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. We propose that visualization library components should preserve continuity and equivariance, which we define as

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 data transformations 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

In **??** we discuss how data continuity influences visualization library architecture and in **??** we summarize how visualization researchers have approached equivariance.

2.1 Continuity

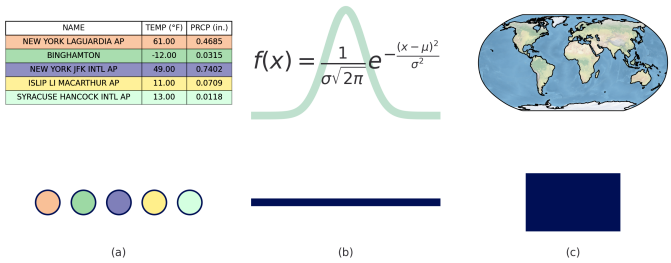


Fig. 1: Continuity is how elements in a data set are connected to each other, which is distinct from how the data is structured. The rows in (a) are discrete, therefore they have discrete continuity as illustrated by the discrete dots. The gaussian in (b) is a 1D continuous function, therefore the continuity 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

- *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.

its 6 nearest cardinal neighboring points (NW, N, NE, E, SE, 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 [?].

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.

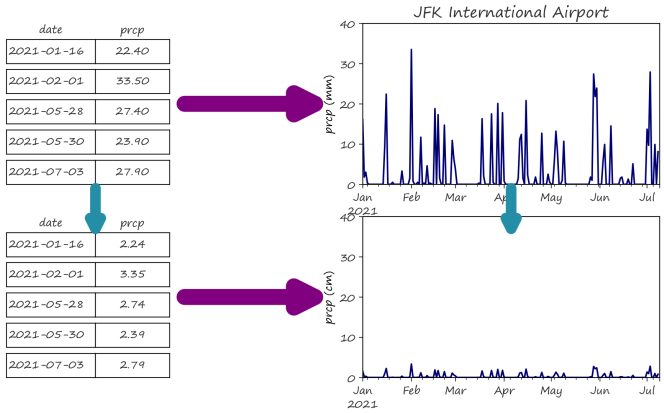


Fig. 3: Equivariance is that a transformation on the data has a corresponding transformation in the graphical representation. For example, in this figure the data is scaled by a factor 10. Equivalently the line plot is scaled by factor of 10, resulting in a shrunken line plot. Either a transformation 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.

3 Algebraic Topology & Category Theory

potentially move this to appendix, if so push Butler Vickers up to RW

Category theory can be used to express software specifications because it provides a method of expressing structure and how that structure can be composed [?] algebraic topology provides a rich language for expressing data continuity. In this section, we introduce concepts from algebraic topology and category theory that we will use in ?? to develop a formal model of specifying visualization component.

3.1 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.

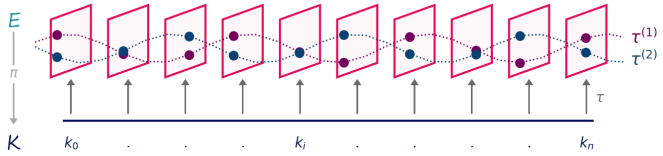


Fig. 4: A fiber bundle is mathematical construct that allows us to express the relationship between data and continuity. The total space Eis the topological space in which the data is embedded. The fiber space F is embedded in E and is the set of all possible values that any add big rectangle E

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

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

Definition 3.1. 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 3.2. 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 [?], [?]

The fibers can be thought of as encapsulating data types according to Spivak, who proposes that databases can be represented as fiber bundles [?], [?]. Spivak formulates the fiber as a (field name, data domain) simple schema where field names and field types can be formally mapped to the set of values associated with the field type, for example $\mathbb{R}$ for a `float` column named *temperature*.

Definition 3.3. 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 $E = K \texttt{times}_T F$ [?], [?]. The twisting refers to the fiber F not aligning in the same direction over all $U \subseteq K$.

There is a map from a point on the base space K which returns values in the fiber space F. This map is called a section $\tau : K \to E$ of the fiber bundle

$$\begin{array}{ccc} F & \hookrightarrow & E \\ & & \pi \downarrow \quad \tau \in \Gamma(K, E) \\ & & K \end{array} \tag{2}$$

and the set of all sections over K is denoted $\Gamma(K, E)$. Sections can also be defined locally, meaning over an open set $U \subseteq E$ and the set of all sections over an arbitrary open set is $\Gamma(U, E)$.

$$\Gamma(U, E|_U) \coloneqq \left\{ \tau : U \to E|_U \;\middle|\; \pi(\tau(k)) = k \; \forall k \in U \right\} \tag{3}$$

Butler et al proposed that [?], [?] sections could be used to model data sets since they are maps from a point on the continuity to a point in the fiber, and therefore they encapsulate both. The set of sections is the space of all data sets with the same continuity and fields.

3.2 Sheaves

Sheaves can be thought of as an "algebraic data structure" [?] for data that lives over topological spaces. A presheaf $\mathcal{O}$ is a map from the open set to the set of sections $\mathcal{O} : U \to \Gamma(U, E|_U)$ [?], [?], [?]. The presheaf preserves inclusion maps ι between open sets and the inclusion map ι^* between the sets of sections.

$$\begin{array}{ccccc} \Gamma(U_1, E|_{U_1}) & & \xleftarrow{\iota^*} & & \Gamma(U_2, E|_{U_2}) \\ \uparrow \mathcal{O}_E & & \uparrow \mathcal{O}_E & & \uparrow \mathcal{O}_E \\ U_1 & & \xrightarrow{\iota} & & U_2 \end{array} \tag{4}$$

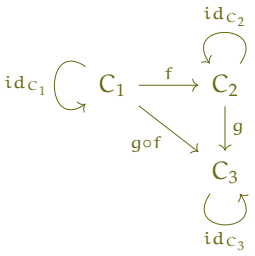
This means that a smaller space U_1 is included in a larger space U_2 and a function $\tau \in \Gamma(U, E|_U)$ that is continuous over a larger space U_2 is continuous over a subspace $U_1 \subset U_2$. Sheaves are presheafs where sets of sections are defined over unions of open sets over a topology $\bigcup_{j \in I} U)_j \in K$ [?], [?]. Sheaves provide a mathematical abstraction of data, the space it lives over, and relationships between subsets.

3.3 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 **??**.

Definition 3.4. A category $\mathcal{C}$ consists of a collection of objects c with identity $\mathrm{id}_c : C \to C$ and the set of morphisms between every two objects $f : C_1 \to C_2$, [?], [?].

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 $f \circ \mathrm{id}_{C_1} = f = \mathrm{id}_{C_2} \circ f$

Definition 3.5. An opposite category $\mathcal{C}^{op}$ is a category with all the same objects of category $\mathcal{C}$ and morphisms that 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 3.6. A functor is a morphism between objects of any two categories $F : \mathcal{C} \to \mathcal{D}$. 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 **[?]**. For example, the functor $F : \mathcal{C} \to \mathcal{D}$ maps an object in $\mathcal{C}$ into an object in $\mathcal{D}$

$$\begin{array}{ccc} c & \xrightarrow{F} & F(c) \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle F(f)} \\ c' & \xrightarrow[F]{} & F(c') \end{array} \tag{5}$$

such that for every morphism $f : c \to c' \in \mathrm{Hom}(\mathcal{C})$ there is a compatible morphism $F(f) : F(c) \to F(c') \in \mathrm{Hom}(\mathcal{D})$. Functors $F : \mathcal{C}^{op} \to \mathcal{D}$ are called contravariant functors.

Definition 3.7. A presheaf $\mathcal{O}$ as introduced in **??**, is a functor $\mathcal{O} : \mathcal{C}^{op} \to \mathbf{Set}$. **[?]**

Definition 3.8. A natural transformation is a morphism of functors $\alpha : F \to G$ where F, G are functors from $\mathcal{C}$ to $\mathcal{D}$. **[?]**, **[?]**

$$\begin{array}{ccc} & \overset{F}{\curvearrowright} & \\ \mathcal{C} & \Downarrow \alpha & \mathcal{D} \\ & \underset{G}{\curvearrowright} & \end{array} \tag{6}$$

The natural transformation composes with a functor such that the output is a compatible functor between the same categories. This means that for every morphism $f : c \to c'$ between objects in $\mathcal{C}$, there is a commutating naturality square for the corresponding objects in $\mathcal{D}$ **[?]**.

$$\begin{array}{ccccc} & & \overset{\alpha_c}{\curvearrowright} & & \\ F(c) & \xleftarrow{F} & c & \xrightarrow{G} & G(c) \\ {\scriptstyle F(f)}\downarrow & & {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle G(f)} \\ F(c') & \xleftarrow[F]{} & c' & \xrightarrow[G]{} & G(c') \\ & & \underset{\alpha_{c'}}{\curvearrowright} & & \end{array} \tag{7}$$

unpack this a drop more

4 Artist

In this section we describe the properties a data to visualization transform must satisfy to be considered equivariant and continuity preserving. We propose that by explicitly defining these properties, we can better incorporate them into visualization library design.

We formulate the visualization transformation as a functor, which we call the Artist A[1]. The artist A converts data, which we denote $\mathcal{E}$ to graphics $\mathcal{H}$. We formulate this association as

$$A : \mathcal{E} \to \mathcal{H} \tag{8}$$

In this section we encapsulate the structure of the data and graphic spaces as categories. We then use these definitions to express the structure that an artist preserves. Finally, we introduce rules for composing artists in a way where the structure between the inputs to the individual artists is preserved.

4.1 Continuity

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 continuity of the data, for example if the data are discrete points, continuous lines, images, or networks.

Definition 4.1. The category of open sets $\mathcal{K}$ consists of

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

We also introduce an analogous category $\mathcal{S}$ that encapsulates the topology of the graphic, which is the topology of the rendering in the display space; for example, this rendering can be a 2D or volumetric image or a 3D print.

Definition 4.2. The category of open sets $\mathcal{S}$ in the topological space $(K, \mathcal{T})$ consists of open set *objects* $W \in \{W\}$, including the empty set $\varnothing$ and the maximum set H and *morphisms* $\iota : W_i \to W_j$.

The data space K is a deformation retract of the graphic topology Sthe letter that regions in S collapse to points in K such that the relative positions of the points are preserved **[?]**. One method of constructing the graphic space is by expanding the dimensionality of the data K to match the spatial dimensionality of the rendering by multiplying the data space by intervals I.

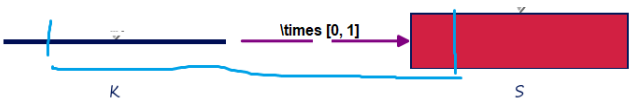


Fig. 5: Given a 2D screen display, the graphic space S must be 2D. It is constructed by multiplying the line K by the interval [0,1] such that every vertical band in S corresponds to a unique point $k \in K$.

For example, as illustrated in **??**, the display space is a 2D screen and the data space is a 1D line K. For that line to have any thickness, it must be rendered as a ban. One method of obtaining this graphic continuity while preserving the

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

constraint that it is a deformation retract is by multiplying the base space K by an interval $I = 0, 1 such that S = K \times I^1$. This construction ensures that every region $(k, \alpha)|_{\alpha \in I} \subset S$ maps to a corresponding point $k \in K$.

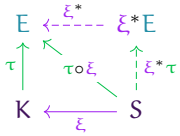
4.1.1 Data

clean up why sheaf As mentioned in **??**, sheaves are a powerful tool for keeping track of data that lies over topological spaces. Therefore, we model the type of data an artist expects as a sheaf $\mathcal{O}(E) : \mathcal{K}^{op} \to \mathbf{Set}$.

$$\begin{array}{c} \Gamma(U, E\restriction_U) \in \mathrm{Ob}(\mathbf{Set}) \\ \uparrow \mathcal{O}(E) \\ U \in \mathrm{Ob}(\mathcal{K}^{op}) \end{array} \tag{9}$$

The objects of **Set** in this sheaf are the sets of sections $\Gamma(K, E)$ introduced in **??** and the morphisms are the inclusion maps (ι, ι^*) introduced in **??**.

To establish a mapping between graphic and data base spaces, we introduce the surjective map $\xi : S \to K$.



(10)

such that the data continuity K is a deformation retract of S, as expressed in **??**.

As shown in **??**, the map between spaces ξ can be composed with section map τ such that there is a map from the graphic base space S to the fiber bundle in which the data lives E. Since there is a mapping from the graphic space

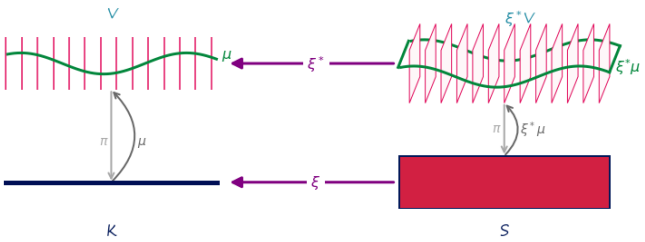


Fig. 6: The map ξ^* is a map from a repetition of the 1D fiber over every point $k \in K$ in data space, turning the fiber into a 2D plane of repeating values over the corresponding region $k = \xi(s)$ in the graphic space S.

to the total bundle $\tau \circ \xi : S \to E$, the mapping ξ can be pulled through the composition such that there is a map ξ^* between a total space over graphics $\xi^* E$ and its equivalent space over data E. As illustrated in the example given in **??**, a bundle E over a point $k \in K$ is pulled back through ξ **[?]** such that it is copied over every point in the corresponding region in graphic space $\xi(s) = k, s \subset S$. The pullback ξ^* can then be composed with the elements of the set of sections

$$\Gamma(K, E) \xrightarrow{\xi^*} \Gamma(S, \xi^* E) \tag{11}$$

to generate a set of equivalent sections over the graphic space $\Gamma(S, \xi^* E)$. As a consequence of **??**, there exists a data sheaf over graphic space

$$\mathcal{O}_{\xi^*}(E) : \mathcal{S}^{op} \to \Gamma(S, \xi^* E) \tag{12}$$

which is equivalent to

$$\mathcal{O}_{\xi^*}(E) : \mathcal{S}^{op} \to \mathbf{Set} \tag{13}$$

4.1.2 Graphic

We model the space of all graphic generating functions over a specific graphic topology $W \subseteq S$ and targeting a specific display as a set of sections

$$\Gamma(W, H\restriction_W) \coloneqq \left\{ \rho : W \to H\restriction_W \;\middle|\; \pi(\rho(s)) = s \; \forall s \in W \right\} \tag{14}$$

The set of sets of sections $\{\Gamma(W, H)\}|_{W \in gbase}$ is subcategory of **Set**; therefore the full space of graphics is a sheaf $\mathcal{O}(H)$

$$\begin{array}{c} \Gamma(W, H\restriction_W) \supset \left\{ \rho : W \to H\restriction_W \;\middle|\; A(\tau) = \rho \,, \tau \in \mathcal{O}(E) \right\} \\ \mathcal{O}(H) \quad\quad A(\mathcal{O}(E)) \coloneqq \mathcal{O}_A(H) \\ W \end{array} \tag{15}$$

The set of graphics reachable through an artist function is a subset of sections of the full set of graphics $\{\rho\} \subset \Gamma(W, H\restriction_W)$. These sets of reachable graphics over open sets W are also objects of **Set**. This allows us to formulate the graphic output by the artist as a sheaf

$$\mathcal{O}_A(H) : \mathcal{S}^{op} \to \mathbf{Set} \tag{16}$$

from the graphic base space to the set of reachable graphics.

4.1.3 Data to Graphic

Following from **??** and **??**, we propose that the artist function A is a natural transformation

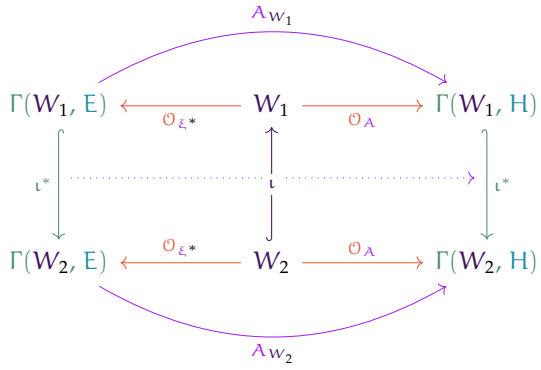
$$\mathcal{S}^{op} \;\; \overset{\mathcal{O}_{\xi_*}(E)}{\underset{\mathcal{O}_A(H)}{\Downarrow A}} \;\; \mathbf{Set} \tag{17}$$

of sheaf functors

$$A : \mathcal{O}_{\xi^*}(E) \to \mathcal{O}_A(H) \tag{18}$$

where each sheaf is a map from an open set in the graphic base space $W \subset S$ into a set of sections of a fiber bundle over that open set. As defined in **??**, the artist is a natural transformation. Following from **??**, the artist specifies that

there are compatible inclusion maps in the data and graphic spaces

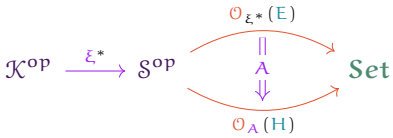


(19)

Given that $W \subseteq S$, **??** can be generalized to sections over S. This fact can be combined with **??** to describe the path from data sections over data space to graphic sections over graphic space

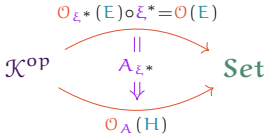
$$\Gamma(K, E) \xrightarrow{\xi^*} \Gamma(S, E) \xrightarrow{A} \Gamma(S, H) \tag{20}$$

which means that the artist can be pulled back through ξ^*[2]



(21)

such that the artist can be constructed over data space



(22)

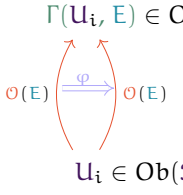
which in turn provides a way to construct the artist as a map from data section in data space to corresponding graphic section

$$A_{\xi^*} : \tau \mapsto \rho \tag{23}$$

such that for every dataset, the artists constructs a corresponding graphic generating function. maybe mention here Munzner, Zeminsky, and the other dude on glyphs map back to data possibly more here as this is the type signature of the thing we implement

4.2 Equivariance

Besides preserving inclusions, as described in **??**, the artist function is constructed such that transformations on τ and ρ are equivariant with respect to the morphism between data sheaf functors φ which was introduced in **??**.



(24)

[2]Definition 3.68 in Seven Sketches in Compositionality by Fong and Spivak

The map $\varphi : \mathcal{O}(E) \to \mathcal{O}(E)$ is any arbitrary morphism between sheaf functors $\mathcal{O}(E)$. Following from **??**, φ is a natural transform, which means all inclusions ι, ι^* are preserved as part of any φ transformation.

Definition 4.3. The fiber category $\mathcal{F}$ is a monoidal category [**?**], which means it is a category equipped with a bifunctur $\otimes : \mathcal{F} \times \mathcal{F} \to \mathcal{F}$

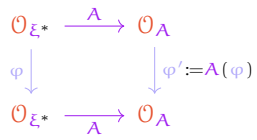
- *object* a monoid object F. A monoid is a set with a binary operator that is associative, closed, and for which the set contains an identity element i [**?**].
- *morphisms* $F \otimes F \to F$ and unit $i \to F$

The bifunctor $\otimes$ combines categories in a way that preserves identity and composition [**?**]. For example, given a pair of categories $\mathcal{F}_\dashv, \mathcal{F}_\lfloor$, the bifunctor yields a set of pairs $\{(F_a, F_b) F_a \in \mathcal{F}_\dashv, F_b \in \mathcal{F}_\lfloor\}$. For the associated morphisms $g : \mathcal{F}_\dashv \to \mathcal{F}_\dashv$ and $h : \mathcal{F}_\lfloor \to \mathcal{F}_\lfloor$, the function $(g \times h) : (\mathcal{F}_\dashv \times \mathcal{F}_\lfloor) \to (\mathcal{F}_\dashv \times \mathcal{F}_\lfloor)$ is pointwise $(g \times h)(F_a, F_b) \coloneqq (g(F_a), h(F_b))$. Because of the bifunctor, objects of the category $\mathcal{F}$ can be the product of any arbitrary number and types of categories, including **Set**, **Graph**, and topological spaces **Top**.

The set of all morphism of the fiber category $Hom_{\mathcal{F}}(F, F)$ are all possible morphisms in a given category. This includes any of the functions that serve as the basis of equivariance, such as the binary operations, group actions, and measurement scale discussed in **??**. The generalization to $Hom_{\mathcal{F}}$ also allows for the inclusion of structures such as monoid actions [**?**], which provide a way of applying partial order relations to data [**?**], such as to build multi-ranked indicators [**?**].

When a fiber bundle is trivial, we can dispense with localization and directly consider the fiber maps $Hom(F_k, F_k)$ over a point k as the basis of our equivariance. This is because a 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 $\mathcal{O}(E)|_k \coloneqq \lim_{U \ni k} \Gamma(U, E)$ [**?**]. 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 $\varphi \in Hom_{\mathcal{F}}(F, F)$ include the types of data transformations described in **??**In **??**, we assume that the fiber bundles are trivial.

$A : \mathcal{O}^{\xi^*} \to \mathcal{O}_A$, it follows from **??** that an arbitrary morphism φ on the data sheaf has a compatible morphism φ' on the graphic sheaf.



(25)

As a consequence of **??**,

4.3 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 $\mathcal{E}$.

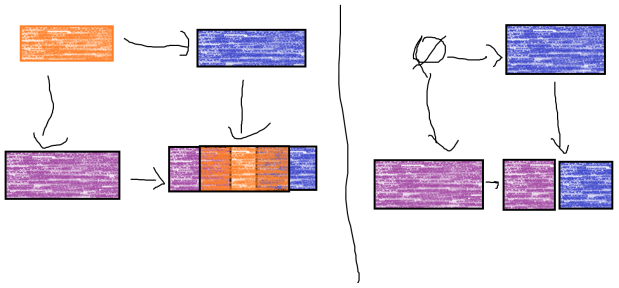


Fig. 7: In **??**, the input object $\mathcal{O}_E$ encodes a continuous function over spaces E_a, E_b and overlapping space E_c. In this figure, the visual transformation is to a point on the line on screen. An artist that preserves the continuity of the function must generate graphics such that functions evaluated $k \in K_a \cap K_b$

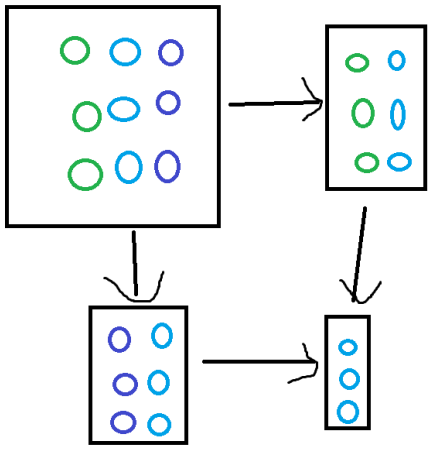


Fig. 8: The fiber spaces F_a, F_b have a shared fiber space F_c. If they are input into the same artist A_1, then an equivariant transformation is one where F_c is transformed to a visual element in a consistent matter. In this figure, F_c is mapped to the x-axis for both F_a and F_b.

4.3.1 *Shared Index*

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 [**?**];

$$\begin{array}{ccc} K_c & \xhookrightarrow{\iota} & K_b \\ \downarrow{\scriptstyle \iota} & & \downarrow{\scriptstyle i_{K_b}} \\ K_a & \xrightarrow{i_{K_a}} & K_a \bigsqcup_{K_c} K_b \end{array} \tag{26}$$

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.

4.3.2 *Shared Fields*

$$\begin{array}{ccc} F_a \times F_b & \xrightarrow{\text{proj}_a} & F_a \\ \downarrow{\scriptstyle \text{proj}_b} & & \downarrow{\scriptstyle \text{proj}_c} \\ F_b & \xrightarrow{\text{proj}_b} & F_c \end{array} \tag{27}$$

?? 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?

5 Construction and Formal Properties of Artists

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

$$A : \mathcal{O}_{\mathcal{E}} \tag{28}$$

that we construct to preserve continuity and equivariance.

Definition 5.1. 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

$$\begin{array}{ccc} E & \xrightarrow{\xi} & V \\ & {\scriptstyle \pi}\searrow & \downarrow{\scriptstyle \pi} \\ & & K \end{array} \qquad \begin{array}{ccc} \mathcal{V} & & \mathcal{H} \\ \uparrow{\scriptstyle \mathcal{O}_V} & \xrightarrow{Q} & \uparrow{\scriptstyle \mathcal{O}_A} \\ \mathcal{K} & \xrightarrow{\xi^*} & \mathcal{S} \end{array} \tag{29}$$

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

5.1 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.

One example of encoding data as a section of a fiber bundle is illustrated in **??**. In this example, the data is a not

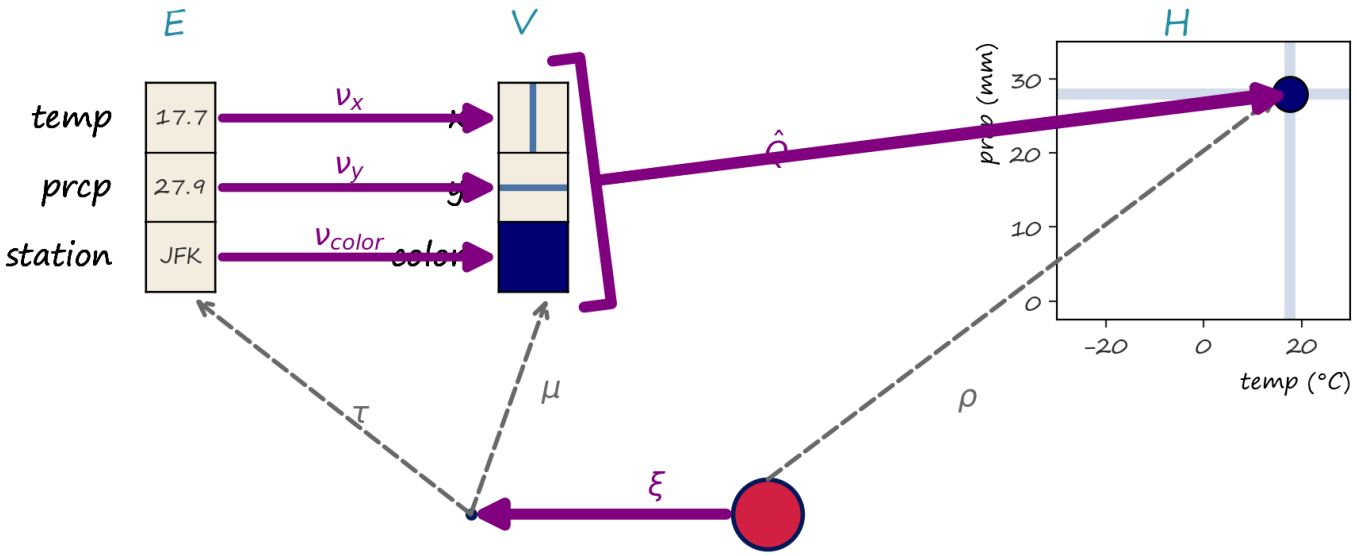


Fig. 9

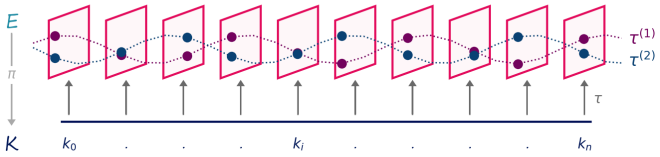


Fig. 10: replace with more concrete

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.

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

5.2 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

$$\begin{array}{ccc} D & \hookrightarrow & H \\ & & \pi \downarrow \uparrow \rho \\ & & S \end{array} \tag{30}$$

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.

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 [**?**].

5.3 Visualization Library Components

5.3.1 *Visual Bundle V*

$$\begin{array}{ccc} P & \hookrightarrow & V \\ & & \pi \downarrow \uparrow \mu \\ & & K \end{array} \tag{31}$$

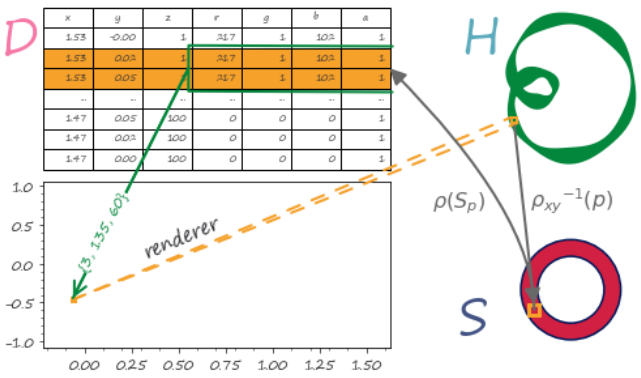
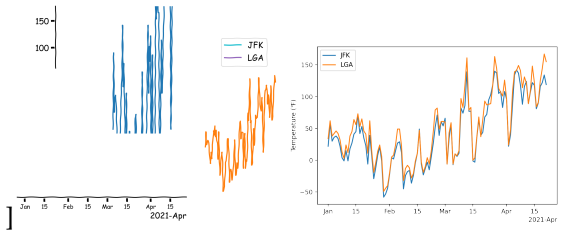


Fig. 11



(a) No shared fields

(b) Shared fields

Fig. 13: 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 **??**

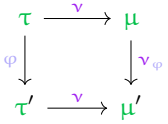
5.3.2 *Data to Visual Encodings:* ν

maybe figure out how to put these side by side nicely

(32)

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

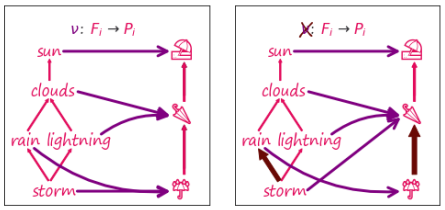


(33)

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.

$$\{\nu_0, \ldots, \nu_n\} : \{\tau_0, \ldots, \tau_n\} \mapsto \{\mu_0, \ldots, \mu_n\} \tag{34}$$

One example of this is



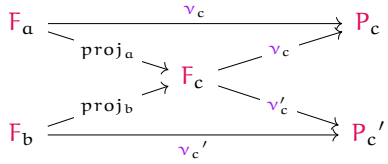
(a) insert title

(b) insert title

Fig. 12: is a weird nu, colors are a but much

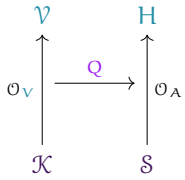
5.3.2.1 Multiview Constraints: The concept that shared data fields should be encoded visually in a consistent is formally expressed by Qu and Hullman [**?**] as the notion that the same field should have the same scale.

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



(35)

5.3.3 *Visual to Graphic:* Q



(36)

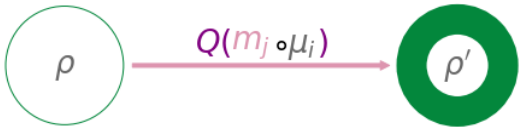
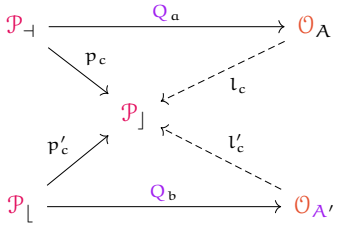
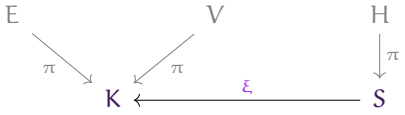


Fig. 14: rework this as a commutative box w/ the r in E row associated w/ this qhat(k)



(37)

5.3.4 *Graphic to Data:* ξ

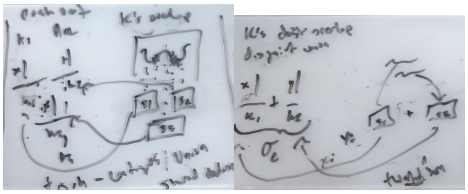


(38)

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

By homotopy lifting?

$$\begin{array}{ccc} & & U_j \\ & \overset{\xi^*}{\nearrow} & \downarrow \iota \\ K & \xleftarrow[\xi]{} & S \end{array} \tag{39}$$



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

Fig. 15: probably doesn't need the k=0

$$\begin{array}{ccccc} K_c & \hookrightarrow & K_b & \xleftarrow[\xi]{} & S_b \\ \downarrow & \overset{\xi}{\searrow} & & & \uparrow \\ K_a & \xleftarrow[\xi]{} & S_a & \hookleftarrow & S_c \end{array} \tag{40}$$

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 [?], [?].

6 Case Study

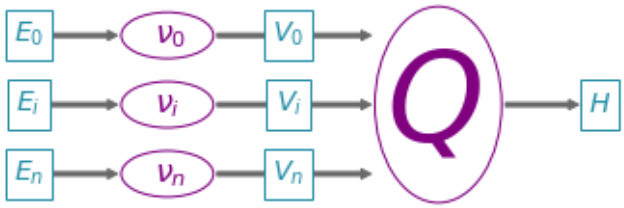


Fig. 16: ν&

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

6.1 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:

```

```

6.1.1 ξ

6.1.2 ν

6.1.3 $\hat{Q}$

7 Discussion

7.1 Limitations

7.2 future work

8 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.