Visualizing Probability Distributions Across Bivariate Cyclic Temporal Granularities

Sayani Gupta\textsuperscript{a}, Rob J Hyndman\textsuperscript{a}, Dianne Cook\textsuperscript{a}, and Antony Unwin\textsuperscript{b}

\textsuperscript{a}Department of Econometrics and Business Statistics, Monash University, Australia; \textsuperscript{b}University of Augsburg, Augsburg, Germany

ABSTRACT

Deconstructing a time index into time granularities can assist in exploration and automated analysis of large temporal datasets. This article describes classes of time deconstructions using linear and cyclic time granularities. Linear granularities respect the linear progression of time such as hours, days, weeks and months. Cyclic granularities can be circular such as hour-of-the-day, quasi-circular such as day-of-the-month, and aperiodic such as public holidays. The hierarchical structure of granularities creates a nested ordering: hour-of-the-day and second-of-the-minute are single-order-up, hour-of-the-week is multiple-order-up, because it passes over day-of-the-week. Methods are provided for creating all possible granularities for a time index. A recommendation algorithm provides an indication whether a pair of granularities can be meaningfully examined together (a “harmony”), or when they cannot (a “clash”). Time granularities can be used to create data visualizations to explore periodicities, associations and anomalies. The granularities form categorical variables (ordered or unordered) which induce groupings of the observations. Assuming a numeric response variable, the resulting graphics are then displays of distributions compared across combinations of categorical variables. The methods implemented in the open source R package gravitas are consistent with a tidy workflow, with probability distributions examined using the range of graphics available in ggplot2. Supplementary files for this article are available online.

1. Introduction

Temporal data are available at various resolutions depending on the context. Social and economic data are often collected and reported at coarse temporal scales such as monthly, quarterly or annually. With recent advancements in technology, more and more data are recorded at much finer temporal scales. Energy consumption may be collected every half an hour, energy supply may be collected every minute, and web search data might be recorded every second. As the frequency of data increases, the number of questions about the periodicity of the observed variable also increases. For example, data collected at an hourly scale can be analyzed using coarser temporal scales such as days, months, or quarters. This approach requires deconstructing time in various possible ways called time granularities (Aigner et al. 2011).

It is important to be able to navigate through all of these time granularities to have multiple perspectives on the periodicity of the observed data. This aligns with the notion of exploratory data analysis (EDA) (Tukey 1977) which emphasizes the use of multiple perspectives on data to help formulate hypotheses before proceeding to hypothesis testing. Visualizing probability distributions conditional on one or more granularities is an indispensable tool for exploration. Analysts are expected to comprehensively explore the many ways to view and consider temporal data. However, the plethora of choices and the lack of a systematic approach to do so quickly can make the task overwhelming.

Calendar-based graphics (Wang, Cook, and Hyndman 2020a) are useful in visualizing patterns in the weekly and monthly structure and are helpful when checking for the effects of weekends or special days. Any temporal data at sub-daily resolution can also be displayed using this type of faceting (Wickham 2016) with days of the week, month of the year, or another sub-daily deconstruction of time. But calendar effects are not restricted to conventional day-of-week or month-of-year deconstructions. There can be many different time deconstructions, based on the calendar or on categorizations of time granularities.

Linear time granularities (such as hours, days, weeks, and months) respect the linear progression of time and are non-repeating. One of the first attempts to characterize these granularities is due to Bettini et al. (1998). However, the definitions and rules defined are inadequate for describing nonlinear granularities. Hence, there is a need to define some new time granularities, which can be useful in visualizations. Cyclic time granularities can be circular, quasi-circular or aperiodic. Examples of circular granularities are hour of the day and day of the week; an example of a quasi-circular granularity is day of the month; examples of aperiodic granularities are public holidays and school holidays.

Time deconstructions can also be based on the hierarchical structure of time. For example, hours are nested within days,
days within weeks, weeks within months, and so on. Hence, it is possible to construct single-order-up granularities such as second of the minute, or multiple-order-up granularities such as second of the hour. The lubricate package (Grolemund and Wickham 2011) provides tools to access and manipulate common date-time objects. However, most of its accessor functions are limited to single-order-up granularities.

The motivation for this work stems from the desire to provide methods to better understand large quantities of measurements on energy usage reported by smart meters in households across Australia, and indeed many parts of the world. Smart meters currently provide half-hourly use in kWh for each household, from the time they were installed, some as early as 2012. Households are distributed geographically and have different demographic properties as well as physical 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 (see Goodwin and Dykes 2012) of usage across households, such as half-hourly total usage by state, because energy companies need to plan for maximum loads on the network. But studying overall energy use hides the distribution of usage at finer scales, and makes it more difficult to find solutions to improve energy efficiency. We propose that the analysis of smart meter data will benefit from systematically exploring energy consumption by visualizing the probability distributions across different deconstructions of time to find regular patterns and anomalies. Although we were motivated by the smart meter example, the problem and the solutions we propose are practically relevant to any temporal data observed more than once per year. In a broader sense, it could be even suitable for data observed by years, decades, and centuries as might be in weather or astronomical data.

This work provides tools for systematically exploring bivariate granularities within the tidy workflow (Grolemund and Wickham 2017). In particular, we

- provide a formal characterization of cyclic granularities;
- facilitate manipulation of single- and multiple-order-up time granularities through cyclic calendar algebra; and
- develop an approach to check the feasibility of creating plots or drawing inferences for any two cyclic granularities.

The remainder of the article is organized as follows: Section 2 provides some background material on linear granularities and calendar algebra for computing different linear granularities. Section 3 formally characterizes different cyclic time granularities by extending the framework of linear time granularities, and introducing cyclic calendar algebra for computing cyclic time granularities. The data structure for exploring the conditional distributions of the associated time series across pairs of cyclic time granularities is discussed in Section 4. Section 5 discusses the role of different factors in constructing an informative and trustworthy visualization. Section 6 examines how systematic exploration can be carried out for a temporal and non-temporal application. Finally, we summarize our results and discuss possible future directions in Section 7.

2. Linear Time Granularities

Discrete abstractions of time such as weeks, months, or holidays can be thought of as “time granularities.” Time granularities are linear if they respect the linear progression of time. There have been several attempts to provide a framework for formally characterizing time granularities, including Bettini et al. (1998) which forms the basis of the work described here.

2.1. Definitions

Definition 1. A time domain is a pair $(T; \leq)$ where $T$ is a nonempty set of time instants (equivalently, moments or points) and $\leq$ is a total order on $T$.

The time domain is assumed to be discrete, and there is unique predecessor and successor for every element in the time domain except for the first and last.

Definition 2. The index set, $Z = \{ z : z \in \mathbb{Z}_{\geq 0} \}$, uniquely maps the time instants to the set of nonnegative integers.

Definition 3. A linear granularity is a mapping $G$ from the index set, $Z$, to subsets of the time domain such that: (1) if $i < j$ and $G(i)$ and $G(j)$ are nonempty, then each element of $G(i)$ is less than all elements of $G(j)$; and (2) if $i < k < j$ and $G(i)$ and $G(j)$ are non-empty, then $G(k)$ is non-empty. Each nonempty subset $G(i)$ is called a granule.

This implies that the granules in a linear granularity are non-overlapping, continuous and ordered. The indexing for each granule can also be associated with a textual representation, called the label. A discrete time model often uses a fixed smallest linear granularity named by Bettini et al. (1998) bottom granularity. Figure 1 illustrates some common linear time granularities. Here, “hour” is the bottom granularity and “day,” “week,” “month,” and “year” are linear granularities formed by mapping the index set to subsets of the hourly time domain. If we have “hour” running from $[0,1,\ldots,t]$, we will have “day” running from $[0,1,\ldots,\lfloor t/24 \rfloor]$. These linear granularities are uni-directional and non-repeating.

2.2. Relativities

Properties of pairs of granularities fall into various categories.

Definition 4. A linear granularity $G$ is finer than a linear granularity $H$, denoted $G \preceq H$, if for each index $i$, there exists an index $j$ such that $G(i) \subset H(j)$.

Definition 5. A linear granularity $G$ groups into a linear granularity $H$, denoted $G \subsetneq H$, if for each index $j$ there exists a (possibly infinite) subset $S$ of the integers such that $H(j) = \bigcup_{i \in S} G(i)$.

For example, both $\text{day} \preceq \text{week}$ and $\text{week} \preceq \text{month}$ hold, since every granule of $\text{week}$ is the union of some set of granules of day and each day is a subset of a week. These definitions are not equivalent. Consider another example, where $G_1$ denotes “weekend” and $H_1$ denotes “week.” Then, $G_1 \preceq H_1$, but $G_1 \not\preceq H_1$. Further, with $G_2$ denoting “days” and $H_2$ denoting “business-week,” $G_2 \not\preceq H_2$. 

A granularity G is periodic with respect to a finite granularity H if: (1) G ⊆ H; and (2) there exist R, P ∈ Z+, where R is less than the number of granules of H, such that for all i ∈ Z≥0, if H(i) = ∪j∈S G(j) and H(i + R) ≠ ∅, then H(i + R) = ∪j∈S G(j + P).

If G groups into H, then it would imply that any granule H(i) is the union of some granules of G, for example, G(a1), G(a2), ..., G(ak). Condition (2) in Definition 6 implies that if H(i + R) ≠ ∅, then H(i + R) = ∪(G(a1 + P), G(a2 + P), ..., G(ak + P)), resulting in a "periodic" pattern of the composition of H using granules of G. In this pattern, each granule of H is shifted by P granules of G. P is called the Period (Bettini, Jajodia, and Wang (2000)).

For example, day is periodic with respect to week with R = 1 and P = 7, while if we ignore leap years day is periodic with respect to month with R = 12 and P = 365 as any month would consist of the same number of days across years. Since the idea of period involves a pair of granularities, we say that the pair (day, week) has period 7, while the pair (day, month) has a period 365 (ignoring leap years).

Granularities can also be periodic with respect to other granularities, “except for a finite number of spans of time where they behave in an anomalous way;” these are called quasi-periodic relationships (Bettini and De Sibi 2000). In a Gregorian calendar with leap years, day groups quasi-periodically into month with the exceptions of the time domain corresponding to February 29 of any year.

Definition 7. The order of a linear granularity is the level of coarseness associated with a linear granularity. A linear granularity G will have lower order than H if each granule of G is composed of lower number of granules of bottom granularity than each granule of H.

With two linear granularities G and H, if G groups into or finer than H then G is of lower order than H. For example, if the bottom granularity is hour, then granularity day will have lower order than week since each day consist of fewer hours than each week.

Granules in any granularity may be aggregated to form a coarser granularity. A system of multiple granularities in lattice structures is referred to as a calendar by Dyreson et al. (2000).

Linear time granularities are computed through “calendar algebra” operations (Ning, Wang, and Jajodia 2002) designed to generate new granularities recursively from the bottom granularity. For example, due to the constant length of day and week, we can derive them from hour using

\[ D(j) = \lfloor H(i)/24 \rfloor, \]
\[ W(k) = \lfloor H(i)/(24 \times 7) \rfloor, \]

where H, D, and W denote hours, days, and weeks, respectively.

3. Cyclic Time Granularities

Cyclic granularities represent cyclical repetitions in time. They can be thought of as additional categorizations of time that are not linear. Cyclic granularities can be constructed from two linear granularities, that relate periodically; the resulting cycles can be either regular (circular), or irregular (quasi-circular).

3.1. Circular Granularities

Definition 8. A circular granularity CB,G relates linear granularity G to bottom granularity B if

\[ C_{B,G}(z) = z \mod P(B, G) \quad \forall z \in Z_{\geq 0}. \]

where z denotes the index set, B groups periodically into G with regular mapping and period P(B, G).

Figure 2 illustrates some linear and cyclical granularities. Cyclical granularities are constructed by cutting the linear granularity into pieces, and stacking them to match the cycles (as shown in b). B, G, H (day, week, fortnight, respectively) are linear granularities. The circular granularity CB,G (day-of-week) is constructed from B and G, while cyclic granularity CB,H (day-of-fortnight) is constructed from B and H. These overlapping cyclic granularities share elements from the linear granularity. Each of CB,G and CB,H consist of repeated patterns \{0, 1, ..., 6\} and \{0, 1, ..., 13\} with P = 7 and P = 14, respectively.

Suppose L is a label mapping that defines a unique label for each index \( \ell \in \{0, 1, \ldots, (P - 1)\} \). For example, the label mapping L for CB,G can be defined as

\[ L : \{0, 1, \ldots, 6\} \rightarrow \{\text{Sunday}, \text{Monday}, \ldots, \text{Saturday}\}. \]

In general, any circular granularity relating two linear granularities can be expressed as

\[ C_{G,H}(z) = \lfloor z/P(G, H) \rfloor \mod P(G, H), \]

where H is periodic with respect to G with regular mapping and period P(G, H). Table 1 shows several circular granularities constructed using minutes as the bottom granularity.
Figure 2. Index sets for some linear and circular granularities (a). Circular granularities can be constructed by slicing the linear granularity into pieces and stacking them (b).

Table 1. Examples of circular granularities with bottom granularity minutes.

| Circular granularity | Expression | Period |
|-----------------------|------------|--------|
| Minute-of-hour        | $C_1 = z \mod 60$ | $P_1 = 60$ |
| Minute-of-day         | $C_2 = z \mod 60 \times 24$ | $P_2 = 1440$ |
| Hour-of-day           | $C_3 = \lfloor z / 60 \rfloor \mod 24$ | $P_3 = 24$ |
| Hour-of-week          | $C_4 = \lfloor z / 60 \rfloor \mod 24 \times 7$ | $P_4 = 168$ |
| Day-of-week           | $C_5 = \lfloor z / 24 \times 60 \rfloor \mod 7$ | $P_5 = 7$ |

NOTES: Circular granularity $C_i$ relates two linear granularities one of which groups periodically into the other with regular mapping and period $P_i$. Circular granularities can be expressed using modular arithmetic due to their regular mapping.

Table 2. Examples of quasi-circular granularities relating two linear granularities with irregular mapping leading to several possible period lengths.

| Quasi-circular granularity | Possible period lengths |
|----------------------------|-------------------------|
| $Q_1 = \text{day-of-month}$ | $P_1 = 31, 30, 29, 28$ |
| $Q_2 = \text{hour-of-month}$ | $P_2 = 24 \times 31, 24 \times 30, 24 \times 29, 24 \times 28$ |
| $Q_3 = \text{day-of-year}$ | $P_3 = 366, 365$ |

3.2. Quasi-Circular Granularities

A quasi-circular granularity cannot be defined using modular arithmetic because of the irregular mapping. However, they are still formed with linear granularities, one of which groups periodically into the other. Table 2 shows some examples of quasi-circular granularities.

Definition 9. A quasi-circular granularity $Q_{B,G}$ is formed when bottom granularity $B$ groups periodically into linear granularity $G'$ with irregular mapping such that the granularities are given by

$$Q_{B,G}(z) = z - \sum_{w=0}^{k-1} |T_w \mod R' |, \quad \text{for } z \in T_k,$$

where $z$ denotes the index set, $w$ denotes the index of $G'$, $R'$ is the number of granules of $G'$ in each period of $(B, G')$, $T_w$ are the sets of indices of $B$ such that $G'(w) = \bigcup_{z \in T_w} B(z)$, and $|T_w|$ is the cardinality of set $T_w$.

For example, day-of-year is quasi-periodic with either 365 or 366 granules of $B$ (days) within each granule of $G'$ (years). The pattern repeats every 4 years (ignoring leap seconds). Hence $R' = 4$. $Q_{B,G}$ is a repetitive categorization of time, similar to circular granularities, except that the number of granules of $B$ is not the same across different granules of $G'$.

3.3. Aperiodic Granularities

Aperiodic linear granularities are those that cannot be specified as a periodic repetition of a pattern of granules as described in Definition 6. Aperiodic cyclic granularities capture repetitions of these aperiodic linear granularities. Examples include public holidays which repeat every year, but there is no reasonably small span of time within which their behavior remains constant. A classic example is Easter (in the western tradition) whose dates repeat only after 5.7 million years (Reingold and Dershowitz 2018). In Australia, if a standard public holiday falls on a weekend, a substitute public holiday will sometimes be observed on the first non-weekend day (usually Monday) after the weekend. Examples of aperiodic granularity may also include school holidays or a scheduled event. All of these are recurring events, but with non-periodic patterns. Consequently, $P_i$ (as given in Table 2) are essentially infinite for aperiodic granularities.

Definition 10. An aperiodic cyclic granularity is formed when bottom granularity $B$ groups into an aperiodic linear granularity $M$ such that the granularities are given by

$$A_{B,M}(z) = \begin{cases} i, & \text{for } z \in T_i \\ 0, & \text{otherwise} \end{cases},$$

where $i$ is the index of $G'$, $R'$ is the number of granules of $G'$ in each period of $M$, and $T_i$ are the sets of indices of $B$ such that $G'(w) = \bigcup_{z \in T_i} B(z)$.
where \( z \) denotes the index set, \( T_i \) are the sets of indices of \( B \) describing aperiodic linear granularities \( M_i \) such that \( M_i(j) = \bigcup_{z \in T_i} B(z) \), and \( M = \bigcup_{i=1}^{n} M_i \), \( n \) being the number of aperiodic linear granularities in consideration.

For example, consider the school semester shown in Figure 3. Let the linear granularities \( M_1 \) and \( M_2 \) denote the teaching and nonteaching stages of the semester respectively. Both \( M_1 \), \( M_2 \) and \( M = M_1 \cup M_2 \) denoting the "stages" of the semester are aperiodic with respect to days (\( B \)) or weeks (\( G \)). Hence, \( A_{B,M} \) denoting day-of-the-stage would be an aperiodic cyclic granularity because the placement of the semester within a year would vary across years. Here, \( Q_{B,M} \) denoting semester-day-of-the-stage would be a quasi-circular granularity since the distribution of semester days within a semester is assumed to remain constant over years. Here semester-day is denoted by "sem day" (\( B' \)) and its granules are only defined for the span of the semesters.

### 3.4. Relativities

The hierarchical structure of time creates a natural nested ordering which can be used in the computation of relative pairs of granularities.

**Definition 11.** The nested ordering of linear granularities can be organized into a hierarchy table, denoted as \( H_n : (G, C, K) \), which arranges them from lowest to highest in order. It shows how the \( n \) granularities relate through \( K \), and how the cyclic granularities, \( C \), can be defined relative to the linear granularities. Let \( G_\ell \) and \( G_m \) represent the linear granularity of order \( \ell \) and \( m \), respectively, with \( \ell < m \). Then \( K \equiv P(\ell, m) \) represents the period length of the grouping \( (G_\ell, G_m) \), if \( C_{G_\ell, G_m} \) is a circular granularity and \( K \equiv k(\ell, m) \) represents the operation to obtain \( G_m \) from \( G_\ell \), if \( C_{G_\ell, G_m} \) is quasi-circular.

![Figure 3. Quasi-circular and aperiodic cyclic granularities illustrated by linear (a) and stacked-displays (b) progression of time. The linear display shows the granularities days (\( B \)), weeks (\( G \)), semester days (\( B' \)), and stages of a semester (\( M \)) indexed over a linear representation of time. The granules of \( B' \) is only defined for days when the semester is running. Here a semester spans 18 weeks and 2 days, and consists of 6 stages. It starts with a week of orientation, followed by an in-session period (6 weeks), a break (1 week), the second half of semester (7 weeks), a 1-week study break before final exams, which spans the next 16 days. This distribution of semester days remains relatively similar for every semester. \( Q_{B,M} \) with \( P = 128 \) is a quasi-circular granularity with repeating patterns, while \( A_{B,M} \) is an aperiodic cyclic granularity as the placement of the semester within a year varies from year to year with no fixed start and end dates.

| Linear (G) | Single-order-up cyclic (C) | Period length/conversion operator (K) |
|------------|-----------------------------|--------------------------------------|
| Kin        | Kin-of-uinal                | 20                                   |
| Uinal      | Uinal-of-tun                | 18                                   |
| Tun        | Tun-of-katun                | 20                                   |
| Katun      | Katun-of-baktun             | 20                                   |
| Baktun     |                             | 1                                    |

For example, Table 3 shows the hierarchy table for the Mayan calendar. In the Mayan calendar, one day was referred to as a kin and the calendar was structured such that 1 kin = 1 day; 1 uinal = 20 kin; 1 tun = 18 uinals (about a year); 1 katun = 20 tun (20 years) and 1 baktun = 20 katuns. Like most calendars, the Mayan calendar used the day as the basic unit of time (Reingold and Dershowitz 2018). The structuring of larger units, weeks, months, years, and cycle of years, though, varies substantially between calendars. For example, the French revolutionary calendar divided each day into 10 "hours", each "hour" into 100 "minutes" and each "minute" into 100 "seconds", the duration of which is 0.864 common seconds. Nevertheless, for any calendar, a hierarchy table can be defined. Note that it is not always possible to organize an aperiodic linear granularity in a hierarchy table. Hence, we assume that the hierarchy table consists of periodic linear granularities only, and that the cyclic granularity \( C_{G(\ell), G(m)} \) is either circular or quasi-circular.

### 3.4. Relativities
In the Mayan calendar (Table 3), kin-of-tun or kin-of-baktun are examples of multiple-order-up granularities and single-order-up granularities are kin-of-unial, unial-of-tun etc.

3.5. Computation

Following the calendar algebra of Ning, Wang, and Jajodia (2002) for linear granularities, we can define cyclic calendar algebra to compute cyclic granularities. Cyclic calendar algebra comprises two kinds of operations: (i) single-to-multiple (the calculation of multiple-order-up cyclic granularities from single-order-up cyclic granularities) and (ii) multiple-to-single (the reverse).

3.5.1. Single-to-Multiple Order-up

Methods to obtain multiple-order-up granularity will depend on whether the hierarchy consists of all circular single-order-up granularities or a mix of circular and quasi-circular single-order-up granularities. Circular single-order-up granularities can be used recursively to obtain a multiple-order-up circular granularity using

\[ C_{G_1,G_m}(z) = \sum_{i=0}^{m-\ell-1} P(\ell, \ell + i) C_{G_{i+1},G_{i+1}}(z), \] (4)

where \( \ell < m - 1 \) and \( P(i, i) = 1 \) for \( i = 0, 1, \ldots, m - \ell - 1 \), and \( C_{B,G}(z) = z \mod P(B, G) \) as per Equation (1).

For example, the multiple-order-up granularity \( C_{\text{unial},k\text{atun}} \) for the Mayan calendar could be obtained using Equation (7)

\[ C_{\text{unial},\text{baktun}}(z) = C_{\text{unial},\text{tun}}(z) + P(\text{unial, tun}) C_{\text{tun},k\text{atun}}(z) \\
+ P(\text{unial, k\text{atun}}) C_{k\text{atun},\text{baktun}}(z) \\
= C_{\text{unial},\text{tun}}(z) + 18 \times C_{\text{tun},k\text{atun}}(z) \\
+ 18 \times 20 \times C_{k\text{atun},\text{baktun}}(z), \]

where \( z \) is the index of the bottom granularity \( \text{kin} \).

Now consider the case where there is one quasi-circular single order-up granularity in the hierarchy table while computing a multiple-order-up quasi-circular granularity. Any multiple-order-up quasi-circular granularity \( C_{\ell,m}(z) \) could then be obtained as a discrete combination of circular and quasi-circular granularities.

Depending on the order of the combination, two different approaches need to be employed leading to the following cases:

- \( C_{G_1,G_{m'}} \) is circular and \( C_{G_{m'},G_m} \) is quasi-circular
  \[ C_{G_1,G_m}(z) = C_{G_1,G_{m'}}(z) + P(\ell, m') C_{G_{m'},G_m}(z) \] (5)

- \( C_{G_1,G_{m'}} \) is quasi-circular and \( C_{G_{m'},G_m} \) is circular
  \[ C_{G_1,G_m}(z) = C_{G_1,G_{m'}}(z) + \sum_{w=0}^{C_{G_{m'},G_m}(z)-1} (|T_w|) \] (6)

where, \( T_w \) is such that \( G_{m'}(w) = \bigcup_{\ell \in T_w} G_\ell \) and \( |T_w| \) is the cardinality of set \( T_w \).

For example, the Gregorian calendar (Table 4) has day-of-month as a single-order-up quasi-circular granularity, with the other granularities being circular. Using Equations (5) and (6), we then have:

\[ C_{\text{hour},\text{month}}(z) = C_{\text{hour},\text{day}}(z) + P(\text{hour, day}) \times C_{\text{day},\text{month}}(z) \]

\[ C_{\text{day},\text{year}}(z) = C_{\text{day},\text{month}}(z) + \sum_{w=0}^{C_{\text{month},\text{year}}(z)-1} (|T_w|), \]

where \( T_w \) is such that month(\( w \)) = \( \bigcup_{\ell \in T_w} \text{day}(\ell) \).

3.5.2. Multiple-to-Single Order-Up

Similar to single-to-multiple operations, multiple-to-single operations involve different approaches for all circular single-order-up granularities and a mix of circular and quasi-circular single-order-up granularities in the hierarchy. For a hierarchy table \( H_n : (G, C, K) \) with only circular single-order-up granularities and \( \ell_1, \ell_2, m_1, m_2 \in 1, 2, \ldots, n \) and \( \ell_2 < \ell_1 \) and \( m_2 > m_1 \), multiple-order-up granularities can be obtained using (7).

\[ C_{G_1,G_{m_1}}(z) = \left\lfloor \frac{C_{G_{m_1},G_{m_2}}(z)}{P(\ell_2, \ell_1)} \right\rfloor \mod P(\ell_1, m_1) \] (7)

For example, in the Mayan Calendar, it is possible to compute the single-order-up granularity tun-of-katun from unial-of-baktun, since \( C_{\text{unial},\text{katun}}(z) = \left\lfloor C_{\text{unial},\text{baktun}}(z)/18 \right\rfloor \mod 20 \).

3.5.3. Multiple Order-Up Quasi-Circular Granularities

Single-order-up quasi-circular granularity can be obtained from multiple-order-up quasi-circular granularity and single/multiple-order-up circular granularity using Equations (5) and (6).

4. Data Structure

Effective exploration and visualization benefit from well-organized data structures. Wang, Cook and Hyndman (2020b) introduced the tidy “tsibble” data structure to support exploration and modeling of temporal data. This forms the basis of the structure for cyclic granularities. A tsibble comprises an index, optional key(s), and measured variables. An index is a variable with inherent ordering from past to present and a key is a set of variables that define observational units over time. A linear granularity is a mapping of the index set to subsets of the time domain. For example, if the index of a tsibble is days, then a linear granularity might be weeks, months or years. A bottom granularity is represented by the index of the tsibble.

All cyclic granularities can be expressed in terms of the index set. Table 5 shows the tsibble structure (index, key, measurements) augmented by columns of cyclic granularities. The total number of cyclic granularities depends on the number of
linear granularities considered in the hierarchy table and the presence of any aperiodic cyclic granularities. For example, if we have \( n \) periodic linear granularities in the hierarchy table, then \( n(n - 1)/2 \) circular or quasi-circular cyclic granularities can be constructed. Let \( N_C \) be the total number of contextual circular, quasi-circular and aperiodic cyclic granularities that can originate from the underlying periodic and aperiodic linear granularities. Simultaneously encoding more than a few of these cyclic granularities when visualizing the data overwhelsms human comprehension. Instead, we focus on visualizing the data split by pairs of cyclic granularities \((C_i, C_j)\). Dataset of the form \( <C_i, C_j, v> \) then allow exploration and analysis of the measured variable \( v \).

### 4.1. Harmonies and Clashes

The way granularities are related is important when we consider data visualizations. Consider two cyclic granularities \( C_i \) and \( C_j \), such that \( C_i \) maps index set to a set \( \{A_k \mid k = 1, \ldots, K\} \) and \( C_j \) maps index set to a set \( \{B_\ell \mid \ell = 1, \ldots, L\} \). Here, \( A_k \) and \( B_\ell \) are the levels/categories corresponding to \( C_i \) and \( C_j \) respectively. Let \( S_{kj} \) be a subset of the index set such that for all \( s \in S_{kj} \), \( C_i(s) = A_k \) and \( C_j(s) = B_\ell \). There are \( KL \) such data subsets, one for each combination of levels \((A_k, B_\ell)\). Some of these sets may be empty due to the structure of the calendar, or because of the duration and location of events in a calendar.

**Definition 13.** A clash is a pair of cyclic granularities that contains empty combinations of categories.

**Definition 14.** A harmony is a pair of cyclic granularities that does not contain any empty combinations of its categories.

Structurally empty combinations can arise due to the structure of the calendar or hierarchy. For example, let \( C_i \) be day-of-month with 31 levels and \( C_j \) be day-of-year with 365 levels. There will be \( 31 \times 365 = 11315 \) sets \( S_{kl} \) corresponding to possible combinations of \( C_i \) and \( C_j \). Many of these are empty. For example, \( S_{131} \) is empty because the first day of the month can never correspond to the fifth day of the year. Hence the pair (day-of-month, day-of-year) is a clash.

Event-driven empty combinations arise due to differences in event location or duration in a calendar. For example, let \( C_i \) be day-of-week with 7 levels and \( C_j \) be working-day/non-working-day with 2 levels. While potentially all of these 14 sets \( S_{kl} \) can be nonempty (it is possible to have a public holiday on any day of-week), in practice many of these 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, Australia.

An example of harmony is where \( C_i \) and \( C_j \) denote day-of-week and month-of-year, respectively. So \( C_i \) will have 7 levels while \( C_j \) will have 12 levels, giving \( 12 \times 7 = 84 \) sets \( S_{kl} \). All of these are non-empty because every day-of-week can occur in every month. Hence, the pair (day-of-week, month-of-year) is a harmony.

### 4.2. Near-Clashes

Suppose \( C_i \) denotes day-of-year and \( C_j \) denotes day-of-week. While any day of the week can occur on any day of the year, some combinations will be very rare. For example, the 366th day of the year will only coincide with a Wednesday approximately every 28 years on average. We refer to these as “near-clashes.”

### 5. Visualization

The purpose is to visualize the distribution of the continuous variable \( v \) conditional on the values of two granularities, \( C_i \) and \( C_j \). Since \( C_i \) and \( C_j \) are factors or categorical variables, data subsets corresponding to each combination of their levels form a subgroup and the visualization amounts to having displays of distributions for different subgroups. The response variable \( v \) is plotted on the \( y \)-axis and the levels of \( C_i(C_j) \) on the \( x \)-axis, conditional on the levels of \( C_j(C_i) \). This means, carrying out the same plot corresponding to each level of the conditioning variable. This is consistent with the widely used grammar of graphics which is a framework to construct statistical graphics by relating the data space to the graphic space (Wilkinson 1999, Wickham (2016)).

#### 5.1. Data Summarization

There are several ways to summarize the distribution of a dataset such as estimating the empirical distribution or density of the data, or computing a few quantiles or other statistics. This estimation or summarization could be potentially misleading if it is performed on rarely occurring categories (Section 4.2). Even when there are no rarely occurring events, the number of observations may vary greatly within or across each facet, due to missing observations or uneven locations of events in the time domain. In such cases, data summarization should be used with caution as sample sizes will directly affect the accuracy of the estimated quantities being displayed.

#### 5.2. Display Choices for Univariate Distributions

The basic plot choice for our data structure is one that can display distributions. For displaying the distribution of a continuous univariate variable, many options are available. Displays based on descriptive statistics include boxplots (Tukey 1977) and its variants such as notched boxplots (McGill, Tukey, and Larsen 1978) or other variations as mentioned in Wickham and Stryjewski (2012). They also include line or area quantile plots which can display any quantiles and not only quartiles like in a boxplot. Plots based on kernel density estimates include violin plots (Hintze and Nelson 1998), summary plot (Potter et al. 2010), ridge line plots (Wilke 2020), and the highest density region (HDR) plots (Hyndman 1996). The less commonly used Letter-Value plots (Hofmann, Wickham, and Kafadar 2017) is midway between boxplots and density plots. Letter values are order statistics with specific depths, for example, the median.
(M) is a letter value that divides the dataset into halves. Each of the next letter values splits the remaining parts into two separate regions so that the fourths (F), eighths (E), sixteenths (D), etc. are obtained. They are useful for displaying the distributions beyond the quartiles especially for large data, where boxplots mislabel data points as outliers. One of the best approaches in exploratory data analysis is to draw a variety of plots to reveal information while keeping in mind the drawbacks and benefits of each of the plot choices. For example, boxplots obscure multimodality, and interpretation of density estimates and histograms may change depending on the bandwidth and bin-widths, respectively. In R package gravitas (Gupta et al. 2020), boxplots, violin, ridge, letter-value, line and area quantile plots are implemented, but it is potentially possible to use any plots which can display the distribution of the data.

5.3. Comparison Across Sub-Groups Induced by Conditioning

5.3.1. Levels
The levels of cyclic granularities affect plotting choices since space and resolution may be problematic with too many levels. A potential approach could be to categorize the number of levels as low/medium/high/very high for each cyclic granularity and define some criteria based on human cognitive power, available display size and the esthetic mappings. Default values for these categorizations could be chosen based on levels of common temporal granularities like days of the month, days of the fortnight, or days of the week.

5.3.2. Synergy of Cyclic Granularities
The synergy of the two cyclic granularities will affect plotting choices for exploratory analysis. Cyclic granularities that form clashes (Section 4.1) or near-clashes lead to potentially ineffectual graphs. Harmonies tend to be more useful for exploring patterns. Figure 4(a) shows the distribution of half-hourly electricity consumption through letter value plots across months of the year conditional on quarters of the year. This plot does not work because quarter-of-year clashes with month-of-year, leading to empty subsets. For example, the first quarter never corresponds to December.

5.3.3. Conditioning variable
When \( C_i \) is mapped to the x position and \( C_j \) to facets, then the \( A_k \) levels are juxtaposed and each \( B_l \) represents a group/facet. Gestalt theory suggests that when items are placed in close proximity, people assume that they are in the same group because they are close to one another and apart from other groups. Hence, in this case, the \( A_k \)'s are compared against each other within each group. With the mapping of \( C_i \) and \( C_j \) reversed, the emphasis will shift to comparing \( B_l \) levels rather than \( A_k \) levels. For example, Figure 4(b) shows the letter value plot across weekday/weekend partitioned by quarters of the year and Figure 4(c) shows the same two cyclic granularities with their mapping reversed. Figure 4(b) helps us to compare weekday and weekend within each quarter and Figure 4(c) helps to compare quarters within weekend and weekday.

6. Applications

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 trials (Department of the Environment and Energy 2018) conducted as part of the Smart Grid Smart City project in Newcastle and parts of Sydney provides customer level data on energy consumption for every half hour from February 2012 to March 2014. We can use this dataset to visualize the distribution of energy consumption across different cyclic granularities in a systematic way to identify different behavioral patterns.

6.1.1. Cyclic Granularities Search and Computation
The tsibble object smart_meter10 from R package gravitas (Gupta et al. 2020) includes the variables reading_datetime, customer_id and general_supply_kwh denoting the index, key and measured variable respectively. The interval of this tsibble is 30 min.

To identify the available cyclic time granularities, consider the conventional time deconstructions for a Gregorian calendar that can be formed from the 30-min time index: half-hour, hour, day, week, month, quarter, half-year, and year. In this example, we will consider the granularities hour, day, week and month giving six cyclic granularities “hour_day,” “hour_week,” “hour_month,” “day_week,” “day_month,” and “week_month,” read as “hour of the day,” etc. To these, we add day-type (“wknd_wday”) to capture weekend and weekday behavior. Now that we have a list of cyclic granularities to look at, we can compute them using the results in Section 3.4.

6.1.2. Screening and Visualizing Harmonies
Using these seven cyclic granularities, we want to explore patterns of energy behavior. Each of these seven cyclic granularities can either be mapped to the x-axis or to facets. Choosing 2 of the possible 7 granularities, gives \( T \) = 42 candidates for visualization. Harmonies can be identified among those 42 possibilities to narrow the search. Table 6 shows 16 harmony pairs after removing clashes and any cyclic granularities with

| Facet variable | x-axis variable | Facet levels | x-axis levels |
|---------------|----------------|--------------|--------------|
| day_week      | hour_day       | 7            | 24           |
| day_month     | hour_day       | 31           | 24           |
| week_month    | hour_day       | 5            | 24           |
| wknd_wday     | hour_day       | 2            | 7            |
| hour_day      | day_week       | 24           | 7            |
| day_month     | day_week       | 31           | 7            |
| week_month    | day_week       | 5            | 7            |
| hour_day      | day_month      | 24           | 31           |
| day_week      | day_month      | 7            | 31           |
| wknd_wday     | day_month      | 2            | 31           |
| hour_day      | week_month     | 5            | 24           |
| day_week      | week_month     | 5            | 7            |
| wknd_wday     | week_month     | 2            | 5            |
| hour_day      | wknd_wday      | 24           | 2            |
| day_month     | wknd_wday      | 31           | 2            |
| week_month    | wknd_wday      | 5            | 2            |
more than 31 levels, as effective exploration becomes difficult with many levels (Section 5.3).

A few harmony pairs are displayed in Figure 5 to illustrate the impact of different distribution plots and reverse mapping. For each of Figure 5(b) and (c), $C_i$ denotes day-type (weekday/weekend) and $C_j$ is hour-of-day. The geometry used for displaying the distribution is chosen as area-quantiles and violins in Figure 5(b) and (c), respectively. Figure 5(a) shows the reverse mapping of $C_i$ and $C_j$ with $C_i$ denoting hour-of-day and $C_j$ denoting day-type with distribution geometrically displayed as boxplots.

In Figure 5(b), the black line is the median, whereas the purple (narrow) band covers the 25th to 75th percentile, the orange (middle) band covers the 10th to 90th percentile, and the green (broad) band covers the 1st to 99th percentile. The first facet represents the weekday behavior while the second facet displays the weekend behavior; energy consumption across each hour of the day is shown inside each facet. The energy consumption is extremely skewed with the 1st, 10th, and 25th percentile lying relatively close whereas 75th, 90th, and 99th lying further away from each other. This is common across both weekdays and weekends. For the first few hours on weekdays, median energy consumption starts and continues to be higher for longer compared to weekends.

The same data are shown using violin plots instead of quantile plots in Figure 5(c). There is bimodality in the early hours of the day for weekdays and weekends. If we visualize the same data with reverse mapping of the cyclic granularities (Figure 5(a)), then the natural tendency would be to compare weekend and weekday behavior within each hour and not across hours. Then it can be seen that median energy consumption for the early morning hours is higher for weekdays than weekends. Also, outliers are more prominent in the later hours of the day. All of these indicate that looking at different distribution geometries or changing the mapping can shed light on different aspects of energy behavior for the same sample.

### 6.2. T20 cricket Data of Indian Premier League

Our proposed approach can be generalized to other hierarchical granularities where there is an underlying ordered index. We illustrate this with data from the sport cricket. Cricket is played with two teams of 11 players each, with each team taking turns batting and fielding. This is similar to baseball, wherein the batsman and bowler in cricket are analogous to a batter and pitcher in baseball. A wicket is a structure with three sticks, stuck into the ground at the end of the cricket pitch behind the batsman. One player from the fielding team acts as the bowler, while another takes up the role of the wicket-keeper (similar to a catcher in baseball). The bowler tries to hit the wicket with a ball, and the batsman defends the wicket using a bat. At any one time, two of the batting team and all of the fielding team are on the field. The batting team aims to score as many runs as possible,
JOURNAL OF COMPUTATIONAL AND GRAPHICAL STATISTICS

Figure 5. Energy consumption of a single customer shown with different distribution displays, and granularity arrangements: hour of the day; and weekday/weekend. (a) The side-by-side boxplots make the comparison between day types easier, and suggest that there is generally lower energy use on the weekend. Interestingly, this is the opposite to what might be expected. Plots (b) and (c) examine the temporal trend of consumption over the course of a day, separately for the type of day. The area quantile emphasizes time, and indicates that median consumption shows prolonged high usage in the morning on weekdays. The violin plot emphasizes subtler distributional differences across hours: morning use is bimodal.

Table 7. Hierarchy table for cricket where overs are nested within an innings, innings nested within a match and matches within a season.

| Linear (G) | Single-order-up cyclic (C) | Period length/conversion operator (K) |
|------------|-----------------------------|--------------------------------------|
| Over       | Over-of-inning              | 20                                   |
| Inning     | Inning-of-match             | 2                                    |
| Match      | Match-of-season             | k(match, season)                     |
| Season     |                             | 1                                    |

while the fielding team aims to successively dismiss 10 players from the batting team. The team with the highest number of runs wins the match.

Cricket is played in various formats and Twenty20 cricket (T20) is a shortened format, where the two teams have a single innings each, which is restricted to a maximum of 20 overs. An over will consist of 6 balls (with some exceptions). A single match will consist of 2 innings and a season consists of several matches. Although there is no conventional time component in cricket, each ball can be thought to represent an ordering over the course of the game. Then, we can conceive a hierarchy where the ball is nested within overs, overs nested within innings, innings within matches, and matches within seasons. Cyclic granularities can be constructed using this hierarchy. Example granularities include ball of the over, over of the innings, and ball of the innings. The hierarchy table is given in Table 7. Although most of these cyclic granularities are circular by the design of the hierarchy, in practice some granularities are aperiodic. For example, most overs will consist of six balls, but there are exceptions due to wide balls, no-balls, or when an innings finishes before the over finishes. Thus, the cyclic granularity ball-of-over may be aperiodic.

The Indian Premier League (IPL) is a professional T20 cricket league in India contested by eight teams representing eight different cities in India. The IPL ball-by-ball data are provided in the cricket dataset in the gravis package for a sample of 214 matches spanning 9 seasons (2008–2016).

Many interesting questions could be addressed with the cricket dataset. For example, does the distribution of total runs depend on whether a team bats in the first or second innings? The Mumbai Indians (MI) and Chennai Super Kings (CSK) appeared in the final playoffs from 2010 to 2015. Using data from these two teams, it can be observed (Figure 6(a)) that for the team batting in the first innings there is an upward trend of runs per over, while there is no clear upward trend in the median and quartile deviation of runs for the team batting in the second innings after the first few overs. This suggests that players feel mounting pressure to score more runs as they approach the end of the first innings, while teams batting second have a set target in mind and are not subjected to such mounting pressure and therefore may adopt a more conservative run-scoring strategy.

Another question that can be addressed is if good fielding or bowling (defending) in the previous over affects the scoring rate in the subsequent over? To measure the defending quality,
we use an indicator function on dismissals (1 if there was at least one wicket in the previous over, 0 otherwise). The scoring rate is measured by runs per over. Figure 6(b) shows that no dismissals in the previous over leads to a higher median and quartile spread of runs per over compared to the case when there has been at least one dismissal in the previous over. This seems to be unaffected by the over of the innings (the faceting variable). This might be because the new batsman needs to “play himself in” or the dismissals lead the (not-dismissed) batsman to adopt a more defensive playstyle. Run rates will also vary depending on which player is facing the next over and when the wicket falls in the previous over.

Here, wickets per over is an aperiodic cyclic granularity, so it does not appear in the hierarchy table. These are similar to holidays or special events in temporal data.

7. Discussion

Exploratory data analysis involves many iterations of finding and summarizing patterns. With temporal data available at ever finer scales, exploring periodicity can become overwhelming with so many possible granularities to explore. This work provides tools to classify and compute possible cyclic granularities from an ordered (usually temporal) index. We also provide a framework to systematically explore the distribution of a univariate variable conditional on two cyclic time granularities using visualizations based on the synergy and levels of the cyclic granularities.

The gravitas package provides very general tools to compute and manipulate cyclic granularities, and to generate plots displaying distributions conditional on those granularities.
A missing piece in the package gravitas is the computation of cyclic aperiodic granularities which would require computing aperiodic linear granularities first. A few R packages including almanac (Vaughan 2020) and gs (Laird-Smith 2020) provide the tools to create recurring aperiodic events. These functions can be used with the gravitas package to accommodate aperiodic cyclic granularities. We propose producing plots based on pairs of cyclic granularities that form harmonies rather than clashes or near-clashes. A future direction of work could be to further refine the selection of appropriate pairs of granularities by identifying those for which the differences between the displayed distributions is greatest and rating these selected harmony pairs in order of importance for exploration.

Acknowledgments

The Australian authors thank to the ARC Centre of Excellence for Mathematical and Statistical Frontiers (ACEMS) for supporting this research. Thanks to Data61 CSIRO for partially funding Sayani’s research and Dr Peter Toscas for providing useful inputs on improving the analysis of the smart meter application. We thank to Nicholas Spyrison for many useful discussions, sketching figures and feedback on the manuscript. The package gravitas was built during the Google Summer of Code, 2019. More details about the package can be found at sayami07.github.io/gravitas. The Github repository, github.com/Sayani07/paper-gravitas, contains all materials required to reproduce this article and the code is also available online in the supplemental materials. This article was created with knitr (Xie 2020, 2015) and rmarkdown (Xie, Allaire, and Grolemund 2018).

ORCID

Rob Hyndman  
http://orcid.org/0000-0002-2140-5352
Dianne Cook  
http://orcid.org/0000-0002-3813-7155

Supplementary Materials

Data and scripts: Datasets and R code to reproduce all figures in this article (main.R).

R-package: The ideas presented in this article have been implemented in the open-source R (R Core Team 2020) package gravitas (Gupta et al. 2020), available from CRAN. The R-package facilitates manipulation of single and multiple-order-up time granularities through cyclic calendar algebra, checks feasibility of creating plots or drawing inferences for any two cyclic granularities by providing list of harmonies and recommends possible visual summaries through factors described in the article. Version 0.1.3 of the package was used for the results presented in the article and is available on Github (https://github.com/Sayani07/gravitas).

References

Aigner, W., Miksch, S., Schumann, H., and Tominski, C. (2011), Visualization of Time-Oriented Data, London: Springer Science & Business Media. [14]

Allaire, J., Xie, Y., McPherson, J., Luraschi, J., Ushey, K., Atkins, A., Wickham, H., Cheng, J., Chang, W., and Iannone, R. (2020), Rmarkdown: Dynamic Documents for R, R package version 2.1. Available at https://github.com/rstudio/rmarkdown

Bettini, C., and De Sibi, R. (2000), “Symbolic Representation of User-Defined Time Granularities,” Annals of Mathematics and Artificial Intelligence, 30, 53–92. [16]

Bettini, C., Dyreson, C. E., Evans, W. S., Snodgrass, R. T., and Wang, X. S. (1998), “A Glossary of Time Granularity Concepts,” in Temporal Databases: Research and Practice, eds. O. Etzion, S. Jajodia, and S. Sri-pada, Berlin: Springer Berlin Heidelberg, pp. 406–413. [14,15]

Bettini, C., Jajodia, S., and Wang, S. (2000), Time Granularities in Databases, Data Mining, and Temporal Reasoning, Springer Science & Business Media. Berlin: Springer. [16]

Department of the Environment and Energy. (2018), Smart-Grid Smart-City Customer Trial Data, Australian Government, Department of the Environment and Energy. Available at https://data.gov.au/dataset/421de3a-9b87-4610-94c7-158a077907ef [21]

Dyreson, C., Evans, W., Lin, H., and Snodgrass, R. (2000), “Efficiently Supporting Temporal Granularities,” IEEE Transactions on Knowledge and Data Engineering, 12, 568–587. [16]

Goodwin, S., and Dykes, J. (2012), “Visualising Variations in Household Energy Consumption,” in 2012 IEEE Conference on Visual Analytics Science and Technology (VAST), Seattle, WA: IEEE, pp. 217–218. [15]

Grolemund, G., and Wickham, H. (2011), “Dates and Times Made Easy With Lubridate,” Journal of Statistical Software, 40, 1–25. Available at http://www.jstatsoft.org/v40/i03/ [15]

Hofmann, H., Wickham, H., and Kafadar, K. (2017), “Letter-Value Plots: Boxplots for Large Data,” Journal of Computational and Graphical Statistics, 26, 469–477. [20]

Hyndman, R. J. (1996), “Computing and Graphing Highest Density Regions,” The American Statistician, 50, 120–126. [20]

Laird-Smith, J. (2020), Gs: A Grammar of Recurring Calendar Events (R package version 0.0.9000). Available at https://github.com/jameslairdsmith/gs [25]

McGill, R., Tukey, J. W., and Larsen, W. A. (1978), “Variations of Box Plots,” The American Statistician, 32, 12–16. [20]

Ning, P., Wang, X. S., and Jajodia, S. (2002), “An Algebraic Representation of Calendars,” Annals of Mathematics and Artificial Intelligence, 36, 5–38. [16,19]

Peters, K., Kniss, I., Riesenfeld, R., and Johnson, C. R. (2010), “Visualizing Summmary Statistics and Uncertainty,” Computer Graphics Forum, 29, 823–832. [20]

R Core Team (2020), R: A Language and Environment for Statistical Computing, Vienna, Austria: R Foundation for Statistical Computing. Available at https://www.R-project.org/ [25]

Reingold, E. M., and Dershawitz, N. (2018), Calendarical Calculations (4th ed.), Cambridge: Cambridge University Press. [17,18]

Tukey, J. W. (1977), Exploratory Data Analysis, Reading, MA: Addison-Wesley. [14,20]

Vaughan, D. (2020), Almanac: Tools for Working With Recurrence Rules (R package version 0.1.1). Available at https://CRAN.R-project.org/package=almanac [25]

Wang, E., Cook, D., and Hyndman, R. J. (2020), “Calendar-Based Graphics for Visualizing People’s Daily Schedules,” Journal of Computational and Graphical Statistics, 29, 490–502. [14]

——— (2020b), “A New Tidy Data Structure to Support Exploration and Modeling of Temporal Data,” Journal of Computational and Graphical Statistics, 29, 466–478. [19]

Wickham, H. (2016), Ggplot2: Elegant Graphics for Data Analysis, New York: Springer-Verlag. [14,20]

Wickham, H., and Stryjewski, L. (2012), “40 Years of Boxplots.” Available at https://vita.had.co.nz/papers/boxplots.html [20]

Wilke, C. O. (2020), Ggridges: Ridgeline Plots in ‘Ggplot2’. (R package version 0.5.2). Available at https://CRAN.R-project.org/package=ggridges [20]

Wilkinson, L. (1999), The Grammar of Graphics, New York: Springer. [20]

Xie, Y. (2013), Dynamic Documents with R and knitr, (2nd ed.), Boca Raton, FL: Chapman and Hall/CRC. [25]

Xie, Y. (2020), knitr: A General-Purpose Package for Dynamic Report Generation in R. (R package version 1.28). Available at https://yihui.org/knitr/ [25]

Xie, Y., Allaire, J., and Grolemund, G. (2018), R Markdown: The Definitive Guide, Boca Raton, FL: Chapman and Hall/CRC. [25]