and support, I thank James Chapman, Johnathan Chiu, Antonio Diez de los Rios, Anneke Kosse, Narayan Bulusu, all other members of the BAP Research team and the Payments Analytics Working Group. The views expressed here are those of the author and not necessarily those of the Bank of Canada. All errors remain my own.

# Abstract

This paper proposes a unique approach to simulate intraday transactions in the Canadian retail payments batch system. Such transactions are currently unobtainable. The simulation procedure, though demonstrated in the realm of payments systems, has tremendous potential for helping with data-deficient problems where only high-level aggregate information is available. The approach uses the concept of integer composition in combinatorics to break down the daily total value and volume (available) into individual data points (unavailable) throughout the day. The algorithm also introduces a technique to incorporate any intraday timing pattern (known or hypothetical) to make simulated data closer to reality. Simulation results show that the probability distribution of individual payment values is remarkably stable through repeated random sampling. This suggests a high degree of accuracy and viability of this simulation method. In addition, the densities of simulated intraday transactions are found to be invariably skewed to the left of the mean payment value, which reflects the nature of retail payments systems.



*Topics: Financial Markets, Payment clearing and settlement systems*

*JEL codes: C, C63, E42, E58*



# Résumé

Cette étude propose une méthode inédite pour simuler les transactions intrajournalières dans le système canadien de traitement par lots des paiements de détail. Il est actuellement impossible d'obtenir des données sur de telles transactions. Bien que la méthode de simulation soit démontrée dans le domaine des systèmes de paiement, elle se révèle très prometteuse pour combler le manque de données détaillées quand on dispose uniquement de renseignements généraux agrégés. La méthode fait appel au concept de composition des nombres entiers en analyse combinatoire pour diviser la valeur et le volume totaux quotidiens (disponibles) en points de données individuels (non disponibles) au cours de la journée. L'algorithme applique également une technique pour intégrer tout profil chronologique intrajournalier (connu ou hypothétique) afin de rapprocher les données simulées de la réalité. Les résultats de la simulation montrent que la distribution des probabilités de la valeur des paiements individuels est remarquablement stable dans les échantillonnages aléatoires répétés. On peut donc supposer que cette méthode de simulation offre un haut niveau de précision et qu'elle est très viable. De plus, les densités des transactions intrajournalières simulées sont invariablement étalées vers la gauche de la valeur moyenne des paiements, ce qui illustre la nature des systèmes de paiement de détail.



*Sujets : Marchés financiers, Systèmes de compensation et de règlement des paiements*
*Codes JEL : C, C63, E42, E58*

# 1 Introduction

In the modern digitized world, data has proven more important than ever and information is the key to success. However, despite the unprecedented efforts to collect, store and manage data, data is not always available because of restricted access, cost concerns, the technical challenges of extracting data from legacy systems, or the simple fact that it is impossible to collect every bit of information that surrounds one. For example, the financial industry lacks data on the linkages between financial institutions and risk exposures across all sector members. In the context of payments systems, especially in retail payments systems, not every single transaction is recorded due to the extremely large number of such transactions. However, such information can be crucial in understanding system characteristics and participants' payment behaviours and in examining intraday risk exposures.

This paper proposes a unique approach of reconstructing unknown individual data entries based on aggregate statistics, such as daily total value and volume. The general objective of this algorithm is to simulate the unknown underlying micro data that generates the aggregate information seen. The study demonstrates this simulation technique using the transaction records in the Canadian Automated Clearing and Settlement System (ACSS).

In ACSS, a tremendous number of paper and electronic-based payments are exchanged continuously every day. Payments are exchanged bilaterally outside the central platform, and for every direct clearer in the system, all of the payments sent to various other system members are then collected into a single payment (i.e., a batch total) at the end of day for clearance and settlement. The system contains $22 \sim 25$ payment streams, and a separate batch total is calculated for each payment stream. For most payment streams, such batch-total entries are then further aggregated into a multilateral net position for each system direct clearer (DC) for that day.[2] Currently, the most detailed levels of data extracted from ACSS are daily total transaction value and volume between bilateral pairs of DCs

[2]There are 13 DCs in ACSS including Bank of Canada, among which 12 are direct participants in the new Canadian large-value payments system, LYNX (having replaced the old system LVTS in August 2022). The number of payment streams in ACSS varies throughout the history of the system. Five payment streams, mostly related to Government of Canada transactions, settle bilaterally directly through LYNX (historically in LVTS); i.e., the bilateral net balances between DCs are not included in the final multilateral ACSS positions. Interested readers can find more detailed information about ACSS in Arjani and McVanel (2006).

in each payment stream on a given business day. Individual retail payments are never obtainable.

The first step in the proposed approach is to break up the total transaction value into a series of smaller values, on the condition that (i) these smaller values sum to the daily total amount and (ii) the number of these smaller values is equal to the daily total volume observed in the data. The second step is to randomly pair up each simulated individual transaction with a transaction time stamp, which is generated according to a given intraday payment timing pattern. In the next step, the payment profiles of every stream between each participating pair of DCs are then combined to create a single transaction data file for the entire system during a given sample period, which is then sorted by transaction time upon merging. By construction, these simulated individual intraday payments satisfy the constraint such that, at any time of day and for any given DC, its intraday maximum net debit position (NDP) never exceeds its daily total value sent.

This approach draws heavily on the theory of integer composition in combinatorics. According to number theory, there are a finite number of ways of writing an integer $n$ as the sum of exactly $m$ number of positive integers. For example, one day in ACSS, in stream $A$, direct clearer $B$ sends three payments having a total value of 5 to direct clearer $C$. There are six possible distinct ways of splitting the 5 into three individual payments, each of which is an integer number of cents: $\{3, 1, 1\}$, $\{2, 2, 1\}$, $\{2, 1, 2\}$, $\{1, 3, 1\}$, $\{1, 2, 2\}$, and $\{1, 1, 3\}$. Note that some of the sequences are identical in value but differ only in the order of elements. This is because in payments systems, the order of payments is influential in a participant's intraday funds positions and liquidity recycling in the system as a whole.

In the method developed in this paper, a random composition is chosen with an equal probability of $1/N$, where $N$ denotes the total number of all possible compositions of a given integer $n$. An efficient search algorithm is utilized in finding where dividing lines can be drawn to generate a sequence of smaller-value integers that satisfy all of the economic constraints in the known data. Transaction values, formulated as dollars and cents in the input data are first converted into an integer in denomination of cents only, and then, in the last step of algorithm, changed back to dollars and cents in the output.

The results of this remarkably stable simulation suggest a high degree of accuracy, especially in the scenarios where the daily total volume of transactions is very

large relative to the daily total value. The random selection process is repeated $1,000$ times, and the density of payment value is examined across the sampling distribution. The results show that the probability of individual payment values within a small window appearing in any randomly chosen composition is quite stable and that probability becomes more stable as the daily total volume (i.e., the number of individual payments we are aiming to break up the total value into) increases. The intuition behind this finding is simple and rather obvious. For example, if a total of \$10 is split into three payments, e.g., $\{\$2, \$3, \$5\}$, $\{\$1, \$2, \$7\}$, or $\{\$3, \$3, \$4\}$, then the probability of having \$1 as the value of one of the three individual payments ranges from 0 to 58.3%. However, if, instead of three payments, a composition of \$10 into eight individual payments is sought, then the probably of having \$1 as one part of the composition has become a certainty—and, in fact, the probability of having any number greater than 3 in the composition is also a certainty: equal to 0. To illustrate, there are only two possible partitions (Recall that compositions are permutations of partitions.):

$$\begin{aligned} 10 &= 1+1+1+1+1+1+1+3 \\ &= 1+1+1+1+1+1+2+2 \end{aligned}$$

This unique method can be a viable, useful and powerful tool for simulating a level of detail in the data that is otherwise unobtainable, or simply un-collectable (e.g., due to sheer quantity) when aggregate information is somehow available. The algorithm proposed here executes reasonably quickly. In this study, more than 4.6 billion individual transactions were generated in a time frame of about 90 minutes. Furthermore, this approach can be applied to any data set in any field of study, not only to payments systems in the financial sector.

The fundamental difference between this paper and Saiz et al. (2018), which also looked into profiling ACSS intraday payments, is that the simulation here generates every individual payment with an associated time stamp rather than snapshots of participants' intraday net funds positions at various times of day. This provides the first advantage of the method, as complete and high time-resolution transaction data (e.g., available for LYNX) proves to be extremely useful and important in conducting research and analyses that otherwise are not feasible, such as intraday liquidity distribution, participants' payment behaviours, and intraday risk exposures. Saiz

et al. (2018) argue that in ACSS, each individual payment follows a distribution with unknown mean and variance. However, based on the Central Limit Theorem (CLT), it can be derived that the sum of payments sent by a participant over a period of time, which is essentially this participant's net funds position at a certain point of time, follows a normal distribution characterized by the parameters that can be computed from the observable data (e.g., the total number of payments in that given time period, the daily average net funds positions, and the variance of these net positions). Random draws are then made to generate intraday net positions for various lengths of time intervals ranging from 1 minute to 6 hours.

The second advantage of this simulation technique is its updatability and flexibility. It is able to incorporate additional information (e.g., if new data surfaces later on) to produce a more representative and accurate data set. For example, if it were known that transactions in one ACSS stream were all in multiples of $1,000, then that can, hypothetically, be easily incorporated into the algorithm to make the simulation results satisfy this additional constraint. It would be much harder to achieve this with the model suggested by Saiz et al. (2018).

The concept of integer composition is used widely in other scientific fields, such as statistical mechanics, string theory in mathematical physics, biology, and non-parametric statistics. To the best of my knowledge, this paper may be the first application of this particular aspect of number theory in payments systems.

The rest of the paper proceeds as follows: section 2 contains a brief description of the concept of integer composition; section 3 provides implementation details of applying this number theory to ACSS data; section 4 presents the simulation results; and section 5 concludes.

## 2 Integer Composition

The composition of an integer $n$ is a characterization of $n$ as a sum of smaller positive integers. Each component element is called a *part* of the composition. Let us use the function $c(n)$ to denote the number of compositions of the integer n: for example,

$c(3) = 4$. Below are all the four compositions of the integer $n = 3$:

$$\begin{aligned} 3 &= 3 \\ &= 2 + 1 \\ &= 1 + 2 \\ &= 1 + 1 + 1 \end{aligned}$$

Partitions of an integer, similar to compositions, are various ways of writing it as a sum of positive integers; however, the order of the parts are ignored. In the example above, $1 + 2$ and $2 + 1$ are two distinct compositions, but they are both considered the same partition of 3.[3] In this paper, I choose integer composition as the base of my approach because the order of transactions is crucial in payments systems. Different ordering can generate ripple effects and result in drastically different outcomes in terms payment timing, liquidity usage, and liquidity recycling for individual participants as well as for the system as a whole. For instance, participant $A$ sends to participant $B$ a payment of \$5 first and \$1 second. Upon receiving \$5, participant $B$, with no money in store, is able to send a payment of \$3 to participant $C$. If $A$ had sent \$1 first, $B$ would not have had enough liquidity to send the \$3 that enables $C$ to send its own payments.

Each positive integer $n$ has $2^{n-1}$ distinct compositions, which contain sequences of parts of various lengths. The number of compositions of a positive integer $n$ into *exactly* $m$ parts is given by the binomial coefficient,

$$c(n, m) = \binom{n-1}{m-1} = \frac{(n-1)!}{(m-1)!(n-m)!}$$

Note that summing over all possible numbers of parts $m$, we get $2^{n-1}$ as the total number of distinct compositions of $n$:

$$\sum_{m=1}^{n} \binom{n-1}{m-1} = 2^{n-1}$$

[3]The theory of integer partitions was first pioneered by Leonhard Euler in 1748, and it is an evolving theory that still has many unsolved problems, such as finding criteria for determining whether the number of partitions of an integer is odd or even. Interested readers can refer to Andrews (1998), Glaisher (1883), and Cohen (1981).

Broadly speaking, there are two distinct methods of obtaining compositions of an integer: the algebraic approach based on generating functions and the combinatorial method. This paper proposes an algorithm that is built on a pure combinatorial foundation; the details of the algorithm are elaborated in section 3.2.

The basic idea of the combinatorial method is simple, and the results of function $c(n)$ can be generated in the following way. If we write $n$ as the sum of 1's, then we have $n-1$ spaces between each pair of adjacent 1's; imagine that in each of the spaces, we can insert a barrier, which grants us a total of $2^{n-1}$ of all possible ways of inserting partitioning barriers. All the individual 1's between any two barriers are summed up and the sum is considered as one part of the composition. For example, $c(3) = 4$ can be illustrated as below:

$$
\begin{aligned}
3 = 1 \quad 1 \quad 1 = & \qquad (3) \\
= 1 \quad \blacklozenge \quad 1 \quad 1 = & \qquad (1,2) \\
= 1 \quad 1 \quad \blacklozenge \quad 1 = & \qquad (2,1) \\
= 1 \quad \blacklozenge \quad 1 \quad \blacklozenge \quad 1 = & \qquad (1,1,1)
\end{aligned}
$$

# 3 Implementations

The simulation of individual intraday transactions in ACSS consists of two steps. First, we develop a complex algorithm that randomly chooses, with equal probability among all possibilities, one composition of a daily total transaction value into a sequence of individual payments, the number of which is equal to the daily total volume. For the purpose of handling integers, transaction values are first converted into cents, and then back to dollars in the simulation output. The challenge is that as the daily total value $n$ and/or daily total volume $m$ increases, the number of all possible compositions of $n$ into exactly $m$ parts, i.e., $c(n,m) = \frac{(n-1)!}{(m-1)!(n-m)!}$, increases exponentially: for example,

$$
\begin{aligned}
c(10,3) &= 36 \\
c(100,3) &= 4,851 \\
c(100,30) &= 8.8e+24
\end{aligned}
$$

On a typical day, in a relatively small payment stream of ACSS, one observes a total transaction volume of 200 and a total transaction value of 68 million; in bigger payment streams, the daily total volume can amount to 5 million and the daily total value reaches 540 million. In such cases, $c(n, m)$ is simply an uncomputable figure, too large to have a name for. A detailed description of data used in this study is provided in section 3.1.

To deal with this challenge, an algorithm is designed and implemented that is sufficiently efficient to carry out such cumbersome calculations in a reasonable amount of time. Furthermore, the algorithm needs to be "smart" in the sense that it does not actually lay out all the possible compositions in memory and literally choose one impartially. Rather, it achieves the goal of choosing a random composition with an equal probability such that the algorithm can be proven mathematically to be theoretically equivalent to the mechanical process. As shown clearly in the examples above, in the cases where $n$ and $m$ are very large, it is absolutely impossible and impractical to list out all the possible compositions, even for computers.

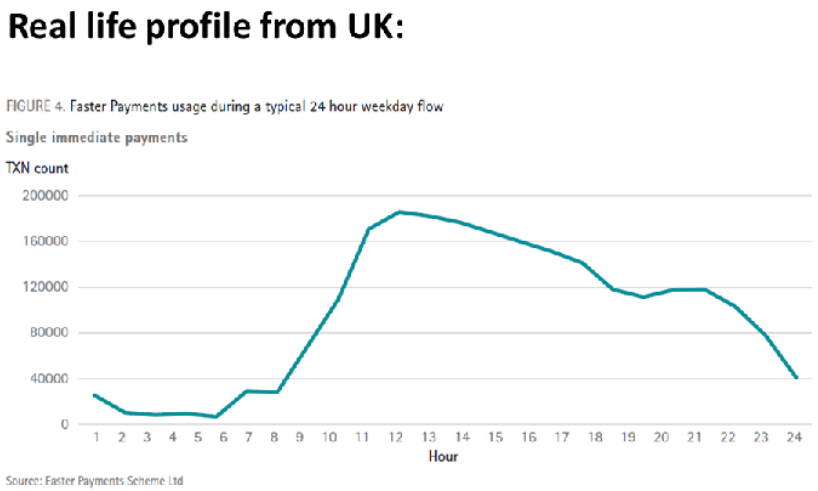
The second step in the simulation is to assign a time stamp to each individual payment, based on a given intraday timing pattern of a retail payments system. Unfortunately, no such data is available yet in ACSS. This study borrows the intraday timing pattern of the U.K. Faster Payments (shown in Figure 1), and this data set can be easily replaced with Canadian data in the future when it becomes available. The timing distribution data illustrates how many transactions are conducted on average during each hour of the day. The fundamental idea of generating time stamps based on the volume time distribution is that when more payments are made at a certain time of day, that time is, in general, considered to be a more probable time for transactions to occur. Details of this simulation process are further explained in section 3.3.

## 3.1 Data

To carry out the simulation in this study, a sample of ACSS data is used that spans from March 2, 2020, to September 3, 2021, containing a total of 383 business days. During this period, more than 12.1 billion transactions worth $11.9 trillion in total settled in the Canadian retail batch payments system.[4] The beginning of

[4]That is, on average each day, ACSS settles about 32 million transactions totaling $31 billion in value. In contrast, in Canada's large value payment system (Lynx), the daily settlement is around

**Figure 1:** Intraday Timing Pattern—U.K. Faster Payments




This graph illustrates timing distribution of payment volume in the U.K. Faster Payments system. The horizontal axis is the time of day, with a resolution of 1 hour. The vertical axis shows the average number of real-time payments made during each hour duration on a typical day. Source: Accenture (Accenture)

**Table 1:** Pairwise Daily ACSS Data per Payment Stream

| Date | Sender | Receiver | Stream | Stream name | Volume | Value ($) |
|---|---|---|---|---|---|---|
| 20200601 | DC A | DC B | C | C-AFT credit | 80,733 | 289,552,539.17 |
| 20210721 | DC C | DC D | D | D-AFT debit | 94,969 | 75,565,233.85 |
| 20201216 | DC E | DC F | E | E-Encoded paper | 109 | 401,144.26 |
| 20210720 | DC G | DC H | N | N-Shared network | 1,796 | 343,280.00 |
| 20201230 | DC I | DC J | O | O-Images | 25,183 | 214,586,801.08 |
| 20210223 | DC A | DC K | Q | Point of service | 2,612 | 181,538.04 |

the sample period is determined by the earliest day on which ACSS weekly data extracts became available, and the end of the sample is around the time when this study was conducted.

The weekly extracts are one of the most comprehensive data sets currently available in ACSS. Each line of data contains the total value and number of transactions sent by one ACSS direct clearer to one other, in one particular payment stream on a specific day. A partial example of this data is given in Table 1.

43,061 in volume and $472 billion in value in the same time period.

## 3.2 Generating Payments: The Algorithm

This section describes the algorithm created to break down a daily total transaction value into a known number of individual payments.

When an integer $n$ is written as a composition of $m$ parts, by construction, the number of all possible positions for inserting partitioning barriers between parts is $n-1$ and the number of barriers needed is $m-1$. For example, the composition of 3 into exactly two parts provides $3-1=2$ places for inserting partitioning barriers, and only $2-1=1$ barrier is needed in this case. Hence, the algorithm consists of two steps. First, we randomly choose $m-1$ partitioning positions with equal probabilities; specifically, the probability of any available position being chosen for inserting a barrier is equal to $1/C$, where $C$ is the total number of vacant candidate positions remaining. Second, we count the number of 1's between each pair of barriers or to the left (right) of the first (last) barrier, and let the sum of 1's be the corresponding part of the composition of $n$ in that position.

### 3.2.1 Positioning Barriers

The algorithm chooses a set of $m-1$ partitioning positions one at a time. Each selection is made from all the remaining vacant positions (i.e., not yet chosen). The first position $p_1$ can be any position from 1 to $n-1$; each subsequent $p_i$ can be any choice among all $n-1-i$ remaining candidate positions, excluding those already occupied by the $i-1$ previous selections.

Let $r_i$ be any position in all remaining $n-1-i$ possibilities, and $r_i$ can be generated randomly from a uniform distribution. An essential part of our integer-composition algorithm involves a bijective mapping function that maps from a local index number (i.e., $r_i$) in a set of all remaining vacant positions to its corresponding global location (i.e., $p_i$) in the entire array of $n-1$ possible partitioning positions. To illustrate, I use an example of composing integer $n=8$ into at least four parts $m \geq 4$. Specifically, there are seven possible positions for inserting barriers, and we need at least three barriers.

Table 2 captures a zoomed-in snapshot of the state of the variables in the interim step of mapping from $r_3$ to $p_3$, after $p_1$ and $p_2$ have already been selected. Previously selected positions are marked by solid diamonds at position #2 and #5 in this example; vacant positions are indicated by grey-coloured diamonds. The first row

**Table 2:** Mapping from $r_3$ to $p_3$ after $p_{1,2}$ Have Been Selected

| | 1 ♦ 1 | ♦ | 1 ♦ 1 ♦ 1 | ♦ | 1 ★ 1 ♦ 1 |
|---|---|---|---|---|---|
| $r_3$ | 1 | Occupied | 2 | 3 | Occupied | 4 | 5 |
| $p_3$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 |

of the table shows that after $p_1$ and $p_2$ are chosen, the remaining vacant positions are indexed by $1, 2, 3, 4$ in the local order. The second row of the table lists the indices of all possible locations for barrier insertion in the global order from 1 to 7. If the third barrier is randomly chosen to be at $r_3 = 4$, marked by a solid star, then the bijective mapping function is used to map from 4 to 6 because $p_3 = 6$.

It can be seen that the value of $p_i$ must be larger than $r_i$ by the cardinality of the set of $p_j$ where $j < i$ and $p_j < p_i$, because the set of $r$ keeps altering each time after a new position is chosen whereas $p$ never changes. As $r$ skips the previously chosen positions, $r_i$ and $p_i$ begin to differ at first by 1 and later on by more. However, when the algorithm operates, the corresponding $p_i$ is not yet known in the scope of the remaining vacant positions; we need to calculate $p_i$ from $r_j$ such that $j \leq i$. This problem is compounded by the fact that the size of $p_i$ can be very large (recall that some large streams have a volume of $5,000,000$)—practically unsolvable. Hence, the following iterative procedure is suggested so that the algorithm runs in a reasonable amount of time for fairly large values of $i$ and $n$. Let $k^{(h)}$ be a candidate position for $p_i$, where $h$ is the iteration index. $L^{(h)}$ is the number of previously chosen positions that precede $k^{(h)}$:

$k^{(0)} \leftarrow r_i$
$L^{(0)} \leftarrow 0$
**repeat** for $h = 1, 2, 3, \cdots$
  $L^{(h)} \leftarrow \{p_j | j < i \wedge p_j \leq k^{(h-1)}\}$
  $k^{(h)} \leftarrow k^{(h-1)} + \left(L^{(h)} - L^{(h-1)}\right)$
**until** $k^{(h)} = k^{(h-1)}$
$p_i \leftarrow k^{(h)}$

Take the previous example of choosing $r_3 = 4$ (i.e., Table 2). The goal of each iteration is to find the number of elements in the set of $\{p_j\}$ that $k^{(h)}$ is greater than or equal to, where $j \leq 3$. Initially, $k^{(0)}$ is set to $r_3 = 4$ and $L^{(0)}$ is set to 0.

Next, a few steps are repeated such that $L^{(1)}$ is set to the number of $\{p_j\}$ that precede 4: There is one such $p_j$, at position #2, therefore $L^{(1)}$ is set to 1. $k^{(1)}$ is then given the value of $k^{(0)}$ plus the difference between $L^{(1)}$ and $L^{(0)}$, hence, $k^{(1)}$ is $4 + 1 = 5$.

In the next iteration, $h = 2$ and $k^{(h-1)} = 5$. As well as position 2, position #5 now precedes $k^{(h-1)}$; therefore, $L^{(2)} = 2$. Since $L^{(2)} - L^{(1)} = 2 - 1 = 1$, $k^{(2)} = 6$. At this point, the algorithm does not yet terminate, because $k^{(2)}$ is larger than $k^{(1)}$.

Iterations continue. Now $h = 3$; with $k^{(2)} = 6$, only position #2 and position #5 precede $k^{(h-1)}$. So $L^{(3)} = 2$, which is the same value as $L_{(2)}$. Since the difference is 0, $k^{(3)} = k^{(2)}$ and $p_3 = k^{(3)}$. The finding of $p_3$ is now complete, with its value being 6.

Note that in the algorithm presented, it can be time consuming to count each element of the set $p_j$ and see if $p_j \leq k^{(h)}$, because it would require time proportional to the magnitude of $i$, a number that can potentially be enormous. Instead, the values of $p_j$ are stored in a red-black tree Bayer (2004) that is designed to have each node contain a subtree size. Each time the tree is modified, i.e., when insertions and/or deletions occur, a new tree will be generated by rearranging and recoloring all the stored information (to either red or black) to make sure the search tree remains balanced. Such re-balancing operation is executed in $log(n)$ time. Specifically, in this study, using a red-black tree allows us to look up each intermediate position $k^{(h)}$ in the amount of time proportional to $log(i)$.

### 3.2.2 Constructing Parts

Once a set $\{p_j | 1 \leq j \leq M\}$ is determined, the composition parts can be found in the spaces between each successive pair of $p$ values, with the exception of the very first and very last value. The first part of the composition is the number of 1's before the first barrier $p_1$ and the last part is the sum of 1's after the last barrier. Formally, the composition part $i$ can be calculated as follows:

$$C(i) = \begin{cases} p_1 & \text{if } i = 1 \\ n - p_{m-1} & \text{if } i = m \\ p_{i+1} - p_i & \text{otherwise} \end{cases}$$

### 3.3 Generating Time Stamps

As mentioned earlier, this study utilizes the timing distribution of transaction volume over the course of day in the U.K. Faster Payments system for the purpose of simulating a time stamp for each individual simulated payment in the Canadian ACSS. One caveat is that timing distributions of two different payments systems in two different countries can be very different; in addition, all payment streams within the ACSS itself may not even exhibit the same timing patterns, e.g., cheques versus point-of-sales. The U.K. timing data was adopted because of the lack of the Canadian equivalent, and the suggested simulation framework can later be easily updated with any new information.

In this study, this timing distribution is considered a probability density function (PDF) of payments in the realm of time. It follows that the transaction volume at a certain time of day represents the probability that this particular time will be chosen as a payment time: the higher the transaction volume at a given time, the more likely a payment in general will be made at that time.

The first step is to normalize the timing distribution of payment volume such that the area under the curve is equal to 1. Next, this normalized PDF is converted into a cumulative distribution function (CDF), because CDFs provide cumulative probabilities of a payment being settled *prior to and up to* a given time, not only exactly at that point of time. As shown in Figure 1, the timing profile of the U.K. Faster Payments System provides transaction volume only during each hour of a day. Positions between two hours are linearly interpolated between the hour values such that a cumulative probability can be constructed for every second of the 24-hour period. Given this piece-wise linear function, we first calculate the area under the curve for each hour duration, and then integrate the areas over the course of day. The integration results in a piece-wise quadratic CDF, and mathematical details are provided in Appendix A.

Finally, for each simulated individual payment, a number $i \in [0, 1]$ is randomly drawn from the uniform distribution, and the inverse CDF is used to map from that random number to a time stamp. After the whole payments profile is created, all payments from all participants are combined in all streams and sorted strictly by the payment times.

# 4 Simulation Results

After the simulation is carried out in its entirety, a complete payment profile of exactly $12,158,962,927$ ACSS transactions is generated.

One of the most interesting aspects of the simulation results is that, given a pair of daily total value ($n$) and daily total volume ($m$) of payments sent by one ACSS direct clearer to the other in a given stream, repeated random sampling yields a remarkably stable distribution of individual payment values across all randomly generated compositions. In this repeated sampling exercise, six lines of data entries (shown in Table 3) were chosen to serve as examples in this paper, such that the daily total payment volumes for the six are the median, $75^{th}$, $90^{th}$, $95^{th}$, $99.99^{th}$ percentiles and the maximum in the sample period, respectively. Payment volume, rather than value, is the main selection criterion in choosing these six examples because the algorithm is essentially a volume-based approach. Recall that we first look for potential positions to insert barriers between parts, and the number of barriers needed is $m - 1$; so the size of $m$ plays a critical role in the proposed algorithm in terms of the number of iterations and the run time of execution.

**Table 3:** Pairs of Payment Value and Volume Chosen for 1,000 Repeated Sampling (Daily totals between two DCs in a given stream)

| Percentiles in volume | Pairwise daily volume | Pairwise daily value ($) |
|---|---|---|
| Median | 204 | 401,371,039.41 |
| $75^{th}$ | 5,393 | 80,112,015.37 |
| $90^{th}$ | 40,259 | 278,591,400.00 |
| $95^{th}$ | 98,414 | 53,866,775.14 |
| $99.99^{th}$ | 3,710,789 | 155,841,997.80 |
| Maximum | 7,888,726 | 61,592,565.34 |

If there are multiple entries containing this same magnitude of daily total volume ($m$) that corresponds to one of the targeted percentiles, the data entry with the maximum daily total value ($n$) is chosen for the given $m$ to serve as the example. The rationale is to maximize the variations in potential compositions, testing the algorithm to the maximum degree based on the given data. Variations in compositions typically increase when the difference between $n$ and $m$ becomes larger. For example, if one wants to split a total of \$2 into two payments, there is only one way of doing so (\$1 + \$1); however, if one seeks two payments out of a total of \$10,

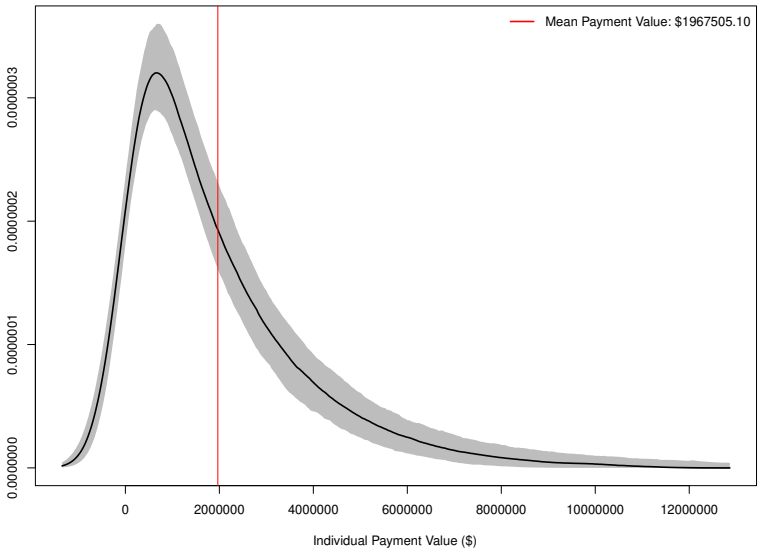
suddenly there are many possibilities: (\$2 + \$8), (\$3 + \$7) or (\$1 + \$9), etc.

For each pair of payment value ($n$) and volume ($m$) in Table 3, the algorithm was run 1,000 times to generate $1,000$ random compositions of $n$ into exactly $m$ parts. Every sample of the compositions provides an empirical distribution of individual payment values that any *one* part of the composition can take; for each of the empirical distributions, a probability density function was estimated that indicates the likelihood that any one part of the composition of $n$ will be a particular payment value $x$. Repeated sampling provides a distribution of probability densities at every level of individual payment values. These density functions are portrayed in Figure 2 for all the six simulation cases.

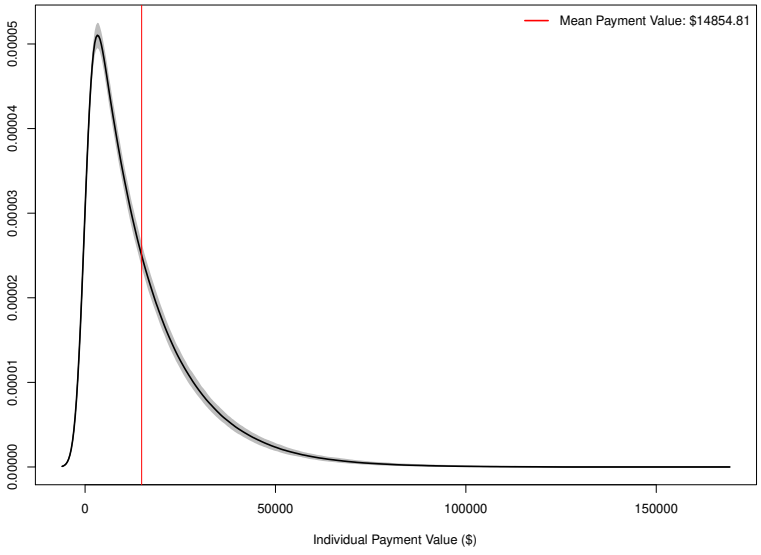
The black lines in the graphs trace out the median probabilities of every payment value in all $1,000$ densities. The grey band represents a 95% confidence interval, with the upper bound specifying the probabilities at the $97.5^{th}$ percentiles and the lower bound corresponding to the $2.5^{th}$ percentiles. These results show that, as the transaction volume increases, the confidence interval grows narrower, indicating that the probability distribution of individual payment values in ACSS becomes increasingly stable. For example, when the daily total transaction volume ($m$) is above the level of its $95^{th}$ percentile in the sample, the probability of any one part of the composition (of the daily total value) taking a given payment value $x$ is almost fixed.

In contrast, in this repeated sampling exercise, we are not controlling for the changes in daily total payment value ($n$), which suggests that the magnitude of $n$ does not seem to contribute to the stability of the composition. The red line marks the mean payment value (known from the data, i.e., daily total value divided by daily total volume) that one ACSS direct clearer sends to another in a given payment stream on a given day. The simulation results show that the probability distribution of individual payment values in ACSS is not only remarkably stable but also invariably skewed to the left of the mean payment value. This is very interesting because it defies the most common and typical presumption that the probability distribution of payment values is likely bell-shaped and centred around the mean. However, it does reflect the very nature of retail payments systems, which is that the majority of the retail transactions are small in value; occasionally, bigger-value payments move the average payment value to the right of the centre of the density.

**Figure 2:** Density of Individual Payment Values—1,000 Repeated Sampling



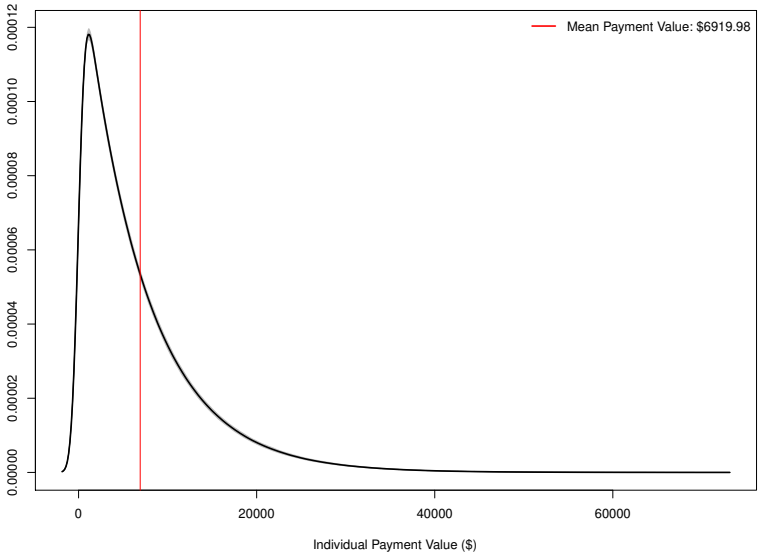

**(a)** Median Volume: 204



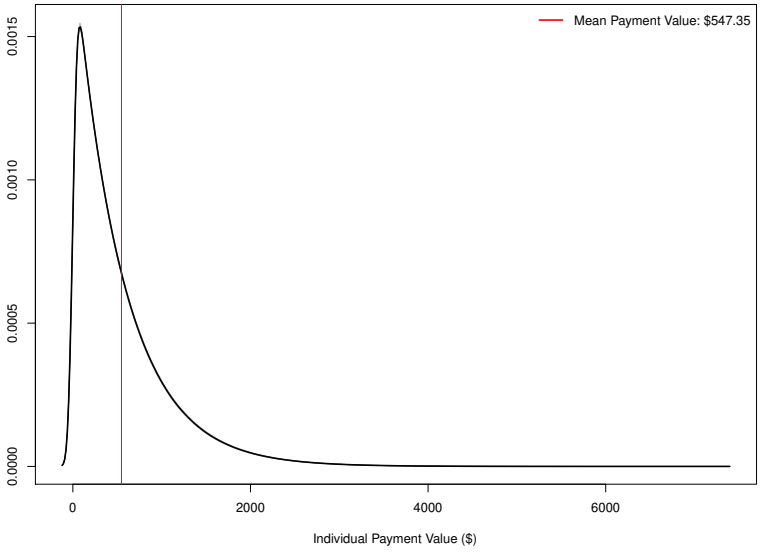

**(b)** $75^{th}$ Percentile Volume: 5,393

This graph illustrates the distribution of probability densities at every level of individual payment values, based on 1,000 compositions generated by repeated random sampling. The horizontal axis is the individual payment value in dollars. The vertical axis is the probabilities. The simulation results interestingly show that the probability distribution of payment values in ACSS is rather stable, and it becomes increasingly so as the volume of transactions expands. In addition, these densities are found to be invariably skewed to the left of the mean payment value, which reflects the nature of retail payments systems.

**Figure 2:** Density of Individual Payment Values—1, 000 Repeated Sampling



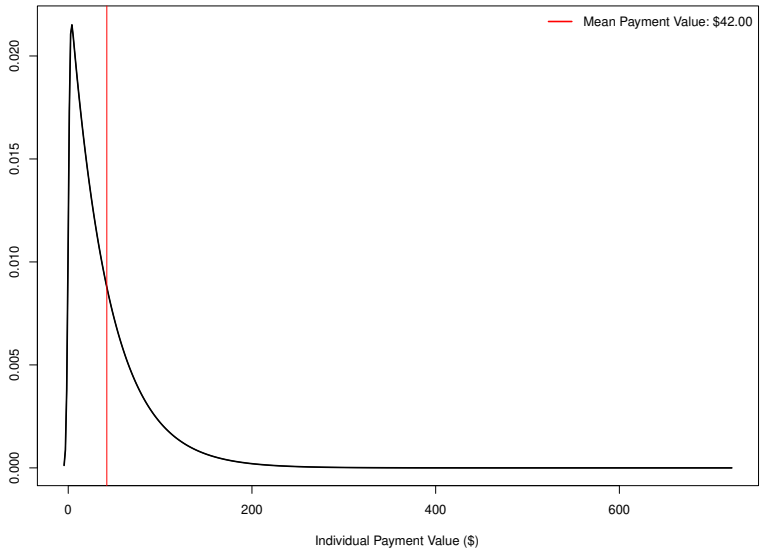

**(c)** $90^{th}$ Percentile Volume: 40,259



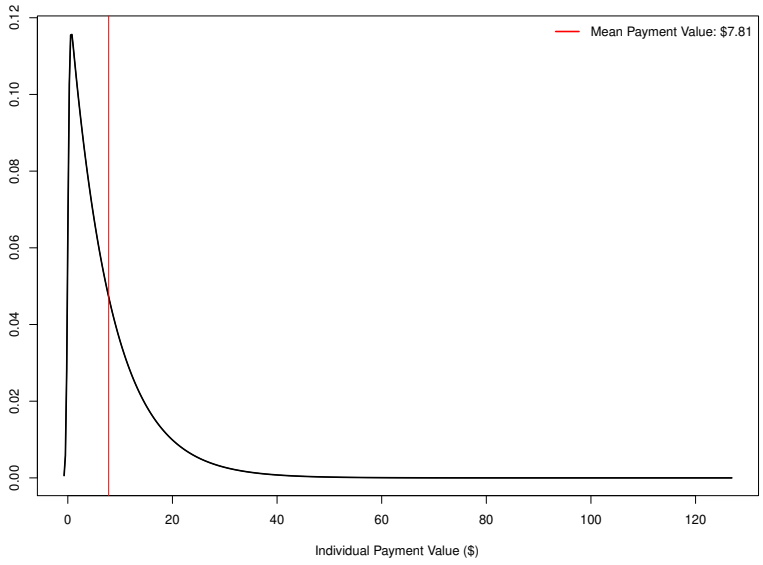

**(d)** $95^{th}$ Percentile Volume: 98,414

This graph illustrates the distribution of probability densities at every level of individual payment values, based on 1, 000 compositions generated by repeated random sampling. The horizontal axis is the individual payment value in dollars. The vertical axis is the probabilities. The simulation results interestingly show that the probability distribution of payment values in ACSS is rather stable, and it becomes increasingly so as the volume of transactions expands. In addition, these densities are found to be invariably skewed to the left of the mean payment value, which reflects the nature of retail payments systems.

**Figure 2:** Density of Individual Payment Values—1, 000 Repeated Sampling




**(e)** $99.99^{th}$ Percentile Volume: 3,710,789




**(f)** Maximum Volume: 7,888,726

This graph illustrates the distribution of probability densities at every level of individual payment values, based on 1, 000 compositions generated by repeated random sampling. The horizontal axis is the individual payment value in dollars. The vertical axis is the probabilities. The simulation results interestingly show that the probability distribution of payment values in ACSS is rather stable, and it becomes increasingly so as the volume of transactions expands. In addition, these densities are found to be invariably skewed to the left of the mean payment value, which reflects the nature of retail payments systems.

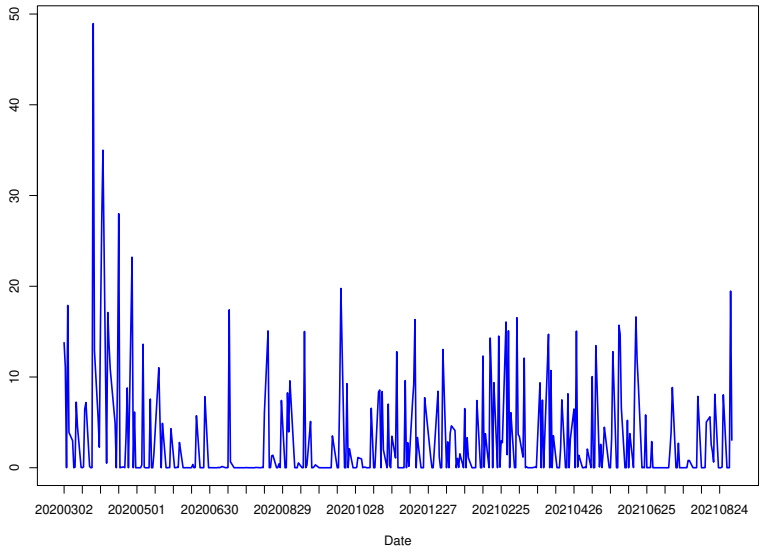
Various checks are also performed to examine how reasonable and reliable this simulated data is. First, typically in a payments system, every participant's outgoing payments are intertwined with their incoming receipts throughout the course of a day; hence, any participant's intraday maximum NDP should not exceed its total value of payments sent, which could result from a highly unlikely scenario where a financial institution sends out all its outgoing payments before receiving anything back from its fellow system participants. In this paper, net debit position is defined as, at any given point of time, the total value of payments sent by a financial institution minus the total value of payments it has received. Figure 3 (as well as Figure 7, presented in Appendix B) shows that for every 11 ACSS direct clearers (excluding the Bank of Canada), the intraday maximum NDP is well below the theoretical threshold throughout the sample, indicating that all the simulated individual transactions are interwoven properly with each other over the course of each day, as they should be in real operations.

Next, we take a closer look at every ACSS direct clearer's intraday max NDP during each hour throughout the day for every day in the data sample. Although we do not have a presumption of what the hourly maximum NDP should look like intraday, these 3-dimensional (3D) visualizations suggest that the simulation results are reasonable and plausible in the general context of any payments system.

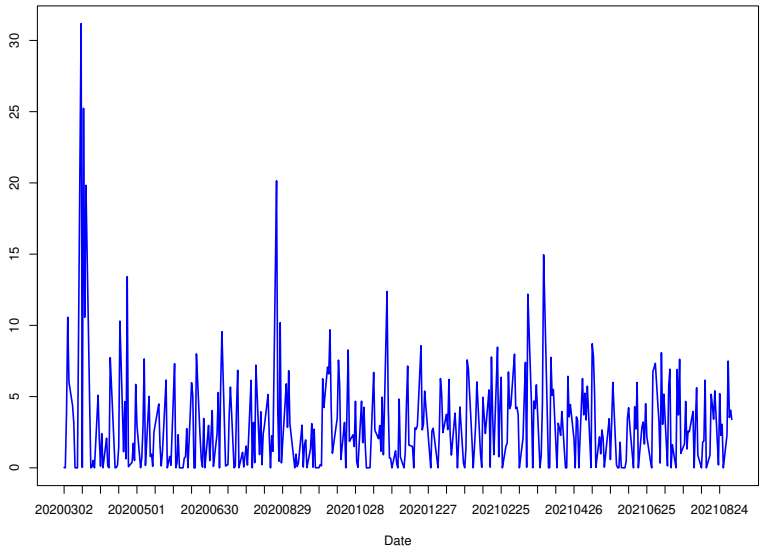
As shown in Figure 4, there are no observed abrupt swings from positive to negative funds positions, or vice versa. Each day, every participant starts with zero dollar in funds balances. At some point of time during the day, after many payment exchanges, an ACSS participant usually enters into an either positive or negative funds position, and typically this state persists throughout the remaining hours of the day. This result reflects one of the defining features of the retail payments system; i.e., on average payments are small in size compared with transactions in large value payments system, for example. In order for a positive NDP at moment $t_i$ ($NDP_{ti}$) to overturn quickly, immediately subsequent incoming payments at time $t_(i+1)$, $t_(i+2)$, and $t_(i+3)$ ... need to be sufficiently large to offset the positive NDP.

It is worth noting that this finding is independent of the structure imposed by the input data of intraday payment timing distribution; the assumption of timing patterns affects only the actual shapes of these NDP curves, e.g., how far up a net positive funds position can go at a given time of day. By construction, three

**Figure 3:** Simulated Intraday Maximum Net Debit Positions of ACSS Direct Clearers (Direct Clearers: E and I)



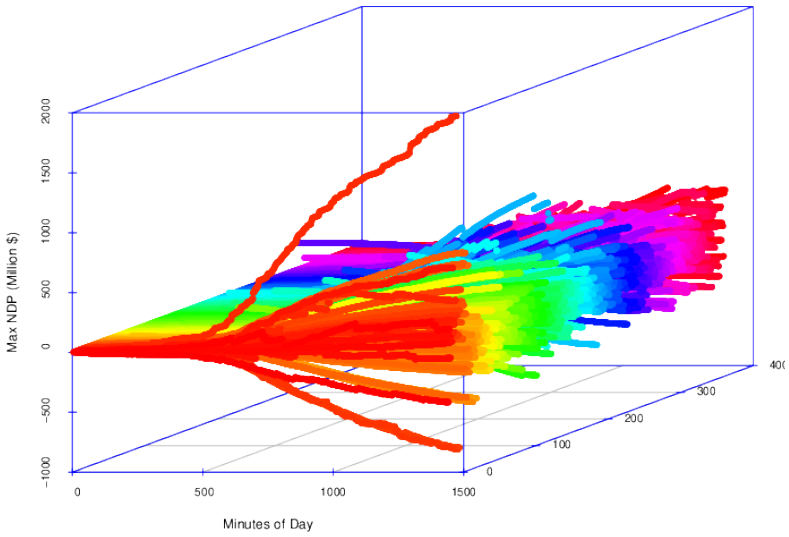

**(a)** Direct Clearer E



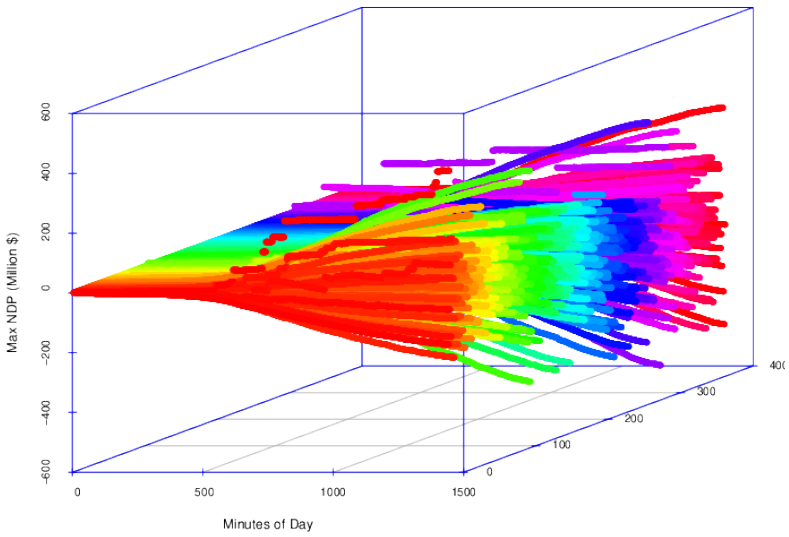

**(b)** Direct Clearer I

The intraday maximum net debit position is expressed as a percentage of the daily total payment value sent. The horizontal axis is 383 days across the sample. The vertical axis is percentage. The simulation results show that simulated transactions are interwoven properly in time throughout the day.

**Figure 4:** Simulated Intraday Maximum Net Debit Positions across All Sampled Days (Direct Clearers: D and H)




**(a)** Direct Clearer D




**(b)** Direct Clearer H

The graph provides a 3-dimensional visualization of an ACSS direct clearer's intraday maximum net debit positions over the course of day and across all the days in the sample period. The vertical axis (z-axis) is the intraday maximum net debit position (NDP) in million dollars. The horizontal axis (x-axis) shows $1,440$ minutes of a day. The y-axis is marked with all the 383 days in the sample. The simulation results show a trend of either steady increase or persistent decline in intraday maximum NDPs, which clearly reflects the fact that in retail payments systems, the majority of transactions are small in value, unlikely to trigger abrupt swings from positive to negative during the course of a day.

variables are fixed in this method: (i) the start of each line, i.e., the initial funds position at time 0 ($NDP_{t0} = 0$); (ii) the end of each line, the end-of-day funds position ($NDP_{EOD} = \sum(out_{EOD}) - \sum(in_{EOD})$, where out and in are, respectively, the cumulative sum of all payments sent and received by an ACSS participant at the end of each day); and (iii) a time window during which variations of the intraday funds positions are allowed to happen. How these intraday net funds positions vary (i.e., the total amount of "wiggling" of the lines) is intrinsically controlled by payment volume, which comes directly from actual data.

The 3D illustrations for Direct Clearer D and H are presented here in the main text (the same results for the remaining nine ACSS direct clearers are included in in Figure 8, Appendix C).

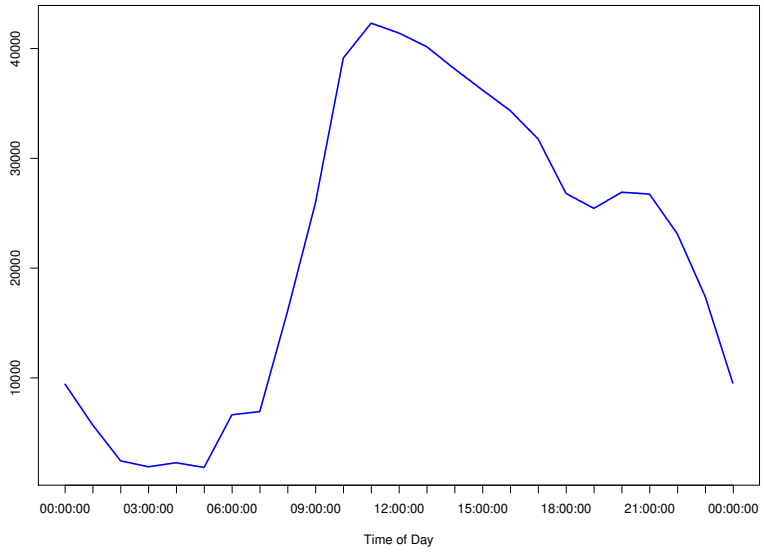
Lastly, Figure 5 illustrates the timing distribution of payment volume over the course of day that is generated on more than 12 billion simulated ACSS intraday transactions. For each 1-minute time interval of the 24 hours each day, we compute the total number of transactions exchanged during that minute and then take the average across all 383 days in the data sample. The graph shows a pattern that is almost identical to the intraday timing distribution of the U.K. Faster Payments System, which is used as an input for the simulation. This indicates that all the steps taken in the algorithm to apply a structure of timing distribution in the process of generating a time stamp for every individual payment are implemented properly.

This is somewhat to be expected; however, the result shows that the randomness introduced in the process of generating the time stamps is not enough to cause the simulated timing pattern to deviate significantly from the underlying shape. The results also suggest that the proposed algorithm is highly flexible and able to incorporate any payment timing pattern into simulation, as long as such data is available.

# 5 Concluding Remarks

In the current era of information technology advancing at lightning speed, data has proven to be more important than ever in businesses, academic research, and activities in virtually every aspect of people's lives. However, data is not always available—for example, because proprietary information cannot be disclosed to third parties or because records do not exist. Situations often arise where aggregated

**Figure 5:** Intraday Timing Pattern—Simulated ACSS Transactions




This graph illustrates the timing distribution of payment volume in Canada's ACSS, based on simulated intraday transactions during the period between March 02, 2020 and September 03, 2021. The timing pattern presented here is almost identical to that of the U.K. Faster Payments system, which is used as an input in the simulation. The horizontal axis is the time of day, with a resolution of 1 minute. The vertical axis shows the average number of payments made during each minute duration on a typical day.

data is easy to find but more detailed information is unobtainable. This is because detailed data frequently contains confidential information in relation to, for example, customers' identities, personal finances or registered trade secrets. The Canadian ACSS, which began operations in 1984, is a good example. The most comprehensive data currently available for this system is aggregate transaction value and volume on a daily basis.

This paper proposes a simple yet unique and useful method to simulate detailed data based on aggregate statistics. Based on integer composition in number theory, the proposed approach breaks down the daily total transaction value in ACSS into a set of intraday individual payments, the number of which is equal to the daily total volume. The suggested algorithm is also able to assign a time stamp to each individual payment according to a known intraday payment timing pattern to create a wholesome payment profile for the system. The simulation results revealed that the distribution of individual payment values is remarkably stable. Specifically, by conducting $1,000$ repeated random sampling on compositions of the same pair of total value and volume, the analysis demonstrated that as the transaction volume

grows larger, the probability of any one part of the composition taking a given payment value $x$ is almost fixed. Moreover, the study showed that the distribution of payment values in a retail payments system like ACSS is invariably skewed to the left of the mean, which is contrary to the common expectations that it might be bell-shaped and centre around the mean. Overall, this analysis strongly suggests that the proposed simulation method is a viable and powerful tool to disaggregate data when the summary statistics are the only information at hand.

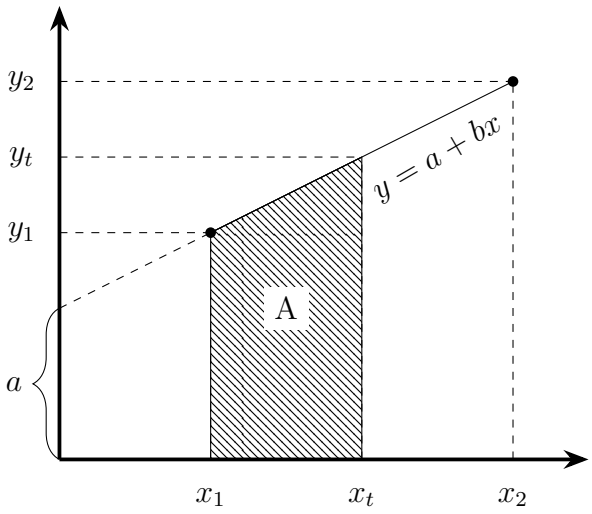



**Figure 6:** Piecewise Linear Function

# A Deriving the Piece-wise Quadratic CDF and Its Inverse

Illustrated in Figure 6 is a linear function $y = a + bx$. $x_1$, $x_2$, $y_1$ and $y_2$ are the known data points. This line segment represents any portion of the graph (Figure 1—the hourly timing distribution of payment volume of the U.K. Faster system), between any two adjacent hours, e.g. between the $11^{th}$ and $12^{th}$ hour.

Point $(x_t, y_t)$ represents one of the many missing elements in data that could potentially provide the time for payments that are settled on a particular second during any given hour range. However, we can construct these missing points using linear interpolation. For instance, in this illustration, $x_t$ represents one of the $3,600$ seconds during the one-hour period between $x_1$ and $x_2$. Let us suppose the area under the line segment, bounded by $x_1$, $x_t$, $y_1$ and $y_t$, is $A$.

Given $y = a + bx$, we know that

$$a = y_1 - bx_1$$

and

$$b = \frac{y_2 - y_1}{x_2 - x_1}.$$

Therefore, the area $A$ can be computed as follows:

$$
\begin{aligned}
A &= \int_{x_1}^{x_t} \big(y_1 - bx_1 + bx\big)dx \\
&= \left\{ \frac{bx^2}{2} + \big(y_1 - bx_1\big)x \right\}\Bigg|_{x_1}^{x_t} \\
&= \frac{b}{2}x_t^2 + \big(y_1 - bx_1\big)x_t + \left( \frac{bx_1^2}{2} - x_1 y_1 \right)
\end{aligned}
$$

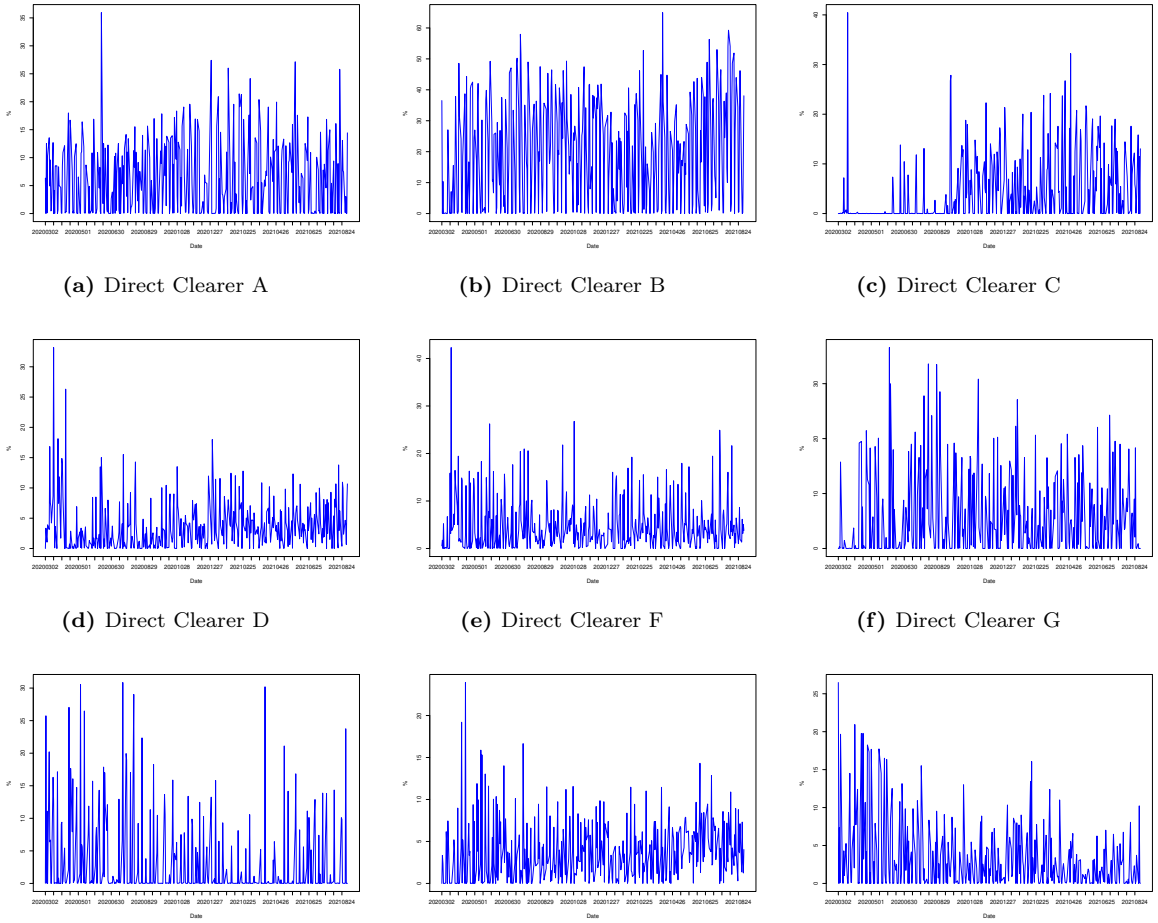
Solving for $x_t$:

$$
x_t = x_1 - \frac{y_1}{b} \pm \frac{\sqrt{y_1^2 + 2bA}}{b} \tag{1}
$$

Equation 1 provides a one-to-one mapping between a defined area under a line segment and the independent variable $x$. In this study, the area under the curve of the timing distribution of payment volume, up to and including $A$ (i.e., between midnight and any given point of time) is the CDF of that time.

# B Simulated Intraday Maximum NDPs

**Figure 7:** Simulated Intraday Maximum Net Debit Positions of ACSS Direct Clearers (Direct Clearers: A, B, C, D, F, G, H, J and K)



**(a)** Direct Clearer A **(b)** Direct Clearer B **(c)** Direct Clearer C

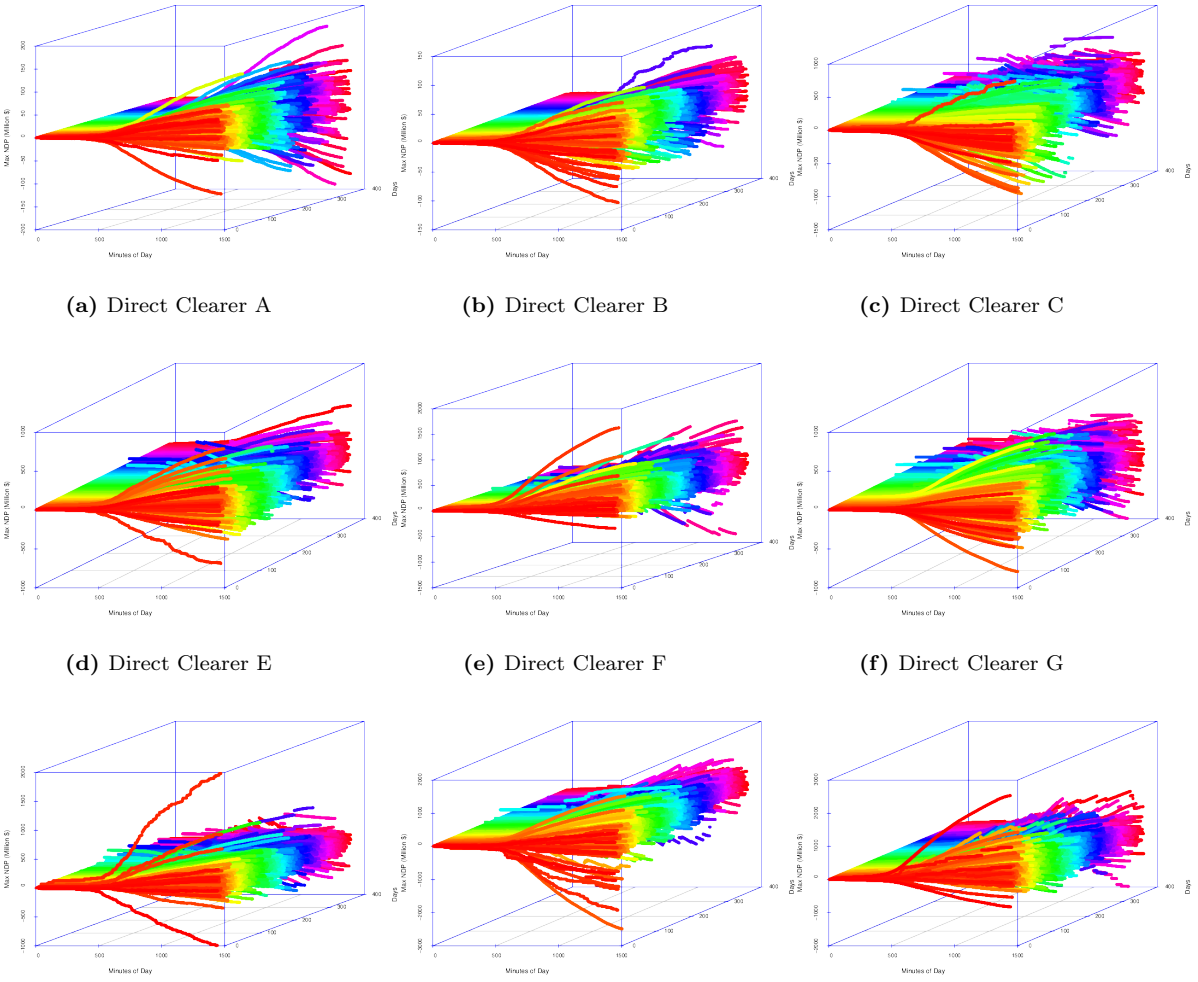
**(d)** Direct Clearer D **(e)** Direct Clearer F **(f)** Direct Clearer G

**(g)** Direct Clearer H **(h)** Direct Clearer J **(i)** Direct Clearer K

The intraday maximum net debit position is expressed as a percentage of the daily total payment value sent. The horizontal axis is 383 days across the sample. The vertical axis is percentage. The simulation results show that simulated transactions are interwoven properly in time throughout the day.

# C Intraday Max NDPs across Days

**Figure 8:** Simulated Intraday Maximum Net Debit Positions across All Sampled Days (Direct Clearers: A, B, C, E, F, G, I, J and K)



**(a)** Direct Clearer A **(b)** Direct Clearer B **(c)** Direct Clearer C

**(d)** Direct Clearer E **(e)** Direct Clearer F **(f)** Direct Clearer G

**(g)** Direct Clearer I **(h)** Direct Clearer J **(i)** Direct Clearer K

The graph provides a 3-dimensional visulization of an ACSS dicrect clearer's intraday maximum net debit positions over the course of day and across all the days in the sample period. The vertical axis (z-axis) is the intraday maximum NDP in million dollars. The horizontal axis (x-axis) shows $1,440$ minutes of a day. The y-axis are marked with all the 383 days in the sample. The simulation results show a trend of either steady increase or persistent decline in intraday maximum NDPs, which very well reflects the fact that in retail payments systems, the majority od transactions are small in value, unlikely to trigger abrupt swings from positive to negative during the course of day.

# References


Accenture. Accenture payment services immediate payments: Seizing the customer opportunity why real-time transfer. https://www.accenture.com/t00010101t000000__w__/mx-es/_acnmedia/accenture/conversion-assets/nonsecureclients/documents/pdf/1/accenture-immediate-payments.pdf.

Andrews, G. (1998). *The Theory of Partitions*. Cambridge mathematical library. Cambridge University Press.

Arjani, N. and D. McVanel (2006, March). A primer on Canada's Large Value Transfer System.

Bayer, R. (2004). Symmetric binary b-trees: Data structure and maintenance algorithms. *Acta Informatica 1*, 290–306.

Cohen, D. I. (1981). Pie-sums: A combinatorial tool for partition theory. *Journal of Combinatorial Theory, Series A 31*(3), 223–236.

Glaisher, J. (1883). A theorem in partitions. *Messenger of Math. 12*(142), 158–170.

Saiz, H. P., S. Untawala, and G. Xerri (2018, January). A calibrated model of intraday settlement. *Bank of Canada Staff Discussion Paper* (2018-03).