# Reframing *Pattern*:
A Comprehensive Approach to a Composite Visual Variable

Tingying He (ID), Jason Dykes (ID), Petra Isenberg (ID), Tobias Isenberg (ID)

Abstract—We present a new comprehensive theory for explaining, exploring, and using *pattern* as a visual variable in visualization. Although patterns have long been used for data encoding and continue to be valuable today, their conceptual foundations are precarious: the concepts and terminology used across the research literature and in practice are inconsistent, making it challenging to use patterns effectively and to conduct research to inform their use. To address this problem, we conduct a comprehensive cross-disciplinary literature review that clarifies ambiguities around the use of "pattern" and "texture". As a result, we offer a new consistent treatment of *pattern* as a composite visual variable composed of structured groups of graphic primitives that can serve as marks for encoding data individually and collectively. This new and widely applicable formulation opens a sizable design space for the visual variable *pattern*, which we formalize as a new system comprising three sets of variables: the spatial arrangement of primitives, the appearance relationships among primitives, and the retinal visual variables that characterize individual primitives. We show how our pattern system relates to existing visualization theory and highlight opportunities for visualization design. We further explore patterns based on complex spatial arrangements, demonstrating explanatory power and connecting our conceptualization to broader theory on maps and cartography. An author version and additional materials are available on OSF: `osf.io/z7ae2`.

Index Terms—Pattern, texture, visual variables, retinal variables, data visualization, Jacques Bertin.

✦

1 INTRODUCTION

Visualization design at its very core relies on the mapping of data values to visual variables. Visual variables, also referred to as visual channels, are attributes of graphical elements—referred to as "marks" or "symbols"—whose appearance can be manipulated to encode data [72]. The visual variables that we draw upon are much studied and include *position*, *hue*, and *size* [5, 6]. Their effectiveness ranking has been the subject of much research and discussion in our field [26, 31, 61, 64, 69] with scope for encoding nominal, ordered, and numeric data [80]. Among the available visual variables is one that researchers have called *pattern* [59], typically featuring repetitive dots or lines ▨▨▨. Visualization designers often use these patterns when color is either limited or already encodes other data dimensions (e. g., [5, 6, 11, 12, 22, 28, 43, 55, 78, 82, 87, 91]).

When patterns ▨▨▨ are described as visual variables, researchers also refer to them as *texture*. This interchangeability of the terms "pattern" and "texture" may arise from the blended use of both terms in everyday language and the use of "texture" as a translation of Bertin's visual variable "grain" in the English translation of his seminal and highly influential work [5, 6]. To add to the confusion, however, the term *texture* has a diverse set of meanings in visualization research that goes beyond an understanding of texture as pattern ▨▨▨. Researchers working on 3D representations, for example, often use *texture* for surface or volume characteristics of 3D objects, represented as realistic images ▨▨▨ [50, 57]. These textures typically have different visual characteristics and encoding goals from the more structured repeating patterns that are used as a visual variable in abstract data representations . Even in the specific context of visual variables used for abstract data representations, researchers may interpret the term *texture* as a variation of a specific dimension of a *pattern*, such as "granularity" (Bertin's "grain" in French) ▨▨▨, the spacing between the repeated elements ▨▨▨, or the elements' shape ▨▨▨. This méli-mélo de

- *T. He (何汀滢) is with Université Paris-Saclay, CNRS, Inria, LISN, France, and the University of Utah, USA. E-mail: hetingying.hty@gmail.com.*
- *J. Dykes is with City, University of London, UK. E-mail: j.dykes@city.ac.uk.*
- *P. and T. Isenberg are with Université Paris-Saclay, CNRS, Inria, LISN, France. E-mail: given_name.family_name@inria.fr.*

terminologie, we argue, prevents the research community from investigating *pattern* as a visual variable or using its encoding effectively. Research on patterns and their practice of use is difficult to compare and situate in the absence of a consistent terminology. We consider this situation to be a theoretical irritant with practical implications—a persistent blind spot that we aim to address with our work.

Our re-reading of the literature [20, 59, 72, 92] leads us to a consistent terminology, in which we use the term "pattern" to describe a composite visual variable ▨▨▨ that consists of graphical primitives that can also serve as marks for data encoding. **Our first contribution** is an in-depth discussion and clarification of the terms *texture* and *pattern* in light of existing interpretations around both terms, to address the inconsistent terminology problem. As **our second contribution**, we introduce a new pattern system that extends the conceptualization of *pattern* and its potential variations. This system builds upon our terminology to describe a design space in ways that can be used for encoding, exploration, and experiments. We identify three sets of attributes of a pattern: the spatial arrangement of primitives, the appearance relationships among primitives, and the individual appearance characteristics of primitives. Furthermore, we examine more complex spatial arrangements of primitives, using the concept of pattern as a theoretical lens to compare, explain, and connect different types of visualizations and to uncover new design opportunities.

2 TEXTURE AND PATTERN

Researchers often use the terms *pattern* (e. g., [53, 59, 88]) or *texture* (e. g., [44, 91, 94]) to describe a visual variable characterized by repeated elements ▨▨▨. Although both terms can make sense and are understandable, Carpendale [20], in her discussion of visual variables, suggests to use the term *texture* for "apparent surface quality of the material like wood or marble" and to use *pattern* for "repetitive use of shape variations." We consider Carpendale's recommendation reasonable and useful[1] due to two main issues associated with the term *texture*: (1) compared to *pattern*, the term *texture* has a broader meaning in visualization and related fields, can refer to different concepts (as we show in Fig. 1 and Fig. 2), making it less precise; and (2), even when *texture* specifically refers to a visual variable, it is subject to different interpretations, even post-Carpendale [20], as can be observed by comparing various publications that use the term [52, 53, 83, 88]. Below we discuss the first of these issues and clarify the use of *texture* and *pattern* in the visualization literature, and explain why *pattern* is a more suitable term for this type of visual variable. We address the second issue in Sec. 3.
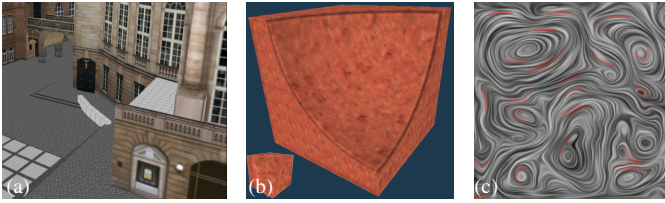
Fig. 1: Textures in (a) surface rendering [14], (b) volume rendering [57], and (c) flow visualization [67]; all images © IEEE; used with permission.
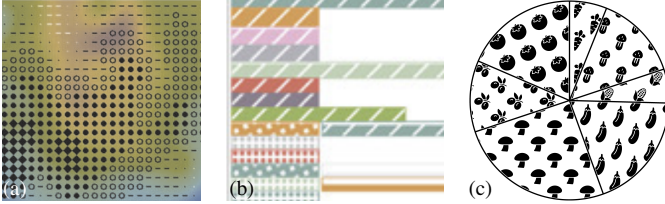

Fig. 2: Patterns used in the visualization community that are called "texture," from (a) [91], (b) [22] , and (c) [44]; (a) and (b) © IEEE, (c) CC BY 4.0; all images used with permission.

2.1 Texture: Surface characteristics

The term *texture* is often used to describe an object's "visual or tactile surface characteristics and appearance" [70]. In computer graphics, especially in work that relates to rendering, *texture* is a widely used concept. In this context, it essentially refers to a data structure that stores characteristic information (visual or other). It is typically represented as a multidimensional array. Through texture sampling, we obtain the necessary data from the texture and map it onto the corresponding location of the object. The visual texture that we ultimately observe on the object is the result of the rendering process [7, 21]. From an appearance standpoint, textures are often closely related to real-world materials, have a sense of depth and realism, and often look continuous.

Leveraging techniques from computer graphics, researchers in 3D visualization use *texture*, for example, to depict materials of a model's surface (e. g., Fig. 1(a)) or to define a volume's visual characteristics (e. g., Fig. 1(b)). In flow visualization, researchers also use techniques that rely on textures, such as Line Integral Convolution (LIC) [16] or spot noise [90], to represent the directionality, magnitude, and other attributes of vectors or tensors (e. g., Fig. 1(c)).

Texture is also used to refer to surface characteristics in other visualization-related fields beyond computer graphics. It is recognized as one of the seven elements of art, denoting the characteristics of an object's material [35]. In the visual arts, visual textures are called *implied textures* (in contrast to *actual textures*, which are tactile), e. g., to create a simulated appearance of physical materials [35]. In computer vision, researchers study *texture analysis* techniques (e. g., *texture segmentation* and *classification*) to enable computers to recognize objects and understand scenes [86]. In the vision sciences, researchers study *texture perception* to understand how humans perceive surface qualities [79].

Carpendale [20] brings multiple perspectives together in her discussion of visual variables for data visualization. She specifically discusses the possibility of using surface materials (i. e., the computer graphics interpretation of *texture*) as a separate visual variable [20, Table 9], and illustrates it with examples using photographic images [20, Table 11]. In this case, a (texture) image is applied to a chart element, and what we read is both the chart element's value and the information in the image. Because both the visual appearance and the information encoding in these examples differ from the visual variable examined in our work, we recommend using the term texture for this visual variable in line with Carpendale's suggestion—yet a more specific term such as "surface material texture" would make for a clearer distinction.

2.2 Pattern: Repetition and structure

A body of research and practice that has its roots in cartography and statistical graphics interprets *texture* in a different manner. In Fig. 2 we show examples of this type of "texture." Researchers map data

dimensions to the graphical features of these textures. These textures typically feature clearer and more distinguishable repetitive elements than those textures used in computer graphics rendering. even though they may carry meaning through mimetic properties, they are generally unrelated to surface materials.

We describe this use of "texture" as *pattern*—a concept that emphasizes aspects different from *texture*. The term "pattern" originated from the same root as "patron," derived from the Latin *patronus*, meaning "protector" or "defender" [27]. It evolved to signify "an example to be copied" [27], emphasizing the repetition of elements—rather than the tactile or perceived feeling of a material (i. e., a texture). Note that the term *pattern* is not limited to visual elements, it is a structural concept that can be applied to the abstract[2] as well as the physical world.[3]

We focus on the visual aspects of *pattern*. The emphasis on repetition and structure makes the concept of *pattern* particularly suitable for describing visual variables in the form of structured marks that repeat to fill space, capturing both their composite encoding and abstract appearance. As Wilkinson [98] in his discussion of visual variables mentioned, "these [visual variables] are not ones customarily used in computer graphics to create realistic scenes. They are not even sufficient for a semblance of realism." Nevertheless, patterns can also characterize a surface, suggesting that we can view patterns as a specific type of "texture," one that describes a surface with distinct primitives and structured arrangement. When the repeated elements in a texture are clearly identifiable, the texture takes on the characteristics of a *pattern*. This overlap between the two concepts may explain why some researchers use the terms *pattern* and *texture* interchangeably.

2.3 Summary

The term "texture" is used differently in different visualization contexts, with meanings derived and used in computer graphics and in abstract data representation having some similarities, but important differences. Both can characterize a surface and add visual complexity. *Texture* often describes the appearance of a surface and its material properties, while *pattern* emphasizes the repetition and structure of elements that involves semiotics and is frequently used in abstract data representations. We acknowledge the overlap of the terms *texture* and *pattern* and are sensitive to other researchers' use of both terms. In our case of using it as a visual variable, however, we suggest that *pattern* is a more precise term than *texture*, as *patterns* rely on repetition and are not usually meant to suggest surface material.

3 PATTERN AS A VISUAL VARIABLE: THREE INTERPRETATIONS UNDER THE TERM OF "TEXTURE"

While *pattern* is a more precise term if we think of it as a visual variable, we frequently see "texture" in lists of visual variables. This use of "texture" to describe a visual variable can be traced back to Bertin's seminal book, *Semiology of Graphics* [5, 6], in which he introduced the first set of visual variables—"texture" among them. Subsequent literature has continued to use this term (e. g., [20, 64]), however, with different interpretations. It has been referred as the variation of granularity in a *pattern* (e. g., [5, 6, 52]), the spacing of a *pattern* (e. g., [29, 83]), the shape variation of a *pattern*, or a *pattern* in its entirety (e. g., [53, 88]).[4] This inconsistency has its roots in the translation of Bertin's book, which we carefully unpick as follows.

3.1 Grain: The original term Bertin used

The French term "grain" is the original word that Bertin used to describe the visual variable [6], which William J. Berg translated to "texture" in the English version of this book [6]. Bertin defined the visual variable as follows: "at a given value, the [granularity][5] represents the number of separable marks within a unit area." In the "texture" palettes from his book that we reproduce in Fig. 3, the variation of granularity along each *horizontal* palette involves changing both the size and spacing of primitives *simultaneously*, while maintaining a given ratio of black to white. As a result, the average *value* of each square stays constant. This effect is similar to what can be achieved by zooming in or out of a pattern or through photographic reduction [5, 6].
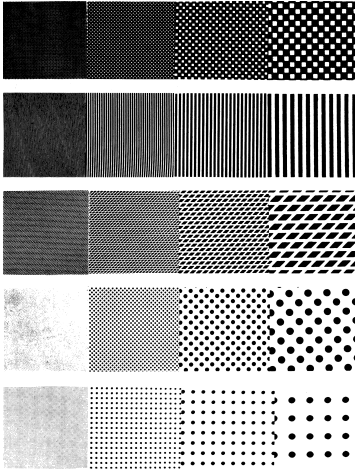
Fig. 3: Bertin's diagram for granularity variation, described in French as "Horizontalement: grain. Verticalement: valeur et forme" [our translation: horizontal: granularity. vertical: value and shape] [5] and in English as "texture is given horizontally; value and shape [pattern] vertically" [6]; image © EHESS, used with permission.

The translator notes that "Bertin draws a rigorous distinction between texture (*grain* in French) and pattern (*texture* in French)" as follows (images recreated similar to those used by the translator):

The translator continues to note that "A and B differ in their texture, but there is no difference in pattern. The elementary shapes are the same. The notion of pattern explains the difference between A and C. The elementary shapes are different. A difference in "pattern" is essentially a difference in shape."
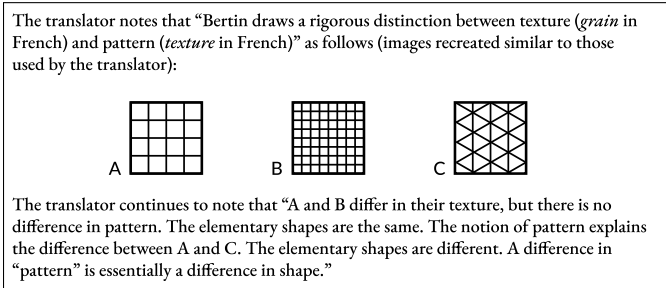
Fig. 4: Citation of the translator's note (with redrawn images similar to the original ones) in the English edition of Bertin's book [5].

Researchers questioned the translation of Bertin's French "grain" variation to the English term "texture," suggesting that "grain" or "granularity" would be more precise translations. MacEachren [59], e. g., suggests that the English term "grain" may be better to describe this variation, as it is similar to the grain in film. Similarly, Wilkinson [98] said that Bertin "really means granularity (as in the granularity of a photograph)." Carpendale [20] also noted that this variation more closely relates to a variation of granularity and directly referred to it using the English term "grain." Between "grain" and "granularity," we recommend using "granularity" in English, based on the rationale that it is not the "grain" itself that varies but rather the size of the grain, which is more accurately described by "granularity." In English, moreover, "grain" can refer to the longitudinal structures of wood fibers (i. e., "wood grain"), potentially conveying a sense of direction. The concept of direction, however, is not implied in Bertin's (French) *grain*, leading us to argue that "granularity" is a better translation.

3.2 Spacing: A misinterpretation in Bertin's book

Another interpretation of the word "texture" refers to the spacing of primitives in a *pattern*. Spacing between primitives can affect *density*— the smaller the spacing, the more densely packed the primitives. This variation is called "spacing" by Brewer [10] and Slocum et al. [83], "density" by Mackinlay [64], or "frequency" by Chung et al. [24].

The interpretation of "texture" to relate to spacing may arise from a misinterpretation in a translator's note in the English version of Bertin's book, which we reproduce in Fig. 4. As we had just discussed, for Bertin changes in *granularity* (French: "grain," translated in Fig. 4 to "texture") require that a constant average value is upheld. The translator's note, however, refers to the difference between Squares A and B as a change in "texture" (i. e., "grain" in French or, for us, *granulatity*). Yet, A and B do NOT share the same average value, as A has a lower black-to-white ratio than B. Therefore, A and B would not constitute a change of the (French) "grain" for Bertin. Instead, if we see the black lines as primitives of the pattern, we can see from the figure that A and B differ only in the *spacing* between primitives, changing the black-to-white ratio while maintaining the *primitive shape* (cf. Fig. 3).

3.3 Pattern: Not only shape variation

"Pattern," largely in the sense we have established in Sec. 2.2, is a third term often used interchangeably with "texture" when referring to a visual variable—a confusion stemming both from the semantic overlap discussed in Sec. 2 and from the translator's interpretation of Bertin's work. In the original French edition [5], Bertin described the *vertical* change between *corresponding "palette" entries* in Fig. 3 is a variation of "value and shape" (French: "valeur et forme"). It is unclear if this interpretation was supported by Bertin, but the English translator of the book amended this statement to "value and shape [pattern]" [6]. This amendment seems reasonable: we can see in Fig. 3 that the differences between palette entries on each column are not just differences in *value* and *shape*, but also include differences in *size* of the elements, *spacing* between the elements, etc.—which are all variations a *pattern* can have. In the translator's note we just mentioned (Fig. 4), however, the translator explained that "a difference in 'pattern' is essentially a difference in shape."[6] Carpendale [20] adopts this interpretation, defining *pattern* as "repetitive use of shape variations (the use of marks upon marks)," and equates the impact of using the visual variable *pattern* in visual interpretative tasks with the visual variable *shape*. This interpretation captures the emphasis of *pattern* on repetition well and the notion of "the use of marks upon marks" touches an apparent inconsistency of Bertin's use of visual variables, which we discuss in Sec. 4.2. The variations of *patterns*, however, should not be limited to the change of shape as we just discussed for Fig. 3.

3.4 Summary and our recommendation

"Texture" has multiple interpretations, and when used without clarification or diagrams in the context of abstract data representation, its meaning is often ambiguous. We therefore recommend reserving "texture" for surface materials and using "pattern" to describe the visual variable that features structure and repetition . Given the many subdimensions of patterns, they should be described comprehensively and consistently. To support this, we propose a consistent and expressive descriptive terminology in the remainder of the paper.

4 ADDITIONAL RELATED WORK ON PATTERN VARIATIONS

To fully understand what truly constitutes a pattern, we first review previous work on identifying subdimensions of *pattern*, then highlight an inconsistency in Bertin's use of visual variables—one that inspires our development of a comprehensive description of *pattern*.

4.1 Pattern description from two perspectives

Researchers have investigated variations of *pattern* from two perspectives, corresponding to the encoding and decoding processes. In encoding, designers use visual variables to represent differences in data; in decoding, readers perceive these variations and interpret them as data differences. The description of *pattern* can thus be approached from two directions: what designers can control and what readers can perceive. Research from disciplines with a design focus has proposed subdimensions of *pattern* from a design perspective, while research from the field of vision science has explored dimensions of *pattern* from a perceptual perspective. Our proposals draw upon both perspectives.

4.1.1 Perception perspective

To be able to use pattern for encoding data effectively, it is vital to understand how the human visual system perceives such visual stimuli. Vision science researchers have tried to identify the most important perceptual dimensions that are useful for humans to judge the difference between appearance of textures (also known as texture features).

Tamura et al. [85] propose six basic texture features, namely, coarseness, contrast, directionality, line-likeness, regularity, and roughness. Amadasun and King [2] approximate five perceptual texture attributes in computational form, namely coarseness, contrast, busyness, complexity, and strength of texture. Rao and Lohse [77] identify a Texture Naming System with the three most significant dimensions in natural texture perception: "repetitive vs. non-repetitive; high-contrast and non-directional vs. low-contrast and directional; granular, coarse and

low-complexity vs. non-granular, fine and high-complexity." Liu and Picard [56] identify three mutually orthogonal dimensions of texture that are important to human texture perception, namely periodicity, directionality, and randomness. Cho et al. [23] extend the perceptual research and reported four texture dimensions: coarseness, contrast, lightness, and regularity. Features such as coarseness, roughness, or strength remind us of the interpretation of texture as a surface characteristic, which is understandable because vision researchers [2, 23, 56, 77, 85] have primarily focused on natural textures (e. g., the photographic textures in Brodatz' album [13]). Yet their work—dedicated to understanding how humans perceive texture—can nevertheless shed light on using pattern for data visualization. In particular, Ware and Knight [95, 96] identify three orderable dimensions for data displays: *orientation*, *size*, and *contrast* (OSC). Healey and Enns [45] build three-dimensional perceptual texture elements, called pexels, for visualizing multidimensional datasets. Pexels can be varied in three separated texture dimensions, which are height, density, and regularity, and the color of each pexel. This perception perspective is highly relevant to the use of *pattern* as a visual variable. A pattern description system developed from the design perspective can be informed by and tested in the context of the perception literature.

4.1.2 Design perspective

Researchers in design, cartography, and visualization have described the composite nature of *pattern*, which has multiple dimensions that can be varied to encode data. From an architectural perspective, Caivano [18, 19] describes a system of patterns using the term "texture," which we adopt when discussing his system. He classifies "simple textures" and "complex textures," defining the former as "the uniform repetition of a certain element" ⬚ ⬚ and the latter as combinations of multiple sets of these primitives ⬚ [19]. His simple textures are essentially described by the relationship of two elements within a tiling (repeating) unit. He describes texture variation as the shape of texture elements, organization (relative positions of the two texture elements in the tiling unit), proportionality (a tiling unit's width-height ratio), and density (overall black-to-white ratio). Cavaino, however, does not intend to use his textures as a visual variable for data encoding. As a result, not all the dimensions he identified are directly manipulable, and his composition of simple textures is therefore unsuitable for our purpose. In addition, we identify two distinct subsets of pattern within his simple texture classification (see Appx. A for a detailed discussion). Below we thus develop our own *pattern* configuration to offer an alternative that covers a wider design space, specifically aimed at encoding data.

A cartographic angle, expressed in MacEachren's definitive *How Maps Work* [59], considers "'pattern' as [a] higher-level visual variable, consisting of units that have shape, size, orientation, texture (in Bertin's sense of grain), and arrangement." In the field of visualization, Harris [41], in his book on information graphics design, suggests that "there are many variations" within patterns and lists factors that make up patterns: "shape of individual elements," "orientation of individual elements," "texture (sometimes referred to as coarseness)," "size of individual elements," and "spacing between individual elements." Wilkinson [98] wrote that "texture includes pattern, granularity, and orientation," but he does not further analyze *pattern*. Instead, he interprets *pattern* as being "similar to fill style in older computer graphics systems, such as GKS (Hopgood et al., 1983) or paint programs" but does not describe the subdimensions of *pattern*. In our own previous work [44] we identify a set of *pattern* properties (as [sic] "textures") but also does not cover all dimensions.

In summary, our reading of this prior work provides us with useful examples for understanding the various dimensions of patterns, but none of the individual treatments are comprehensive and the rationale that supports them is somewhat sketchy. Based on the various authors' suggestions, we thus provide a systematically defined proposal that is comprehensive in its consideration of pattern as a high-level visual variable—developed via a structured, repeated combination of sets of visual primitives that vary in their visual characteristics to encode data.
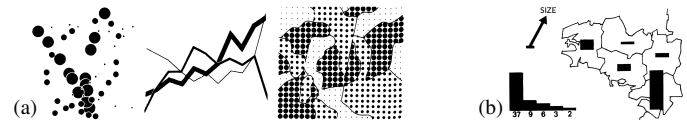

Fig. 5: Size variations from Bertin's book [5, 6]: (a) Size variations for three mark types: left and middle show Bertin's Approach 1, where he directly adjusted the marks' size properties (dot size and line width); right shows Bertin's Approach 2.2, where he added repetitive dots to fill the area mark and used the dots as secondary marks—he varied the dots' size properties (dot size). (b) Size variation for an area mark using Bertin's Approach 2.1, where he added a single secondary mark (rectangle) and varied its size property (rectangle size). All images © EHESS (text translated), used with permission.

4.2 Inspiration from Bertin's apparent incongruity

Bertin himself do not explicitly define or employ the concept of *pattern*, but why then do we see many visual encodings we may intuitively call *pattern* in his charts or maps? One explanation may be that Bertin attempts to address an inherent limitations of area marks. Area marks cannot change in *size*, *shape*, or *orientation* [5, 6, 20, 72], without the area changing its meaning. For example, we cannot encode an additional data attribute into the *size*, *shape*, or *orientation* of a region (an area mark) on a map because these attributes are already used to encode geographic information. To address this limitation, Bertin invokes the cartographic tradition, in using a method of adding a single additional mark with variable visual characteristics or a group of repetitive additional marks with common visual characteristics. When he does the latter, he creates a *pattern* with repetitive tiling of secondary marks (primitives)—which is why Carpendale interprets *pattern* as "the use of marks upon marks." Bertin explains this encoding by the fact that it is possible to change the visual characteristics of the constituents that make up an area: "if the area is visually represented by a constellation of points or lines, these constituent points and lines can vary in size, shape, or orientation without causing the area to vary in meaning" [5]. He also makes a similar point for the constituents of a line mark. This adjustment explains why Bertin could apply all his six retinal visual variables—including *size*, *shape*, and *orientation*—onto line and area marks (see his overview in Fig. 21 in Appx. D), and the "constituents" here equate to the secondary marks (the primitives in the pattern).

Let us take the visual variable *size* as an example to explain how he mixes these two approaches. *Size* is applicable to point and line marks, but not to area marks. When applying *size* to point and line marks, Bertin directly adjusts the mark's intrinsic properties, such as the dot's radius or the line's width—without introducing secondary marks (Fig. 5(a), left and middle). We call this *Approach 1—Direct Encoding*: changing the graphical properties of the mark itself, which aligns with the precise definition of a visual variable. For area marks (such as map regions), however, *size* is not applicable if the integrity of the geographic encodings is to be maintained. In this case, Bertin takes another approach: to introduce new mark(s)—we call this *Approach 2—Secondary Marks*, which includes two options: *Approach 2.1—Single Secondary Marks* adds one constituent (secondary mark), and *Approach 2.2—Repeated Secondary Marks* adds repetitive secondary marks ("constellation") to fill the area or line mark. For example, Bertin shows the application of *size* to an area mark by adding a rectangle of different size to each area mark (Fig. 5(b), which aligns with the single secondary marks approach, and this is essentially the approach favored by Cleveland and McGill in the framed-rectangle charts of their seminal graphical perception paper [25]), or by repetitively filling each area mark with circles of varying sizes (Fig. 5(a), which aligns with the repeated secondary marks approach). In summary, when Bertin can use a direct encoding, he does so. When this is not possible, he automatically switches to using secondary marks, yet without clarification. In his discussion, unfortunately, he does not clearly explain why and when to select single or repeated secondary marks.

The secondary marks that Bertin adds to the marks are typically point- or line-marks, as more visual variables can be applied to them than to area marks. With repeated secondary marks, Bertin, in fact,

creates point-based and line-based patterns as we understand it, and so that is ultimately why we see many *pattern* examples in Bertin's book.

Bertin's mixing of the two methods actually blurs the meaning of visual variables. As Wilkinson [98] points out, "Bertin uses size, shape, and orientation to characterize both the exterior form of objects (such as symbol shapes) and their interior texture pattern (such as cross-hatching)." Bertin's approach does not fully explore or clearly articulate the full emerging possibilities of patterns. We can, in fact, apply *pattern* across all visual variables and mark types, but Bertin reserved *pattern* for situations where visual variables were not applicable to certain types of marks. In addition, Bertin simply keeps the secondary marks arranged in a regular grid and ensures that each secondary mark was exactly repetitive. Similarly, Carpendale [20] equated a variation in patterns to the variation in shapes constituting them, as we discussed in Sec. 3.3. Inspired by these perspectives and having established their apparent inconsistencies, we systematically explore the opportunities for explanation and for visually encoding data offered by this new comprehensive description of 'pattern.'

5 PATTERN AS A VISUAL VARIABLE: A DESIGN SPACE

This position enables us to establish a new pattern system—based on the design perspective and with the goal of identifying and exposing the basic parameters that can be exploited to encode data. We conceptualize this system as follows: A *pattern* has a group of primitives. Primitives are graphical elements whose visual attributes can be manipulated to encode information; they can thus be considered "marks." To differentiate these marks from the graphical elements (mark) to which patterns are applied, we refer to the latter as *host symbols*. We thus define a *pattern* as a **composite** of constituent marks (primitives) applied to a host mark (host symbol). For consistency, throughout the remainder of this paper, we use the terms *primitive* and *host symbol* to refer to these two respective levels of marks. Transitioning from a symbol as a single mark to a symbol that contains a composite of (possibly repeating) primitives that constitute a *pattern* then introduces new visual attributes. These attributes arise from the composite nature of the pattern, specifically the relationships among its primitives.

Our approach sees pattern as a set of rules for describing the primitives, i.e., parameters that define their attributes, variation, and relationships. By following these rules, we can generate expressive patterns that fill a host symbol. We identified three sets of rules that are key to this process and that also serve as attributes by which a pattern can be characterized: (1) *spatial arrangement* of primitives (Sec. 5.1), (2) *appearance relationships* among primitives (Sec. 5.2), and (3) the *retinal visual variables* applied to each individual primitive that define its appearance (Sec. 5.3). We now discuss how these attributes can be varied to characterize a pattern. The fact that they can be deliberately and systematically exploited for encoding data enables us to speculate that they have a scope as "visual variables," and we begin to explore their capacities for representing information.

5.1 Spatial arrangement of primitives: Lattice

Spatial arrangement attributes describe how the primitives of a pattern are spatially positioned and repeated to fill the space occupied by a host symbol. In theory, any algorithmic method for arranging visual elements could define such spatial arrangements. We begin, however, with a pattern type that is commonly found in existing visualizations—characterized by the regular arrangement of primitives along or across the host symbol ▦▨▥. We can use a lattice structure to describe this regular arrangement, which consists of a set of regularly spaced points that can extend infinitely in space. Each point, known as a *lattice point*, represents a predefined position for a primitive within the pattern (Fig. 6–①, rows 1 and 2). We follow a method used in crystallography [36] to define a lattice, whose central idea is to identify the unit cell of the lattice. The unit cell is the smallest unit of a lattice and the entire lattice can be generated by the repetitive tiling of the unit cells. We call the parameters that define the unit cell and thus the lattice structure "lattice parameters." Each lattice has a unique set of parameters, such as θ, a, and b for oblique lattices[7] or $\theta = 120°$ and a for hexagonal lattices. These parameters are the spatial relationship attributes for

characterizing a pattern. Although the discussion of lattices relates to their use in crystallography [36] and in tiling [51], we take a more general approach here and say that all lattice definitions are allowed—including nonuniform ones. In this section, we focus on uniform regular lattices as examples for explaining the spatial relationship attributes of patterns. We discuss patterns composed of primitives arranged in more complex or non-lattice structures in Sec. 7.

5.1.1 Define lattice: Dimensionality and unit cell shape

Lattice dimensionality. The number of lattice dimensions affects the parameters that are required to define the lattice. We can organize the primitives of a pattern for 2D visualizations into either 1D or 2D lattices (rows 1 and 2 in Fig. 6). It is important to note that the lattice dimensionality (1D or 2D) differs from the number of dimensions that the pattern itself occupies (along a line or across an area), as well as from the number of dimensions of the host symbols onto which the pattern is applied (point, line, or area). The lattice dimensionality is determined by the number of directions in which the lattice can extend.

The first two rows of Fig. 6 illustrate the two types of patterns possible in 2D representations, to which we refer as point-based and line-based patterns, respectively. Point-based patterns are based on 2D lattices, in which the lattice points are often equally spaced, extending in two directions (2nd row in Fig. 6). In contrast, line-based patterns rely on 1D lattices but with linear primitives that (usually) extend to infinity (1st row in Fig. 6). Both types of patterns, however, can be used on all three types of host symbols—areas, lines, and points—because all three mark types are represented by a host symbol with areal extent.[8] If we apply the lattice to line host marks (1D elements), we can either treat the line as a linear host symbol with a (small) extent perpendicular to its major direction, or as a form of selection that picks a single direction from a 2D lattice or that limits the lengths of the line primitives for a 1D lattice (also see Fig. 12 in Appx. C).[9]

Unit cell shape. The shape of the unit cell characterizes the lattice type. In a 1D lattice, the unit cell is simply a line segment (essentially a distance) between two adjacent lattice points. In a 2D lattice, the unit cell can take many tessellating shapes, including triangles, rectangles, hexagons, and more complex geometries with nonrepetitive tilings (e.g., [74, 84]). The lattices and patterns that result in the latter case are not grid-based. In this section, we mainly use the square lattice as examples to illustrate different pattern variations. Note that we refer here to the overall geometry of the unit cell. Some shape variations can be derived through geometric transformations (e.g., shearing a square into a parallelogram), which we discuss next.

5.1.2 Transform lattice: Affine transformation

The lattices can further be transformed through affine transformations, such as translation, shearing, scaling, and rotation, to achieve arrangements suitable for specific application contexts (Fig. 6–②). Due to the inherently infinite repetition of a lattice, a translation usually only leads to limited visual changes.[10] Shearing, in contrast, affects the shape of the unit cell, e.g., converting a square lattice into an oblique lattice, and is thus often considered together with the unit cell shape. Among the various affine transformations, lattice scaling and rotation are often used for encoding data, and we discuss them in detail next.

Scale: Size of the unit cell. Scaling changes the spacing between lattice points and, thus, the unit cell size. The spacing can be modified either uniformly across both directions or independently, the latter facilitating directional variation in the pattern. Independent adjustments in the width and length of the unit cell can significantly affect the directionality of the resulting lattice pattern. For a 1D lattice, in contrast, variations in spacing are constrained to a single dimension—along the line of the lattice. In both 1D and 2D cases, however, a variation of the spacing between primitives affects the area of the unit cell, which in turn influences the density of primitives in a pattern.[11]

Rotate: Orientation of the lattice. Rotation varies the orientation of the lattice.[12] For a pattern arranged in a 2D lattice (Fig. 6, 2nd row), it is important to distinguish between the orientation of the lattice itself (Fig. 6–②) and the orientation of the primitives within it (Fig. 6–④, retinal variables–orientation).[13] If both the primitives and the lattice
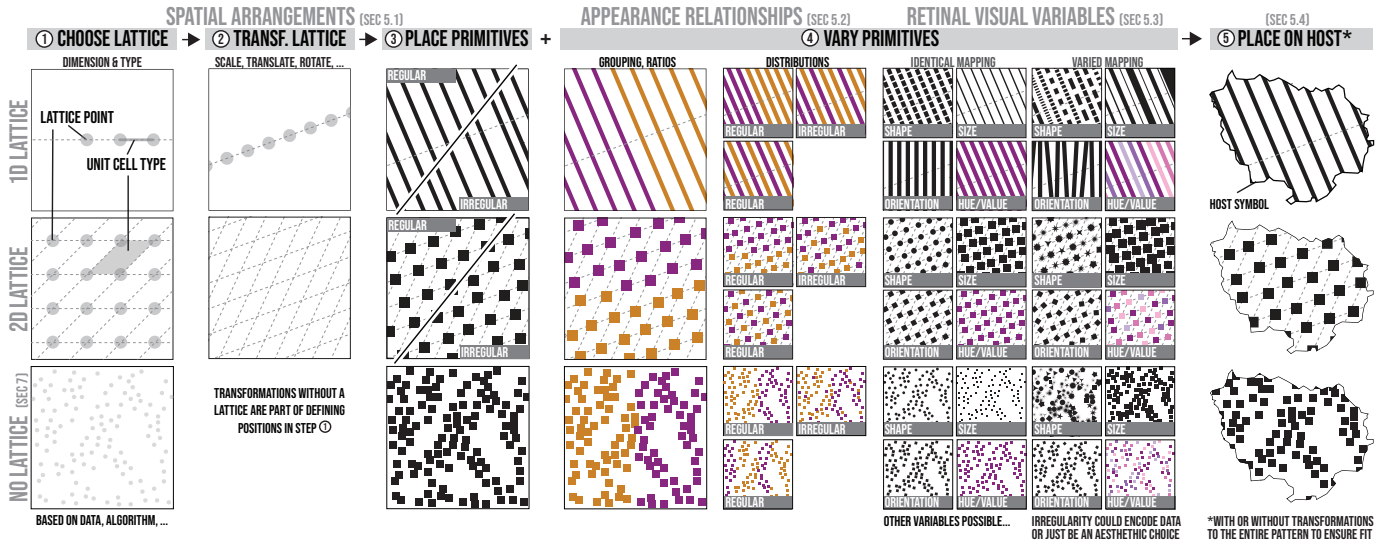
Fig. 6: Procedure for creating a pattern by describing three sets of pattern attributes: (1) the *spatial arrangement* of primitives (Sec. 5.1), (2) the *appearance relationships* among primitives (Sec. 5.2), and (3) the *retinal visual variables* applied to each individual primitive that define its appearance (Sec. 5.3). We illustrate the attributes with pattern samples constructed both with a lattice (Sec. 5) and without one (Sec. 7).

are rotated by the same angle, this can intuitively be described as a rotation of the entire pattern (Fig. 14 in Appx. C).[14]

5.1.3 Place primitives: Positional regularity

So far, we have established a set of predefined position points for the primitives within the pattern—the lattice points. When we place the primitives onto the lattice, however, they may deviate from these predefined positions (Fig. 6–③). To describe the extent of this deviation, we adopt the concept of *positional regularity*, introduced by Morrison [71].[15] This variable describes the degree to which primitives can deviate from lattice points, ranging from strict adherence to the grid via structured irregularity to a fully random placement within the mark. For a 2D lattice, the deviation can occur in either one direction of the unit cell or both directions (Fig. 15 in Appx. C). For 1D lattice patterns, the placement along the lattice line is affected. *Positional regularity* is not an atomic variable and has subdimensions, including its range and its dispersion level. **Range** describes how far we can deviate from the predefined point, and the **dispersion level** can be understood as the standard deviation or entropy of the deviations among all primitives.

5.2 Appearance relationship among primitives

Our intended comprehensive treatment of pattern as a visual variable requires us to consider not only the spatial relationships between primitives but also how primitive groups appear within the pattern. For noncomposite marks, describing their graphical attributes—such as shape, size, and color—is sufficient. These attributes are what Bertin refers to as the "retinal variables." We cannot, however, directly manipulate each primitive's retinal variables in a pattern because patterns are used to fill host symbols. Depending on the size of the host symbol, the same pattern may contain different numbers of primitives. As a result, we cannot set the attributes of each primitive individually. Instead, we define rules for the primitives' attributes, and the pattern is then generated accordingly to fill the host symbol. We thus need to establish rules that describe the relationships between the primitives' appearances, no matter how many of them appear in a pattern. For common patterns with repeated primitives, e. g., this rule is simply "all primitives look the same." Our consideration of patterns as composite marks that vary in defined ways to encode data opens up a large design space, revealing more and new possibilities with combinations of graphical attributes that can serve as new ways to encode data. We discuss these next, describing the internal variation in the appearances of primitives.

5.2.1 Number of primitive groups

Number of primitive groups describes how many distinct combinations of visual variable variation, or styles of primitive, are used within a

pattern, with each group depicted using a unique encoding or encoding combination. Traditionally, patterns with a single repeated primitive have been the most widely used, meaning that one or more visual variables are applied consistently across all primitives used in a pattern to encode data (Fig. 6–④: "identical mapping"). The pattern's composite nature and the enabling nature of today's digital technologies (e. g., SVG), however, allow us to be more expressive than is required by this consistency: We can vary visual variables to associate and differentiate subsets of the primitives in a pattern (Fig. 6–④: column "grouping, ratios"). Intuitively, internal variation in patterns allows us to visualize a new data facet and represent it within the mark. We can use the variable primitive group count to encode the number of categories associated with the facet (e. g., [49]). Its application, however, is broader: as the *number of primitive groups* is an attribute just like any other visual variable, it can encode various types of data. It is ordered as it represents a numerical count, making it useful for encoding ordinal data. When using it, however, it is important to ensure that the number of primitive groups is not excessively large to maintain discriminability.

5.2.2 Ratio between the groups

If a pattern comprises multiple primitive groups (and only then), the primitive count can differ between groups, to which we refer as the *ratio between groups*. Fig. 6–④: column "grouping, ratios") and Fig. 17(b) in Appx. C show examples for variation in this variable. It can encode ordered data due to its numerical nature.

A straightforward way to use this facet is to form primitive groups and encode categories (i. e., keys). All purple primitives, for instance, encode data for category A, and all yellow primitives encode data for category B. If these categories have a certain distribution (e. g., 50% of data items are of type A, and 50% are of type B), then we could reflect this split in the primitive group ratio. Fig. 7(a) (reproduced from Bertin's book [5, 6]) shows a good example of using the variable to encode the distribution of categories. It shows three categories of data for France, with differently colored primitives that encode data by region. Here, the width—1D size—of each colored primitive encodes the proportion of the respective category. Fig. 8 shows another example, employing the number of groups of primitives to encode the number of categories (IsoType, [39, 73]). The ratio between groups (indicated by line width) encodes the percentage of each category.[16]

5.2.3 Distribution style of different primitives

The *distribution style of each group* refers to how we arrange each primitive group within a pattern (Fig. 6–④, column "distributions"). We should differentiate it from the spatial arrangement of primitives as being the allocation of primitive variations to predefined positions in
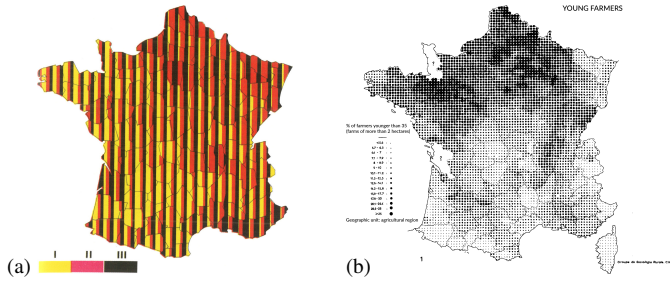
Fig. 7: Examples of using regular arrangement pattern with internal variation. Within the patterns, the variations (a) show a facet of data (described in Sec. 5.2.2), (b) based on geographical data (described in Sec. 7.1); both images © EHESS (text translated), used with permission.
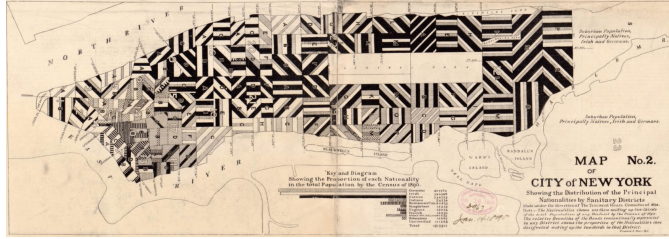

Fig. 8: Map of nationality distribution in NYC [75] from 1895, It depicts the distribution of different nationalities across sanitary districts in Manhattan. The designer used line pattern with variations in orientation to distinguish the city districts and used patterns on each line to distinguish different categories. ⊛ The image is in the public domain.

the pattern. Here, we discuss how to further specify which primitive belongs to each group after having defined all primitives' positions.

Fig. 6–④ (column "distributions", rows 1 and 2) shows different choices of distribution style applied to a consistent spatial arrangement of primitives, both for 1D lattice patterns (row 1) and for 2D lattice patterns (row 2). There can be regular arrangements (left part of column "distributions") or irregularly dispersed ones (right part of the column). Within the former, the primitives within the same group can be clustered together (the top example, respectively), as also in the IsoType image in Fig. 25 in Appx. E. Alternatively, the primitives of the pattern within each group can also be dispersed (bottom examples), resulting in a uniform distribution, as also exemplified in Fig. 8. The regularity, clustering, or dispersion of primitive groups can be used to differentiate patterns, and may also have ordinal or numeric encoding possibilities, ranging from regular to irregular, to clustered, to dispersed.[17]

5.3 Retinal visual variables on each primitive

So far we have used two sets of parameters to describe the relationships between groups of primitives in a pattern: their spatial and group appearance relationships. After establishing these rules, we still need to discuss the choice of retinal variables to characterize the appearance of primitive groups to complete our systematic description of pattern encodings. The patterns ⊞⊞, e.g., are identical in their spatial and appearance relationships (they have the same two pattern groups, same number of primitives in each group, same distribution style, and same primitive positions) but differ in the choice of retinal variables used to characterize the groups. The first ⊞ uses hue to differentiate, which could encode categories; the second ⊞ uses size, which could encode quantities. The choice of these retinal variables has a profound impact on the appearance of the pattern and we thus now explore the application of retinal variables to primitives within a pattern as well as the additional parameters and effects that arise from their use.

5.3.1 Retinal variables for primitives

Bertin [5, 6] used the term "retinal variables" to describe the graphical attributes that "elevate marks above the plane," and pointed out that these variables are independent of position. Following Bertin's definition, we use the term retinal variable to describe nonspatial graphical attributes. Here we apply these attributes, however, to the individual

primitives that we locate on our lattice to create the specific pattern, as distinct from the variables we discussed in the Sec. 5.1 and Sec. 5.2.

Bertin identified six initial retinal variables: *shape*, *size*, *orientation*, *value*, *color*, and *texture* (*granularity*). *Granularity*, notably, comprises two subdimensions: *size* and *spacing*, as we discussed in Sec. 3.1. *Granularity* is thus a composite visual variable rather than an atomic one. Strictly speaking, *size* also comprises two independent components—*width* and *length* (more in higher-dimensional space). Similarly, Bertin's *color* combines *hue*, *saturation*, and *value/lightness*, where the last component is essentially the already included *value*. By removing these ambiguities as well as *granularity*, we are left with *shape*, *size* (1D size), *orientation* (primitive-level orientation), and *color* (*hue*, *saturation*, *value/lightness*). Unlike the spatial arrangement and group appearance relationship discussed before, these retinal variables are not new variables specific to *patterns*. This list can thus be extended to include any visual variable that is applicable to individual marks. For instance, researchers have added variables such as *resolution*, *transparency*, and *crispness*. For non-static charts, the list can also include *motion* parameters [59, 93]. Past work (e. g., [5, 6, 59, 72, 80]) has investigated the use of these variables and proposed guidelines on their syntactics for mapping (such as which variable is suitable for which type of data). When applied repetitively to primitives, however, these variables can produce new effects, as we discuss next and in Sec. 6.

5.3.2 Regularity of retinal variables

Similar to positional regularity (Sec. 5.1.3), any of the mentioned retinal variables can be applied with a varying degree of regularity. These regularities are, in fact, a secondary characteristic for each of the retinal variables at the primitive level, for which we show examples in Fig. 6 (column "varied mapping"). For variables that can carry numerical values (e. g., size, orientation, value, lightness), similar to positional regularity, we can quantify the range using the maximum deviation and the dispersion level using the standard deviation. For variables that do not carry numerical values (e. g., hue, shape) the quantification of range and dispersion level depends on the variable. For example, we can use entropy to describe the degree of their regularity.[18]

5.4 Transform and fit pattern

We can now apply transformations to the pattern, such as stretching or wrapping[19] and fit it to the host symbol (Fig. 6–⑤). For doing the latter, we may add further steps such as adjusting the relative position between pattern and host symbol, cropping, omitting incomplete primitives, or adding a halo next to the border of the host symbol.

6 INTERACTIONS AMONG VARIABLES

Beyond these three sets of variables that characterize the pattern we also need to discuss the interactions among the variables.

6.1 Dependency between variables

Retinal variables are independent of variables in spatial and appearance relationships, but when they are used in the context of patterns they are constrained by the texture context. Size, e. g., is influenced by the spatial arrangement. Theoretically, a graphical element's *size* can range from 0 to infinity, but in the context of a pattern a primitive cannot have 0 size and increasing the size of primitives beyond a certain point (dependent on lattice configuration and primitive shape) will lead to overlap, which can mask the retinal properties of the primitives or make their characteristics difficult to determine, and ultimately may result in a solid fill (2D or 1D) or thick line (1D).[20]

Dependencies also exist between retinal variables. A primitive's *shape*, e. g., affects both the range and the possible step size of its *orientation*. A round primitive is invariant to *orientation*. The more elongated the shape is, however, the better we can perceive its orientation [5, 6]. Orientation variation on line patterns thus works well.[21] Reliable hue detection, however, is challenging for small primitives.

6.2 Using multiple visual variables to encode data

Bertin [5, 6] introduced the concept of combination of variables but primarily focused on retinal variable combinations. The application

of multiple variables in patterns beyond retinal variables, however, extends his explicit discussion; we can find examples in his own work. Bertin's *grain*, e. g., is a combination of *primitive size* and *spacing*, with the latter two varying consistently across all primitives. As we introduce the concept of multiple groups of primitives within a pattern (Sec. 5.2.1), however, when there is more than one group, we can either covary multiple retinal variables using the same method of primitive grouping (e. g., Fig. 19(b) in Appx. C), or we can decrease covariation by using different methods of primitive grouping (e. g., Fig. 19(c) in Appx. C). With each additional visual variable, we gain an "extent of co-variation" channel that can convey information. This can be an emergent phenomenon (discussed in the next section) and can implicitly encode which visual variables are more closely correlated. In addition, when discussing the retinal variable *shape*, Bertin presents patterns embedded with semantic meaning (Fig. 22 in Appx. D). These examples, however, vary not only primitive shapes but multiple visual variables. *Hatching* patterns used in technical and architectural drawing similarly use multiple variables, such as some of those in Fig. 23 in Appx. D.

6.3 Emergent phenomena

Directly manipulating the pattern attributes we discussed may affect the appearance of the pattern beyond the attributes we control. These *emergent phenomena* are a result of the composite nature of pattern and can affect value, shade and shape and result in optical illusions.[22]

Patterned area value: Following Bertin's definition of value while reserving this concept for individual marks or primitives, we propose to use the term "patterned area value" to specifically describe the ratio of black (or colored) to white—across an entire applied pattern. Bertin controlled *value* akin to traditional halftoning [89], which creates the illusion of various shades of gray by adjusting the lattice size and primitive shapes of numerous black dots on a white background. Given its composite nature, a pattern can inherently produce a *patterned area value* that aligns with the logic of *value* as perceived from halftoning (or from stippling [30, 65]). Even though *patterned area value* is thus emergent and cannot be controlled directly,[23] we need to pay attention to it when encoding data—specifically because *value* variation is a dominant variation for conveying order [5, 6]. It is important to identify which independent pattern parameters affect the *patterned area value*.

Among the variables we discussed, both *orientation* (at the primitive and lattice levels) and carefully designed *shape* variation (i. e., maintaining a constant number of black pixels) can keep the area value constant. In addition, employing combinations of variables can also preserve *patterned area value*, such as simultaneously adjusting primitive size and spacing (i. e., Bertin's *granularity* variation). *Size* variation by itself usually affects *patterned area value*, only the special case of changing the subparameters *width* and *height* in opposite directions can maintain constant *value*. Conversely, an isolated variation of *spacing* or of the individual primitive's *value* also directly affects *patterned area value*.

Understanding which visual variables can be used without causing *patterned area value* variation can help us to reason about the encoding of data. One recommended use of patterns [92], for instance, is to overlay a *pattern* on a *color* encoding to represent a bivariate scalar field, with one data dimension mapped to *pattern* and the other to *color*. Using this concept, Retchless and Brewer [78] compared eight ways to show uncertainty (Fig. 26 in Appx. E). Among them, most participants preferred the design with a dot *pattern* overlaid on *color* (Fig. 26(g)) using *positional regularity* to encode uncertainty. Yet Ware [92] pointed out that, when *patterns* are overlaid on *color*, the bandwidth of *luminance* is shared between the two. When using a pattern to represent one of the bivariate variables; therefore, the (ordered) data dimension should be represented by a *pattern* variation with a constant *patterned area value*, to minimize its impact on the perceived value of the *color* layer, e. g., instead of .

Regional shade: If all primitives in a pattern are the same color, the regional shade is simply the color of the primitives mixed with the (white) background. If there are multiple color primitives, however, we introduce a *regional shade* to the pattern. Incorporating internal *hue* variations among different primitives (e. g., some primitives are blue whereas others are yellow) can result in a different *regional hue* (e. g.,
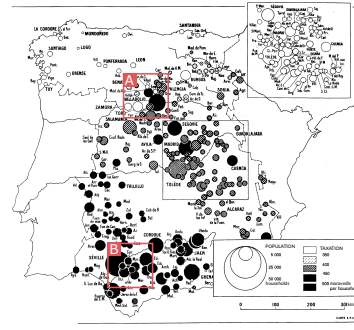

Fig. 9: Symbol map example, edited version (regions A and B added and changes for space efficiency: legend moved moved to inside the figure) based on a map about Population and Taxation in Castille from Bertin's book [5, 6]. A and B can be considered as patterns, whose primitives encode geographic information. Image from [6], © EHESS (text translated), used with permission.

green) due to color mixing, similar to color halftoning .

7 MORE COMPLEX SPATIAL ARRANGEMENTS

Our exploration of the pattern design space also helps us see gaps for exploration and situate and relate existing methods and practices. In our discussion so far we explored patterns based on a regular arrangement of primitives. The regular lattice provides a convenient and consistent framework for expressing the spatial arrangement of primitives within a pattern. Bertin explored this approach and it is broadly accepted in the community for encoding data using patterns—yet many other possibilities exist. We can use, for instance, irregular or nonuniform lattices: Many line dot patterns used to encode lines [9] are defined by irregular spacing and, while lattice crystallography is constrained by nature, visualization design is not.[24] Fundamentally, we thus only need a set of rules for organizing primitives in space—a lattice or any other algorithm that can define the positions of primitives across a given area. In this section, we discuss two representative examples of more complex spatial arrangements that go beyond regular lattice-based configurations: data-driven patterns and nested patterns.

7.1 Data-driven patterns

When we use a lattice to arrange primitives, their position has not been used to encode data—but our pattern system does not prevent us from doing so: we can use each primitive position to encode data directly.

One intuitive approach is to encode geographical position directly (Fig. 6–last row) as the position of each primitive, yielding symbol maps. Yet the created appearance is less intuitively a "pattern" in the canonical sense. Consider, e. g., the symbol map in Fig. 9, in which each dot represents a city and its geographical location on the map. When we read an individual dot we can interpret a city's population (from the dot's *size*) and its tax rate (from the dot's value). When we look at regions, such as Region A, highlighted with a red frame on the map in Fig. 9, we see a pattern emerging consisting of a group of dots—the pattern's primitives. We can see that the pattern of Region A has internal variation and conveys comprehensive regional information. From the pattern, we can discern (1) **where** the cities are located in this region (from the dot *positions*), (2) **how many** cities there are (from the dot *density*), and (3) **what** their characteristics are (with dot *size* representing population and *value* representing tax rate). (1) and (2) are emergent variables that come from geographic information, which uses arrangement-level visual variables, whereas (3) is directly encoded on the primitive-level. This pattern also allows us to compare different regions. We can see, e. g., that the pattern in Region A is different from the one in Region B, and we can find that the taxation levels within Region A may have a greater diversity compared to those in Region B. We can also seek 'patterns' across the map that have similar visual characteristics to that of Region A. When we read the map, we can thus visually select different patterns at multiple scales and multiple places concurrently to understand geospatial data based on the emergent visual patterns. This process aligns with MacEachren's [60] goal of cartographic visualization: to "assist an analyst in discovering patterns and relationships in the data" at multiple locations and multiple scales.

We note that data-driven arrangements are not limited to accurate geographic locations (Fig. 31(a)); they can also be inaccurate geographic (i. e., transformations applied to accurate geographic locations, e. g., to avoid overlap between dots, Fig. 31(b)), or gridded geographic (e. g.,

Fig. 7(b) and 31(c) show an application of this type of pattern). In addition, such arrangements can even be nongeographic—derived from statistical data (e. g., [38]). Fundamentally, we encode the positional information of primitives relative to coordinate axes. In maps, we use geographical locations, but for a scatterplot, e. g., the point positions are the data's x- and y-dimensions. We can thus use a similar way to interpret emergent "patterns" on scatterplots. Our pattern system allows us to describe and differentiate patterns in such cases and facilitates further research on how people can accomplish data analysis tasks at these different pattern levels and with different pattern characteristics.

7.2 Nested patterns

As we discussed in Sec. 5, our pattern system characterizes a pattern by setting the rules for: (1) *spatial relationships* of primitives (Sec. 5.1), (2) *appearance relationships* among primitives (Sec. 5.2), and (3) *retinal visual variables* that define the appearance of each individual primitive (Sec. 5.3). For the last of these rules we can also treat patterns themselves as retinal variables and use a pattern as primitives of another pattern—a "nested pattern." The primitives of the base pattern then serve as host symbols for the new pattern. We can characterize the new added pattern using the same three sets of rules from our overall system and, for the retinal variables of its primitives, we can again add another layer of pattern. We thus provide the possibility of adding multiple layers of additional patterns ad infinitum.[25]

By varying how spatial relationships are defined, we can construct both lattice-based and data-driven nested patterns. Notably, either type can serve as the spatial rule for both the base pattern and the added pattern. A two-layer nested pattern thus yields four possible configurations: (1) lattice pattern on lattice pattern (e. g., Fig. 8); (2) lattice pattern on data-driven pattern (e. g., Pattern A in Fig. 9); (3) data-driven pattern on lattice pattern (this configuration is less common, but we illustrate it in Fig. 30 in Appx. C: a regular grid pattern hosts a dot pattern based on geographical data); and (4) data-driven pattern on data-driven pattern (e. g., Fig. 29 in Appx. E), where we can consider the entire map as a base pattern, and the small maps are the added pattern, and both of which use position to encode aspects of geography).

Our new pattern system thus offers a theoretical lens through which we can not only discover new design possibilities but also compare, explain, and relate different designs, to theoretically connect and perhaps understand idioms that we usually separate with different names.

8 BRINGING IT ALL TOGETHER

We deeply engaged with and discussed Bertin's use of texture and its various translations, interpretations, and applications and noted inconsistencies. This leads us to recommend avoiding the term "texture" in lists of visual variables, and to instead discuss the use of granularity, spacing, and shape of *pattern* primitives. Patterns are ultimately a composite visual variable that consists of a series of expressive components whose design space we explored. We are not the first to have attempted this exploration but our pattern system provides a new unifying angle. Specifically, we identified three sets of attributes of patterns. First, *spatial arrangement* relationships of pattern primitives distinguish how pattern primitives are arranged, either through a static, dynamic, or irregular lattice or modified by data parameters. This conceptualization of arrangements encapsulates previously proposed pattern parameters [10, 24, 64, 71, 83] such as density or regularity. Second, *group appearance relationships* introduce internal variation within patterns as a novel concept that includes the number of primitive groups, ratio between groups, and distribution style. Our group appearance relationships allow us to categorize and discuss pattern visualizations by Bertin and others. Examples in Fig. 17(b) and 18(a) in Appx. C also remind us of waffle charts, BallotMaps [100], or proportion visualizations more generally. Finally, we discussed the use of *retinal variables on pattern primitives*. Although retinal variables as such have been investigated before, we introduce new variables made possible by patterns, specifically the regularity of the retinal variables and the visual effects caused by using retinal variables repeatedly. Based on our three attribute sets we further discussed the use of multiple variables and the creation of patterns through more complex spatial arrangements.

Our design space is deeply generative—it is easy to conceive of pattern variations using any of our dimensions. Existing examples (e. g., Fig. 7(a) and 8 as well as Fig. 25 in Appx. E) demonstrate the potential of patterns in encoding data, making them worthy of further exploration. Creative design experiments are needed to establish possibilities and rigorous empirical work required to understand how these variations are perceived. One aspect that we have not at all touched upon, however, is the use of patterns decidedly ***not*** for encoding data. Think, for example, about the use of non-lattice arrangements or intentional irregularity in a lattice-based pattern being used simply for aesthetic purposes—to make the visualization more interesting. Fig. 35 and 37, for instance, vary the dot placement in an even yet irregular fashion, which shows the overall density well yet in a visually more interesting fashion than a regular lattice-based placement—linking our system to approaches used in non-photorealistic rendering on stippling [30, 65] and non-repetitive patterns (e. g., [4, 47, 65, 81, 99]). Also parameters other than position can be manipulated purely for aesthetic purposes, such as line width to simulate hand-drawn lines as in traditional data graphics [3, 102].

The degree to which such uses of pattern for aesthetics [42] are effective should be further investigated, as well as their effect on readability [15]. In addition, we need to study the encoding size of types of pattern variations—that is, how many different variations of a texture are even noticeable by a viewer. Any of these metrics can be applied to a number of specific questions. For example, it would be interesting to explore whether regular patterns create more readable visualizations than random placements or if and how semantically resonant patterns can be created with abstract pattern designs [58]. Specifically intriguing is the use of combinations of patterns with other retinal variables as well as nested patterns. We see few examples of their application in practice. Is it because they are a bad idea? Because common visualization libraries do not allow or facilitate pattern encoding? Or because people have lacked the conceptual framing to explore and advocate for or study their use? Ultimately, our design space is thus also evaluative. It aids in the design of studies comparing different pattern parameters to ultimately build up a more comprehensive understanding of what makes patterns interesting, effective and appreciated. In addition, given the limited pattern support in current visualization tools (Appx. F), one could build a more flexible pattern library for based on our pattern system to support pattern design and empirical studies.

Finally, having described a wide design space, we do not know the various limits of the use of patterns. As we use more parameters and visual variables for encoding in composites, the resulting intricate patterns are unlikely to be easily interpretable or reliably decoded. Like hand-drawn stipple images, or efforts to convey surface material characteristics, the properties of *patterns* may then become more akin to those of natural *textures* [62, 63], suggesting that our system may have potential for bridging conceptual, linguistic, and perhaps practical gaps in data visualization design.

SUPPLEMENTAL MATERIAL POINTERS

We share our additional material (author version of the paper including the appendix with footnotes, self-created figures) at osf.io/z7ae2.

IMAGES/FIGURES LICENSE/COPYRIGHT

REFERENCES

[1] C. Alexander, S. Ishikawa, and M. Silverstein. *A Pattern Language: Towns, Buildings, Construction.* Oxford UP, New York, 1977.

[2] M. Amadasun and R. King. Textural features corresponding to textural properties. *IEEE Trans Syst Man Cybern*, 19(5):1264–1274, 1989. doi: 10/b48tfv

[3] B. Bach, P. Dragicevic, S. Huron, P. Isenberg, Y. Jansen, C. Perin, A. Spritzer, R. Vuillemot, W. Willett, and T. Isenberg. Illustrative data graphics in 18th–19th century style: A case study. In *Posters of IEEE VIS*, 2013. url: hal.science/hal-00849079.

[4] P. Barla, S. Breslav, J. Thollot, F. Sillion, and L. Markosian. Stroke pattern analysis and synthesis. *Comput Graph Forum*, 25(3):663–671, 2006. doi: 10/cv8vz7

[5] J. Bertin. *Sémiologie Graphique.* Éditions de l'EHESS, Paris, 3rd ed., 1998. url: editions.ehess.fr/ouvrages/ouvrage/semiologie-graphique/.

[6] J. Bertin. *Semiology of Graphics: Diagrams, Networks, Maps.* Esri Press, Redlands, 2011. url: esri.com/en-us/esri-press/browse/semiology-of-graphics-diagrams-networks-maps.

[7] J. F. Blinn and M. E. Newell. Texture and reflection in computer generated images. *Commun ACM*, 19(10):542–547, 1976. doi: 10/dct2v6

[8] R. Brath. Dashed and patterned lines for visualization (aka 1D texture). Blog post: richardbrath.wordpress.com/2021/05/27/dashed-and-patterned-lines-for-visualization-aka-1d-texture/, 2021. Accessed: Mar. 2025.

[9] R. Brath, M. Peters, and R. Senior. Visualization for communication: The importance of aesthetic sizzle. In *Proc. IV*, pp. 724–729. IEEE Comp. Soc., Los Alamitos, 2005. doi: 10.1109/IV.2005.145

[10] C. Brewer. *Designing Better Maps: A Guide for GIS Users.* Esri Press, Redlands, 2016. urn:oclc:record:1012107199.

[11] W. C. Brinton. *Graphic Methods for Presenting Facts.* The Engineering Magazine Company, New York, 1914. urn:oclc:record:1045528209.

[12] W. C. Brinton. *Graphic Presentation.* Brinton Assoc., New York, 1939. urn:oclc:record:1045601113.

[13] P. Brodatz. *Textures: A Photographic Album for Artists and Designers.* Dover Publications, New York, 1966. urn:oclc:record:1225861155.

[14] H. Buchholz and J. Döllner. View-dependent rendering of multiresolution texture-atlases. In *Proc. Visualization*, pp. 215–222. IEEE Comp. Soc., Los Alamitos, 2005. doi: 10/bcn554

[15] A.-F. Cabouat, T. He, P. Isenberg, and T. Isenberg. PREVis: Perceived readability evaluation for visualizations. *IEEE Trans Vis Comput Graph*, 31(1):1083–1093, 2025. doi: 10/njnz

[16] B. Cabral and L. C. Leedom. Imaging vector fields using line integral convolution. In *Proc. SIGGRAPH*, pp. 263–270. ACM, New York, 1993. doi: 10/drnq5c

[17] A. Cairo. Download the Datasaurus: Never trust summary statistics alone; always visualize your data. Archived blog post/web site: thefunctionalart.com/2016/08/download-datasaurus-never-trust-summary.html, 2016. Accessed: Mar. 2025.

[18] J. Caivano. Towards an order system for visual texture. *Lang Des*, 2(1):59–84, 1994. url: colorysemiotica.files.wordpress.com/2015/11/1994lang.pdf.

[19] J. L. Caivano. Visual texture as a semiotic system. *Semiotica*, 80(3/4):239–252, 1990. doi: 10/c9tqj3

[20] M. S. T. Carpendale. Considering visual variables as a basis for information visualisation. Technical Report 2001-693, University of Calgary, Canada, 2003. hdl: 1880/45758.

[21] E. Catmull. *A Subdivision Algorithm for Computer Display of Curved Surfaces.* PhD thesis, University of Utah, Salt Lake City, 1974. url: collections.lib.utah.edu/ark:/87278/s6cg2j21.

[22] G. Y.-Y. Chan, P. Xu, Z. Dai, and L. Ren. ViBr: Visualizing bipartite relations at scale with the minimum description length principle. *IEEE Trans Vis Comput Graph*, 25(1):321–330, 2019. doi: 10/gnd8bg

[23] R. Y. Cho, V. Yang, and P. E. Hallett. Reliability and dimensionality of judgments of visually textured materials. *Percept Psychophys*, 62(4):735–752, 2000. doi: 10/c7q45j

[24] D. H. S. Chung, D. Archambault, R. Borgo, D. J. Edwards, R. S. Laramee, and M. Chen. How ordered is it? On the perceptual orderability of visual channels. *Comput Graph Forum*, 35(3):131–140, 2016. doi: 10/f8v3ps

[25] W. S. Cleveland and R. McGill. Graphical perception: Theory, experimentation, and application to the development of graphical methods. *J Am Stat Assoc*, 79(387):531–554, 1984. doi: 10/cbtq7v

[26] W. S. Cleveland and R. McGill. Graphical perception and graphical methods for analyzing scientific data. *Science*, 229(4716):828–833, 1985. doi: 10/fkhq59

[27] J. Cresswell. *Oxford Dictionary of Word Origins.* Oxford UP, UK, 2010. doi: 10/gtn794

[28] J. de Brouwer, J. Eekhout, A. Besse-Lototskaya, A. Hoitink, C. Ter Braak, and P. Verdonschot. Flow thresholds for leaf retention in hydrodynamic wakes downstream of obstacles. *Ecohydrology*, 10(7), article no. e1883, 10 pages, 2017. doi: 10/gb46nc

[29] B. D. Dent, J. Torguson, and T. W. Hodler. *Cartography: Thematic Map Design.* McGraw-Hill, Boston, 6th ed., 2008. urn:oclc:record:1280806442.

[30] O. Deussen and T. Isenberg. Halftoning and stippling. In P. Rosin and J. Collomosse, eds., *Image and Video based Artistic Stylisation*, vol. 42, chap. 3, pp. 45–61. Springer, London, 2013. doi: 10/kt29

[31] D. DiBiase, A. M. MacEachren, J. B. Krygier, and C. Reeves. Animation and the role of map design in scientific visualization. *Cartogr Geogr Inf Syst*, 19(4):201–214, 1992. doi: 10/cv3jm4

[32] N. A. Dodgson. Regularity and randomness in Bridget Riley's early Op Art. In *Proc. CAe*, pp. 107–114. EG Assoc., Goslar, 2008. doi: 10/n47x

[33] G. Elber. Rendering with parallel stripes. *IEEE Comput Graph Appl*, 21(3):44–52, 2001. doi: 10/cngmdr

[34] M. FC, T. L. Davis, and ggplot2 authors. ggpattern: 'ggplot2' pattern geoms. GitHub repo., github.com/coolbutuseless/ggpattern, 2022.

[35] J. A. Gatto, A. W. Porter, and J. Selleck. *Exploring Visual Design.* Davis Publications, Worcester, 1978. urn:oclc:record:1310593305.

[36] C. Giacovazzo, ed. *Fundamentals of Crystallography.* International Union of Crystallography, 1992. urn:oclc:record:1280758523.

[37] U. Grenander and M. I. Miller. *Pattern Theory: From Representation to Inference.* Oxford UP, New York, 2007. doi: 10/pr7x

[38] J. Görtler, M. Spicker, C. Schulz, D. Weiskopf, and O. Deussen. Stippling of 2D scalar fields. *IEEE Trans Vis Comput Graph*, 25(6):2193–2204, 2019. doi: 10/gfz5hr

[39] S. Haroz, R. Kosara, and S. L. Franconeri. Isotype visualization: Working memory, performance, and engagement with pictographs. In *Proc. CHI*, pp. 1191–1200. ACM, New York, 2015. doi: 10/gf65d5

[40] S. Haroz and D. Whitney. How capacity limits of attention influence information visualization effectiveness. *IEEE Trans Vis Comput Graph*, 18(12):2402–2410, 2012. doi: 10/f4fwcw

[41] R. Harris. *Information Graphics: A Comprehensive Illustrated Reference.* Oxford UP, New York, 2000. urn:oclc:record:1310734651.

[42] T. He, P. Isenberg, R. Dachselt, and T. Isenberg. BeauVis: A validated scale for measuring the aesthetic pleasure of visual representations. *IEEE Trans Vis Comput Graph*, 29(1):363–373, 2023. doi: 10/kt3n

[43] T. He, P. Isenberg, and T. Isenberg. Data embroidery with black-and-white textures. In *Proc. alt.VIS*, article no. 3, 5 pages, 2023. url: hal.science/hal-04197527.

[44] T. He, Y. Zhong, P. Isenberg, and T. Isenberg. Design characterization for black-and-white textures in visualization. *IEEE Trans Vis Comput Graph*, 30(1):1019–1029, 2024. doi: 10/gtkwg3

[45] C. G. Healey and J. T. Enns. Building perceptual textures to visualize multidimensional datasets. In *Proc. Visualization*, pp. 111–118. IEEE Comp. Soc., Los Alamitos, 1998. doi: 10/dz9223

[46] J. D. Hunter. Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007. doi: 10/drbjhg

[47] T. Hurtut, P.-E. Landes, J. Thollot, Y. Gousseau, R. Drouillhet, and J.-F. Coeurjolly. Appearance-guided synthesis of element arrangements by example. In *Proc. NPAR*, pp. 51–60. ACM, New York, 2009. doi: 10/dfvq9v

[48] T. C. Inglis and C. S. Kaplan. Op Art rendering with lines and curves. *Comput Graph*, 36(6):607–621, 2012. doi: 10/f38rf2

[49] J. Jo, F. Vernier, P. Dragicevic, and J.-D. Fekete. A declarative rendering model for multiclass density maps. *IEEE Trans Vis Comput Graph*, 25(1):470–480, 2019. doi: 10/gd868w

[50] S. Johnson, F. Samsel, G. Abram, D. Olson, A. J. Solis, B. Herman, P. J. Wolfram, C. Lenglet, and D. F. Keefe. Artifact-based rendering: Harnessing natural and traditional visual media for more expressive and engaging 3D visualizations. *IEEE Trans Vis Comput Graph*, 26(1):492–502, 2020. doi: 10/gghbxx

[51] C. S. Kaplan. *Introductory Tiling Theory for Computer Graphics.* Springer, Cham, 2009. doi: 10/n5g9

[52] M. Kraak and F. Ormeling. *Cartography: Visualization of Geospatial Data.* CRC Press, Boca Raton, 4th ed., 2020. doi: 10/gtn795

[53] J. Krygier and D. Wood. *Making Maps: A Visual Guide to Map Design for*

GIS. Guilford Press, New York, 3rd ed., 2016. urn:oclc:record:1194418017.

[54] A. Kumpf, M. Rautenhaus, M. Riemer, and R. Westermann. Visual analysis of the temporal evolution of ensemble forecast sensitivities. *IEEE Trans Vis Comput Graph*, 25(1):98–108, 2019. doi: 10/g8rfm9

[55] A. Kumpf, B. Tost, M. Baumgart, M. Riemer, R. Westermann, and M. Rautenhaus. Visualizing confidence in cluster-based ensemble weather forecast analyses. *IEEE Trans Vis Comput Graph*, 24(1):109–119, 2018. doi: 10/gcp9ng

[56] F. Liu and R. W. Picard. Periodicity, directionality, and randomness: Wold features for image modeling and retrieval. *IEEE Trans Pattern Anal Mach Intell*, 18(7):722–733, 1996. doi: 10/fjb4ss

[57] A. Lu and D. S. Ebert. Example-based volume illustrations. In *Proc. Visualization*, pp. 655–662. IEEE Comp. Soc., Los Alamitos, 2005. doi: 10/bhfj9w

[58] Z. Lu, T. He, J. Hong, L. Yao, and T. Isenberg. Designing semantically-resonant abstract patterns for data visualization. arXiv preprint 2505.14816, arXiv.org, May 2025. doi: 10/pn6g

[59] A. M. MacEachren. *How Maps Work: Representation, Visualization, and Design*. Guilford Press, New York, 2004. urn:oclc:record:1409557208.

[60] A. M. MacEachren and J. H. Ganter. A pattern identification approach to cartographic visualization. *Cartographica*, 27(2):64–81, 1990. doi: 10/d689wm

[61] A. M. MacEachren, R. E. Roth, J. O'Brien, B. Li, D. Swingley, and M. Gahegan. Visual semiotics & uncertainty visualization: An empirical study. *IEEE Trans Vis Comput Graph*, 18(12):2496–2505, 2012. doi: 10/f4fvbf

[62] R. Maciejewski, T. Isenberg, W. M. Andrews, D. S. Ebert, and M. C. Sousa. Aesthetics of hand-drawn vs. computer-generated stippling. In *Proc. CAe*, pp. 53–56. EG Assoc., Goslar, 2007. doi: 10/kt4w

[63] R. Maciejewski, T. Isenberg, W. M. Andrews, D. S. Ebert, M. C. Sousa, and W. Chen. Measuring stipple aesthetics in hand-drawn and computer-generated images. *IEEE Comput Graph Appl*, 28(2):62–74, 2008. doi: 10/chrcpn

[64] J. Mackinlay. Automating the design of graphical presentations of relational information. *ACM Trans Graph*, 5(2):110–141, 1986. doi: 10/dxdkkp

[65] D. Martín, G. Arroyo, A. Rodríguez, and T. Isenberg. A survey of digital stippling. *Comput Graph*, 67:24–44, 2017. doi: 10/gc6xf5

[66] J. Matejka and G. Fitzmaurice. Same stats, different graphs: Generating datasets with varied appearance and identical statistics through simulated annealing. In *Proc. CHI*, pp. 1290–1294. ACM, New York, 2017. doi: 10/gdtg2w

[67] V. Matvienko and J. Krüger. Explicit frequency control for high-quality texture-based flow visualization. In *Proc. SciVis*, pp. 41–48. IEEE Comp. Soc., Los Alamitos, 2015. doi: 10/k8sg

[68] MDN contributors. Mozilla developer network web docs – SVG tutorial: Patterns. developer.mozilla.org/en-US/docs/Web/SVG/Tutorial/Patterns, 2023. Accessed: Mar. 2025.

[69] C. M. McColeman, F. Yang, T. F. Brady, and S. Franconeri. Rethinking the ranks of visual channels. *IEEE Trans Vis Comput Graph*, 28(1):707–717, 2022. doi: 10/gnqjcb

[70] Merriam-Webster. texture. In *Merriam-Webster.com dictionary*. url: merriam-webster.com/dictionary/texture/; Accessed: Mar. 2025.

[71] J. L. Morrison. A theoretical framework for cartographic generalization with the emphasis on the process of symbolization. In *International Yearbook of Cartography*, vol. 14, pp. 115–127. 1974.

[72] T. Munzner. *Visualization Analysis and Design*. CRC Press, Boca Raton, 2014. doi: 10/gd3xgq

[73] O. Neurath. *From Hieroglyphics to Isotype: A Visual Autobiography*. Hyphen Press, London, 2010. Edited by M. Eve and C. Burke, url: perpensapress.com/books/from-hieroglyphics-to-isotype.

[74] R. Penrose. Pentaplexity: A class of non-periodic tilings of the plane. *Math Intell*, 2(1):32–37, 1979. doi: 10/b7pbxd

[75] F. E. Pierce. The Tenement-House committee maps. *Harpers Weekly*, 39(1987):60–62, 1895. url: lccn.loc.gov/2006629793.

[76] Plotly Technologies Inc. Collaborative data science. Plotly Technologies Inc., Montréal. Software; url: plot.ly, 2015.

[77] A. R. Rao and G. L. Lohse. Towards a texture naming system: Identifying relevant dimensions of texture. *Vision Res*, 36(11):1649–1669, 1996. doi: 10/ffnq8n

[78] D. P. Retchless and C. A. Brewer. Guidance for representing uncertainty on global temperature change maps. *Int J Climatol*, 36(3):1143–1159, 2015. doi: 10/f8c645

[79] R. Rosenholtz. Texture perception. In *The Oxford Handbook of Perceptual Organization*, chap. 9, pp. 167–186. Oxford UP, 2015. doi: 10/gtn6ps

[80] R. E. Roth. Visual variables. In D. Richardson, N. Castree, M. F. Goodchild, A. Kobayashi, W. Liu, and R. Marston, eds., *International Encyclopedia of Geography: People, the Earth, Environment and Technology*. Wiley, 2017. doi: 10/gm53xj

[81] M. P. Salisbury, S. E. Anderson, R. Barzel, and D. H. Salesin. Interactive pen-and-ink illustration. In *Proc. SIGGRAPH*, pp. 101–108. ACM, New York, 1994. doi: 10/b549m9

[82] H. H. Shenas and V. Interrante. Compositing color with texture for multivariate visualization. In *Proc. GRAPHITE*, p. 443–446. ACM, New York, 2005. doi: 10/fqhj4d

[83] T. A. Slocum, R. B. McMaster, F. C. Kessler, and H. H. Howard. *Thematic Cartography and Geovisualization*. CRC Press, Boca Raton, 2022. doi: 10/gtkwks

[84] D. Smith, J. S. Myers, C. S. Kaplan, and C. Goodman-Strauss. An aperiodic monotile. *Comb Theory*, 4(1), article no. 6, 91 pages, 2024. doi: 10/njvf

[85] H. Tamura, S. Mori, and T. Yamawaki. Textural features corresponding to visual perception. *IEEE Trans Syst Man Cybern*, 8(6):460–473, 1978. doi: 10/d3b8fz

[86] M. Tuceryan and A. K. Jain. Texture analysis. In *Handbook of Pattern Recognition and Computer Vision*, pp. 235–276. World Scientific, 1993. doi: 10/fww58x

[87] E. R. Tufte. *The Visual Display of Quantitative Information*. Graphics Press, Cheshire, 2nd ed., 2001. urn:lcp:visualdisplayofq0000edwa:lcpdf :23aa8e83-25fc-4744-825e-0c38dee32975.

[88] J. Tyner. *Principles of Map Design*. Guilford Publications, New York, 2017. url: guilford.com/books/Principles-of-Map-Design/Judith-Tyner/9781462517121.

[89] R. Ulichney. *Digital Halftoning*. MIT Press, Cambridge, 1987. urn:oclc :record:1033645312. doi: 10/pdt5

[90] J. J. van Wijk. Spot noise texture synthesis for data visualization. In *Proc. SI3D*, pp. 309–318. ACM, New York, 1991. doi: 10/bj3xtn

[91] C. Ware. Quantitative texton sequences for legible bivariate maps. *IEEE Trans Vis Comput Graph*, 15(6):1523–1530, 2009. doi: 10/bx9ccv

[92] C. Ware. *Information Visualization: Perception for Design*. Elsevier, Cambridge, MA, 4th ed., 2020. doi: 10/mj55

[93] C. Ware and R. Bobrow. Motion to support rapid interactive queries on node-link diagrams. *ACM Trans Appl Percept*, 1(1):3–18, 2004. doi: 10/b35p7d

[94] C. Ware and C. Kastrisios. Evaluating countable texture elements to represent bathymetric uncertainty. In *EuroVis Short Papers*, pp. 1–5. EG Assoc., Goslar, 2022. doi: 10/gtmxrx

[95] C. Ware and W. Knight. Orderable dimensions of visual texture for data display: Orientation, size and contrast. In *Proc. CHI*, pp. 203–209. ACM, New York, 1992. doi: 10/bxpc4x

[96] C. Ware and W. Knight. Using visual texture for information display. *ACM Trans Graph*, 14(1):3–20, 1995. doi: 10/c578cv

[97] A. Wilkins, J. Emmett, and G. Harding. Characterizing the patterned images that precipitate seizures and optimizing guidelines to prevent them. *Epilepsia*, 46(8):1212–1218, 2005. doi: 10/dqxqz8

[98] L. Wilkinson. *The Grammar of Graphics*. Springer, New York, 2nd ed., 2005. doi: 10/bhsbp7

[99] G. Winkenbach and D. H. Salesin. Computer-generated pen-and-ink illustration. In *Proc. SIGGRAPH*, pp. 91–100. ACM, New York, 1994. doi: 10/cqcbjm

[100] J. Wood, D. Badawood, J. Dykes, and A. Slingsby. BallotMaps: Detecting name bias in alphabetically ordered ballot papers. *IEEE Trans Vis Comput Graph*, 17(12):2384–2391, 2011. doi: 10/frhfn7

[101] J. Wood, J. Dykes, and A. Slingsby. Visualisation of origins, destinations and flows with OD maps. *Cartogr J*, 47(2):117–129, 2010. doi: 10/fhg6mf

[102] J. Wood, P. Isenberg, T. Isenberg, J. Dykes, N. Boukhelifa, and A. Slingsby. Sketchy rendering for information visualization. *IEEE Trans Vis Comput Graph*, 18(12):2749–2758, 2012. doi: 10/f4fwrn

[103] Y. Zhang, R. Jiang, L. Xie, Y. Zhao, C. Liu, T. Ding, S. Chen, and X. Yuan. OldVisOnline: Curating a dataset of historical visualizations. *IEEE Trans Vis Comput Graph*, 30(1):551–561, 2024. doi: 10/pd88

[104] Y. Zhong, T. Isenberg, and P. Isenberg. Black-and-white textures for visualization on E-ink displays. In *Posters at IEEE VIS*, 2020. url: hal.science/hal-02944212.

Reframing *Pattern*:
A Comprehensive Approach to a Composite Visual Variable

Appendix

In this appendix we provide additional figures and some detailed discussion that we could include in the main paper due to space limitations or because it was not essential for explaining our approach.

A CAIVANO'S SYSTEM FOR TEXTURE DESCRIPTION

From the field of architecture, Caivano [18, 19] adopts a design approach to describe patterns (although under the term "texture"). He classifies *simple textures* and *complex textures*, defining the former as "the uniform repetition of a certain element" and the latter as combinations of multiple sets of simple textures [19]. His simple textures are essentially two elements within a tiling unit. Caivano constructs his simple texture through the tiling of a *texture unit* (the minimal entity for repetition; see Fig. 10 (b)). In Caivano's model, a texture unit comprises a pair of texturing elements (see Fig. 10 (c)). He then treats texture as a tripartite variable, including size of the texture elements, directionality (the unit's width-height ratio), and density (the overall black-to-white ratio). Later [18], he refines his theory to describe pattern variation through the shape of texture elements, organization (the relative positions of the two texture elements within the tiling unit), proportionality (the tiling unit's width-height ratio), and density (the overall black-to-white ratio).

Cavaino, however, did not intend to use texture as a visual variable for data encoding. As a result, not all the dimensions he identifies are directly manipulable, and his composition of simple textures is unsuitable for our purpose. In addition, we interpret Caivano's classification such that a simple texture should be the most basic form of texture—without any subsets of textures ("uniform repetition of a certain element"). If a texture is a combination of multiple sets of textures, we should categorize it as a complex texture. Upon analysis of Caivano's simple texture composition, however, we identify two subsets of texture within it, which appears to contradict our interpretation of his definition of a simple texture. Fig. 10(d) illustrates the two subtextures identified in a simple texture according to Caivano's composition, with blue and red highlighting, respectively. We thus develop our own pattern configuration and offer an alternative that covers a wider design space of *pattern*, specifically aimed at encoding data, which we present in Sec. 5.

B TERMS TRANSLATED AS "PATTERN" IN BERTIN'S BOOK

In the translator's note (Fig. 4), the translator mentions that "pattern" is the English equivalent for the French word "texture." However, in the book, the term "pattern" is not actually translated from "texture" in French. The translator translated the following French terms from Bertin's book [5] into English as "pattern":

"Semis," which means "seedbed." Bertin extends this term to refer to a "dot pattern." He consistently uses it in phrases like "semis régulier (a regular pattern)" or "semis de point (a dot pattern)," but it always refers to something similar, as shown in Fig. 11. In the Lexicon of the book [5], "semis" is explained as "Type d'imposition qui disperse, sur le plan, les éléments d'une variable," which translates to a type of graphic that spreads the elements of a variable across a plane. "Imposition" in Bertin's book refers to a graphic type (e. g., a map). This reflects the characteristic regularity of patterns, which are governed by rules of arrangement. Here, "semis" serves as the rule for generating a pattern in terms of the spatial arrangement of graphical primitives across a plane.

"Baguettage/ligne" refers to a single line but is translated as "line pattern." Similarly, "pointillé" has been translated as "dot pattern," although it actually means "dotted" and, by extension, can denote a dotted line."

"Trame" is a term used in drawing or technical drawing, with a meaning similar to "hatching," a specific type of pattern used
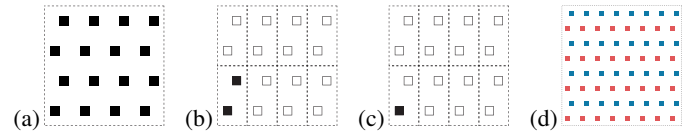

Fig. 10: Recreated schematic drawing based on Caivano's diagram of composition of a simple texture [18, 19]. (a) A texture, (b) a texture unit, (c) a texture element, (d) two subsets of textures identified from this simple texture composition, colored blue and red, respectively.
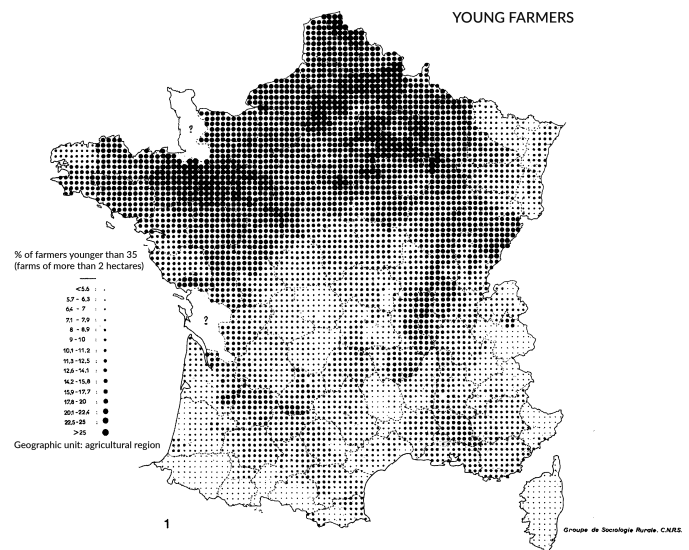

Fig. 11: Examples of "semis" from Bertin's book. [5, 6]; © EHESS (text translated), used with permission.

in technical drawing for indicating materials. The word "trame" also conveys a sense of a grid. In Bertin's book, it appears in phrases such as "des trames mécaniques" (mechanical hatching) and "des trames préfabriquées" (prefabricated hatching), referring to preprinted hatchings. Fig. 23 shows examples of these preprinted hatchings.

C ADDITIONAL FIGURES FOR OUR NEW PATTERN DEFINITION

Fig. 12–20 illustrate some aspects of our new pattern definition in more detail, beyond our summary in Fig. 6.

D ADDITIONAL FIGURES FROM BERTIN'S BOOK THAT WE USED IN OUR DISCUSSION

This section includes supplemental examples (Fig. 21–24) from Bertin's book [5, 6], to which we referred in our discussion in the main paper. Specifically, Fig. 21 shows Bertin's six retinal visual variables as they are applied onto line and area marks (as well as point marks). Next, Fig. 22 shows an example by Bertin for patterns with an embedded semantic meaning—here from the field of cartography—that have achieved the status of symbols in this domain. More generally, specific *hatching* patterns are often used in technical and architectural drawing as we show in Fig. 23 from Bertin's book. Finally, Fig. 24 shows an example from Bertin for repeated, regular patterns that can cause a visual sense of instability, or Moiré effect.
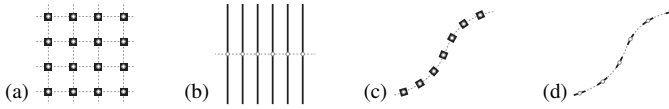
Fig. 12: Configuration of *pattern* with (a) 2D primitives on a 2D lattice, tiling across a 2D area—2×2×2; (b) 1D primitive on a 1D lattice, tiling across a 2D area—1×1×2; (c) 2D primitive on a 1D lattice, tiling along a 1D line (that fills a 2D area)—2×1×1; and (d) 1D primitive on a 1D lattice, tiling along a 1D line (that fills a 2D area)—1×1×1. We can apply (a) and (b) to area symbols, or to point symbols with area (e. g., circles) and line symbols with width. We can apply (c) and (d) only to line symbols, as their lattices follow the line symbol's direction. Here, the primitives are black. The gray dashed lines and white dots represent the lattice and the lattice points, which we use only for descriptive purposes and they are not part of the *pattern* itself.
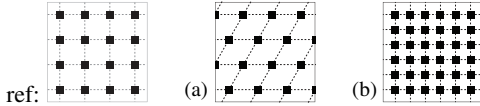
Fig. 13: Variation on (a) shape and (b) size of unit cells.
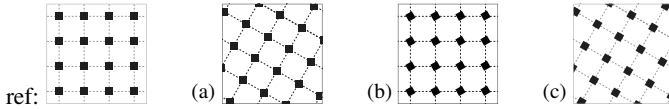
Fig. 14: Orientation at different levels, compared to the left: (a) at arrangement-level, (b) at primitive level, and (c) at both levels (we can call it orientation of the whole pattern), all with same degrees.
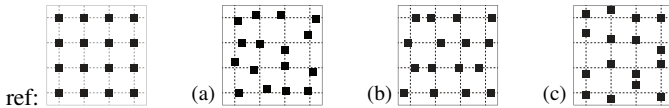
Fig. 15: Positional regularity variation, compared to the reference on the left: (a) in both directions or (b, c) only in one direction.
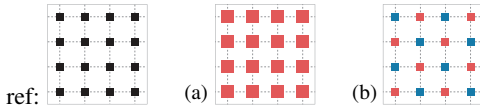
Fig. 16: The reference (left) has one primitive group: (a) global encoding with hue & size, with one primitive group; (b) pattern with internal variation for hue (one subset blue, another subset red), with two primitive groups.
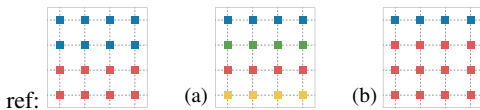
Fig. 17: Compared to the reference (left), (a) increase of primitive groups from 2 to 4, with even primitive count per group; (b) variation of the ratio between each group (from 1:1 to 1:3), with a constant number of primitive groups (2). Here, the different primitive groups are differentiated by hue, but any primitive-level variable can be used, i. e., size, shape, etc.
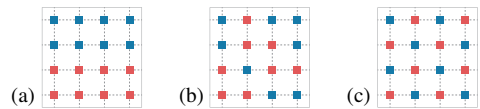
Fig. 18: Pattern with internal variation with different primitive group arrangement: (a) grouped, (b) interspersed, and (c) dispersed.

E MORE EXAMPLES OF PATTERNS

This section includes additional examples (Fig. 25–48) from authors other than Bertin that were partially already discussed in the main paper—some of which we found via OldVisOnline [103]. Specifically, the historical example in Fig. 25—created before the the availability of computers—illustrates the use of pattern groups and of the relationship of their respective appearances to encode data (by shape and color in Fig. 25; another example that uses line orientation and secondary line
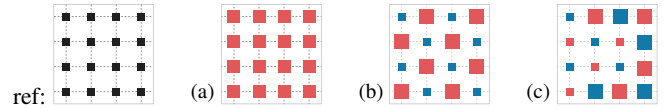
Fig. 19: The reference (left) has one primitive group: (a) global encoding with hue and size, with one primitive group ("combination of variables" in Bertin's book); (b) Primitives' hue and size covary in the pattern, with same two primitive groups for the two variables; (c) Primitives' hue and size do not covary in the pattern, two primitive groups for hue (i. e., two hues) and size (i. e., two sizes), but they are not the same two groups.
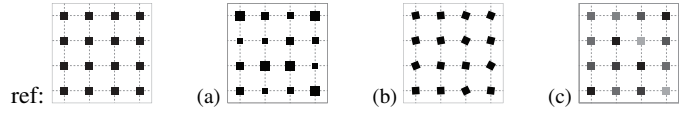

Fig. 20: Primitive regularity variation, compared to the left: (a) for size, (b) for orientation, and (c) for value.
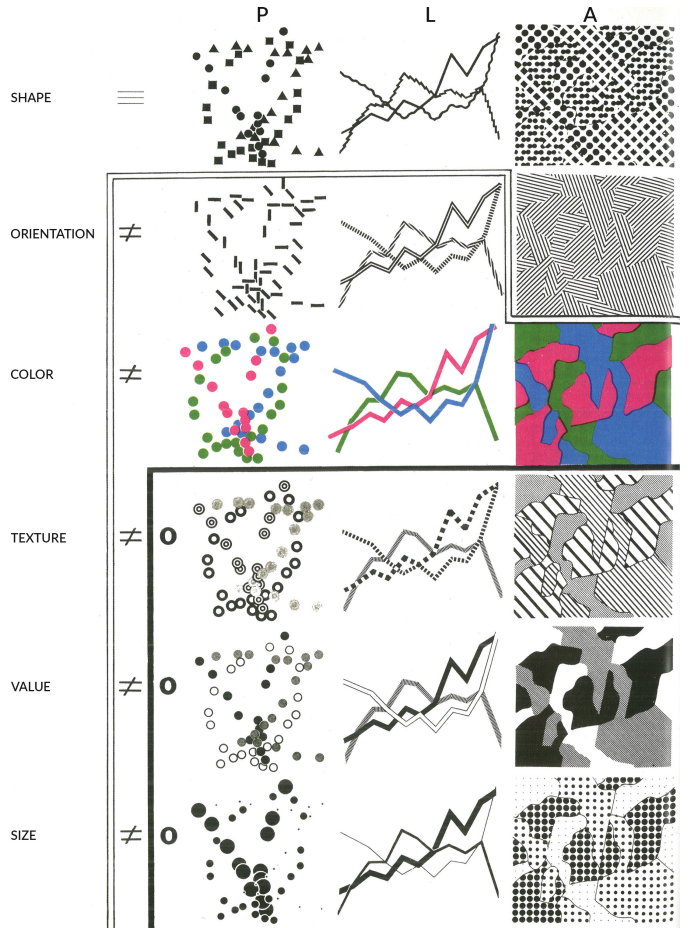

Fig. 21: Bertin's diagram for visual variables across three mark types [5,6]. From left to right, the columns represent point mark, line mark, and area mark; image © EHESS (text translated), used with permission.
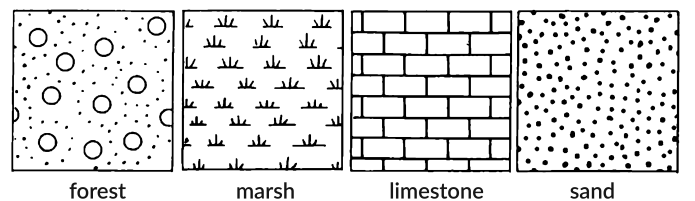

Fig. 22: Bertin's examples of "patterns have achieved the status of symbols" from [5,6]; image © EHESS (text translated), used with permission.

width is included in the main paper as Fig. 8). Fig. 26, in contrast, is a recent example that illustrates the use of retinal visual variables on
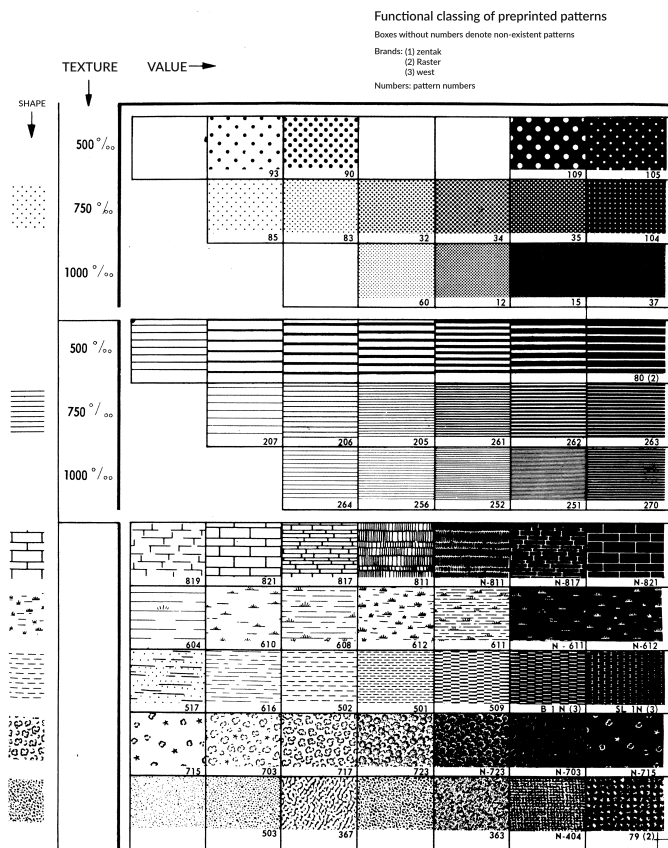
Fig. 23: Examples of pre-printed hatchings from Bertin's book [5, 6]; image © EHESS (text translated), used with permission.
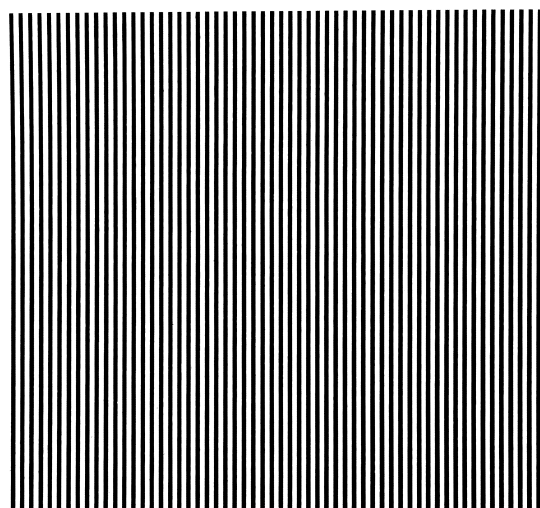

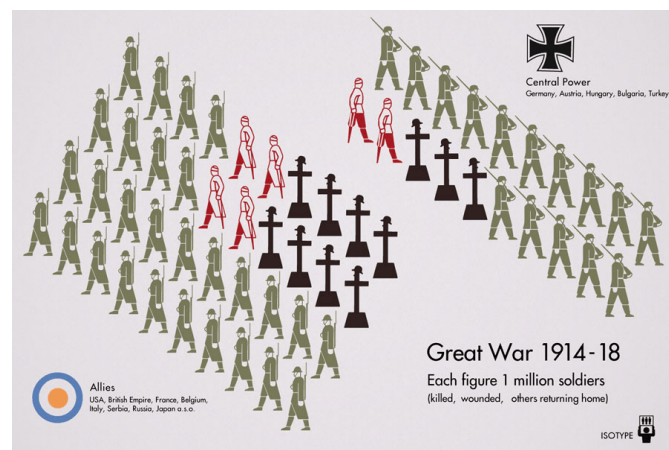Fig. 24: An example of Moiré effect from Bertin's book [5, 6]; image © EHESS, used with permission.

Fig. 25: Unit visualization (IsoType, [73]) that can be considered to be using internal variation. Image 'The Great War' by Otto Neurath; ⓢ the image is in the public domain.
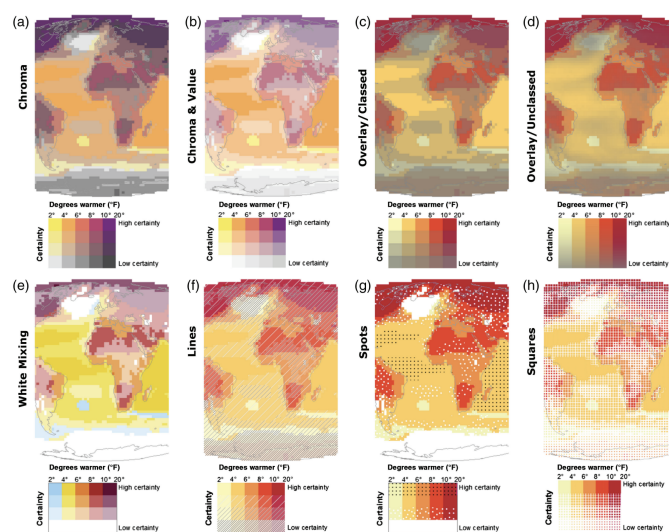

Fig. 26: Comparison of eight uncertainty representations by Retchless and Brewer. Most participants prefer (g). Image reproduced from [78, Fig. 11], used under the Creative Commons Attribution-NonCommercial 4.0 International (ⓒ ⓘ ⓢ CC BY-NC) license.

each of the pattern's primitives.

Fig. 27 and 28 demonstrate what we refer to as patterns based on 1D lattices (Fig. 27) and 2D lattices (Fig. 28). Fig. 29–31 then showcase various examples of non-lattice-based patterns.

Fig. 32–34 show historical examples of pattern legends to be used in visual representations, for example Fig. 32–33 based on *density* for encoding increasing value ranges. Fig. 35–37 and 39 use the *density* of simple dot patterns to encode values, with interesting aspects such as a (possible) transition from a *density* to a *position* encoding (Fig. 35) or the use of dot sizes that accurately represent the dot's referents in a map (Fig. 37). Fig. 38 then technically does not show a pattern but each (minute) mark gets only one (square) primitive, but due to the

composition of the marks we see patterns at a coarser (hour) data scale.

Fig. 40 shows a contemporary example, where a pattern is used as a secondary data layer to a color-based visualization. We found examples such as this one, e. g., to show uncertainty for the color-encoded data, or as in this case a meaningful subarea of the plot was marked.

Fig. 41 shows a pattern design for a diverging scale, but we would argue that it could be improved for a better support of preattentive value reading. We argue, e. g., that the neutral point should be white for both the positive and the negative part of the scale, and then the negative part could use an increasing line width or increasing line frequency of diagonal lines ▨ (but with a negative slope orientation to suggest negativity), and the positive part could either use crossing diagonal lines ▩ or, potentially even better, crossing horizontal and vertical lines ▦ to suggest the shape of a plus—again with an increasing line width or increasing line frequency for more positive values. Both "sides" of the scale should be designed in such a way that their perceived *value* at any given absolute level is the same, regardless of whether it is the positive or the negative side of the scale. Ultimately, further design and empirical research would be needed to identify suitable scales (and not only for diverging scales), to result in something along the lines of what ColorBrewer [10] provides us with for color scales in visualization.

Fig. 42 comes from a tutorial (see below) on how to fill plots with patterns in Matplotlib, and demonstrates cleverly designed patterns

that clearly indicate where two data marks overlap—in a similar way as transparent colors would do for overlapping marks. Fig. 43 shows again a historic example where two layers of patterns have been used to encode data. The first layer completely fills the mark, but the second layer relies on additional circular area marks being overlaid (or inset) on each main mark (which, of course, requires the main marks to be of sufficient size). Next, Fig. 44–46 show contemporary examples of combining color maps with patterns to either make categories more distinguishable (Fig. 44) or to visualize two layers of information at the same time (Fig. 45, 46). Fig. 47 demonstrates Jo et al.'s [49] declarative rendering model for multiclass density maps, some of whose approaches rely on patterns to make the different data classes distinguisheable from each other. The larger example on the right side of the figure demonstrates the ability of this approach to reproduce Bertin's technique of color-based categorical encoding for geographic data we showed in Fig. 7(a) and discussed in Sec. 5.2.2. Finally, Fig. 48 show contemporary examples of using stippling as a non-lattice arrangement.

F Discussion of current solutions for implementing patterns in visualizations

Current visualization tools and graphical drawing libraries offer limited support for patterns. Most tools provide only a few options to vary patterns, so users cannot fully use all pattern attributes. Achieving complex patterns often requires programming skills, making their implementation challenging, especially for designers with limited technical backgrounds. For example, one of the most popular visualization tools, Tableau, does not officially support pattern fills at this point in time. There have been requests for this feature in the forums since 10 years ago. For example, such as the posts "Pattern fill" (community.tableau.com/s/question/0D54T00000C5nGlSAJ/pattern-fill) and "Fill Patterns (Dots and Stripes)" (community.tableau.com/s/question/0D54T00000C5s3GSAR/fill-patterns-dots-and-stripes). This functionality, however, has not yet been integrated. Users can use workarounds. For example, A. McCann shared two tutorials in 2018: "Multiple pattern fill bar charts" (duelingdata.blogspot.com/2018/06/multiple-pattern-fill-bar-charts.html) and "Pattern fill bar chart in Tableau" (duelingdata.blogspot.com/2018/06/pattern-fill-bar-chart-in-tableau.html). These operations, however, are far more complicated than using other visual variables such as color, size, or shape. Some graphical drawing libraries offer pattern fills, but users can only select from default patterns or create repetitive tilings of shapes on a grid. Examples include Matplotlib [46] and Plotly [76] for Python, and ggpattern [34] for R. Below we list additional tools or source code resources we found during our explorations for creating patterns that offer more flexibility:

- SVG's <pattern> Element: If we want to create patterns in charts, one of the most flexible options is using the SVG <pattern> element, which offers a high degree of customization. It is described, however, as "arguably one of the more confusing fill types to use in SVG" [68]. It is based on, and thus confined to, the repetitive tiling of shapes in vertical and horizontal directions. Therefore, even the creation of a frequently used diagonal line pattern such as ▨ does not follow the native logic of SVG <pattern> and can confuse people. For example, a question on Stack Overflow highlights this issue: "Simple fill pattern in SVG: diagonal hatching" (stackoverflow.com/questions/13069446/simple-fill-pattern-in-svg-diagonal-hatching).

- Textures.js—SVG patterns for Data Visualization: riccardoscalco.it/textures and github.com/riccardoscalco/textures; this software package aims to make SVG's <pattern> easier to use, but it still requires extensive manual coding and does not fully break the constraint of repetitive tiling shapes. In addition, its options are limited. For example, although it supports line patterns, the lines cannot fully rotate 180 degrees and have only several predefined rotation options, such as 3/8.

- *Design Characterization for Black-and-White [sic!] Textures*: our own tool for the generation of simple iconic and abstract patterns from our previous work on patterns [44, 104]; github.com/tingying-he/design-characterization-for-black-and-white-textures-in-visualization; it offers greater freedom and ease of use compared to existing libraries and tools. Since we developed this tool earlier than we proposed our pattern design space in this paper, however, the tool does not cover all possible pattern variations. In addition, the tool is currently built around a web design interface, with which users can only design patterns in the given charts on the web, rather than integrate patterns into their own visualizations (Fig. 2(c)).

- Jo et al.'s [49] *Declarative Rendering Model for Multiclass Density Maps*: jaeminjo.com/Multiclass-Density-Maps and github.com/e-/Multiclass-Density-Maps; this tool focuses on maps and some of its techniques rely on patterns (Fig. 47)

- *Line Textures* by Richard Brath: source code for creating line patterns using D3 and dasharray; observablehq.com/@richardbrath/line-textures

- *Remaking Figures from Bertin's Semiology of Graphics* by Nicolas Kruchten: code in Python, using Plotly Express, to recreate some of Bertin's patterns; nicolas.kruchten.com/semiology_of_graphics

- *Generating different spatial patterns in R and their visualization using ggplot2* by Muhammad Mohsin Raza: datawim.com/post/generating-different-patterns-in-r

- *How To Fill Plots With Patterns In Matplotlib* by Elena Kosourova: towardsdatascience.com/how-to-fill-plots-with-patterns-in-matplotlib-58ad41ea8cf8 (Fig. 42)

G Footnotes

[1]We agree that *texture* may primarily be used for materials, but argue that the *pattern* concept extends beyond the repetitive use of shape variations (Sec. 3.3).

[2]It is used as the basis for theoretical constructs in architecture [1], mathematics [37], and beyond.

[3]In our daily life, e. g., *pattern* can refer to many physical items and abstract concepts that include repetition, such as a social/behavioral patterns, sound patterns, language structures, or chronological orders.

[4]These examples are all from cartography textbooks with lists of visual variables. By observing how they interpret "texture" in various ways we found inconsistencies in the understanding of the term "texture" as a visual variable.

[5]In contrast to the official translation of the book, which uses the term "texture," we intentionally changed the translation here to use "granularity" and also not "grain," for reasons that we explain further below.

[6]In this translator's note (Fig. 4), the translator also mentioned that "pattern" is the English equivalent for the French word "texture." In the book, however, the term "pattern" is not actually translated from "texture" in French. The translator translated the following French terms from Bertin's book into English as "pattern": "semis," "baguettage/ligne," and "trame." We discuss these terms in detail in Appx. B.

[7]The parameters a and b describe the spacing between primitives within the lattice. The simplest form of a 2D lattice is a square lattice where $a = b$ and $\theta = 90°$.

[8]For example, a point host symbol, finally, is not a theoretical point but is represented by a small area (often a circle) with a size, which in turn has an area that can be filled with a 2D or 1D lattice.

[9]Brath [8], in his blog, characterized patterns in the form of Fig. 12(d) by their length, gaps, rhythm, and randomness. Using the terminology of our pattern system, length and rhythm define the *appearance of primitives*; gaps define the *spatial relationships* between the primitives—the *lattice*; and randomness determines the *positional regularity* when placing the primitives onto the lattice.

[10]As we discuss later in Sec. 5.4, however, once primitives are placed on the lattice, we can translate the entire pattern when we apply it to a host symbol.

[11]Theoretically, the range of possible spacings—or unit cell sizes—extends from zero up to the size of the entire marking. Practically, it is crucial to use a sufficient number of primitives within the visible area to ensure that the pattern

and the extent of the symbol are discernible. If the primitives on the pattern are too sparse, both the symbol and the pattern may become difficult to perceive.

[12]Theoretically, the lattice can be rotated by any angle between 0° and 360°. Often, the center of rotation is the center of the symbol to which the pattern is applied, although it can be set to other points if needed.

[13]Wilkinson [98] describes the orientation of a mark as "rotation" and the orientation of primitives in a pattern (he called it "texture") as "orientation." He illustrates the concept of "orientation" exclusively with examples of line patterns (Fig. 12(b)) and does not address the orientation variable of the lattice, which we add here. Moreover, we argue that the distinction between "rotation" and "orientation" should not be based on whether they apply to marks or primitives, as Wilkinson suggests. Instead, the key difference lies in their relationship as method and result: an outcome of a given orientation of primitives is achieved through the method of rotation. We thus recommend to use both terms in their traditional meaning: rotation as the action and orientation as the status; both terms applied to either lattice, primitives, or both together.

[14]Patterns arranged in a 1D lattice can also be manipulated both by rotation of the lattice and by rotation of the primitives. In the case of line primitives, however, the rotation of the primitives w.r.t. the base line may also lead to a perceived change in the spacing between them.

[15]Morrison [71] first introduced this concept of *positional regularity* into visual variables. He referred to it as "arrangement" and added it as an additional visual variable to Bertin's list.

[16]Unit-based visualization with subgroups within each category can also be considered as using patterns with internal variation—where the variable *shape* is used to represent the categories. Fig. 25 in Appx. E shows an example of such a unit-based visualization. Here, if we view each diamond region as a pattern, it exhibits internal variation derived from a new facet, "type of soldiers." Within each pattern, the number of groups of primitives encodes number of categories and the ratio between groups represents the percentage of each category.

[17]Haroz et al.'s work [40] would suggest likely perceptual interactions between efforts to encode with distribution and ratio concurrently.

[18]The (ir)regularity can also be intended to not be used as a visual variable but simply as an aesthetic criterion, such as in non-photorealistic rendering (NPR). Here, the goal is the generation of (human-drawing-like) non-repetitive patterns (e. g., [4,47,65,81,99]), which have non-regularity both in their primitive shapes and in their placement.

[19]Such transformations can theoretically be achieved by modifying the lattice primitives directly as we described in Sec. 5.1. In practice, however, it is often more intuitive to apply an additional transformation to the entire pattern after its design, because this allows the designer to adjust the parameters with all decisions having been made already.

[20]Drawing on Gestalt principles, even as primitives merge and lose their individuality, our brain is often still capable of perceiving the shape of primitives to some extent through mental completion. Regardless of the specific primitive shape, however, as their size continues to increase the pattern ultimately becomes completely saturated and turns into a solid fill. The range of size variation that can be effectively used in visualization thus spans from just noticeable differences to a given threshold, at which the pattern is no longer identifiable. In some specific cases, such as perfectly aligned squares or dashed lines in a line pattern, primitives that touch each other immediately form a seamless tessellation and directly convert the pattern into a solid fill.

[21]For lines placed on 1D lattices, a change of orientation at the lattice level and at the primitive level lead to visually very similar results, with the latter also affecting the perceived density of the resulting pattern.

[22]Repeated patterns can cause a sense of instability—the Moiré effect [5,6] (e. g., Fig. 24 in Appx. D)—or even have neurological effects for some people [97]. There are also many Op artworks based on patterns (e. g., Movement in Squares by Bridget Riley) as well as non-photorealistic (NPR) recreations of them (e. g., [32,33,48]), which vary the size and spacing of pattern primitives to create a perception of regions (as we intend with patterns) or movement (which would be detrimental in our case).

[23]Indirect control via machine leaning may work, akin to Datasaurus [17,66].

[24]Visualization design may be constrained by convention, which is itself driven by technical capability, which changes. Our efforts here are to open up a broad theoretical design space that is not constrained by convention and is as such somewhat unoccupied and unknown. And thus somewhat exciting.

[25]It is important to note, however, that, while our systematic description of the pattern design space allows us to describe this recursive application of pattern, its utility may be limited as, e. g., levels of visual complexity will likely be high and interpretation may well be extremely challenging.
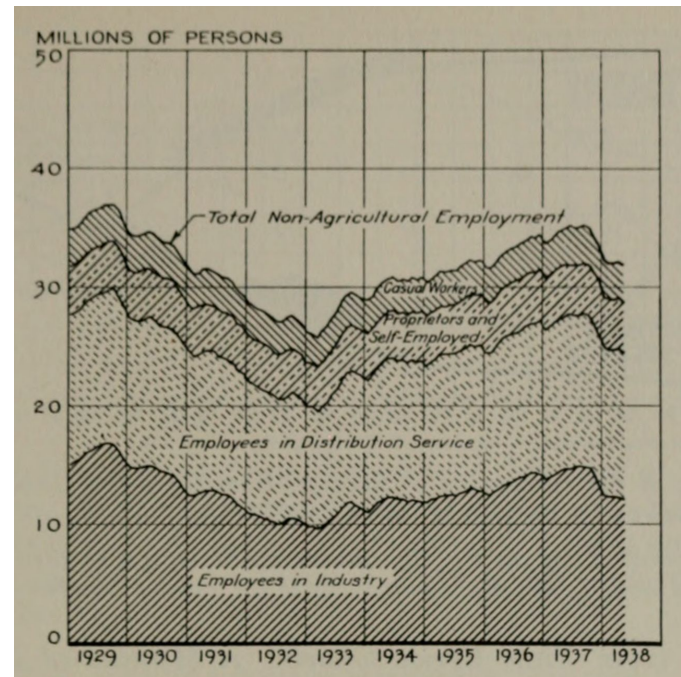

Fig. 27: Historic example of multiple 1D lattice-based patterns used for qualitatively marking regions; by Willard C. Brinton [12, page 297]; Ⓢ the image is in the public domain.
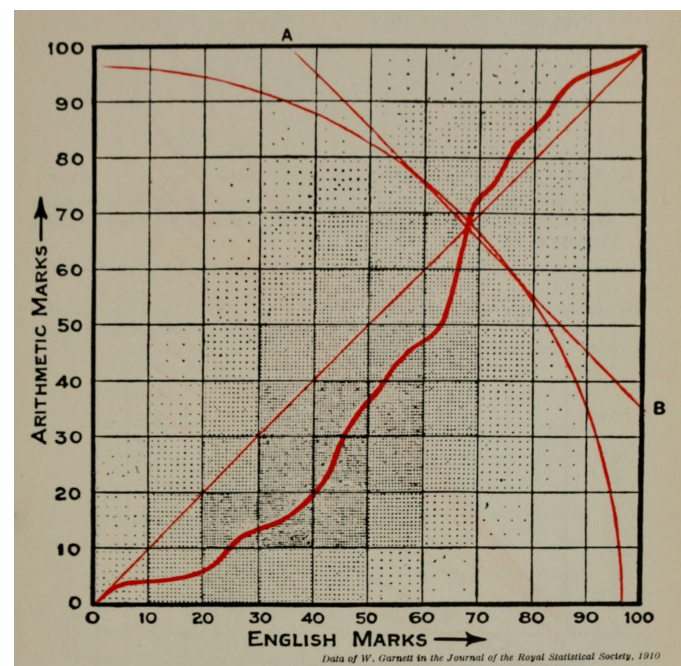

Fig. 28: Historic example of 2D lattice-based pattern for encoding data values (notice that the reproduction of the pattern is partially flawed, likely due to problems with the used plate); by Willard C. Brinton [12, page 327]; Ⓢ the image is in the public domain.

IMAGES/FIGURES LICENSE/COPYRIGHT
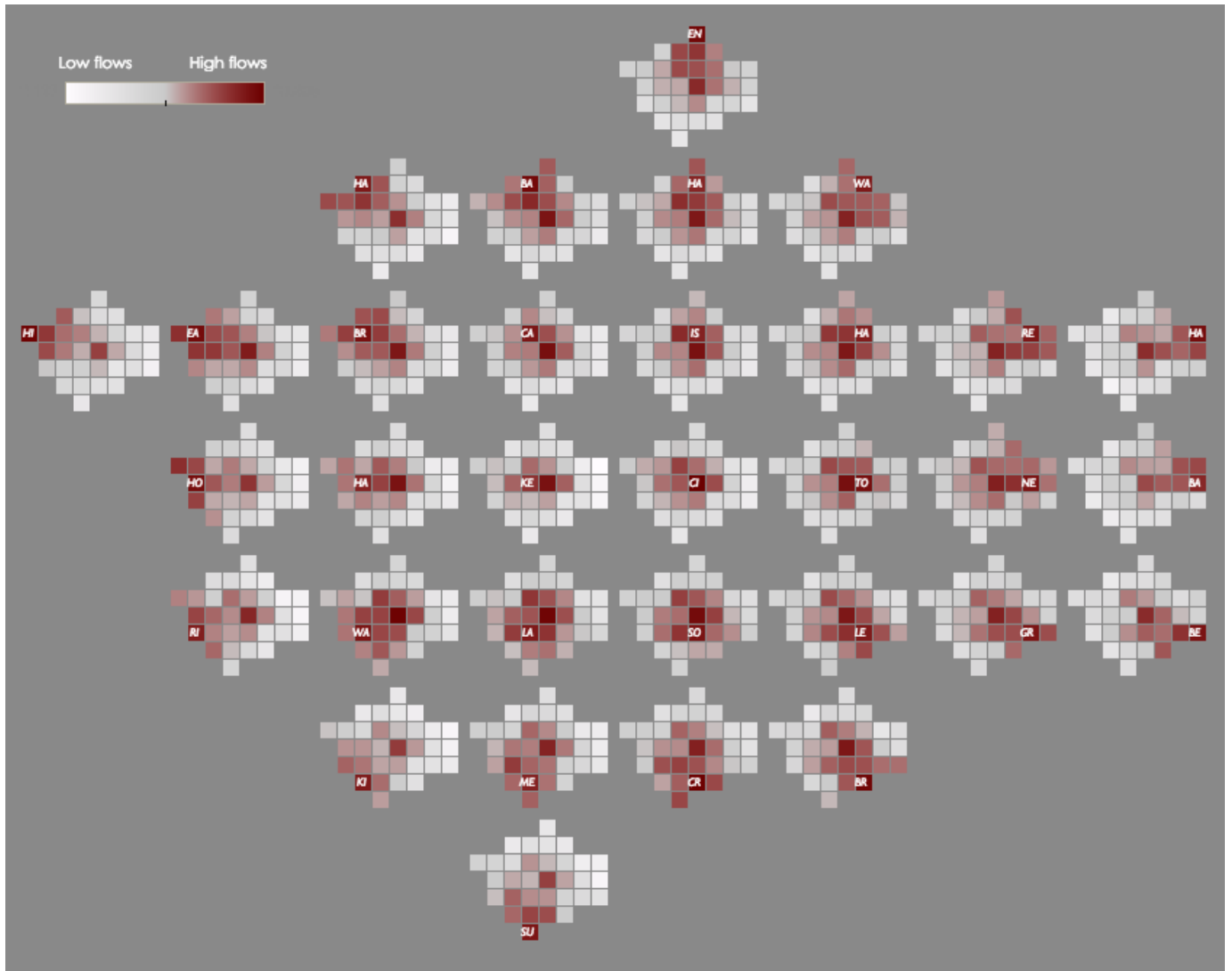
Fig. 29: Origin Destination Maps (OD Maps) [101] of London Commuting: an example of a nested pattern in which both the top-level pattern and the embedded patterns are data-driven. Image © by Robert Radburn, used with permission.
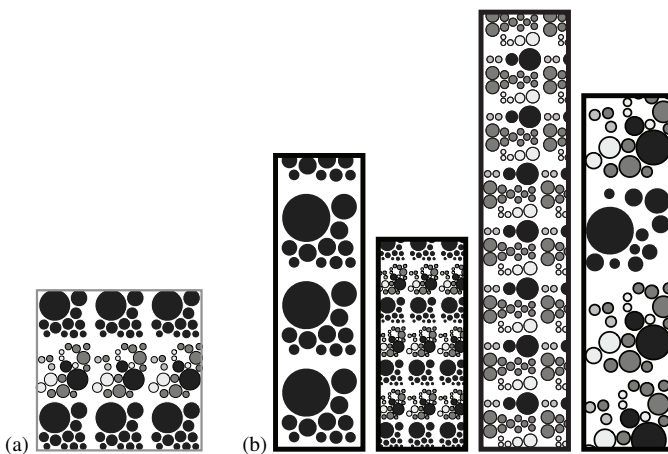

Fig. 30: (a) An example of a type of nested pattern: adding data-driven patterns on top of a lattice-based pattern.(b) A potential application of the nested pattern shown in (a) within a bar chart. This design shows a possibility opened by our pattern system; future work can explore which types of data are best suited for this encoding format.
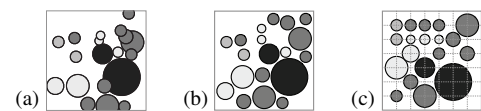
Fig. 31: Patterns with primitives based on geographically data-driven spatial arrangements: (a) accurate geographic data; (b) inaccurate geographic data, with dot overlap avoided; (c) gridded geographic data.
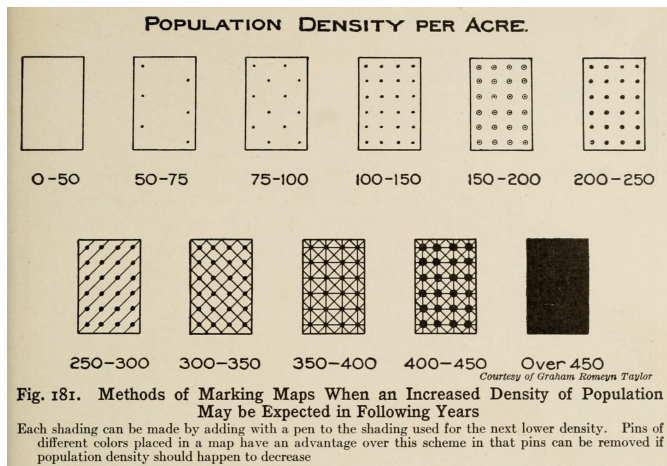
Fig. 32: Historic example pattern with *lattice size* (density) sequence by Willard C. Brinton [11, page 221]; ⓢ the image is in the public domain.
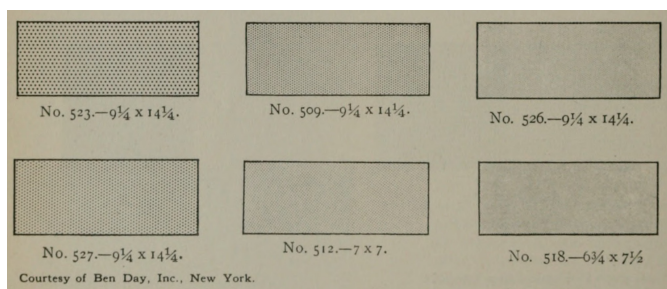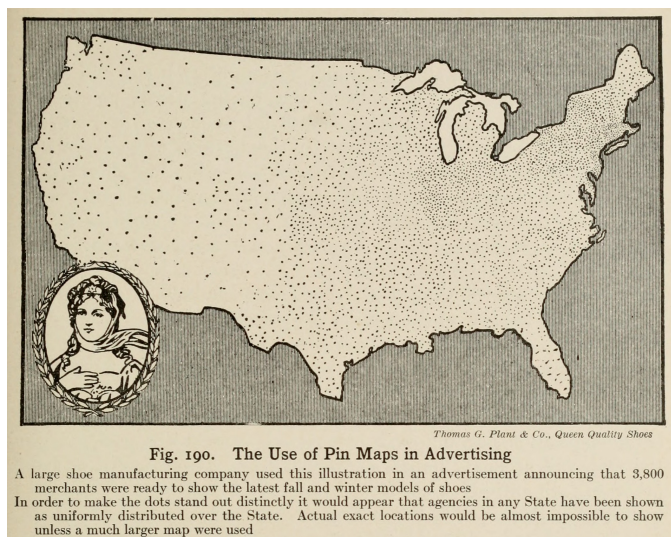

Fig. 35: Historic example of using *lattice size* (density) to illustrate the, in fact, density of merchants of a given product in the US; "pin map" by Willard C. Brinton [11, page 233]; notice that the use of the pattern can transition from a density pattern encoding in regions with high merchant density to a location-based encoding with discrete, actual locations in regions with lower merchant density (but we are not actually sure whether this was done in this illustration); ⓢ the image is in the public domain.

Fig. 33: Historic example pattern with *lattice size* (density) sequence by Willard C. Brinton [12, page 422]; ⓢ the image is in the public domain.

Fig. 34: Historic example pattern with *granularity* sequence by Willard C. Brinton [12, page 420]; ⓢ the image is in the public domain.
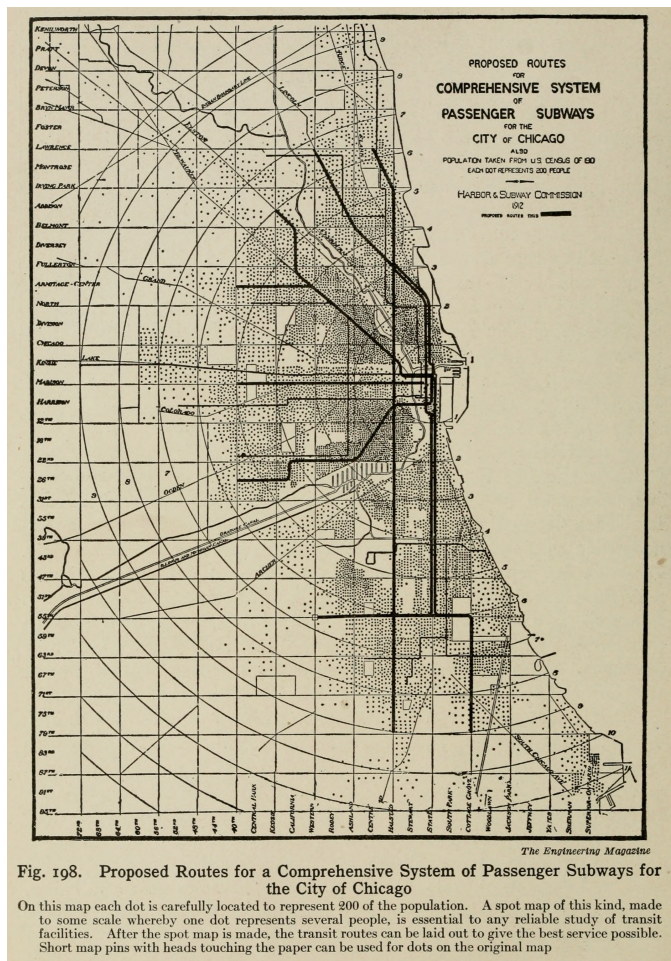

Fig. 36: Another historic example of using *lattice size* (density) to illustrate a density of people in a city (Chicago in the US) by Willard C. Brinton [11, page 246]; ⓢ the image is in the public domain.
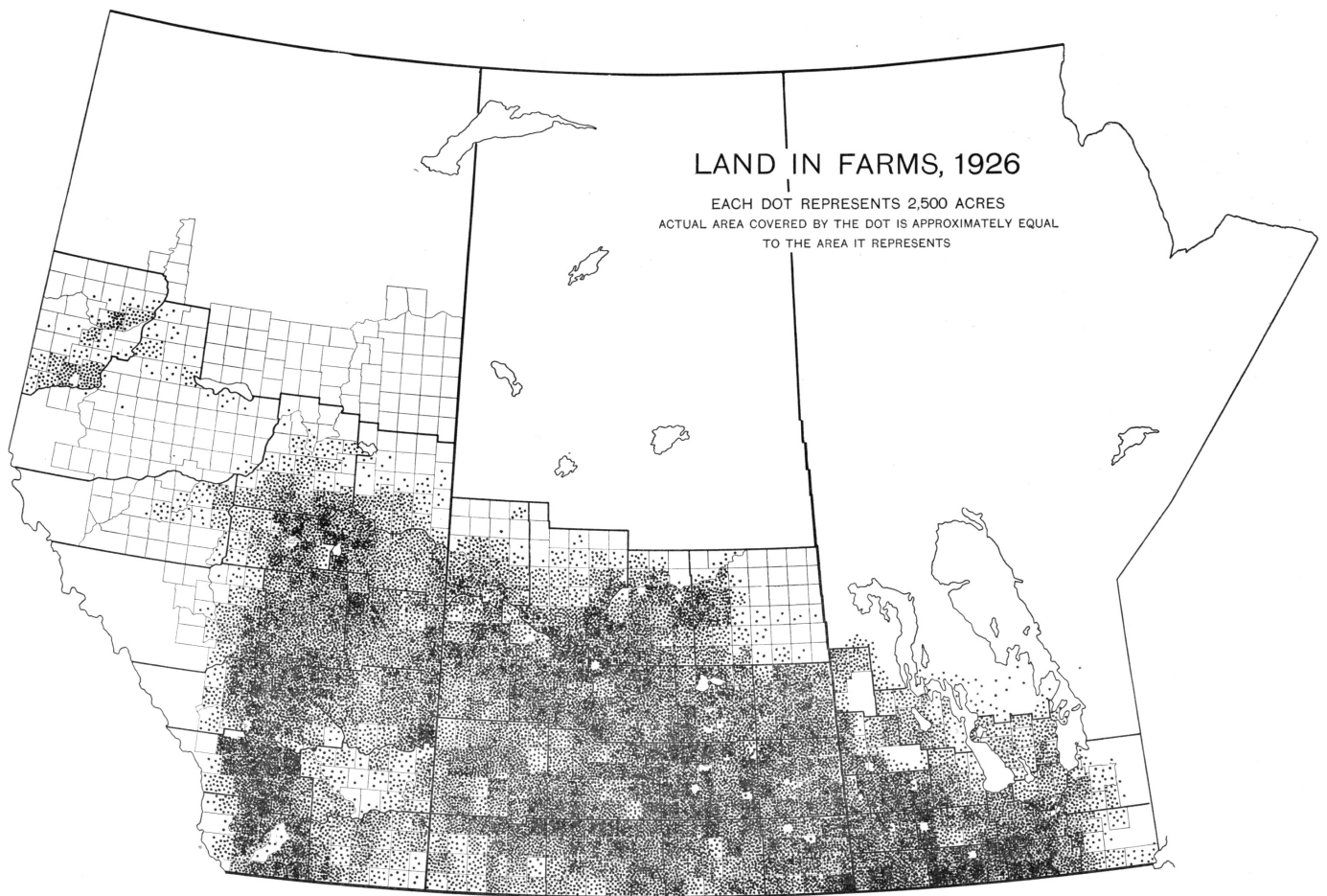
Fig. 37: Another historic example of dot patterns based on *lattice size* (density); notice the remark that the size of each dot represents approximately the area of the actual space to which it refers; image published in 1931 by the Canada Dominion Bureau of Statistics, Ottawa (Fig. 29 on page 28 of their "Agriculture, Climate and Population of the Prairie Provinces of Canada" statistical atlas), Ⓢ the image is in the public domain.
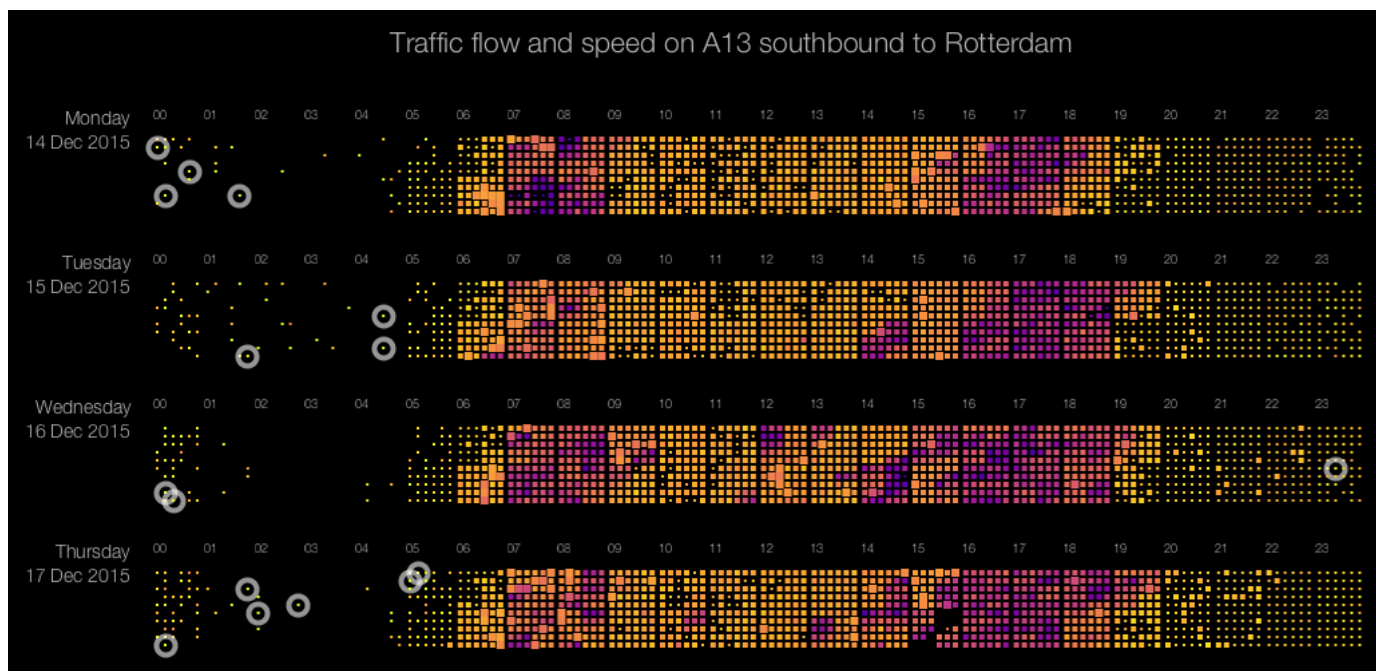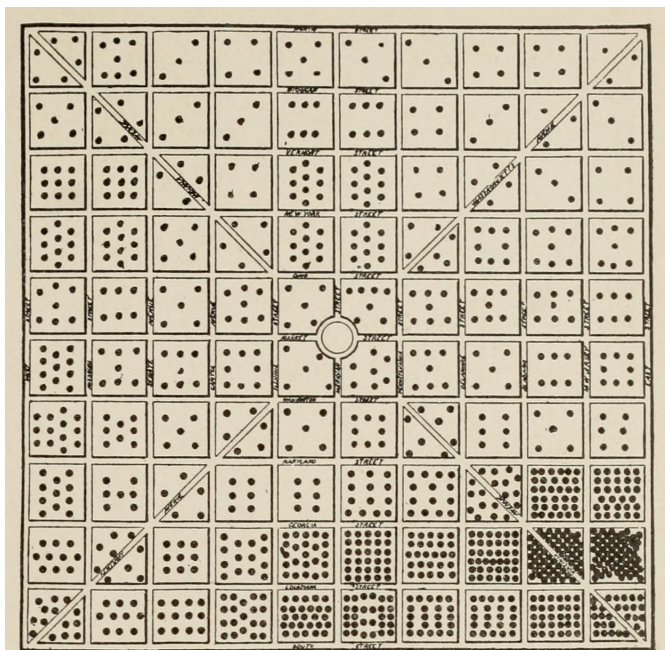

Fig. 38: Contemporary pixel-based visualization of traffic data, technically not a pattern (because the data marks are each individual minutes in the day, to which a size and a color hue is mapped based on that minute's actually captured data), this visualization still has characteristics of a pattern if viewed, e. g., per hour. Image © Erik Boertjes, used with permission.

Fig. 39: Historic example of grid-based (pollution) data for city blocks in Indianapolis in 1912 being visualized by *lattice size* (density); by Willard C. Brinton [11, page 245]; ⓢ the image is in the public domain.
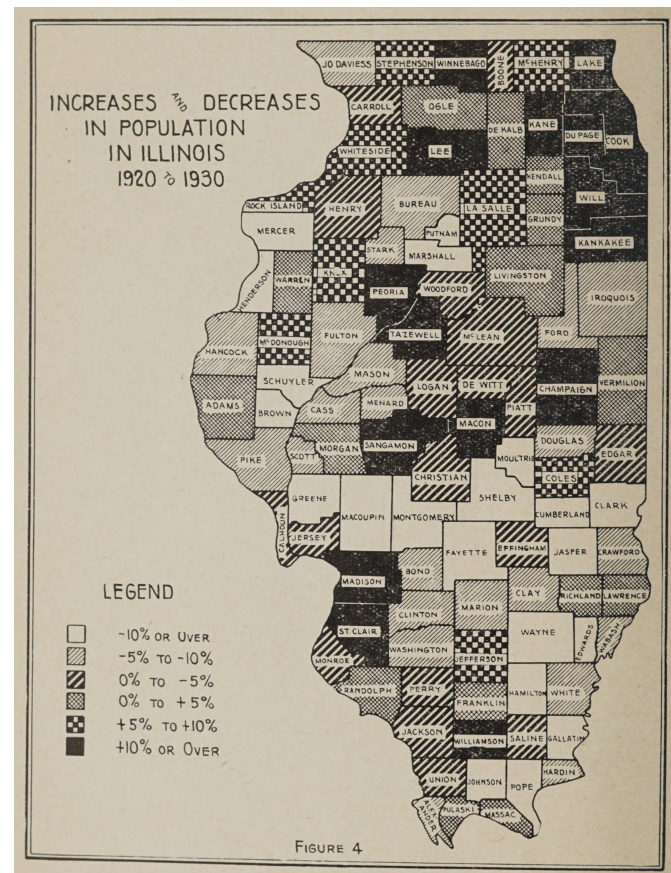

Fig. 41: Historic example of a pattern being used to encode a diverging scale: line patterns for negative values and crossing lines resp. dot patterns for positive values; image published in 1933 by the State of Illinois, Department of Public Works and Buildings, Division of Highways (Fig. 4 on page 24 of their "Economic Survey of Illinois"), ⓢ the image is in the public domain.
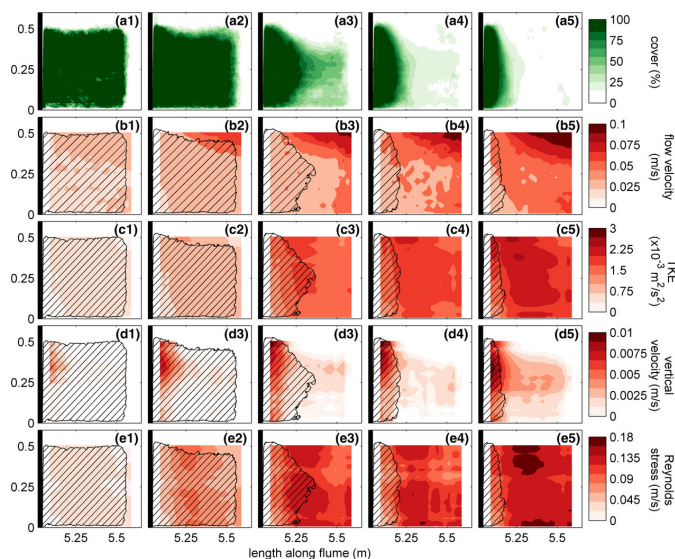

Fig. 40: Contemporary example of a pattern being overlaid on a color-based representation to encode a second quantity (in this example the pattern only has two levels—with pattern and without it). Image reproduced from [28, Fig. 5], used under the Creative Commons Attribution 4.0 International (ⓒⓘ CC BY-NC) license.
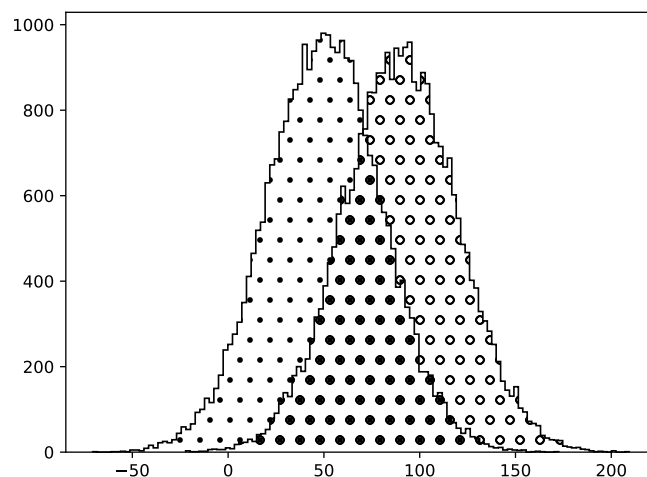

Fig. 42: Contemporary example of two merging area marks, with cleverly designed patterns that highlight the overlap between both marks; image created with Python and Matplotlib [46], following the tutorial by Elena Kosourova, see Appx. F.

Fig. 43: Historic example for a dual encoding with patterns, where one pattern fills the area mark, while another fills an inset circle (which, of course, requires large-enough area marks to begin with); image *"Mouvement de la mortalité de 1840-49 à 1857-66 comparé (parecusson de Rappel) à la mortalité absolue en chaque départemens les écussons rappelant la mortalité absolue des deux sexes de 0 à 1 an : Carte VII. Par le docteur Bertillon. 1872. – Carte VIII. Par le docteur Bertillon. 1872."*; ⓢ the image is in the public domain.

Fig. 44: Contemporary example of combining color scales with patterns to improve the discriminability of a categorical encoding. Image reproduced from [22, Fig. 6]; © IEEE, used with permission.
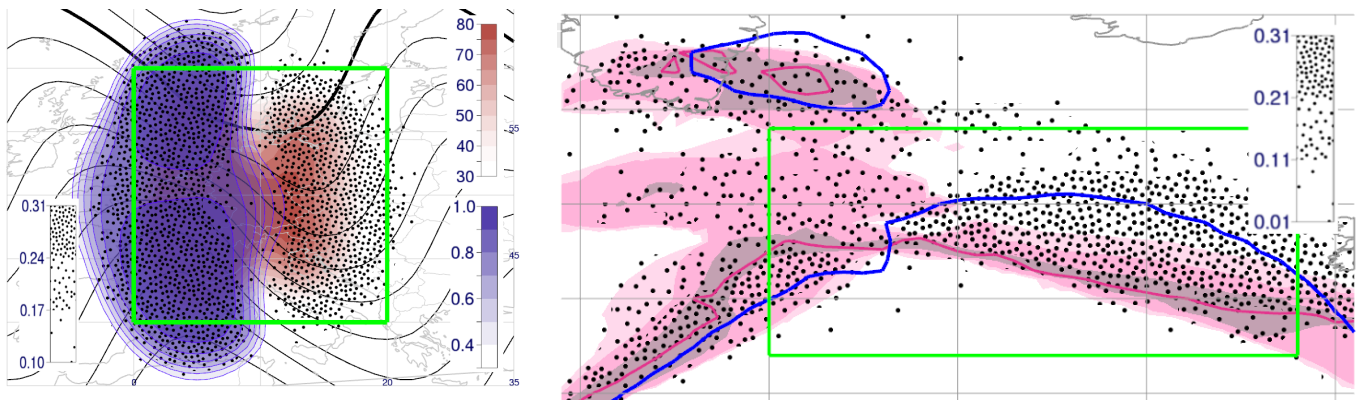
Fig. 45: Contemporary examples of combining color scales with patterns (here point densities of a stippling pattern) to encode an additional spatially changing attribute (here: robustness of the shown cluster). Images reproduced from [55, Fig. 11(a) and 16(a)]; © IEEE, used with permission.
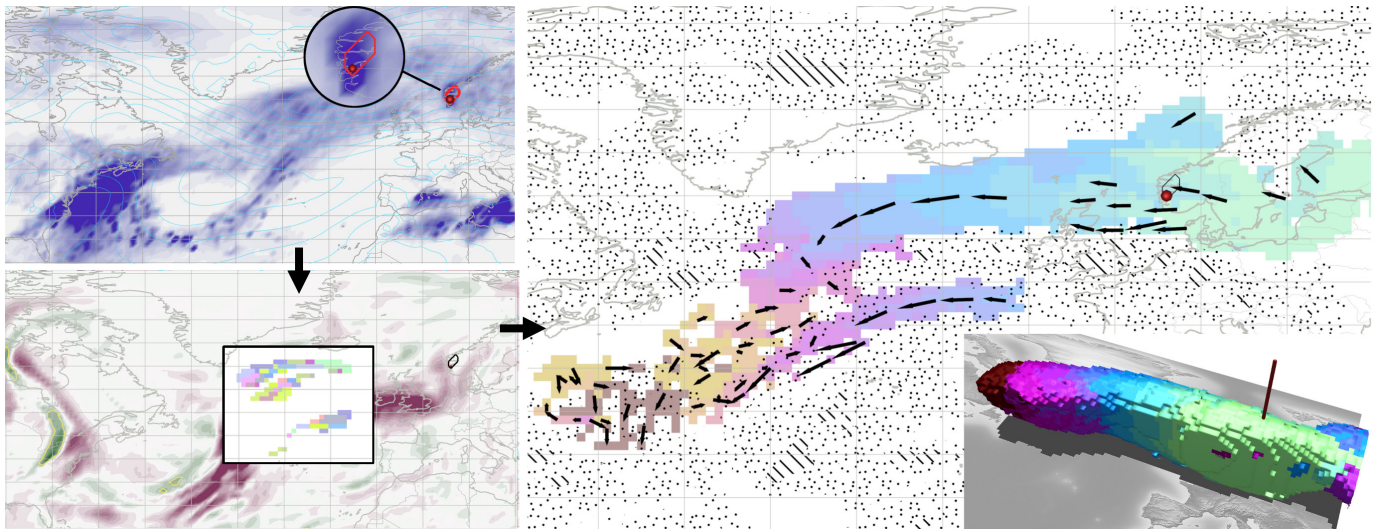

Fig. 46: Contemporary example of combining color scales with patterns to visualize two levels of information in weather forecast time series. Image reproduced from [54, Fig. 1]; © IEEE, used with permission.
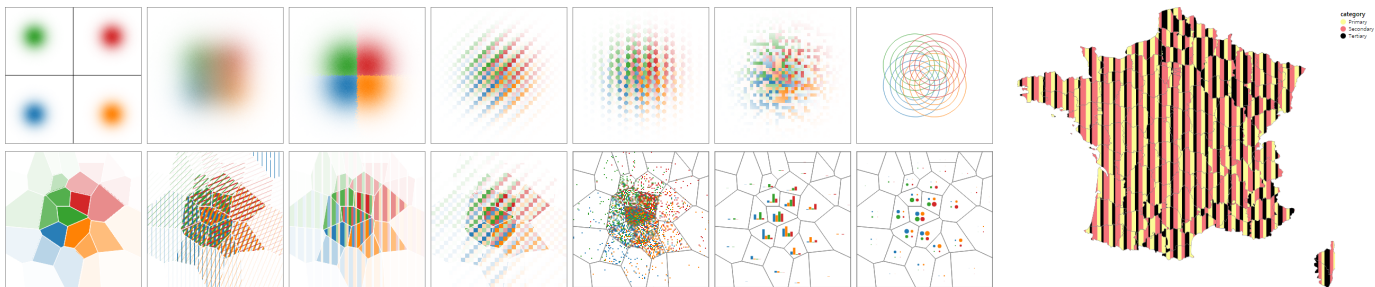

Fig. 47: Contemporary examples of Jo et al.'s [49] declarative rendering model for multi-class density maps, some of which rely on patterns (see Appx. F). The larger example on the right demonstrates the ability of the approach to reproduce Bertin's example we showed in Fig. 7(a). Images reproduced from [49, Fig. 1 and 6(c)]; © IEEE, used with permission.

(a)   (b)   (a) Scatter plot   (b) Stippled scatter plot
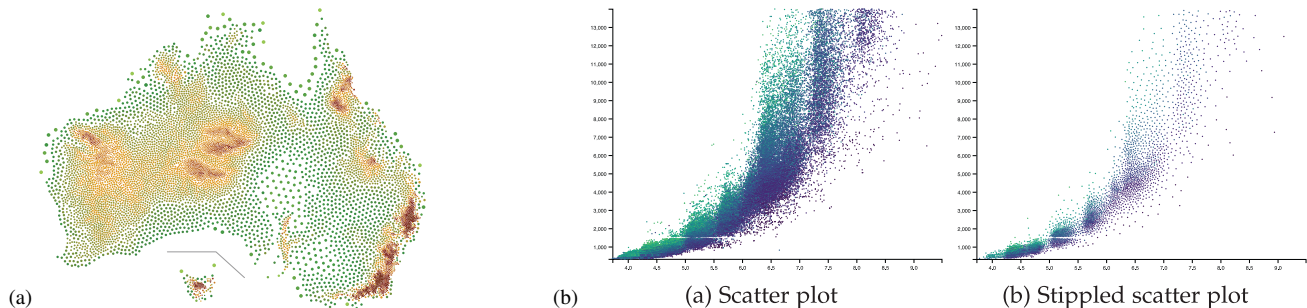
Fig. 48: Contemporary examples by Görtler et al. [38] of using stippling [30, 65] to show data, both for (a) spatial and (b) nonspatial datasets—similar to the historic examples in Fig. 35–37. Images reproduced from [38, Fig. 8 and 11]; © IEEE, used with permission.