Topological Equivariant Artist Model for Visualization Library Architecture

Hannah Aizenman, Thomas Caswell, and Michael Grossberg, *Member, IEEE,*

Abstract—The abstract goes here.

Index Terms—

1 Introduction

This paper uses methods from algebraic topology and category theory to develop a model of the transformation from data to graphical representation. This model provides a language to specify how data is structured and how this structure is carried through in the visualization and serves as the basis for a functional approach to implementing visualization library components. Topology allows us to describe the structure of the data and graphics in a generalizable, scalable, and trackable way. Category theory provides a framework for separating the transformations implemented by visualization libraries from the various stages of visualization and therefore can be used to describe the constraints imposed on the library components [**?**], [**?**]. Well constrained modular components are inherently functional [**?**], and a functional framework yields a library implementation that is likely to be shorter, clearer, and more suited to distributed, concurrent, and on demand tasks [**?**]. Using this functional approach, this paper contributes a practical framework for decoupling data processing from visualization generation in a way that allows for modular visualization components that are applicable to a variety of data sets in different formats. is it OK that this is something reviewer 4 wrote

2 Related Work

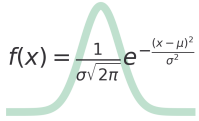
This work aims to develop a model for describing visualization transformations that can serve as guidance for how to architecture a general purpose visualization library. We define a general purpose visualization library as one that provides non domain specific building block components [**?**] for building visualizations, for example functions for converting data to color or encoding data as dots. In this section, we describe how visualization libraries attempt this goal and discuss work that formally describes what properties of data should be preserved in a visualization. We restrict the properties of data that should be preserved to

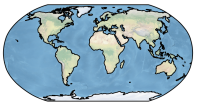
continuity how elements in a dataset are organized, e.g. discrete rows in a table, networked nodes, pixels in an image, points on a line

equivariance functions on data that have an equivalent effect on the graphical representation, e.g. rotating a matrix has a matching rotation of the image, translating the points on a line has a matching visual shift in the line plot

2.1 Continuity

NAME	TEMP (°F)	PRCP (in.)
NEW YORK LAGUARDIA AP	61.00	0.4685
BINGHAMTON	-12.00	0.0315
NEW YORK JFK INTL AP	49.00	0.7402
ISLIP LI MACARTHUR AP	11.00	0.0709
SYRACUSE HANCOCK INTL AP	13.00	0.0118





(a) (b) (c)

Fig. 1: Continuity is how elements in a data set are connected to each other, which is distinct from how the data is structured. The rows in (a) are discrete, therefore they have discrete continuity as illustrated by the discrete dots. The gaussian in (b) is a 1D continuous function, therefore the continuity of the elements of the gaussian can be represented as a line on an interval (0,1). In (c), every element of the globe is connected to its nearest neighbors, which yields a 2D continuous continuity as illustrated by the square.

Continuity describes elements in a data set are organized; this concept is termed topological properties by Wilkinson [**?**]. Wilkinson provides the examples of values that are isolated from each other, and therefore discrete, and values lying on a continuum or in a compact region. For example, in **??**, each station record in the table is independent of the others; therefore, the continuity of the table is discrete. The data provided by the gaussian are points sampled along the curve, therefore the continuity of the points on the line is 1D continuous. Every point on the globe is connected to its 6 nearest cardinal neighboring points (NW, N, NE, E, SE,

- *H. Aizenman and M. Grossberg are with the department of Computer Science, City College of New York.*
  E-mail: haizenman@ccny.cuny.edu, mgrossberg@ccny.cuny.edu
- *Thomas Caswell is with National Synchrotron Light Source II, Brookhaven National Lab*
  E-mail: tcaswell@bnl.gov

Manuscript received X XX, XXXX; revised X XX, XXXX.

S, SW, W). We propose that a robust model of continuity provides a way to develop library components that can work on the data in small pieces in a manner where the overall continuity of the data is preserved in the visual transformation.

Fig. 2: Continuity is implicit in choice of visualization rather than explicitly in choice of data container. The line plots in (b) are generated by a 2D table (a). Structurally this table can be identical to the 2D matrix (a) that generates the image in (c).

The preservation of continuity can be made explicit, as in the transformation of table to parallel coordinates in Ruchikachorn and Mueller [?], but is often expressed implicitly in the choice of visual algorithm (visualization type), as explored in taxonomies by Tory and Möller [?] and Chi [?].

For example, in **??** the same table can be interpreted as a set of 1D continuous curves when visualized as a collection of line plots or as a 2D surface when visualized as an image. This means that often there is no way to express data continuity independent of visualization type, meaning most visualization libraries will allow, for example, visualizing discrete data as a line plot or an image. General purpose visualization libraries-such as Matplotlib [?], Vtk [?], [?], and D3 [?]-carry distinct data models as part of the implementation of each visual algorithm. The lack of unified data model means that each plot in a linked [?], [?] visualization is treated as independent, as are the transforms converting each field in the data to a visual equivalent.

Domain specific libraries can often guarantee consistency because they have a single model of the data in their software design, as discussed in Heer and Agarwal [?]'s survey of visualization software design patterns. For example, the relational database is core to tools influenced by APT, such as Tableau [?], [?], [?] and the Grammar of Graphics [?] inspired ggplot [?], Vega [?] and Altair [?]. Images underpin scientific visualization tools such as Napari [?] and ImageJ [?] and the digital humanities oriented ImagePlot [?] macro, the need to visualize and manipulate graphs has spawned tools like Gephi [?], Graphviz [?], and Networkx [?].

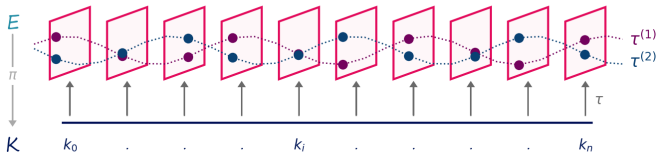


Fig. 3: A fiber bundle is mathematical construct that allows us to express the relationship between data and continuity. The total space Eis the topological space in which the data is embedded. The fiber space F is embedded in E and is the set of all possible values that any add big rectangle E

2.1.1 Fiber Bundles

The model described in this work provides a model for expressing data sets with different topological properties. We obtain this generality by using a mathematical structure called a fiber bundles as the basis of our abstraction, as proposed by Butler, Bryson, and Pendley [?], [?]. As described by them, a fiber bundle is a formal model of the mapping between data points and the underlying topological space they lie in. For example, nodes on a network and the graph of the network, an image and the underlying grid at which the image is sampled, or table columns and the table index.

Formally, a fiber bundle is a mathematical structure (E, K, π, F)

$$F \hookrightarrow E \xrightarrow{\pi} K \tag{1}$$

Definition 2.1. The base space K is a topological space, which means it is a set K with a collection of open sets [?] surrounding each element in the set. Open sets are a collection of subsets $\{U\}$ in a set K, including the empty set, the whole set K, and every union and intersection of its member subsets. For each point $k \in K$, there is a collection of subsets
$U\} \subset K$ such that $k \in U \subseteq K$ [?], [?]

intuition about base space

Definition 2.2. The fiber space F is a topological space such that for every $k \in K$, the fiber over that point is isomorphic to the preimage of that point $F_k \cong \pi^{-1}(\{k\})$. All fibers F_k are homeomorphic to each other [?], [?]

intuition about fiber

Definition 2.3. The total space E is the space reachable via the projection map π. The total space is always locally trivial, meaning $E = K \times F$ over any open neighborhood U_k.

The fiber bundle is trivial when $E = K \times F$ is true globally; a non-trivial bundle can be thought of as the twisted fiber product $E = Ktimes_T F$ [?], [?]. The twisting refers to the fiber F not aligning in the same direction over all $\subseteq K$.

$$\begin{array}{ccc} F & \hookrightarrow & E \\ & & \pi \downarrow \circlearrowright \tau \\ & & K \end{array} \tag{2}$$

How these spaces fit together is illustrated in **??**, wherein values lives in the fiber $F \subseteq E$. In this example, the fiber F is the cartesian product of two sets $F_0 \times F_1$ where each fiber F_i is ....

2.1.2 Sheaves

The set of all sections sections $\{\tau^i : U_j \to E\}|_{i\in I}$ over an arbitrary open set U_j is denoted $\Gamma(U_k, E)$. The set of all sections over the total space K is denoted $\Gamma(K, E)$.

The map from the open set to the set of sections is called a presheaf $\mathcal{O} : U_j \to \Gamma(U_j, E)$ [?], [?], [?]. The presheaf preserves inclusion maps between the open sets open sets ι and the inclusion map ι^* between the sets of sections.

$$\begin{array}{ccc} \Gamma(U_1, E) & \xleftarrow{\iota^*} & \Gamma(U_2, E) \\ \uparrow{\scriptstyle \mathcal{O}_E} & \uparrow{\scriptstyle \mathcal{O}_E} & \uparrow{\scriptstyle \mathcal{O}_E} \\ U_1 & \xrightarrow[\iota]{} & U_2 \end{array} \tag{3}$$

This means that a smaller space U_1 is included in a larger space U_2 and a function that is continuous over a larger space U_2 is continuous over a subspace U_1U_2. Sheaves are presheafs where sets of sections Γ are defined over unions of open sets over a topology $\bigcup_{j\in I} U)_j \in E$ [?], [?]. Sheaves are often used as an abstraction for keeping track of how data over topological spaces is glued together [?] because they model the data as sets of sections, continuity of the data in the base, and how subsets fit together through inclusion maps.

I've seen this convention before, but I also cite all these people above For more information on fiber bundles and sheaves, see Hatcher [?], Munkres [?], Spanier [?] and Ghrist [?], [?].

2.2 Equivariance

When introducing the retinal variables, Bertin informally specifies that continuity is preserved in the mark and defines equivariance constraints in terms of data and visual variables being selective, associative, ordered, or quantitative [?]. In the *A Presentation Tool*(APT) model, Mackinlay embeds the continuity constraint in the choice of visualization type and generalizes the equivariance constraint to preserving a binary operator from one domain to another. The algebraic model of visualization [?], proposed by Kindlmann and Scheidegger, restricts equivariance to invertible transformations.

2.2.1 Category Theory

In this work, we propose that equivariance constraints can be expressed using category theory. Vickers et. al [?] provide a brief introduction to category theory for visualization practitioners, but their work focuses on reasoning about visualization design, while this paper aims to provide guidance on designing visualization library components. Briefly we introduce concepts in category theory that we use to describe our model in ?? and ??.

the notation here may need to be X and Y

Definition 2.4. A category C consists of a collection of objects C with identity $id_c : C \to C$ and the set of morphisms between every two objects $f : C_1 \to C_2$, [?], [?].

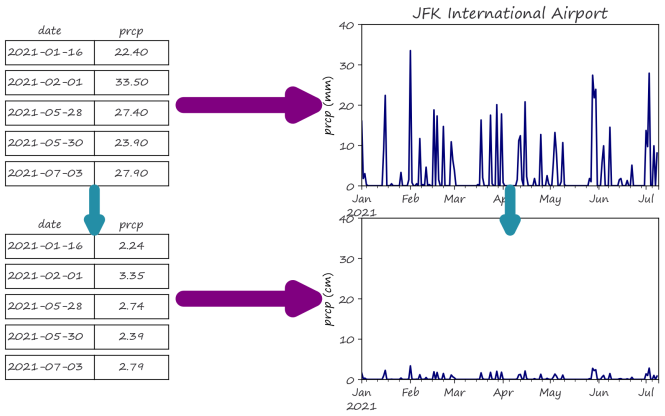
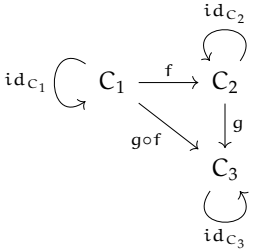


Fig. 4: Equivariance is that a transformation on the data has a corresponding transformation in the graphical representation. For example, in this figure the data is scaled by a factor 10. Equivalently the line plot is scaled by factor of 10, resulting in a shrunken line plot. Either a transformation on the data side can induce a transformation on the visual side, or a transformation on the visual side indicates that there is also a transformation on the data side.

The set of morphisms between objects is termed the hom set $\hom_{\mathcal{C}}(C_1, C_2)$. The morphisms on the category compose



and are constructed such that the following axioms hold [?]:

associativity if $f : C_1 \to C_2$, $g : C_2 \to C_3$ and $h : C_3 \to C_4$ then $h \circ (g \circ f) = (h \circ g) \circ f$

identity for every $f : C_1 \to C_2$ there exists identity morphisms $f \circ id_{C_1} = f = id_{C_2} \circ f$

Definition 2.5. An opposite category $\mathcal{C}^{op}$ is a category with all the same objects of category $\mathcal{C}$ and but the morphisms are reversed. For example $f : C_1 \to C_2$ in $\mathcal{C}$ is $f : C_2 \to C_1$ in $\mathcal{C}^{op}$.

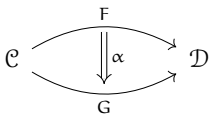
Definition 2.6. A functor is a morphism between categories $\mathfrak{F} : \mathcal{C} \to \mathcal{D}$. A functor has the properties of identity and composition [?]

A functor preserves the structure of the category, namely morphisms and the source and target objects of those morphisms, composition of morphisms, and identity morphisms [?]. Contravariant functors are functors where the morphisms in $\mathcal{C}$ go in the opposite direction from the morphisms in $\mathcal{D}$.

Definition 2.7. A presheaf $\mathcal{O}$ as introduced in **??**, is a functor $\mathcal{O} : \mathcal{C}^{op} \to$ **Set**. [?]

Definition 2.8. A natural transformation is a morphism of functors $\alpha : F \to G$ where F, G are functors from $\mathcal{C}$ to $\mathcal{D}$. [?], [?]

The natural transformation acts as a wrapper of sorts



(4)

such that $\alpha(F : \mathcal{C} \to \mathcal{D}) = G : \mathcal{C} \to \mathcal{D}$ commutes for any morphism $f : c \to c'$ [?]. Arbitrary morphisms on sheaves are natural transforms [?], [?].

For more information on category theory, see Barr and Wells [?], Fong and Spivak [?], Riehl [?] and Bradley et. al. [?].

3 Artist

In this section, we use category theory to formally express the implicit assumptions that visualization library components make in transforming data into graphical representations. We propose that the visualization transformation can be modeled as a functor, which we call the Artist A[1]. The artist A converts data, which we model as the sheaf $\mathcal{O}_E$ to graphics in the sheaf $\mathcal{O}_H$. We formulate this association as

$$A : \mathcal{O}_E \to \mathcal{O}_H \tag{5}$$

In this section we describe the structure of the data and graphic spaces, which we formulate as objects in categories and the morphisms between those objects. We then use these definitions to express the structure that an artist must preserve. Finally, we propose that artists can be composed by manufacturing new artists based on various types of input data.

3.1 Categorical Artist

Fiber bundles, as introduced in **??**, serve as the model of data and graphic spaces in out model. In this section, we introduce categorical formulations of the topological spaces that make up a fiber bundle.

We introduce the category $\mathcal{K}$ to encapsulate the subsets of the continuity and the inclusion maps that glue them together.

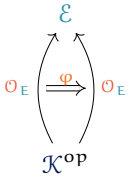
Definition 3.1. The category of open sets $\mathcal{K}$ consists of

- *objects* open sets $U_j \in \{\text{openset}\}$ in the topological space $(K, \mathscr{T})$), including the empty set $\varnothing$ and the maximum set K.
- *morphisms* $\iota : U_j \hookrightarrow U_k$ for all $U_j, U_k \subset E$

Definition 3.2. The category E is a subcategory of **Set** and consists of

- *objects* sets of sections $\Gamma(U_j, E)$ for all $U_j \subset E$
- *morphisms* $\iota^* : \Gamma(U_k, E) \hookrightarrow \Gamma(U_j, E)$ for all $U_j, U_k \subset E$

We model data as the sheaf $\mathcal{O}_E : \mathcal{K}^{op} \to \mathcal{E}$



(6)

[1]We call this transformation the artist because the `Artist` object in Matplotlib [?]

where the map φ is any arbitrary function between sheafs. The sheaf $\mathcal{O}_E$ over a limit of open sets that contain a point $k \in K$ is approximately the same as a sheaf over a point k and is called the stalk $\mathcal{O}_{\mathcal{E}}|_k \coloneqq \lim_{U \ni k} \Gamma(U, E)$ [?].

Definition 3.3. The fiber category $\mathcal{F}$ is a product category [?], [?] of zero or more arbitrary categories. In the two category case, the category $\mathcal{F}$ consists of

- *objects* ordered pairs (x, y), x is an object of $\mathcal{X}$ and y is an object of $\mathcal{Y}$
- *morphisms* ordered pairs $(f : x \to x', g : y \to y')$

The composition of morphisms is defined component wise by composition in $\mathcal{X}$ and $\mathcal{Y}$.ss

The fiber category $\mathcal{F}$ can encode a large variety of collections of data fields because it is a product category of arbitrary categories. For example, a lists of strings is an instance of an object in **Set**, networks are an instance of **Graph**, and images are vector spaces which are a specific type of topological space **Top**. The morphisms $\mathrm{Hom}_{\mathcal{F}}$ are functions from the data to itself, for example the binary operations, group actions, and measurement scales discussed in **??**. These functions could also be monoid actions [?], which provide a way of applying partial order relations to data [?], such as to build multi-ranked indicators [?].

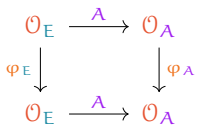
note sure this note applies anymore: We note that a one object category is a monoid [?], [?], which is an algebraic structure that lends itself to computer library design because they are an abstraction of function composition [?].

The fiber space over a point is embedded in the stalk over that point $F_k \hookrightarrow \mathscr{F}_k$. The sheaf maps φ restricted to the stalk are members of $\hom_{\mathcal{F}}(F, F)$. The maps $\varphi \in \mathrm{Hom}_{\mathcal{F}}(F, F)$ include the types of data transformations described in **??**. When a fiber bundle is trivial, we can dispense with localization and directly consider the fiber maps $\mathrm{Hom}(F_k, F_k)$ over a point k. In **??**, we assume that the fiber bundles are trivial.

The domain of the artist function is the space of all possible graphics. As with the domain $\mathcal{O}_E$, the graphic space is modeled with fiber bundle equivalent categories:

fiber bundle	data	graphic
base space	K	S
total space	E	H
fiber space	F	D
π^{-1}	$\mathcal{O}_{\mathcal{E}}$	$\mathcal{O}_{\mathcal{H}}$

The category $\mathcal{H}$ has as objects the sets of functions that generate every arbitrary graphic over the open sets in $\mathcal{H}$. While $\mathcal{H}$ is the domain of the artist A, the range of A is the subset of graphics $\mathcal{O}_A \subset \mathcal{O}_{\mathcal{E}}$ such that structure imposed by φ_E commutes



(7)

The equivariance condition expressed in **??** is that a graphic generated by transforming and then visualizing data $A(\varphi_E(\mathcal{O}_{\lceil \sqcup \wr \sqcup \dashv \updownarrow \rceil}))$ can equivalently be generated by visualizing the data and then transforming the graphic $\varphi_A(A(\mathcal{O}_{\mathcal{E}}))$. This is a formal statement of the equivariance illustrated in **??**

In **??**, we show a method of constructing the artist such that the constraints of **??** and **??** can be satisfied, thereby ensuring equivariance and the preservation of continuity. We then use this formulation to guide the development of visualization library components in **??**.

3.2 Composition of Artists

Visualizations generally consist of more than one visual element, for example line plots, a legend, axis ticks, spines, and labels. We propose that we can express expected consistency between these elements as a preservation of morphisms between objects of different data categories $\mathcal{E}$.

3.2.1 *Shared Index*

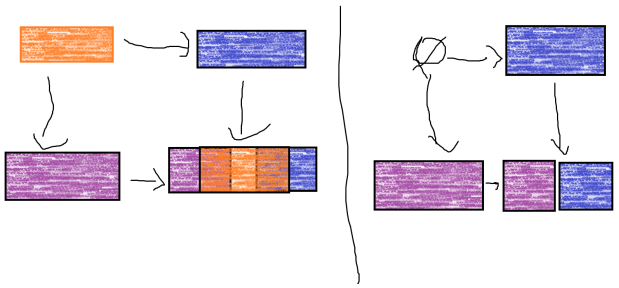


Fig. 5: In **??**, the input object $\mathcal{O}_E$ encodes a continuous function over spaces E_a, E_b and overlapping space E_c. In this figure, the visual transformation is to a point on the line on screen. An artist that preserves the continuity of the function must generate graphics such that functions evaluated $k \in K_a \cap K_b$

An example of overlapping indexing is a parallelized version of a sliding window algorithm where window overlaps must resolve to the same value at the same position **[?]**;

$$\begin{array}{ccc} K_c & \xrightarrow{\iota} & K_b \\ \downarrow{\scriptstyle \iota} & & \downarrow{\scriptstyle i_{K_b}} \\ K_a & \xrightarrow{i_{K_a}} & K_a \bigsqcup_{K_c} K_b \end{array} \tag{8}$$

where the projection functions i_{K_a}, i_{K_b} behave such that $i_{K_a}(k)|_{k\in K_b} = [k]$, $i_{K_b}(k)|_{k\in K_b} = [k]$, meaning the projection functions yield equivalent points in the disjoint union $K_a \bigsqcup_{K_c} K_b$. Data that is completely disjoint in the index, as shown in **??**, is a special case where $K_c = \varnothing$ and therefore there is no constraint on the artist with regards to how this data is rendered.

3.2.2 *Shared Fields*

$$\begin{array}{ccc} F_a \times F_b & \xrightarrow{\text{proj}_a} & F_a \\ \downarrow{\scriptstyle \text{proj}_b} & & \downarrow{\scriptstyle \text{proj}_c} \\ F_b & \xrightarrow{\text{proj}_b} & F_c \end{array} \tag{9}$$

?? and **??** can be composed to specify how different aspects of the structure of a multivariate nested dimensional dataset, such as a spatio-temporal weather dataset, should be preserved in a visualization. this is probably the power of this model and should maybe get a figure?

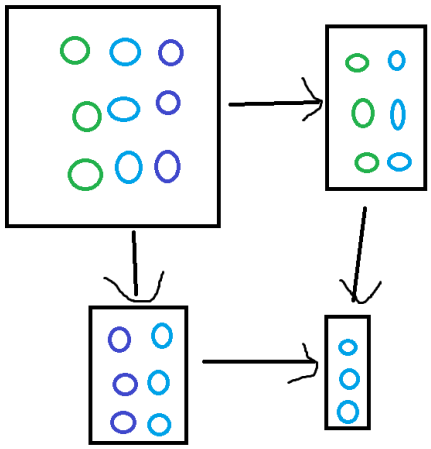


Fig. 6: The fiber spaces F_a, F_b have a shared fiber space F_c. If they are input into the same artist A_1, then an equivariant transformation is one where F_c is transformed to a visual element in a consistent matter. In this figure, F_c is mapped to the x-axis for both F_a and F_b.

4 Construction and Formal Properties of Artists

Following from **??**, the artist A is a natural transformation

$$A : \mathcal{O}_{\mathcal{E}} \tag{10}$$

that we construct to preserve continuity and equivariance.

Definition 4.1. We define the natural transformation Artist A as the tuple $(\xi, \nu, Q, \mathcal{E}, \mathcal{V}, \mathcal{H})$ where

1) $\nu : E \to V$ is a bundle map from data values to the visual variables they are mapped to
2) $Q : \mathcal{O}_{\mathcal{V}\to\mathcal{O}_{\mathcal{A}}}$ is a sheaf map that builds a graphic generating function parameterized by the visual variables
3) $\xi : S \to K$ is a surjective map from the graphic topological base to the data topological base

and $\mathcal{E}$, $\mathcal{V}$, and $\mathcal{H}$ are the categorical representations of fiber bundles that model the data, visual variable, and graphic space of the data to visual transformation. We construct the artist to have two stages

$$\begin{array}{ccc} E & \xrightarrow{\xi} & V \\ & {\scriptstyle \pi}\searrow & \downarrow{\scriptstyle \pi} \\ & & K \end{array} \qquad \begin{array}{ccc} \mathcal{V} & & \mathcal{H} \\ \uparrow{\scriptstyle \mathcal{O}_V} & \xrightarrow{Q} & \uparrow{\scriptstyle \mathcal{O}_A} \\ \mathcal{K} & \xrightarrow{\xi^*} & \mathcal{S} \end{array} \tag{11}$$

which are the data to visual variable map ν at the bundle level and the visual variable to graphic map Q at the sheaf level. Usage of the bundle directly in an equation or diagram - , V, H - denotes that the function can be evaluated pointwise over $k \in K$. Use of the categorical form – $\mathcal{E}, \mathcal{V}, \mathcal{H}$ – denotes that the function is evaluated over an openset

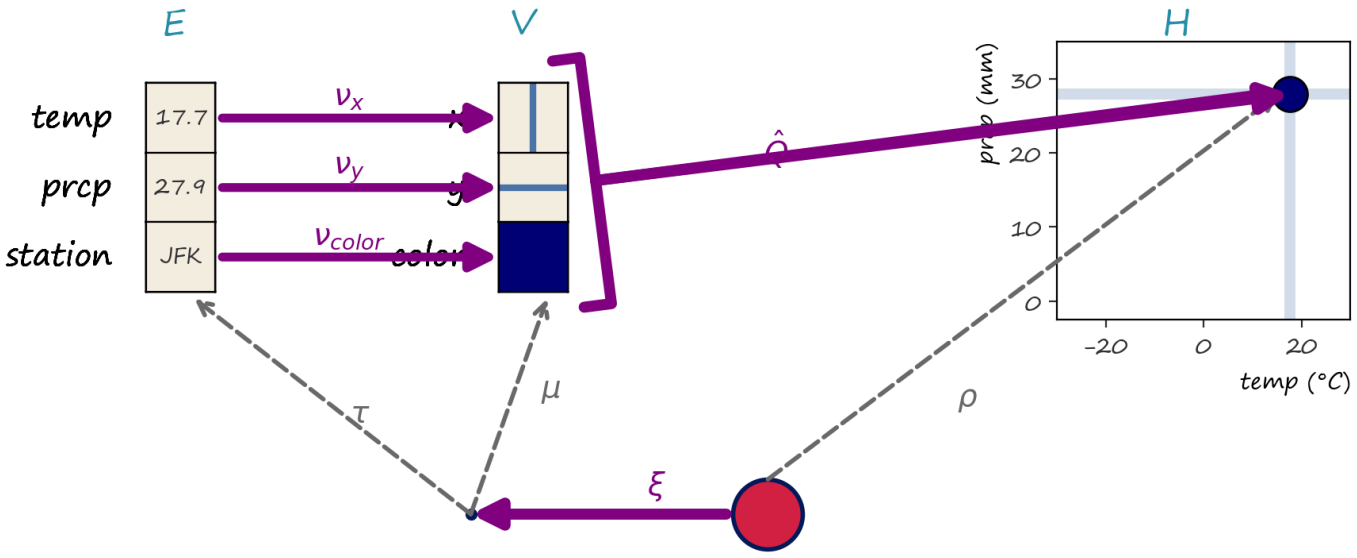


Fig. 7

object, meaning that the function needs knowledge of both a value over a point and some information about neighboring values.

4.1 Data Domain

We model data as sections of a fiber bundle (E, π, K, F). We encode the continuity of the data as the base space K. We model the data as the section τ because, as described in **??**, the section is the map from the indexing space K to the space of possible data values F. We adopt Spivak's formulation of the fiber as a (column name, data domain) simple schema [**?**], [**?**]. Spivak formally maps column names and field types to the set of values associated with the field type, for example $\mathbb{R}$ for a `float` column named *temperature*. This allows for field based selection of values, while inclusion allows for continuity (index) based selections.

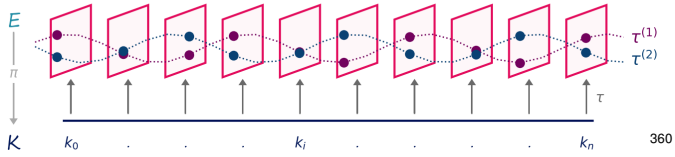


Fig. 8: replace with more concrete

One example of encoding data as a section of a fiber bundle is illustrated in **??**. In this example, the data is a not totally decided yet table of weather station data. Here we are interested in the time series of temperature values in the data, so we encode the continuity as the 1D interval K. In this multivariate data set, the fields we want to visualize are `time`, `temperature`, and `station`. The fiber space F is the cartesian cross product of the fibers of each field

$$F = F_{time} \times F_{temperature} \times F_{station}$$

where each field fiber is the set of values that are valid for the field:

$$\begin{aligned} F_{time} &= \mathbb{R} \\ F_{temperature} &= \mathbb{R} \\ F_{station} &= \{s_0, s_1, \cdots, s_i, \cdots, s_n\} \end{aligned}$$

The section τ is the abstraction of the data being visualized. The section at a point $k \in K$ in the base space returns a value from each field in the fiber.

$$\tau(k) = ((time, t_k); (precipitation, p_k); (station\ name, n_k))$$

4.2 Graphic Codomain

The object of the graphic category $\mathcal{H}$ is the fiber bundle H. The bundle H has the same structure as the data bundle E

$$\begin{array}{ccc} D & \hookrightarrow & H \\ & & \pi \downarrow \uparrow \rho \\ & & S \end{array} \tag{12}$$

with a fiber space D embedded in the total space H and a section map $\rho : S \to H$. The attributes of the graphic bundle (H, π, D, S) encode attributes of the graphic and display space

base space S continuity of display space (e.g. screen, 3D print)

fiber space D attributes of the display space (e.g a pixel = (x,y,r,g,b,a))

section ρ graphic generating function

.

We represent the graphic output as a fiber bundle because it is an abstraction that is generalizable to various output mediums (screens, 3D prints) and types of graphics.

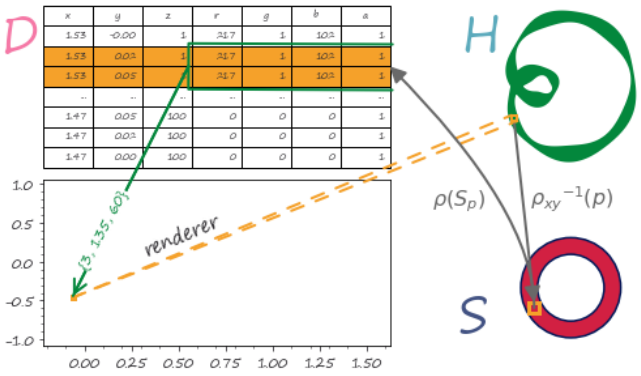


Fig. 9

As illustrated in **??**, in this work, H assumes the the display is an idealized 2D screen. include some more about the figure The graphic section ρ is an abstraction of rendering. For example, ρ can be a specification such as PDF [**?**], SVG [**?**] or an OpenGL scene graph [**?**], or a rendering engine such as Cairo [**?**] or AGG [**?**].

4.3 Visualization Library Components

4.3.1 *Visual Bundle* V

$$\begin{array}{ccc} P & \hookrightarrow & V \\ & {\scriptstyle\pi}\downarrow\uparrow{\scriptstyle\mu} & \\ & K & \end{array} \tag{13}$$

4.3.2 *Data to Visual Encodings:* ν

maybe figure out how to put these side by side nicely

$$\begin{array}{ccc} E & \xrightarrow{\nu} & V \\ {\scriptstyle\pi}\downarrow & \nearrow_{\pi} & \\ K & & \end{array} \qquad \begin{array}{ccc} F & \xrightarrow{\nu} & P \\ {\scriptstyle\tau}\uparrow & \nearrow_{\mu} & \\ K & & \end{array} \tag{14}$$

Since the functor ξ is bundle wise, it can be evaluated at the bundle at a point, which is the fiber space; therefore ξ is a functors from the product category $\mathcal{F}$ to the product categrory $\mathcal{P}$. This constraint is expressed in our construction of ν

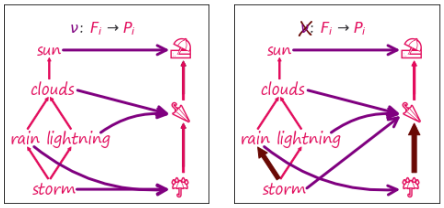
which means equivariance of

$$\begin{array}{ccc} \tau & \xrightarrow{\nu} & \mu \\ {\scriptstyle\varphi}\downarrow & & \downarrow{\scriptstyle\nu_\varphi} \\ \tau' & \xrightarrow{\nu} & \mu' \end{array} \tag{15}$$

Since the functor ν acts on product categories, it can be decomposed into nu_i component functors that act on corresponding sections τ_i of the fiber F_i.

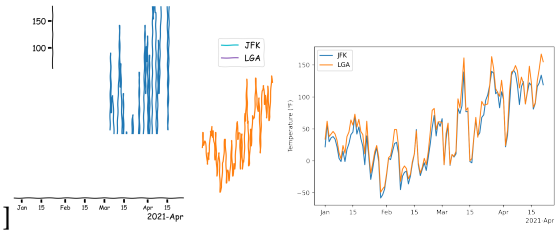
$$\{\nu_0, \ldots, \nu_n\} : \{\tau_0, \ldots, \tau_n\} \mapsto \{\mu_0, \ldots, \mu_n\} \tag{16}$$

One example of this is



(a) insert title (b) insert title

Fig. 10: is a weird nu, colors are a but much



(a) No shared fields (b) Shared fields

Fig. 11: In **??**, the input object $\mathcal{O}_E$ encodes a continuous function over spaces E_a, E_b and overlapping space E_c. In this figure, the visual transformation is to a point on the line on screen. An artist that preserves the continuity of the function must generate graphics such that functions evaluted $k \in K_a \cap K_b$ replace a) w/ fiber cross diagram? - use first half introduced in **??**

4.3.2.1 Multiview Constraints: The concept that shared data fields should be encoded visually in a consistent is formally expressed by Qu and Hullman [**?**] as the notion that the same field should have the same scale.

We enforce the equivariance constraint specified in **??** by constructing artists that apply ν encoders such that

$$\begin{array}{ccccc} F_a & & \xrightarrow{\nu_c} & & P_c \\ & {\scriptstyle proj_a}\searrow & & \nearrow{\scriptstyle\nu_c} & \\ & & F_c & & \\ & {\scriptstyle proj_b}\nearrow & & \searrow{\scriptstyle\nu'_c} & \\ F_b & & \xrightarrow[\nu_{c'}]{} & & P_{c'} \end{array} \tag{17}$$

4.3.3 *Visual to Graphic:* Q

$$\begin{array}{ccc} \mathcal{V} & & H \\ {\scriptstyle\mathcal{O}_V}\uparrow & \xrightarrow{Q} & \uparrow{\scriptstyle\mathcal{O}_A} \\ \mathcal{K} & & \mathcal{S} \end{array} \tag{18}$$

$$\begin{array}{ccccc} \mathcal{P}_\dashv & & \xrightarrow{Q_a} & & \mathcal{O}_A \\ & {\scriptstyle p_c}\searrow & & \nearrow{\scriptstyle l_c} & \\ & & \mathcal{P}_\rfloor & & \\ & {\scriptstyle p'_c}\nearrow & & \searrow{\scriptstyle l'_c} & \\ \mathcal{P}_\lfloor & & \xrightarrow{Q_b} & & \mathcal{O}_{A'} \end{array} \tag{19}$$

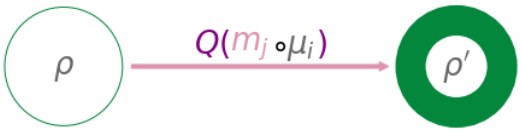


Fig. 12: rework this as a commutative box w/ the r in E row associated w/ this qhat(k)

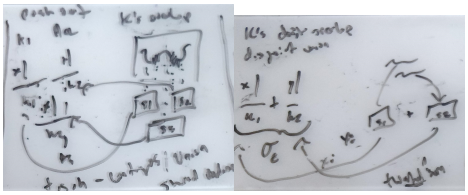
4.3.4 *Graphic to Data: ξ*

$$\begin{array}{ccccc} E & & V & & H \\ & {\scriptstyle\pi}\searrow \quad \swarrow{\scriptstyle\pi} & & & {\scriptstyle\pi}\downarrow \\ & K & \xleftarrow{\quad\xi\quad} & & S \end{array} \tag{20}$$

The functor ξ is a deformation retract, which means.... [?], [?]

By homotopy lifting?

$$\begin{array}{ccc} & & U_j \\ & {\scriptstyle\xi^*}\nearrow & \downarrow{\scriptstyle\iota} \\ K & \xleftarrow[\xi]{} & S \end{array} \tag{21}$$



(a) $K_a \sqcup_{K_c} K_b$ (b) $K_c = \varnothing$

Fig. 13: probably doesn't need the k=0

$$\begin{array}{ccccc} K_c & \hookrightarrow & K_b & \xleftarrow{\xi} & S_b \\ \downarrow & \overset{\xi}{\searrow} & & & \uparrow \\ K_a & \xleftarrow{\xi} & S_a & \leftarrow & S_c \end{array} \tag{22}$$

talk about xi even though is not explicitely implemented

The mapping between graphic and data space expresses how... This is what allows the artist to generate graphics where the subset of data on view is dynamically updated, such as pan and zoom navigation [?] and sliding windows on streaming data [?], [?].

5 Case Study

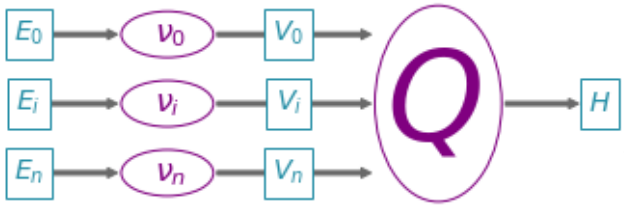


Fig. 14: ν&

We implement the **arrows** in **??**. `axesArtist` is a parent artist that acts as a screen. This allows for the composition described in **??**

5.1 A

```
for local_tau in axesArtist.artist.data.query(screen_bounds, dpi):
    mu = axesArtist.artist.graphic.mu(local_tau)
    rho = axesArtist.artist.graphic.qhat(**mu)
    H = rho(renderer)
```

where the artist is already parameterized with the ξ functions and which fibers they are associated to:

```

```

5.1.1 ξ

5.1.2 ν

5.1.3 Q̂

6 Discussion

6.1 Limitations

6.2 future work

7 Conclusion

The conclusion goes here.

Acknowledgments

The authors would like to thank...

Hannah Aizenman Biography text here.

Thomas Caswell Biography text here.

Michael Grossberg Biography text here.