Exploring probability distributions for bivariate temporal granularities

Abstract

Recent advances in technology greatly facilitates recording and storing data at much finer temporal scales than was previously possible. As the frequency of time-oriented data increases, the number of questions about the observed variable that need to be addressed by visual representation also increases. We propose some new tools to explore this type of data, which deconstruct time in many different ways. There are several classes of time deconstructions including linear time granularities, circular time granularities and aperiodic calendar categorizations. Linear time granularities respect the linear progression of time such as hours, days, weeks and months. Circular time granularities accommodate periodicities in time such as hour of the day, and day of the week. Aperiodic calendar categorizations are neither linear nor circular, such as day of the month or public holidays.

The hierarchical structure of linear granularities creates a natural nested ordering resulting in single-order-up and multiple-order-up granularities. For example, hour of the week and second of the hour are both multiple-order-up, while hour of the day and second of the minute are single-order-up.

Visualizing data across granularities which are either single-order-up or multiple-order-up or periodic/aperiodic helps us to understand periodicities, pattern and anomalies in the data. Because of the large volume of data available, using displays of probability distributions conditional on one or more granularities is a potentially useful approach. This work provides tools for creating granularities and exploring the associated time series within the tidy workflow, so that probability distributions can be examined using the range of graphics available in (Wickham 2016).

Contents

1 Introduction

Temporal data are available at various resolution depending on the context. Social and economic data like GDP are often collected and reported at coarser temporal scales like monthly, quarterly or annually. With recent advancement in technology, more and more data are recorded and stored at much finer temporal scales. Energy consumption is collected every half an hour, while energy supply is collected every minute

and web search data might be collected at every second. As the frequency of data increases, the number of questions about periodicity of the observed variable also increases. For example, data collected at an hourly scale can be analyzed using coarser temporal scales like days, months or quarters. This approach requires deconstructing time in various possible ways. Calendar-based graphics[] can unpack the temporal variable, at different resolutions, to digest multiple seasonalities, and special events. They mainly pick out patterns in the weekly and monthly structure well and are capable of checking the weekends or special days. Any sub-daily resolution temporal data can also be displayed using this type of faceting [reference trellis plot] with days of the week, month of the year and another sub-daily deconstruction of time.

But calendar effects are not restricted to conventional day-of-week, month-of-year ways of deconstructing time. A temporal granularity which results from such a deconstruction may be intuitively described as a sequence of time granules, each one consisting of a set of time instants[]. There can be several classes of time deconstructions including linear time granularities, circular time granularities and aperiodic calendar categorizations. Linear time granularities respect the linear progression of time such as hours, days, weeks and months. Circular time granularities accommodate periodicities in time such as hour of the day, and day of the week. Aperiodic calendar categorizations are neither linear nor circular, such as day of the month or public holidays. Also, the hierarchical structure of time creates a natural nested ordering. For example, hours are nested within days, days within weeks, weeks within months, and so on. Hence, we can construct single-order-up granularities like second of the minute or multiple-order-up granularities like second of the hour.

It is important to be able to navigate through all of these temporal granularities to have multiple perspectives on the observed data. This idea aligns with the notion of EDA (Tukey 1977) which emphasizes the use of multiple perspectives on data to help formulate hypotheses before proceeding to hypothesis testing.

The motivation for this work comes from the desire to provide methods to better understand large quantities of measurements on energy usage reported by smart meters in household across Australia, and indeed many parts of the world. Smart meters currently provide half-hourly use in kWh for each household, from the time that they were installed, some as early as 2012. Households are distributed geographically, and have different demographic properties such as the existence of solar panels, central heating or air conditioning. The behavioral patterns in households vary substantially, for example, some families use a dryer for their clothes while others hang them on a line, and some households might consist of night owls, while others are morning larks.

It is common to see aggregates of usage across households, total kWh used each half hour by state, for example, because energy companies need to understand maximum loads that they will have to plan ahead to accommodate. But studying overall energy use hides the distributions of usage at finer scales, and making it more difficult to find solutions to improve energy efficiency.

We propose that the analysis of probability distributions of smart meter data at finer or coarser scales can be benefited from the approach of Exploratory Data Analysis (EDA). EDA calls for utilizing visualization and transformation to explore data systematically. It is a process of generating hypothesis, testing them and consequently refining them through investigations.

This paper utilizes the nestedness of time granularities to obtain multiple-order-up granularities from single-order-up ones.

Finally, visualizing data across single/multiple order-up granularities help us to understand periodicities, pattern and anomalies in the data. Because of the large volume of data available, using displays of probability distributions conditional on one or more granularities is a potentially useful approach. However, this approach can lead to a myriad of choices all of which are not useful. Analysts are expected to iteratively visualize these choices for exploring possible patterns in the data. But too many choices might leave him bewildered.

This work provides tools for systematically exploring bivariate granularities within the tidy workflow through proper study of what can be considered a prospective graphic for exploration. Pairs of granularities are categorized as either a *harmony* or *clash*, where harmonies are pairs of granularities that aid exploratory data analysis, and clashes are pairs that are incompatible with each other for exploratory analysis. Probability distributions can be examined using the range of graphics available in the ggplot2 package.

In particular, this work provides the following tools.

- Functions to create multiple-order-up time granularities. This is an extension to the lubridate package, which allows for the creation of some calendar categorizations, usually single-order-up.
- Checks on the feasibility of creating plots or drawing inferences from two granularities together. Pairs of granularities can be categorized as either a *harmony* or *clash*, where harmonies are pairs of granularities that aid exploratory data analysis, and clashes are pairs that are incompatible with each other for exploratory analysis.

2 Definitions of time granularities

Often we partition time into months, weeks or days to relate it to data. Such discrete abstractions of time can be thought of as time granularities (Aigner et al. 2011). Examples of time abstractions may also include day-of-week, time-of-day, week-of-year, day-of-month, month-of-year, working day/non-working day, etc which are useful to represent different periodicities in the data.

2.1 Arrangement: Linear vs. Circular vs. aperiodic

The arrangement of the time domain can result in deconstructing time in linear, circular or nearly circular ways. Time granularities are **linear** if they respect the linear progression of time. Examples include hours, days, weeks and months. However, periodicity is very common in all kinds of data. Time granularities, which are constructed by using two linear granularities, can be utilized to explain such periodicities. The mappings between linear granularities can be regular or irregular. For example, a regular mapping exists between minutes and hours, where 60 minutes always add up to 1 hour. On contrary, an irregular mapping exists between days and months, since one might can have days ranging from 28 to 31. Hence, time granularities can also be **circular** or **aperiodic** depending on if the mapping of the linear granularities is regular or irregular. Examples of circular time granularities include hour of the day, and day of the week, whereas examples for aperiodic time granularities can be day of the month or public holidays.

Providing a formalism to some of these abstractions is important to model a time series across differently grained temporal domains.

2.1.1 Linear

There has been several attempts to provide the framework for formally characterizing time-granularities and identifying their structural properties, relationships and symbolic representations. One of the first attempts occur in (Bettini et al. 1998) with the help of the following definitions:

Definition: A **time domain** is a pair $(T; \leq)$ where T is a non-empty set of time instants and $\leq$ is a total order on T.

A time domain can be **discrete** (if there is unique predecessor and successor for every element except for the first and last one in the time domain), or it can be **dense** (if it is an infinite set). A time domain is assumed to be discrete for the purpose of our discussion.

Definition: A linear **granularity** is a mapping G from the integers (the index set) to subsets of the time domain such that:

(C1) if $i < j$ and $G(i)$ and $G(j)$ are non-empty, then each element of $G(i)$ is less than all elements of $G(j)$, and
(C2) if $i < k < j$ and $G(i)$ and $G(j)$ are non-empty, then $G(k)$ is non-empty.

Definition: Each non-empty subset $G(i)$ is called a **granule**, where i is one of the indexes and G is a linear granularity.

The first condition implies that the granules in a linear granularity are non-overlapping and their index order is same as time order. Figure 1 shows the implication of this condition. If we consider the bottom linear

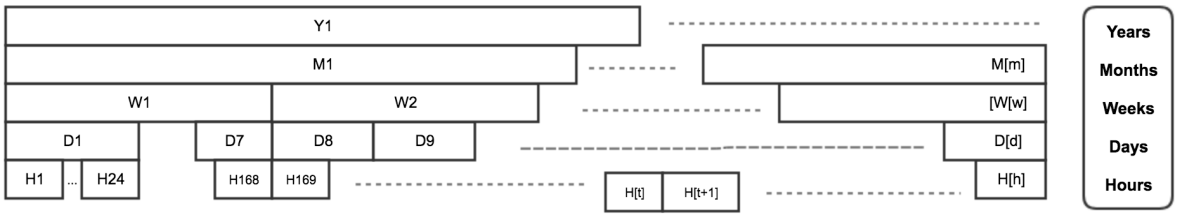


Figure 1: The time domain distributed as linear granularities

granularity (Aigner et al. 2011) as hourly and the entire horizon has T hours then it will have $\lfloor T/24 \rfloor$ days, $\lfloor T/(24*7) \rfloor$ weeks and so on.

The definitions and rules for linear granularities are inadequate to reflect periodicities in time, like weekly, monthly or yearly seasonality. Hence, there is a need to define circular time granularities in a different approach.

2.1.2 Circular

Relationship between two linear granularities can be expressed in numerous ways. **Definition: Groups Into** : A linear granularity G groups into a linear granularity H, denoted

$$G \trianglelefteq H,$$

if for each index j there exists a (possibly infinite) subset S of the integers such that

$$H(j) = \bigcup_{i \in S} G(i) \tag{1}$$

According to this definition,

$$G \trianglelefteq H,$$

if each granule H (i) is the union of some granules of G. For example,

$$Days \trianglelefteq Months,$$

since a week is composed of 7 days. This relationship, however, is not sufficient to fully describe a granularity in terms of another one. For example, it is clear that

$$Days \trianglelefteq Months,$$

, since each month is a grouping of a number of days, however it is also a periodical grouping. If leap year is ignored, the periodicity of the grouping is 1 year, since each month would be defined as the grouping of the same number of days (31, 28, or 30, depending on the month) every year. Considering leap years and all their exceptions, the period becomes 400 years, but the grouping is always periodic. In order to define a granularity in terms of another granularity including the notion of periodicity, a particular case of the groups into relationship is considered.

Definition: Periodical A granularity H is periodical with respect to a granularity G if (1)

$$G \trianglelefteq H,$$

, and (2) there exist $R, P \epsilon Z+$, where R is less than the number of granules of H, such that for all $i \epsilon Z$, if $H(i) = \bigcup_{j \in S} G(j)$ and $H(i+R) \neq \phi$ then $H(i+R) = \bigcup_{j \in S} G(j+P)$.

A granularity H which is periodical with respect to G is specified by: (i) the R sets of indexes of G S0,. . . ,SR-1 describing the granules of H within one period; (ii) the value of P; (iii) the indexes of first and last granules in H, if their value is not infinite. Then, if S0,. . . ,SR-1 are the sets of indexes of G describing H (0), . . . , H (R - 1), respectively, then the description of an arbitrary granule H (j) is given by: $\bigcup_{i \in S_j} G(P * \lfloor j/R \rfloor + i)$.

This formula can be explained observing that H (j) is either one of H (0), . . . , H (R - 1) or, for the periodicity of H, is one of these granules "shifted" ahead or behind on the time line of a finite number of periods. If $H(j) \epsilon H(0), ..., H(R-1)$, then the formula 'j mod R' defines the index (among those in {0,. . . ,R - 1}) of the granule that must be shifted to obtain H (j).The number of periods each granule of G composing H (j mod R) should be shifted is given by $\lfloor j/R \rfloor$.

Many common granularities are in this kind of relationship, for example, Years is periodical with respect to both Days and Months.

As mentioned in the introduction, we are also interested in granularities which are periodical with respect to other granularities, except for a finite number of spans of time where they behave in an anomalous way.

Definition: Quasi Periodical A granularity H is quasi-periodical with respect to a granularity G if (1)

$$G \trianglelefteq H,$$

, and (2) there exist a set of intervals E1,. . . ,Ez (the granularity exceptions) and positive integers R,P, where R is less than the minimum of the number of granules of H between any 2 exceptions, such that for all $i \epsilon Z$, if $H(i) = \bigcup_{k \in [0,k]} G(j_r)$ and $H(i+R) \neq \phi$ and $i + R < min(E)$, where E is the closest existing exception after $H(i)^2$, then $H(i+R) = \bigcup_{k \in [0,k]} G(j_r + P)$.

Intuitively, the definition requires that all granules of H within the span of time between two exceptions have the same periodical behavior, characterized by R and P.

Most practical problems seem to require only a granularity system containing a set of time granularities which are all periodical (or quasi-periodical) with respect to a basic granularity.The assumption is there should be one bottom granularity and all the other granularities are generated from the bottom granularity by calendar algebraic operations.

The formalism of circular granularities is attempted through the tsibble (Wang, Cook, and Hyndman 2019) framework of organizing temporal data. Suppose we have a tsibble with a time index in one column and keys and variables in other columns, a time domain, as defined by Bettini, is essentially a mapping of the index set to the time index.

A linear granularity is a mapping of the index set to subsets of the time domain. For example, if the time index is days, then a linear granularity might be weeks, months or years. Circular granularity, joining two linear granularities, therefore is a mapping between these two subsets of time domain. We use modular arithmetic to define circular granularity and use the following definitions:

Definition: Equivalence class Let $m \in N \backslash 0$. For any $a \in Z$ (set of integers), $[a]$ is defined as the equivalence class to which a belongs if $[a] = \{b \in Z | a \equiv (b \mod m)\}$.

The set of all equivalence classes of the integers for a modulus m is called the ring of integers modulo m, denoted by Z_m. Thus $Z_m = \{[0], [1], ..., [m-1]\}$. However, we often write $Z_m = \{0, 1, ..., (m-1)\}$, which is the set of integers modulo m.

Definition: A **circular granularity** C, joining two linear granularity with a modular period m is defined to be a mapping from the integers Z (Index Set) to Z_m, such that

$C(s) = (s \mod m)$ for $s \in Z$.

For example, suppose C is a circular granularity denoting hour-of-day and we have hourly data for 100 hours. The modular period $m = 24$, since each day consists of 24 hours and C is a mapping from $1, 2, \ldots, 100$ to $0, 1, 2, \ldots, 23$ such that $C(s) = s \mod 24$ for $s \in 1, 2, \ldots, 100$.

2.1.3 Aperiodic

Definition: An **Aperiodic circular granularity** can not be defined using modular arithmetic in a similar fashion. The modulus for these type of calendar categorizations are not constant due to unequal length of some linear granularities. For example, please refer to the table below:

HOM:	$C_3(s) = s \mod 720$ (approximately)	$n_3 = 744$
HOY:	$C_4(s) = s \mod 8760$ (except for leap years)	$n_4 = 8784$
DOM:	$C_6(s) = \lfloor s/24 \rfloor \mod 30$ (approximately)	$n_6 = 31$
DOY:	$C_7(s) = \lfloor s/24 \rfloor \mod 365$ (except for leap years)	$n_7 = 366$
WOM:	$C_8(s) = \lfloor s/168 \rfloor \mod 4$ (approximately)	$n_8 = 5$
WOY:	$C_9(s) = \lfloor s/168 \rfloor \mod 52$ (approximately)	$n_9 = 53$
MOY:	$C_{10}(s) = \lfloor s/720 \rfloor \mod 12$ (approximately)	$n_{10} = 12$

Table 1: Illustrative aperiodic circular granularities with time index in hours

To consider the exhaustive set of temporal regularities that might exist in the data, we can also define temporal granularities based on the hierarchical structure of a calendar.

2.2 Order: Single vs. Multiple

The hierarchical structure of time creates a natural nested ordering which can produce **single-order-up** or **multiple-order-up** granularities. We shall use the notion of a hierarchy table to define them.

Consider a **hierarchy table** to be consisting of two columns:

- The first column represents the (temporal) units in ascending order of hierarchy. Any two units, which are essentially the linear granularities, can join together to form a circular/aperiodic granularity.
- The second column represents the relationship between subsequent (temporal) units. For periodic granularities, this could be represented by a constant, whereas, for a aperiodic one, there is no single constant which can be used to represent their relationship.

and, **order**, is defined as the position of the units in the hierarchical table.

We refer to granularities which are nested within multiple levels as **multiple-order-up** granularities and those concering a single level as **single-order-up** granularities. Let us look at few calendars to see examples of single and multiple order-up granularities.

So far we have used the Gregorian calendar as it is the most widely used calendar. But it is far from being the only one. All calendars fall under three types - solar, lunar or lunisolar/solilunar but the day is the basic unit of time underlying all calendars. Various calendars, however, use different conventions to structure days into larger units: weeks, months, years and cycle of years. Any civil day is divided into 24 hours and each hour into 60 minutes and each minute into 60 seconds. There are exceptions where in London, for example, length of each "hour" varies from about 39 minutes in December to 83 minutes in June. The French revolutionary calendar divided each day into 10 "hours", each "hour" into 100 "minutes" and each "minute" into 100 "seconds". Nevertheless, for any calendar a hierarchy can be defined. For example, in Mayan calendar, one day was referred to as 1 kin and the calendar was structured as follows:

- 1 kin = 1 day
- 1 uinal = 20 kin
- 1 tun = 18 uinal
- 1 katun = 20 tun
- 1 baktun = 20 katun

Thus, the hierarchy table for the Mayan calendar would look like the following:

Units	Conversion factor

Units	Conversion factor
kin	20
uinal	18
tun	20
katun	20
baktun	1

Examples of multiple-order-up granularities can be kin of the tun or kin of the baktun whereas examples of single-order-up granularities may include kin of the uinal, uinal of the tun etc.

In the next section, we discuss the computation of any single-order-up and multiple-order-up granularities in a periodic and aperiodic set up.

3 Computation of time granularities

An algebraic representation for time granularities, which we call the calendar algebra. It is assumed that there exists a "bottom" granularity known to the system. Calendar algebra operations are designed to generate new granularities from the bottom one or recursively, from those already generated. Thus, the relationship between the operand(s) and the resulting granularities are encoded in the operations.

The calendar algebra consists of two kinds of operations: grouping-oriented and granule-oriented operations. The grouping-oriented operations combine certain granules of a granularity together to form the granules of the new granularity, while the granule-oriented operations do not change the granules of a granularity, but rather make choices of which granules should remain in the new granularity.

- **grouping-oriented** : Let G be a full-integer labeled granularity, and m a positive integer. The grouping operation $Group_m(G)$ generates a new granularity G, by partitioning the granules of G into m-granule groups and making each group a granule of the resulting granularity. More precisely, $G = Group_m(G)$ is the full-integer labeled granularity such that for each integer i, $G(i) = \bigcup_{j=(i-1)m+1}^{im} G(j)$. Granularity week is defined by $week = Group_7(day)$. Note that, we assume that the day labeled 1 starts a week.

-**granule-oriented** : Let G1, G2 be full-integer labeled granularities, and l, k, m integers, where G2 partitions G1, and $1 \leq l \leq m$. The altering-tick operation $Alter^m_{l,k}(G2, G1)$ generates a new full-integer labeled granularity by periodically expanding or shrinking granules of G1 in terms of granules of G2. Since G2 partitions G1, each granule of G1 consists of some contiguous granules of G2. The granules of G1 can be partitioned into m-granule groups such that G1(1) to G1(m) are in one group, G1(m + 1) to G1(2m) are in the following group, and so on. The altering-tick operation modifies the granules of G1 so that the lth granule of each group has |k| additional (or fewer when k < 0) granules of G2. For example, if G1 represents 30-day groups (i.e., G1 = Group_30(day)) and we want to add a day (i.e., k = 1) to the 5th one in each group of 12 months (i.e., l = 5 and m = 12), we may have $Alter^{12}_{5,1}(day, G1)$. Intuitively, this will make May have 31 days.

Now, let us dicuss the computation of circular granularities.

3.1 Single-order-up granularities

Suppose, z denotes the index of the tsibble, x,y are two units in the hierarchy table with $order(x) < order(y)$. Let $f(x, y)$ denotes the accessor function for computing the single-order-up granularity relating x and y and $c(x, y)$ is a constant which relates x and y.

Then $f(x, y)$ can be computed using modular arithmetic as follows:

$$f(x,y) = \lfloor z/c(z,x) \rfloor \mod c(x,y)$$

where $y = x + 1$

See table[] for an illustration of computing single order up granularities for Mayan calendar.

Single-order-up granularities $kin\_uinal := z \mod 20$
$uinal\_tun := \lfloor z/20 \rfloor \mod 18$
$tun\_katun := \lfloor z/20 * 18 \rfloor \mod 20$
$katun\_baktun := \lfloor z/20 * 18 * 20 \rfloor \mod 20$

Table 3: Illustrative single-order-up granularities for Mayan calendar with kin as the index

3.1.0.1 Aperiodic single-order-up granularities

Aperiodic single-order-up granularities refers to the granularities which are formed with units that does not repeat its values in regular intervals or periods. In Gregorian calendar, examples may include days of the month where each month may consists of 28, 29, 30 or 31 days. So there is no single number that can be used for converting days to months. We cannot compute the aperiodic single-order-up granularities using the index of the tsibble and modular arithmetic like in the periodic case using just the hierarchy table and relationships of units.

3.2 Multiple order-up granularities

Computation of multiple-order-up granularities will differ if the units in the hierarchy table are periodic or aperiodic. We split the method of computation into two cases - one concerining all periodic single-order-up granularities and the second with mixed single-order-up granularities.

3.2.1 All single-order-up granularities are periodic

z is the index set of the tsibble and x, y are two linear granularities with $order(x) < order(y)$. Also, $f(x,y)$ denotes the accessor function for computing circular granularity which relates x and y and $c(x,y)$ is a constant which relates x and y. It is easy to see that for $order(x+1) = order(y)$, the function is same as the single-order-up granularities.

Then, the accessor function f can be used recursively to obtain any multiple-order-up granularities as follows:

$$\begin{aligned}
f(x,y) &= f(x,x+1) + c(x,x+1)(f(x+1,y) - 1) \\
&= f(x,x+1) + c(x,x+1)[f(x+1,x+2) + c(x+1,x+2)(f(x+2,y) - 1) - 1] \\
&= f(x,x+1) + c(x,x+1)(f(x+1,x+2) - 1) + c(x,x+1)c(x+1,x+2)(f(x+2,y) - 1) \\
&= f(x,x+1) + c(x,x+1)(f(x+1,x+2) - 1) + c(x,x+2)(f(x+2,y) - 1) \\
&\vdots \\
&= \sum_{i=0}^{order(y)-order(x)-1} c(x,x+i)(f(x+i,x+i+1) - 1)
\end{aligned} \tag{2}$$

Let us use the equation to compute the multiple-order-up granularity uinal_katun for Mayan calendar, which is periodic.

From Equation(1), we have

$$\begin{aligned}
f(uinal, baktun) &= f(uinal, tun) + c(uinal, tun) f(tun, katun) + c(uinal, katun) f(katun, baktun) \\
&= \lfloor z/20 \rfloor \mod 18 + 20 * \lfloor z/20 * 18 \rfloor \mod 20 + 20 * 18 * 20 \lfloor z/20 * 18 * 20 \rfloor \mod 20
\end{aligned} \tag{3}$$

(using equations 3)

3.2.2 Mixed single-order-up granularities - circular or aperiodic

Let us turn to Gregorian calendar for addressing this case.

Suppose we have a hierarchy table for Gregorian calendar as **??**. Since months consists of unequal number of days, any temporal unit which is higher in order than months will also have unequal number of days. This is an example of a hierarchy table which has both periodic and aperiodic single-order-up granularities. The single-order-up granularity day_month is aperiodic. Any single-order-up granularities which are formed by units below days are periodic. Similarly, all single-order-up granularities which are formed using units whose orders are higher than months are also periodic.

linear granularities	Conversion factor	format
minute	60	markdown
hour	24	markdown
day	aperiodic	markdown
month	3	markdown
quarter	4	markdown
year	1	markdown

There can be three scenarios for obtaining calendar categorization here: - granularities consisting of two units whose orders are less than day
- granularities consisting of two units whose orders are more than month
- granularities consisting of one unit with order at most day and another with order at least month

The calendar categorization resulting from the first two cases are periodic and has been handled in the earlier section.

The calendar categorization resulting from the last case are aperiodic. Examples might include day of the quarter or hour of the month. In this section, we will see how to obtain aperiodic circular granularities of these types.

$$f_{(hour,month)}(z) = f_{(hour,day)}(z) + c(hour, day) f_{(day,month)}(z) \tag{4}$$

Here, the first part of the equation is a single-order-up granularity which can be obtained using last section. The second part is not single-order-up and can not be broken down further since each month consists of different number of days. In this case, it is important for us to know which month of the year and if the year is a leap year to obtain day of the month.

4 Interaction of time granularities

Before exploring the behavior of a "dependent variable" across time granularities, it is important to know how these granularities interact with each other. If two time granulaties of interest interact, the relationship between each of the interacting variables and the dependent variable will depend on the value of the other interacting variable. To define a general framework, we define harmony and clashes. Harmonies are the pair of granularities that aid the process of exploratory analysis and clashes are pair that interact in a non-compatible ways and hinder the process of exploration of the dependent variable if plotted together.

4.1 Harmony and Clashes

Suppose we have two circular or aperiodic granularities C_1 and C_2, such that C_1 maps row numbers to a set $\{A_1, A_2, A_3, \ldots, A_n\}$, and C_2 maps row numbers to a set $\{B_1, B_2, B_3, \ldots, B_m\}$. That is, let S_{ij} be the set of row numbers such that for all $s \in S_{ij}$, $C_1(s) = i$ and $C_2(s) = j$. Since, C_1 has n levels and C_2 has m levels

there will be nm such sets S_{ij}. The graphs that don't work are those where many of these 12 sets are empty. Now, many situations can lead to any of these sets being empty. Let us discuss the following cases, where one or more of these sets can be empty.

Firstly, empty combinations can arise due to the structure of the calendar or hierarchy. These are called "structurally" empty combinations. Let us take a specific example, where C_1 maps row numbers to Day-of-Month and C_2 maps row numbers to Week-of-Month. Here C_1 can take 31 values while C_2 can take 5 values. There will be $31 \times 5 = 155$ sets S_{ij} corresponding to the possible combinations of WOM and DOM. Many of these are empty. For example $S_{1,5}$, $S_{21,2}$, etc.This is also intuitive since the first day of the month can never correspond to fifth week of the month. In fact, most of these 155 sets will be empty, making the combination of C_1 and C_2 in a graph unhelpful. These are structurally empty sets in that it is impossible for them to have any observations.

Secondly, empty combinations can turn up due to differences in event location or duration in a calendar. These are called "event-driven" empty combinations. Again, let us consider a specific example to illustrate this. Let C_1 be DOW and C_2 be WorkingDay/NonWorkingDay. Here C_1 can take 7 values while C_2 can take 2 values. So there are 14 sets S_{ij} corresponding to the possible combinations of DOW and WD/NWD. While potentially all of these can be non-empty (it is possible to have a public holiday on any DOW), in practice many of these combinations will probably have very few observations. For example, there are few if any public holidays on Wednesdays or Thursdays in any given year in Melbourne.

Thirdly, empty combinations can be a result of how granularities are constructed. Let C_1 maps row numbers to "Business days", which are days from Monday to Friday except holidays and C_2 is Day-of-Month. Then the weekends in Days-of-Month would not correspond to any Business days and would have missing observations due to the way the granularities are constructed. This is different from the structurally empty combinations because structure of the calendar does not lead to these missing combinations, but the construction of the granularity does. Hence, they are referred to as "build-based" empty combinations.

An example when there will be no empty combinations could be where C_1 maps row numbers to Day-of-Week and C_2 maps row numbers to Month-of-Year. Here C_1 can take 7 values while C_2 can take 12 values. So there are $12 \times 7 = 84$ sets S_{ij} corresponding to the possible combinations of DOW and MOY. All of these are non-empty because every DOW can occur in every month. So graphics involving C_1 and C_2 are potentially useful.

Therefore, combinations of circular/aperiodic granularities which lead to structurally, event-driven or build-based empty-combinations are referred to as to as **clashes**. And the ones that do not lead to any missing combinations are called **harmonies** as they promote the exploratory analysis through visualization.

5 Visualization

With huge amount of data being available, mean, median or any one summary statistic might not enough to understand a data set. Soon enough following questions become more interesting:

- Are values clustered around mean/median or mostly around tails? In other words, what is the combined weight of tails relative to the rest of the distribution?
- Does values rise very quickly between 25th percentile and median but not as quickly between median and 75th percentile? More generally, how the variation is the data set changes across different percentiles/deciles?
- Is the tail on the left hand side longer than that on the right side? Or are they equally balanced around mean/median?

This is when displaying a probability distribution becomes a potentially useful approach. The entire distribution can be visualized or some contextual summary statistics can be visualized to emphasize certain properties of the distribution. These properties can throw light on central tendency, skewness, kurtosis, variation of the distribution and can also be useful in detecting extreme behavior or anomalies in the data set.

Graphical output devices are flat. If there are only two variables, a cartesian graph of one variable against the other can show the configuration of points. As soon as the data move to three variables, we need to infer multi-dimensional structure by a two-dimensional medium[Cleveland, elements of graphing data]. Any attempt to encode all these temporal granularities together to develop insights on periodicity might fail or become clumsy. Instead, the big problem can be broken down into smaller pieces by focusing on specific tasks per visual representation in a 2D space.

`ggplot2` (Wickham, 2016) is an excellent tool that facilitates the process of mapping different variables to a 2D frame through grammar of graphics. One aspect of ggplot2 is faceting, inspired by Trellis plots (Cleveland, 1993). These are based on conditioning the values taken on by one or more of the variables in a data set. This means creating same plots using data subsets corresponding to each level of the conditioned variable. This is a powerful tool for exploratory data analysis as we can rapidly compare patterns in different parts of the data.

We will see the distribution of the time series across bivariate temporal granularities through faceting approach with one temporal granularity plotted along the x-axis, the other one across facets and the dependent time series variable on the y-axis.

Now, different distribution plots might be appropriate depending on which features of the distrbution we are interested to look at, the levels of time granularities being plotted, how granularities interact and number of observations available. We will look at each of these aspects separately and analyze the effect of each on visualization.

5.1 Choice of Plots

There are several ways to plot statistical distributions. The displayed probabilities in these plots are either computed using kernel density estimation methods or emperical statistical methods. We will discuss the benefits and challenges of few conventional and recent ways to plot distributions using both of these methods.

5.1.1 Emperical methods

Most commonly used techniques to display distribution of data include the histogram (Karl Pearson), which shows the prevalence of values grouped into bins and the box-and-whisker plots (Tukey 1977) which convey statistical features such as the median, quartile boundaries, hinges, whiskers and extreme outliers. The box plot is a compact distributional summary, displaying less detail than a histogram. Due to wide spread popularity and simplicity in implementation, a number of variations are proposed to the original one which provides alternate definitions of quantiles, whiskers, fences and outliers. Notched box plots (Mcgill, Tukey, and Larsen 1978, 1978) has box widths proportional to the number of points in the group and display confidence interval around medians aims to overcome some drawbacks of box plots. The standard box plot and all of these variations are helpful to get an idea of the distribution at a glance. However, we do not have an idea if the distribution is multimodal. Moreover, for data less than 1000 observations, detailed estimates of tail behavior beyond the quartiles are not trustworthy. Also, the number of outliers is large for larger data set since number of outliers is proportional to the number of observations.

The letter-value box plot (Hofmann, Wickham, and Kafadar 2017, 2006) was designed to adjust for number of outliers proportional to the data size and display more reliable estimates of tail.It shows only actual data values and no smoothed summaries. However, the letter values are shown till the depths where the letter values are reliable estimates of their corresponding quantiles. These might lead to a lot of letter values being shown and leading to overload of information in one plot. Also, multimodality is difficult to spot.

Quantile plots visually portray the quantiles of the distribution of sample data. Much like the quartiles divide the data set equally into four equal parts, extensions might include dividing the data set even further. These plots are referred to as “quantile” plots with, where number of quantiles 9 for deciles and 99 for percentiles. No distributional assumptions are made about the data since empirical deciles are plotted. It avoids much clutter and just enable us to focus on specific probabilities. These plots do not allow us to see the shape, skewness or nature of the tail of the distribution with so much clarity as violin plots, ridge plots or letter value plots. Also, a large data set is required for the extreme percentiles to be estimated with any accuracy.

5.1.2 Kernel density estimation methods

A density plot which uses a kernel density estimate to show the probability density function of the variable can show the entire distribution, unlike discretizing the distribution and showing only parts of it.

Violin plots (Hintze and Nelson 1998, 1998) adds the information available from local density estimates to summary statistics provided by box plots. The shape of the violin represents the density estimate of the variable. The more data points in a specific range, the larger the violin is for that range. Adding two density plots gives a symmetric plot which makes it easier to see the magnitude of the density and compare across categories. Howevr, the density in violin are estimated through kernel density estimation and thus makes assumptions when selecting kernel or bandwidth. As a result, the plot shows smooth summaries based on the assumptions and not only on actual observations.

The summary plot (Potter et al. 2010, 2010) combines a minimal box plot with glyphs representing the first five moments (mean, standard deviation, skewness, kurtosis and tailings), and a sectioned density plot crossed with a violin plot (both color and width are mapped to estimated density), and an overlay of a reference distribution. This suffers from the same problem as boxplots or violin plot, as it is combination of those two.

A Ridge line plot (sometimes called Joy plot) shows the distribution of a numeric value for several groups. Distribution can be represented using histograms or density plots, all aligned to the same horizontal scale and presented with a slight overlap. A clear advantage over boxplots is that these plots allow us to see multimodality in the distribution. However, these plots can be obscuring when there is overlap of distribution for two or more categories of the y-axis. Also, if there are lot of categories, it is difficult to compare the height of the densities across categories.

The highest density region (HDR) box plot proposed by (Hyndman 1996) displays a probability density region that contains points of relatively highest density. The probabilities for which the summarization is required can be chosen based on the requirement. These regions do not need to be contiguous and help identify multi-modality.

As a general rule, the quantile plots are really useful for picking patterns, whereas, other more involved methods of plotting are useful for studying anomalies or outlier behavior. It is good to be aware of the benefits and challenges while choosing any distribution plots, depending on the context.

5.1.3 Levels of temporal granularities

Choice of plots depend on the levels(very high/high/medium/low) of the two granularities plotted. The criteria for different levels could be based on usual cognitive power while comparing across facets and display size available to us.Space and resolution become a problem if the number of levels of facet variables is too high. Hence, plot choices will also vary depending on which granularity is placed on the x-axis and which one across facets.

Levels are categorized as very high/high/medium/low each for the facet variable and the x-axis variable. Default values for these levels are chosen based on levels of common temporal granularities like day of the month, day of a fortnight or day of a week. Any levels above 31 can be considered as very high, any levels between 14 to 31 can be taken as high and that between 7 to 14 can be taken as medium and below 7 as low. 31, 14 and 7 are the levels of days-of-month, days-of-fortnight and days-of week respectively.

Space and cognitive power both are constrained if the number of levels of facet variables is very high. So as a general rule, we should avoid plotting a temporal granularity with very high levels across facets.

- If levels of both the granularities are low, then any distribution plots might be chosen depending on which feature of the distribution who is interested to look at.
- If level of the granularity plotted across x-axis is more than medium, ridge plots should be avoided to escape overlap of categories.
- If level of the granularity plotted across x-axis is low, then any distribution plots might be chosen depending on which feature of the distribution who is interested to look at, irrespective of the number of the levels of the facet variable.

- If level of the granularity plotted across x-axis is more than or equal to high, quantile plots are preferred.

5.2 Effect of interaction

For illustration, distribution of half-hourly electricity consumption of Victoria is plotted across different time granularities in each of the panel in Figure 2. Figure 2 (a) shows the letter value plot across days of the month faceted by months like January, March and December. Figure 2 (c) shows box plot across days of the year by the 1st, 15th, 29th and 31st days of the month. Figure 2 (d) showing violin plot across days of the month faceted by week of the month. Figure 2 (e), variations across week of the year conditional on week of the month can be observed through a ridge plot and Figure 2 (f) shows decile plots across day of the year and month of the year.

Clearly, in Figure 2, we observe that some choices of time granularities work and others do not. In Figure 2 (c), there will be no observations for some combinations "Day-of-Month" and "Day-of-Year". In particular, the 1st day of the month can never correspond to 2nd, 3rd or 4th day of the year. On the contrary, for Figure 2 (a), we will not have any combinations with zero combinations because every "Day-of-Week" can occur in any "Month-of-Year". Thus the graphs that don't work are those where many of the combination sets are empty. In other words, if there are levels of time granularities plotted across x-axis which are not spanned by levels of the time granularities plotted across facets or vice versa, we will have empty sets leading to potential ineffective graphs. We hypothesize that the interaction of these granularities are in play while deciding if the resulting plot would be a good candidate for exploratory analysis.

We should avoid plotting clashes as they hinder the exploratory process by having missing combinations of time granularities in each panel.

5.3 Effect of number of observation

5.3.1 Rarely occuring events

Even with harmonies, problems might occur due to rarely occuring categories.

Suppose we have T hourly observations, and C_1 with n_1 categories and C_2 with n_1 categories. Each element of C_1 occurs approximately T/n_1 times while each element of C_2 occurs approximately T/n_2 times. There are no structurally empty combinations, and each combination will occur on average an equal number of times as $T \to \infty$, so the average number of observations per combination is $T/(n_1 n_2)$. If we require at least k observations to create a meaningful panel, then provided $T \geq n_1 n_2 k$, the visualization will be acceptable. The value of k will depend on what type of visualization we are producing. For emperical estimates, even $k = 10$ may be acceptable, but for density estimates, we probably need $k \geq 30$.

Let us take a specific example to illustrate this. Rarely occurring categories such as the 366th day of the year, or the 31st day of the month can suffer from such problem.

5.3.2 Unevenly distributed events

Even when there are no rarely occuring events, when number of observations are not evenly distributed between different categories of the time granularities.Large differences between number of observations between and across facets, might have an impact on the estimation of dfistribution. We use Gini's measure to compute if differences in number of observations across or within facets are significantly different.

6 Case studies

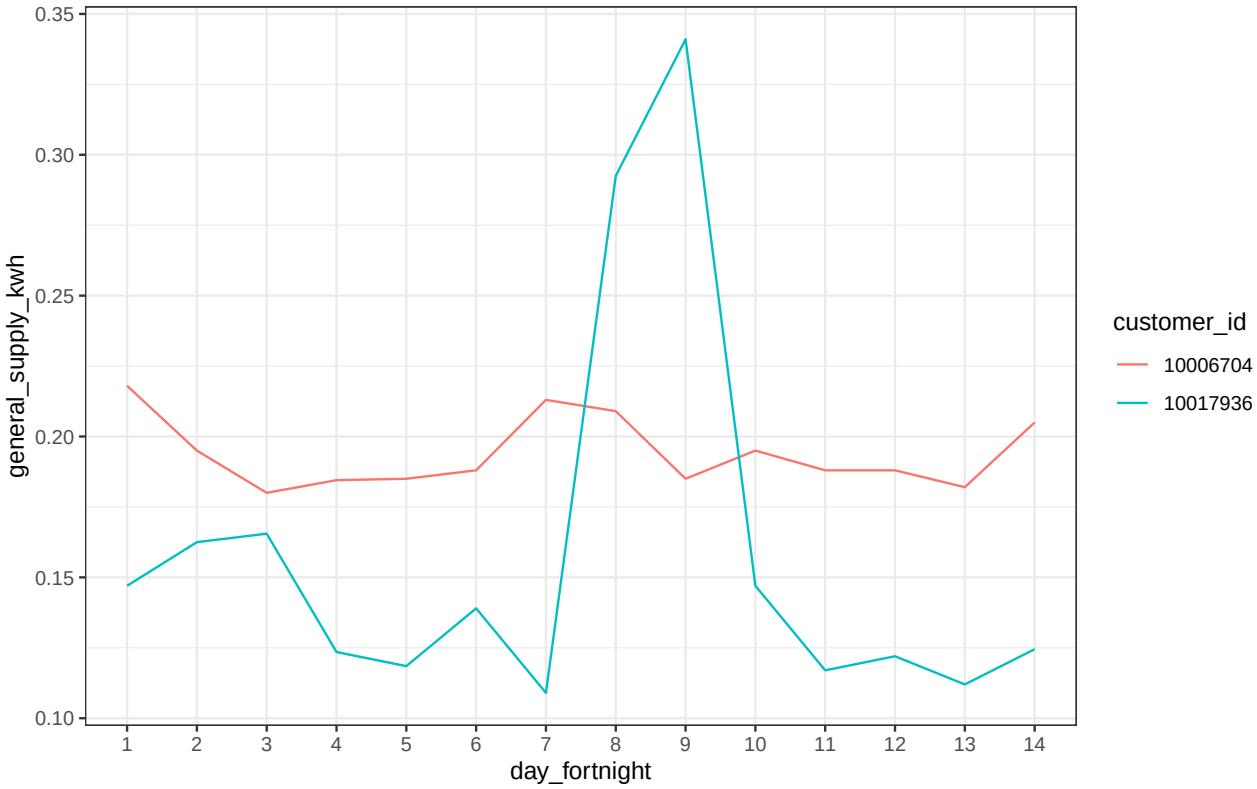
6.1 Smart meter data of Australia

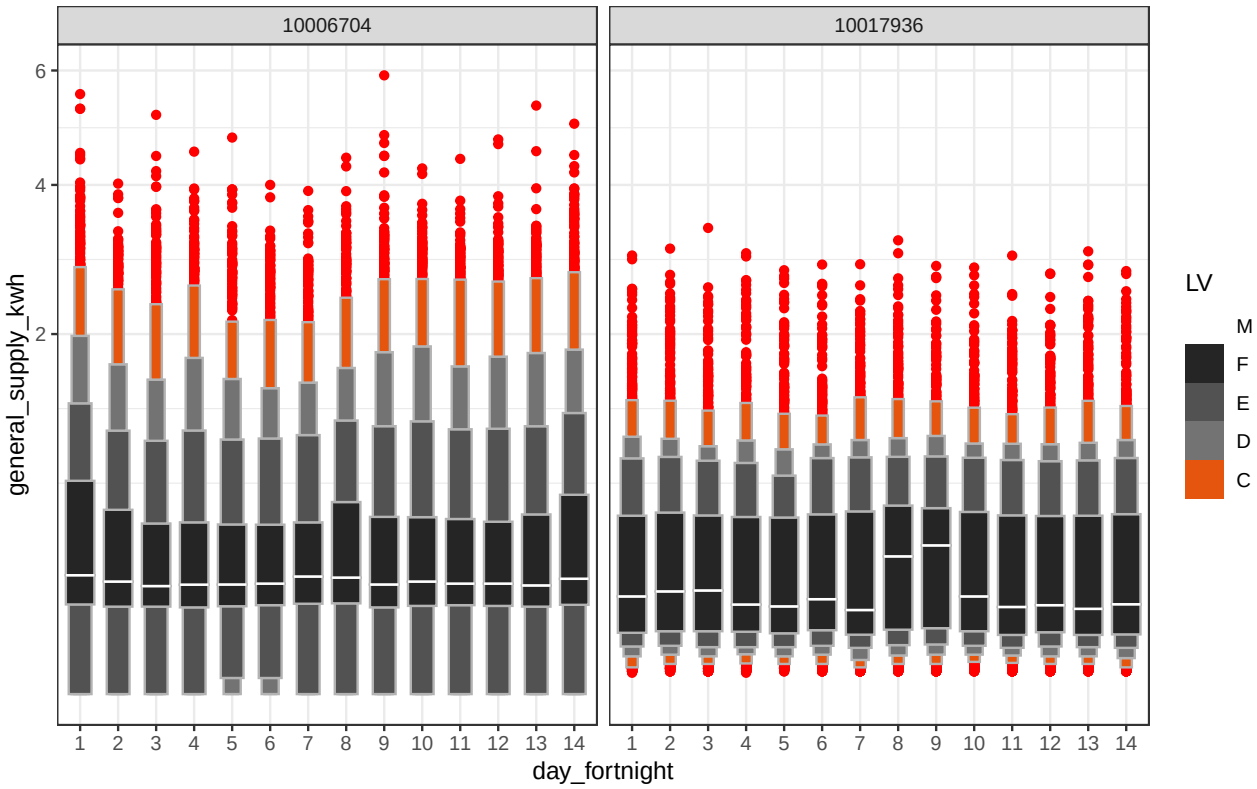
Smart meters provide large quantities of measurements on energy usage for households across Australia. One of the customer trial (Department of the Environment and Energy 2018) conducted as part of the Smart Grid Smart City (SGSC) project (2010-2014) in Newcastle, New South Wales and some parts of Sydney provides customer wise data on half-hourly energy usage and detailed information on appliance use, climate,

retail and distributor product offers, and other related factors. It would be interesting to explore the energy consumption distribution for these customers and gain more insights on their energy behavior which are otherwise lost either due to aggregation or looking only at coarser temporal units. The idea here is to show how looking at the time across different granularities together can help identify different behavioral patterns and identify the extreme and regular households amongst these 50 households.

–>

In Figure **??** and **??** , the smart meter data is filtered for two customers to illustrate what kinds of insights can be drawn for the energy beahvior of these two customers. In Figure **??** it can be seen that for most days of the fortnight, the second household has much less consumption than the first one. However, Figure **??** adds to the beahvior by looking at the distribution. If we consider letter value F as a regular behavior and letter values beyond F as not-so-regular behavior, we can conclude that the regular behavior of the first household is more stable than the second household. However, the distribution of tail of the first household is more variable, observed through distinct letter values, implying that their not-so-regular behavior is quite extreme. This shows, how looking at the distribution of the dependent variable can throw more light on the energy behavior of the customers, which are lost using aggregate or summary statistics.





While trying to explore the energy behavior of these customers systematically, the first thing we should have at our disposal for examining periodicities of energy behavior across time granularities is to know the number of time granularities we can look at exhaustively. If we consider conventional time deconstructions for a Gregorian calendar (second, minute, half-hour, hour, day, week, fortnight, month, quarter, semester, year), the following time granularities can be considered for this analysis.

```
#> 30m
```

```
#>  [1] "hhour_hour"          "hhour_day"           "hhour_week"
#>  [4] "hhour_fortnight"     "hhour_month"         "hhour_quarter"
#>  [7] "hhour_semester"      "hhour_year"          "hour_day"
#> [10] "hour_week"           "hour_fortnight"      "hour_month"
#> [13] "hour_quarter"        "hour_semester"       "hour_year"
#> [16] "day_week"            "day_fortnight"       "day_month"
#> [19] "day_quarter"         "day_semester"        "day_year"
#> [22] "week_fortnight"      "week_month"          "week_quarter"
#> [25] "week_semester"       "week_year"           "fortnight_month"
#> [28] "fortnight_quarter"   "fortnight_semester"  "fortnight_year"
#> [31] "month_quarter"       "month_semester"      "month_year"
#> [34] "quarter_semester"    "quarter_year"        "semester_year"
```

The interval of this tsibble is 30 minutes, and hence the default for search_gran in this case, provides temporal granularities ranging from half-hour to year. If these options are considered too many, the default options can be modified to limit the possibilities. For example, the most coarse temporal unit can be set to be a "month".

```
#>  [1] "hhour_hour"      "hhour_day"       "hhour_week"
#>  [4] "hhour_fortnight" "hhour_month"     "hour_day"
#>  [7] "hour_week"       "hour_fortnight"  "hour_month"
#> [10] "day_week"        "day_fortnight"   "day_month"
#> [13] "week_fortnight"  "week_month"      "fortnight_month"
```

This looks better. However, some intermediate temporal units might not be pertinent to the analysis and we might want to remove them from the list of granularities.

```
#> [1] "hour_day"   "hour_week"  "hour_month" "day_week"   "day_month"
#> [6] "week_month"
```

Now that we have the list of granularities to look at, we can visualize the distribution. From the search list, we found that we can look at six granularities, that amounts to analyzing six graphics. However, what happens if we want to see the distribution of energy across two granularities at a time? This is equivalent to looking at the distribution of energy consumption across one granularity conditional on another one. One way can be to plot one of the granularities on the x-axis and another on the facet. Different perspectives of the data can be derived depending on where the granularities are placed.

So, what is the number of pairs of granularities we can look at? It is equivalent to taking 2 granularities from 6, which essentially means we need to examine 30 plots. The good news is, not all time granularities can be plotted together and we do not have to analyze so many plots!

Harmony/clash can be identified to considerably reduce the number of visualizations that can aid exploratory analysis.

```
smart_meter10 %>%
  is_harmony(gran1 = "hour_day",
             gran2 = "day_week")
```

```
#> [1] "TRUE"
```

```
smart_meter10 %>%
  is_harmony(gran1 = "hour_day",
             gran2 = "day_week",
             facet_h = 14)
```

```
#> [1] "FALSE"
```

```
smart_meter10 %>%
  is_harmony(gran1 = "day_month",
             gran2 = "week_month")
```

```
#> [1] "FALSE"
```

Let us now look at all the harmonies that we can examine. Fortunately, we are left with only 13 out of 30 visualizations.

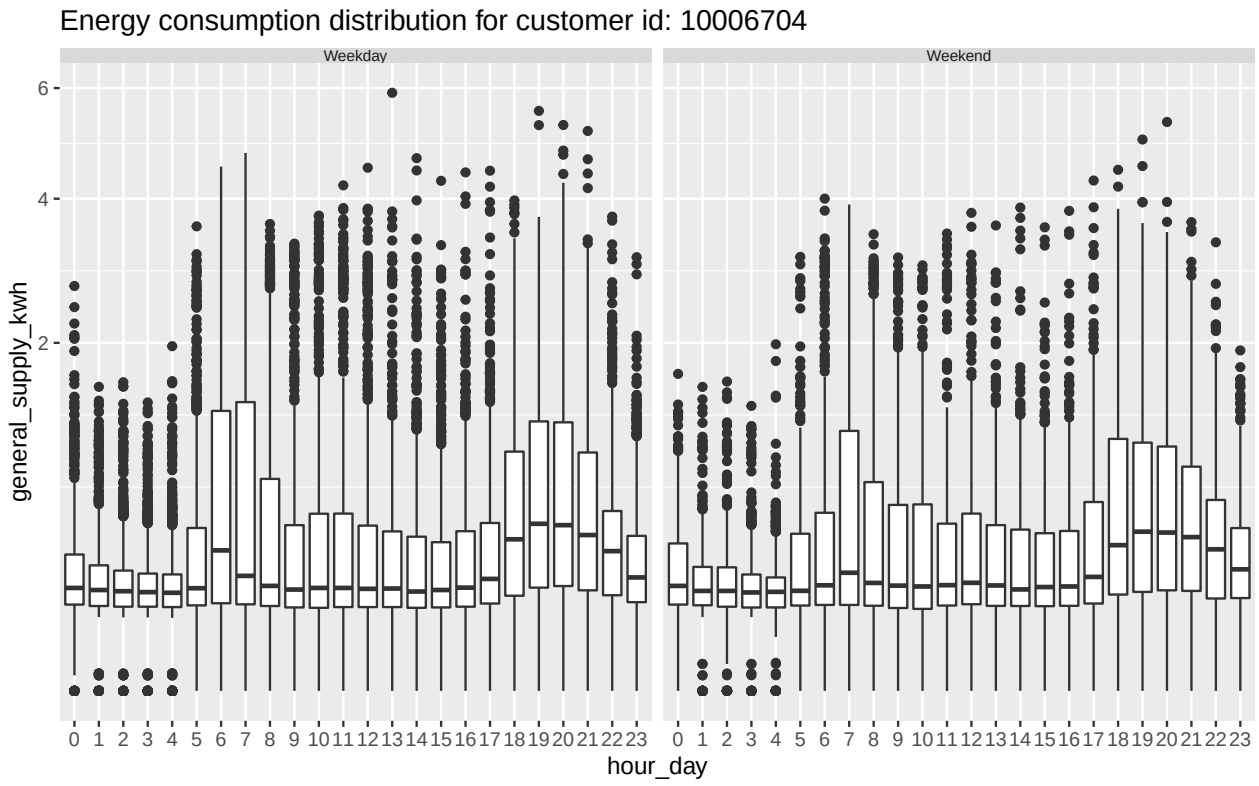
```
smart_meter10 %>% harmony(
  ugran = "month",
  filter_out = c("hhour", "fortnight")
)
```

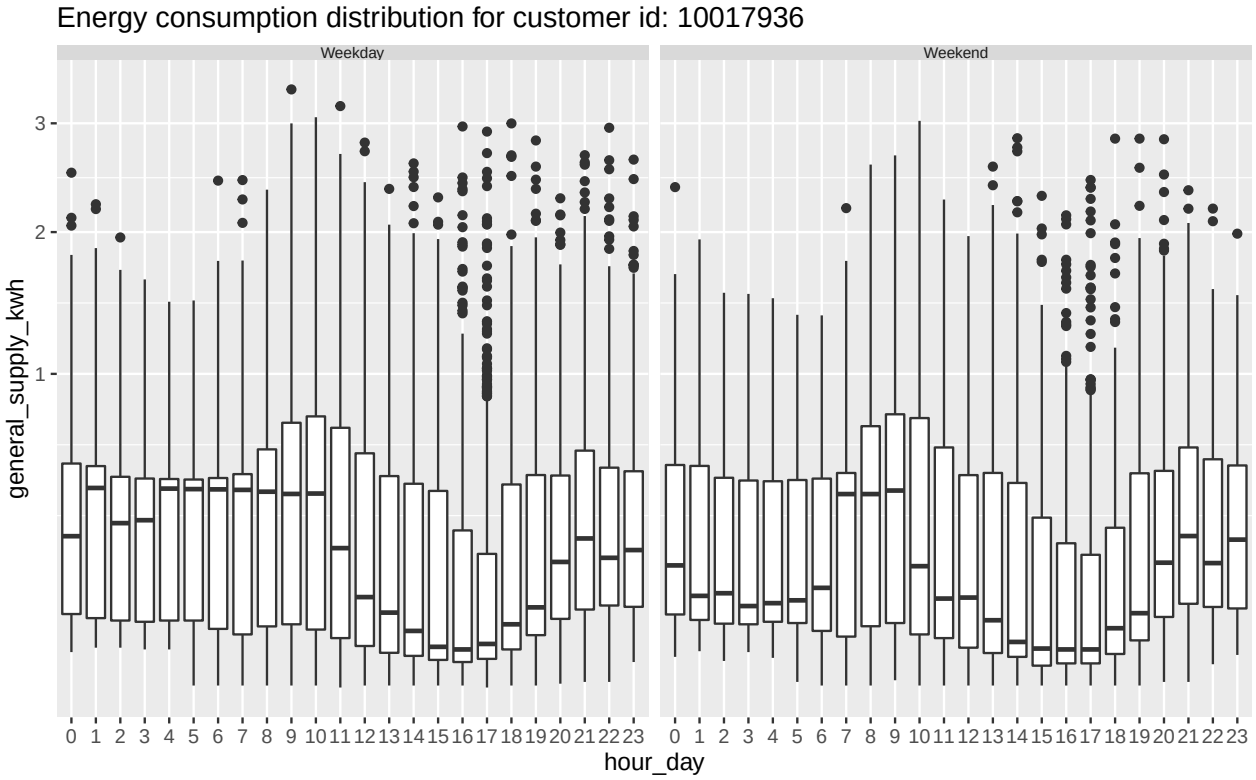
```
#> # A tibble: 13 x 4
#>    facet_variable x_variable facet_levels x_levels
#>    <chr>          <chr>             <int>    <int>
#>  1 day_week       hour_day             7       24
#>  2 day_month      hour_day            31       24
#>  3 week_month     hour_day             5       24
#>  4 day_month      hour_week           31      168
#>  5 week_month     hour_week            5      168
#>  6 day_week       hour_month           7      744
#>  7 hour_day       day_week            24        7
#>  8 day_month      day_week            31        7
#>  9 week_month     day_week             5        7
```

```
#> 10 hour_day       day_month            24       31
#> 11 day_week       day_month             7       31
#> 12 hour_day       week_month           24        5
#> 13 day_week       week_month            7        5
```

In **??** We visualize the harmony pair (wknd_wday, hour_day) through a box plot. Boxplot of energy consumption is shown across wknd_wday (facet) and hour-day (x-axis) for the same two households. For the second household, outliers are less prominent implying their regular behavior is more stable. For the first household, energy behavior is not significantly different between weekdays and weekends. For the second household, median energy consumption for the early morning hours is extremely high for weekends compared to weekdays.

For the second household, outliers are less prominent implying their regular behavior is more stable. For the first household, energy behavior is not significantly different between weekdays and weekends. For the second household, median energy consumption for the early morning hours is extremely high for weekends compared to weekdays.

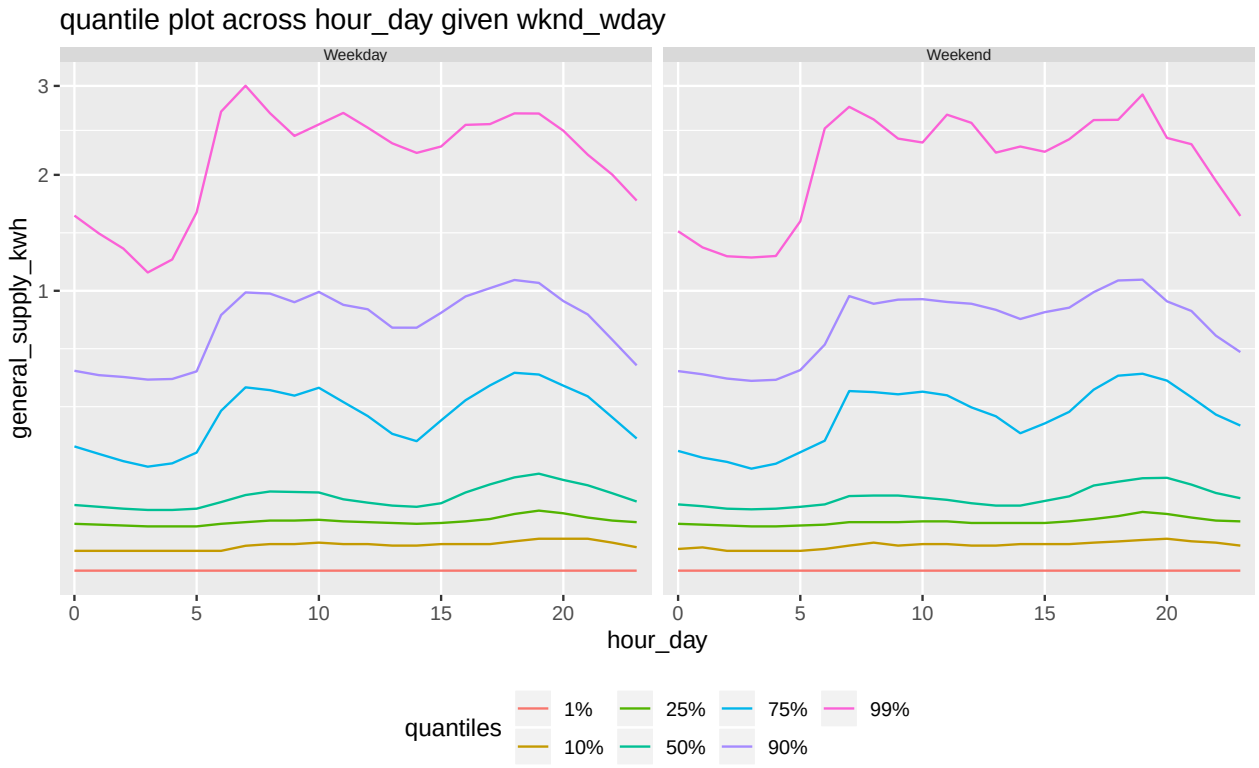


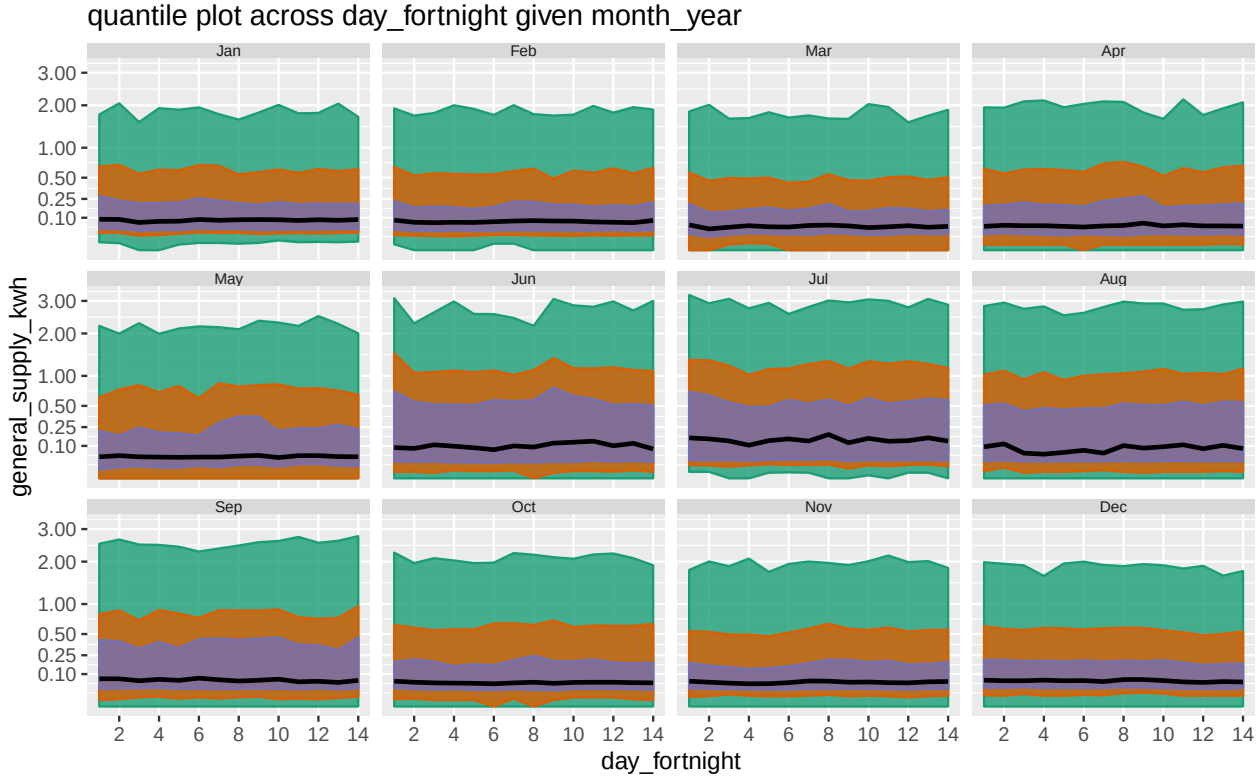


Now, we would like to see how these customers' behavior relate to the rest of the 50 households across these two measures -

- if the regular behavior is regular and extreme behavior extreme or vice versa
- if the weekend and weekday behavior are different for most of these households.

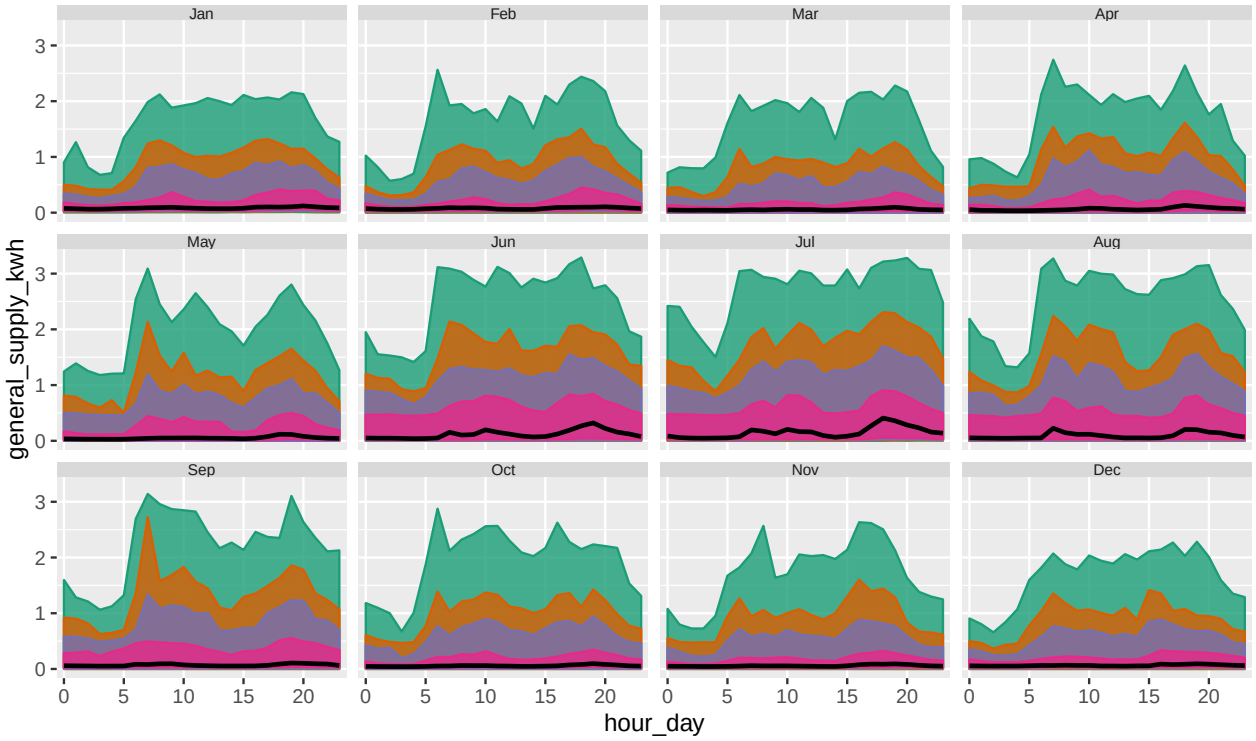
Figure **??** shows the overlay quantile plot of energy consumption of 10 households. The black line is the median, whereas the pink band covers 25th to 75th percentile, the orange band covers 10th to 90th percentile and the green band covers 1st to 99th percentile. It can be observed that the median is very closer to the lower boundaries of all the other bands implying energy consumption for these households are left skewed. Weekend and weekday behavior is not very different for these ten customers, implying the first customer is more typical than the second customer when compared to these ten customers.






```
library(gravitas)
library(tsibble)
```

```
library(ggplot2)
smart_meter10 %>%
  prob_plot("month_year", "hour_day",
    response = "general_supply_kwh",
    plot_type = "quantile",
    quantile_prob = c(0.01, 0.05, 0.1, 0.25, 0.5, 0.75, 0.9, 0.95, 0.99),
    symmetric = TRUE) +
  ggtitle("")
```



6.2 T20 cricket data of Indian Premiere League

Th application is not only restricted to temporal data. We provide an example of cricket to illustrate how this can be generalised in other applications. The Indian Premier League (IPL) is a professional Twenty20 cricket league in India contested by eight teams representing eight different cities in India. With eight teams, each team plays each other twice in a home-and-away round-robin format in the league phase. In a Twenty20 game the two teams have a single innings each, which is restricted to a maximum of 20 overs. Hence, in this format of cricket, a match will consist of 2 innings, an innings will consist of 20 overs, an over will consist of 6 balls. We have sourced the the ball by ball data for IPL season 2008 to 2016 from Kaggle. The dataset contains the information on batting team, bowling team, balls of the over, over of the innings, innings of the match and total runs per ball for 577 matches spanning over 9 seasons (2008 to 2016). It also has information on which team won, winning margin and several other useful information.

A hierarchy like table 4 can be construed for this game format, where index here would refer to ball as the data set contains ball-by-ball data.

There are many interesting questions that can possibly be answered with such a data set, however, we will explore a few and understand how the proposed approach in the paper can help answer some of the questions.

First, we look at the distribution of runs per over across over of the innings and seasons in 3. The distribution of runs per over has not significantly changed from 2008 to 2016. There is no clear pattern/trend that runs

Table 4: A hierarchy table for T20 cricket

units	convert_fct
index	1
ball	6
over	20
inning	2
match	1

per over is increasing or decreasing across seasons. Hence, we work with subsets of seasons and try to see how the strategies of the winning teams differ across seasons or how in a particular season the strategy was different for the winning team and the ones who did not qualify for the playoffs.

Mumbai Indians(MI) and Chennai Super kings(CSK) are considered one of the best teams in IPL with multiple winning titles and always appearing in final 4 from 2010 to 2015. It would be interesting to see the difference in their strategies throughout all matches in the two seasons. The following two questions might help us partially understand their strategies.

- Q1: How their run rates vary depending on if they bat first or 2nd? Is there a chance that they are more likely to win if they bat first?
- Q2: Which team is more consistent in their approach in terms of run rate across different overs of the innings?

```
#> # A tibble: 2 x 3
#>   inning   `0`   `1`
#>   <fct>  <dbl> <dbl>
#> 1 1        240   460
#> 2 2        229   447
```

```
#> # A tibble: 2 x 3
#>   inning   `0`   `1`
#>   <fct>  <dbl> <dbl>
#> 1 1        248   440
#> 2 2        237   429
```

```
#>   win_by_runs
#>  Min.   : 2.00
#>  1st Qu.:18.00
#>  Median :39.00
#>  Mean   :39.95
#>  3rd Qu.:50.00
#>  Max.   :87.00
```

```
#>   win_by_runs
#>  Min.   : 9.00
#>  1st Qu.:22.00
#>  Median :24.00
#>  Mean   :28.02
#>  3rd Qu.:33.00
#>  Max.   :60.00
```

Figure 4 shows that for CSK, irrepective of the fact they won or lost, distribution of run rate in case they are batting in the 2nd innings is always skewed on the right side, implying that they always manage to give a good fight to win even if they are losing. Run rate can go up to 6 runs per over depending on the situation. For MI, this is not true. When they won, their run rates in the 2nd innings went up to 6 and led to their win, but stayed within 4 whenever they lost. This implies CSK is more of a fighter than MI when it comes to

defending in the 2nd innings. However, we can delve deeper and try to see which overs are the most volatile for CSK and if it is different to that of MI?

In Figure 5, we can observe the letter value plot describing the distribution of run rate per over across over of the innings and innings of the match. It can be observed that for both innings, the run rate for Mumbai Indians rise faster across overs of the innings. There is an upward shift of the distribution in the second part of the innings (after 10th over) and more variability (longer and distinct letter values) as well. CSK has been more uniform in their approach maintaining consistent scores per over. Since CSK's last over in the 2nd innings really get volatile, after a consistent performance in the other overs - this behavior is exhibited as an outlier in Figure 4.

–>

over	1	2
1	55	44
2	55	44
3	55	44
4	55	44
5	55	44
6	55	44
7	55	44
8	55	44
9	55	44
10	55	44
11	55	44
12	55	44
13	55	44
14	55	43
15	55	42
16	55	41
17	55	41
18	55	39
19	55	39
20	55	29

over	1	2
1	39	52
2	39	52
3	39	52
4	39	52
5	39	52
6	39	52
7	39	52
8	39	52
9	39	52
10	39	52
11	38	51
12	38	51
13	37	50
14	36	50
15	36	49
16	36	48
17	36	48

over	1	2
18	36	44
19	36	38
20	36	27

```
#> # A tibble: 13 x 1
#>    batting_team
#>    <chr>
#>  1 Kolkata Knight Riders
#>  2 Royal Challengers Bangalore
#>  3 Chennai Super Kings
#>  4 Kings XI Punjab
#>  5 Delhi Daredevils
#>  6 Rajasthan Royals
#>  7 Mumbai Indians
#>  8 Deccan Chargers
#>  9 Kochi Tuskers Kerala
#> 10 Pune Warriors
#> 11 Sunrisers Hyderabad
#> 12 Rising Pune Supergiants
#> 13 Gujarat Lions
```

Q3: Is run rate set to reduce in overs where fielding is good?

For establishing that the fielder fielded well in a particular over, we can see how many catches and run outs were made in that particular over. If a batsman is bowled out, it does not necessarily signify good fielding. So we only include catches and run out as a measure of fielding. Difference in run rates should be negative if fielding is good. Let us see if this fact is true.Thus, Figure 6 shows the difference between run rate between two subsequent overs are negative when good fielding leads to one or two dismissals in an over.

```
#> # A tibble: 4 x 21
#>   fielding_wckts   `1`   `2`   `3`   `4`   `5`   `6`   `7`   `8`   `9`
#>   <fct>          <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
#> 1 0               1039   997   972   978   962   957  1002  1012   993
#> 2 1                120   145   171   165   180   181   138   124   140
#> 3 2                  5    10     8     8     9    10     6    10     8
#> 4 3                  0     0     0     0     0     0     0     0     0
#> # ... with 11 more variables: `10` <dbl>, `11` <dbl>, `12` <dbl>,
#> #   `13` <dbl>, `14` <dbl>, `15` <dbl>, `16` <dbl>, `17` <dbl>,
#> #   `18` <dbl>, `19` <dbl>, `20` <dbl>
```

Q4: Is run rate set to reduce in overs where number of dot balls are more/number of wickets are more?

A dot ball is a delivery bowled without any runs scored off it. The number of dot balls is reflective of the quality of bowling in the game. Run rate of an over should ideally decrease if the number of dot balls increase or the batsman can likely to go for big shots and get dismissed. Figure 7 shows that for any over, increase in dot balls lead to decrease in run rate.

{r read_data, echo=FALSE}

Aigner, Wolfgang, Silvia Miksch, Heidrun Schumann, and Christian Tominski. 2011. *Visualization of Time-Oriented Data*. Springer Science & Business Media.

Bettini, Claudio, Curtis E Dyreson, William S Evans, Richard T Snodgrass, and X Sean Wang. 1998. "A Glossary of Time Granularity Concepts." In *Temporal Databases: Research and Practice*, edited by Opher Etzion, Sushil Jajodia, and Suryanarayana Sripada, 406–13. Berlin, Heidelberg: Springer Berlin Heidelberg.

Department of the Environment and Energy. 2018. *Smart-Grid Smart-City Customer Trial Data*. Australian Government, Department of the Environment; Energy: Department of the Environment; Energy, Australia. https://data.gov.au/dataset/4e21dea3-9b87-4610-94c7-15a8a77907ef.

Hintze, Jerry L, and Ray D Nelson. 1998. "Violin Plots: A Box Plot-Density Trace Synergism." *Am. Stat.* 52 (2). [American Statistical Association, Taylor & Francis, Ltd.]: 181–84.

Hofmann, Heike, Hadley Wickham, and Karen Kafadar. 2017. "Letter-Value Plots: Boxplots for Large Data." *J. Comput. Graph. Stat.* 26 (3). Taylor & Francis: 469–77.

Hyndman, Rob J. 1996. "Computing and Graphing Highest Density Regions." *Am. Stat.* 50 (2). [American Statistical Association, Taylor & Francis, Ltd.]: 120–26.

Mcgill, Robert, John W Tukey, and Wayne A Larsen. 1978. "Variations of Box Plots." *Am. Stat.* 32 (1). Taylor & Francis: 12–16.

Potter, K, J Kniss, R Riesenfeld, and C R Johnson. 2010. "Visualizing Summary Statistics and Uncertainty." *Comput. Graph. Forum* 29 (3): 823–32.

Tukey, John W. 1977. *Exploratory Data Analysis*. Vol. 2. Reading, Mass.

Wang, Earo, Dianne Cook, and Rob J Hyndman. 2019. "A New Tidy Data Structure to Support Exploration and Modeling of Temporal Data," January.

Wickham, Hadley. 2016. *Ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. http://ggplot2.org.

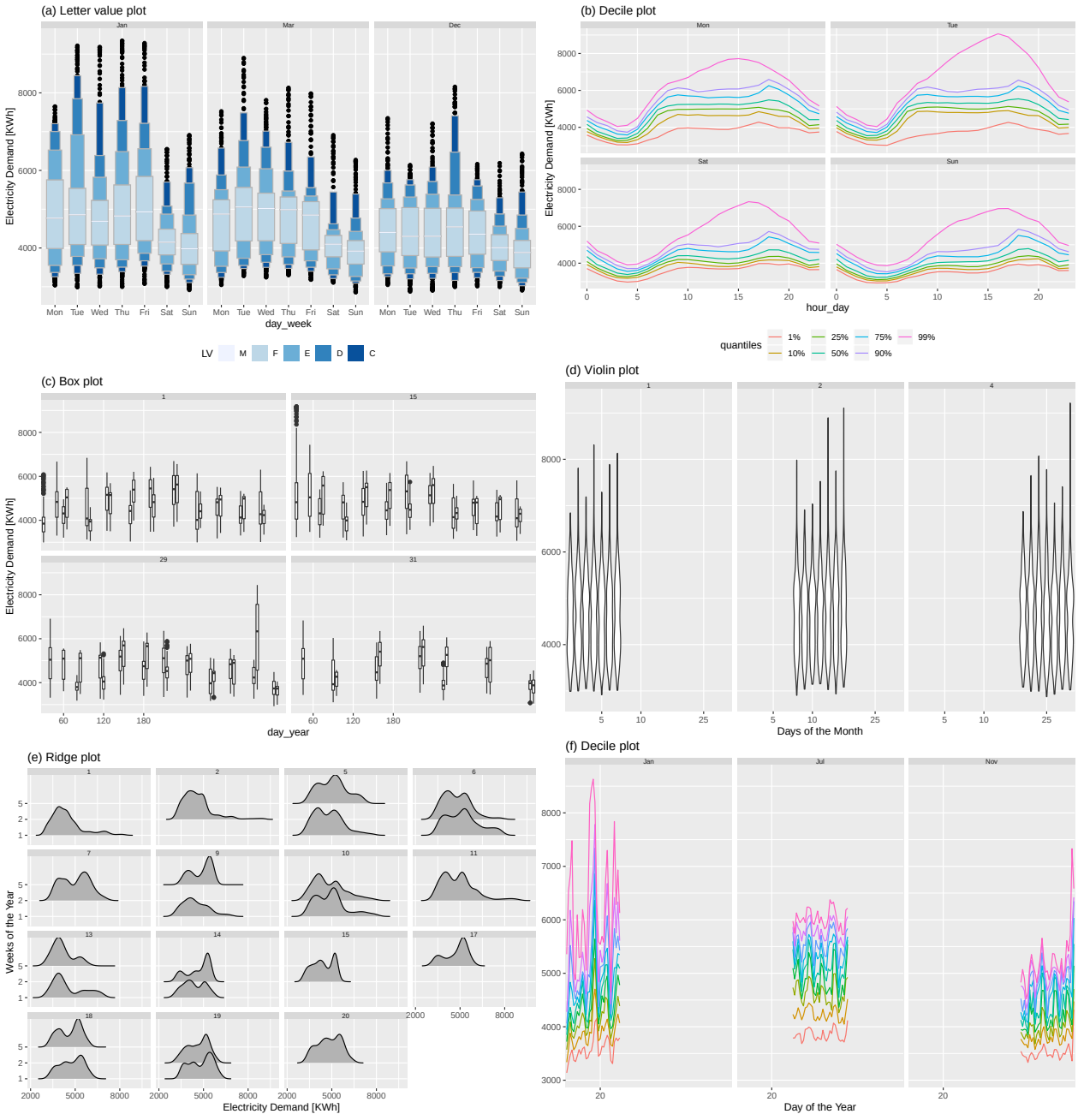


Figure 2: Various probability distribution plots of electricity consumption data of Victoria from 2012 to 2014. (a) Letter value plot by DoM and MoY, (b) Decile plot by HoD and DoW (c) Box plot by DoY and DoM, (d) Violin plot of DoM and WoM, (e) Ridge plot by WoM and WoY, (f) Decile plot by DoY and MoY. Only plots (a) and (b) show harmonised time variables.

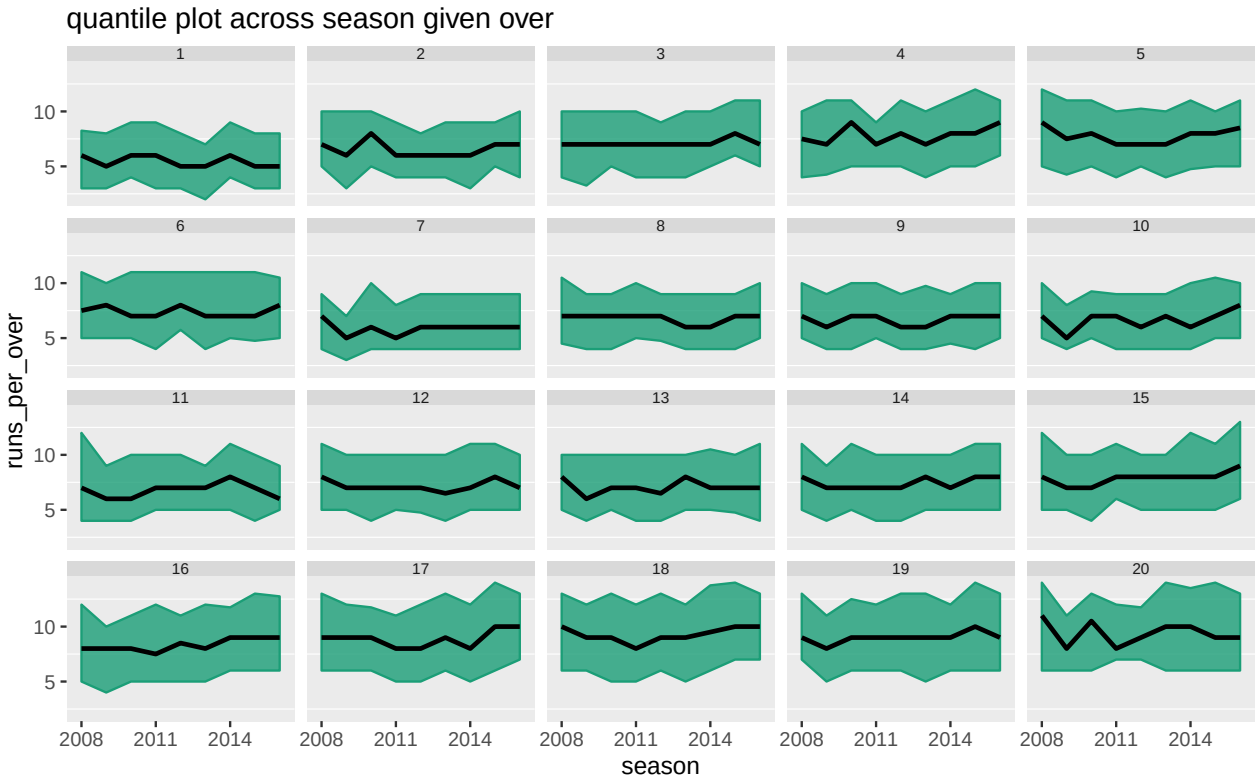


Figure 3: Quantile plot of runs per over across overs of different seasons.

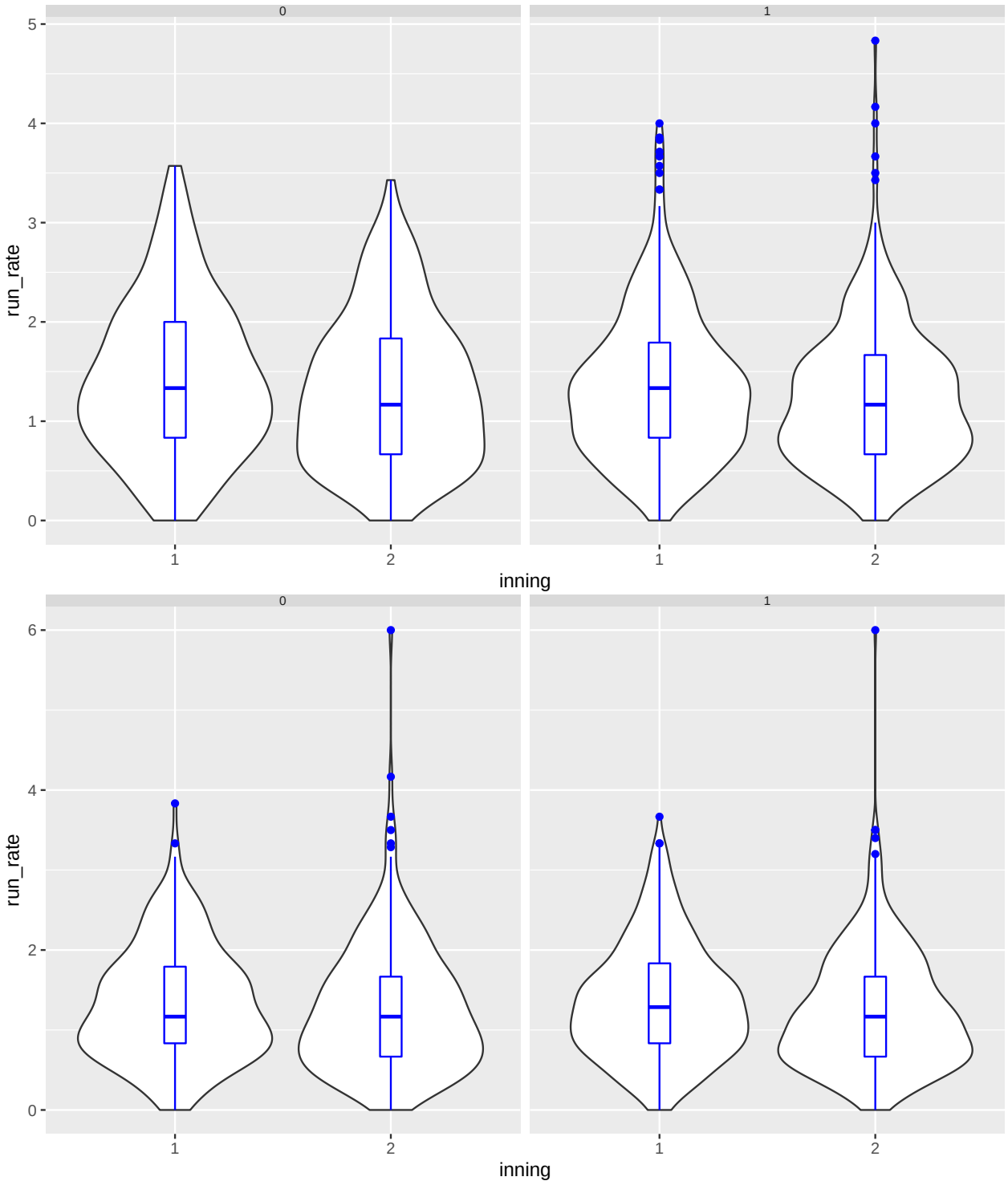


Figure 4: Violin and box plots of run rate across innings of the match and matches won or lost for Mumbai Indians(top) and Chennai Super Kings(bottom).The violins for CSK has long right tails in 2nd innings for cases when they lost or won. MI has long right tail only when they won the match.

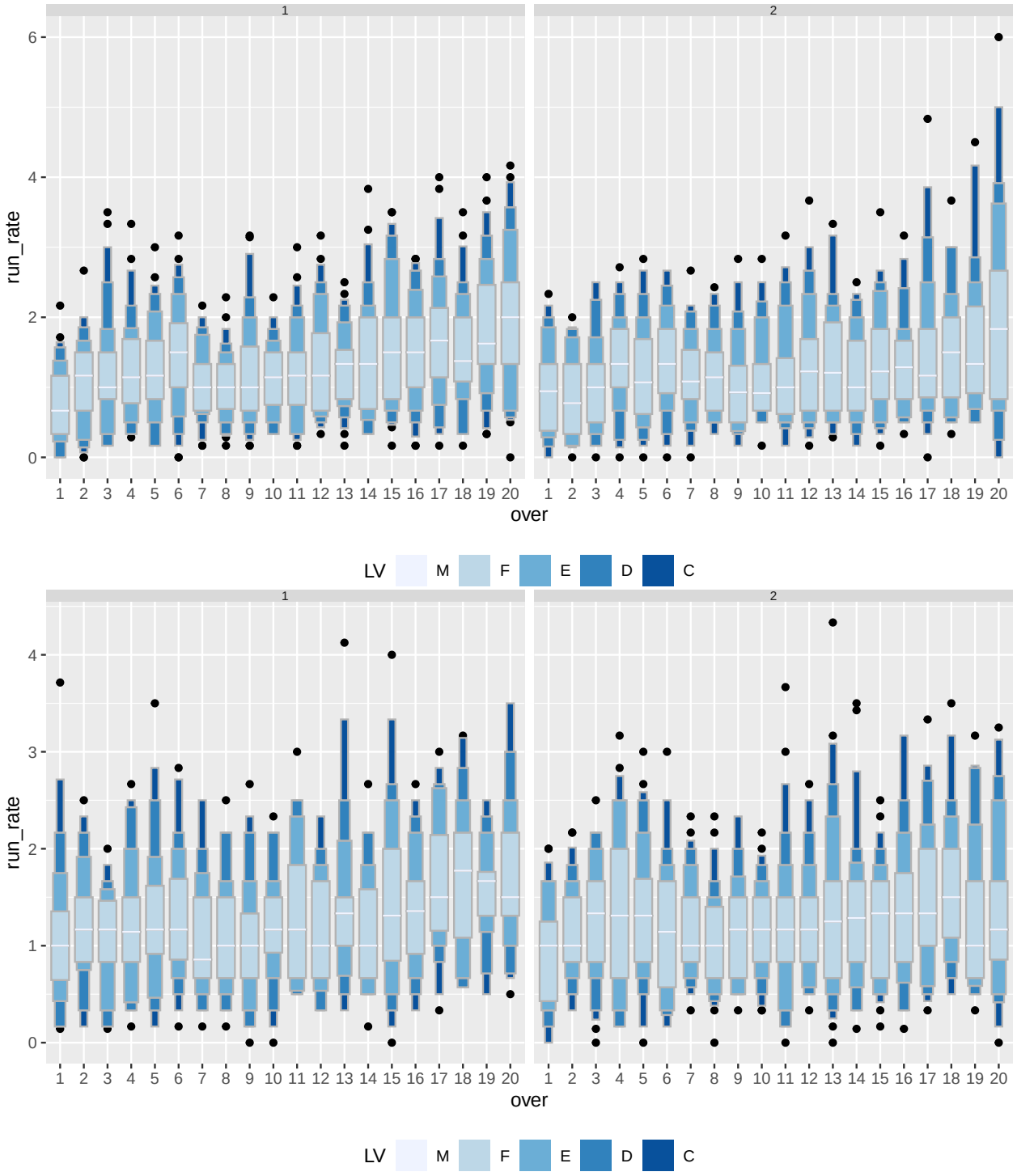


Figure 5: Letter value plot of run rate across overs of an inning and innings of the match for (top) Mumbai Indians and (bottom) Chennai Super Kings.

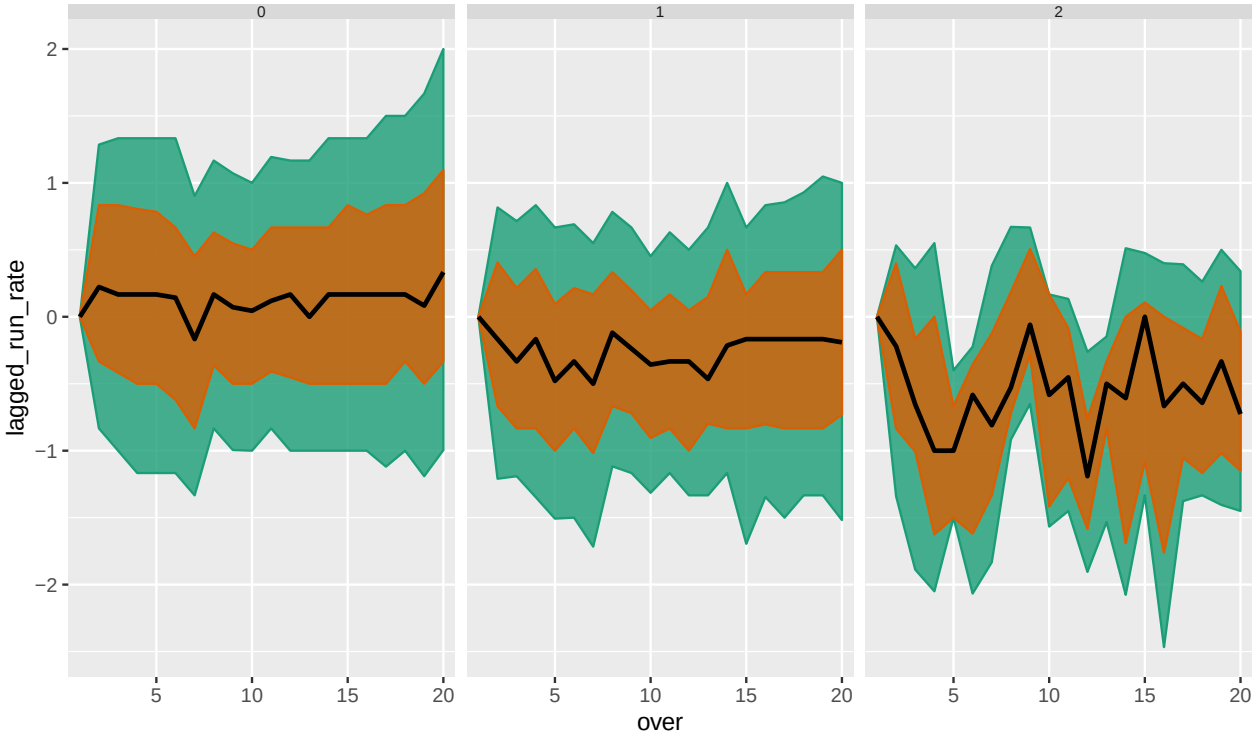


Figure 6: Distribution of lagged run rate across overs of the innings and dismissals by catch and catch and bowled.

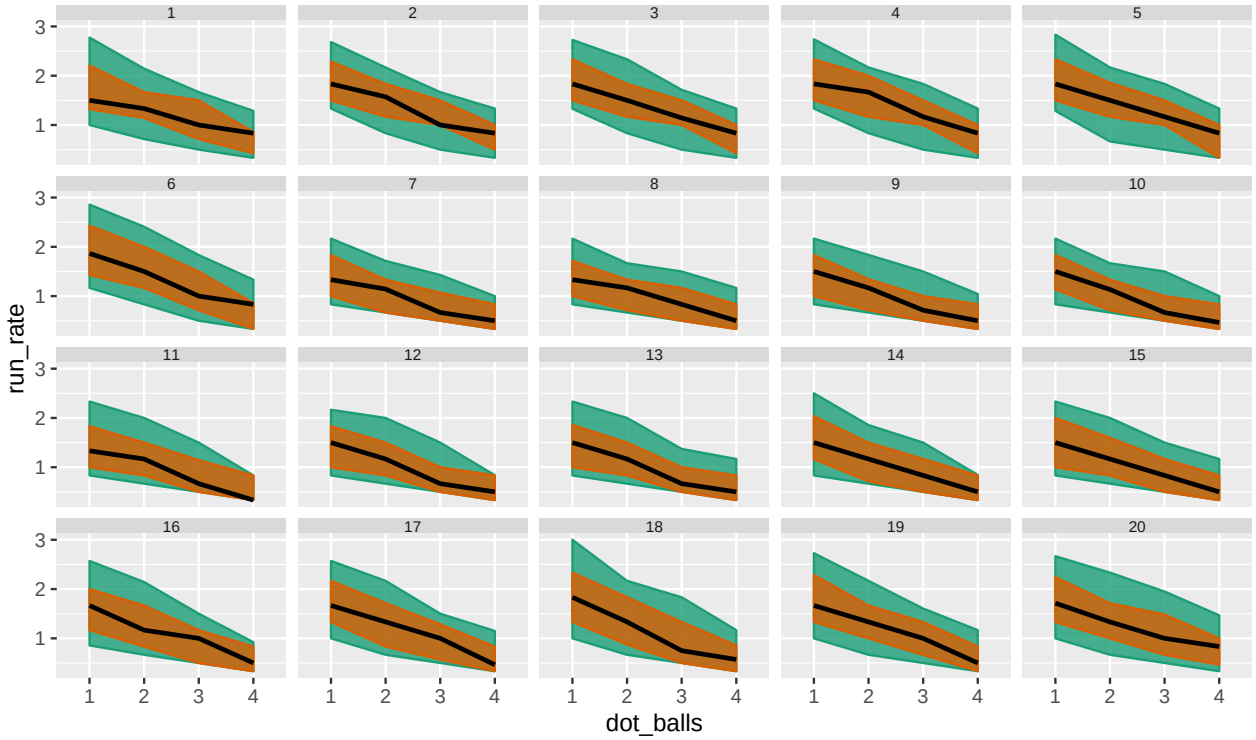


Figure 7: Distribution of run rate across overs of the innings and number of dot balls in an over.