Data Visualization from a Category Theory Perspective

Table of contents

Motivation

This thesis began by trying to reproduce the following plot within the Julia programming language.

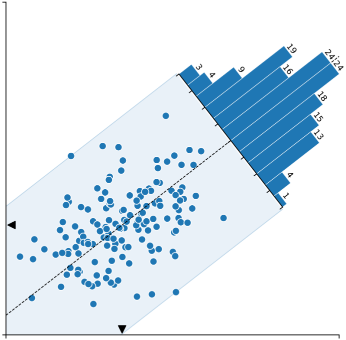


Figure 1: Rotated histogram aligned with second main PCA axis. Figure from **?**].

Introduction

Develope a new data visualizaiton package.

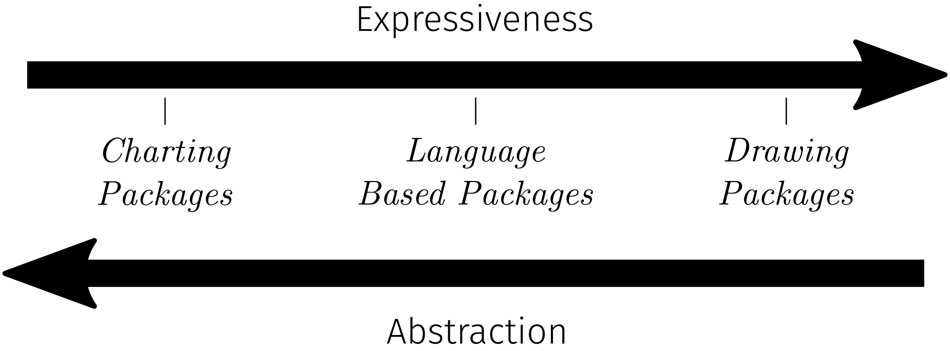


Figure 2: Expressiveness per data visualizaiton package type.

Introduction

Scatter plot and Line plot are programmatically different.

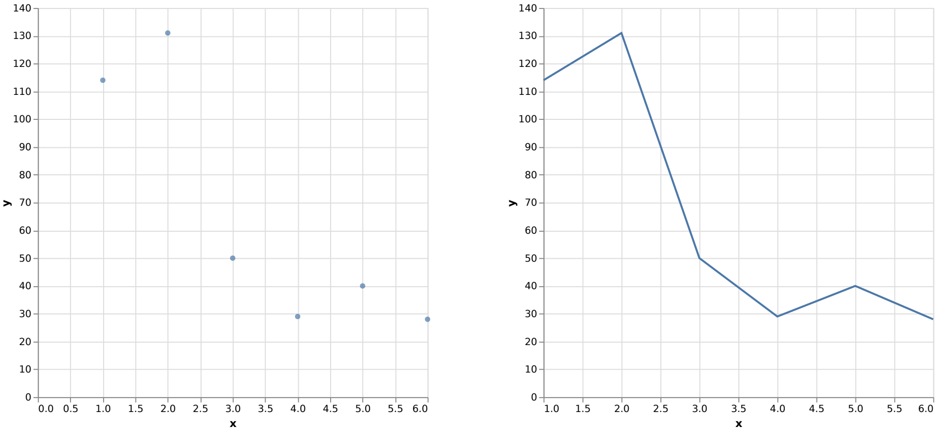


Figure 3: Expressiveness per data visualizaiton package type.

Introduction

Surprisingly, as pointed out by **?**], although graphics are extensively used in many fields, there is still not a lot of substantive theory on the subject from the perspective of data visualization.

From our literature review, we were able to identify three seminal works that contributed to formalize the process of encoding data in a geometrical description:

- Semiology of Graphics [**?**];
- A Presentation Tool [**?**];
- Grammar of Graphics [**?**].

Objective and Challenges

Main Objective: Develop a new formalization framework for data visualization. The purpose of this framework is to guide the development of a general purpose data visualization library by providing sound abstraction without compromising expressiveness.

Challenges

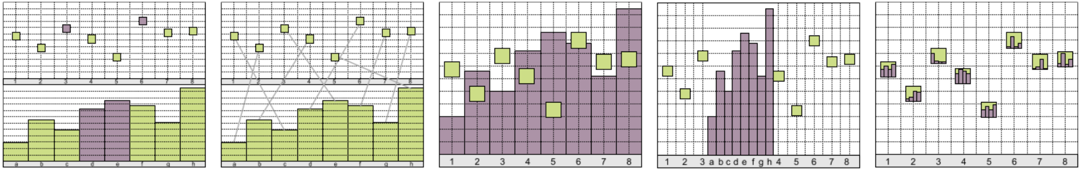
Expressiveness:

- Composite visualizations;
- Customized marks.

Abstraction:

- Mathematical rigour;
- Visualizations constructed with an intuitive logic.

Challenges



(a) Juxtaposed views. (b) Integrated views. (c) Superimposed views. (d) Overloaded views. (e) Nested views.

Figure 4: Example of composing a *scatter plot* and *bar chart*. Taken from **?**].

Challenges

Example of visualization that uses customized marks.

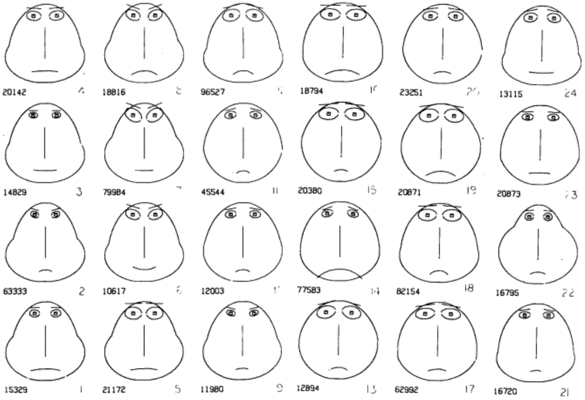


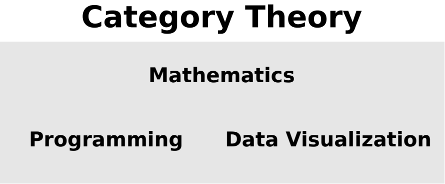
Figure 5: Example of Chernoff faces from **?**].

Why Category Theory?

Category Theory is a branch of mathematics that studies general abstract structures through their relationships.

Origin: Samuel Eilenberg e Saunders Mac Lane - 1940

As pointed by **?**], Category Theory is unmatched in its ability to organize and relate abstractions. We believe that it may serve as a robust framework for bridging mathematics, computer science and data visualization.



Category Theory Brief Introduction

Category Theory is a branch of mathematics that studies general abstract structures through their relationships.

Definition (Category)

A category $\mathcal{C} = \langle \mathrm{Ob}_{\mathcal{C}}, \mathrm{Mor}_{\mathcal{C}} \rangle$ consists of a class of objects $\mathrm{Ob}_{\mathcal{C}}$ and a class of morphisms $\mathrm{Mor}_{\mathcal{C}}$. A morphism $f \in \mathrm{Mor}_{\mathcal{C}}(A, B)$ has a domain $A \in \mathrm{Ob}_{\mathcal{C}}$ and a codomain $B \in \mathrm{Ob}_{\mathcal{C}}$. Every object has an identity morphism. Every morphism can be composed, and this composition is associative.

Category Theory Brief Introduction

The category $\mathbf{1}$ consists of $\text{Ob}_{\mathbf{1}} := \{A\}$ and $\text{Mor}_{\mathbf{1}} = \{id_A\}$. The diagram for such category is shown below.

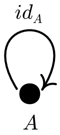


Figure 6: Category $\mathbf{1}$.

The category $\mathbf{2}$ consists of $\text{Ob}_{\mathbf{2}} := \{A, B\}$ and $\text{Mor}_{\mathbf{1}} = \{id_A, id_B, f\}$, where $f : A \to B$. The diagram for such category is shown below.

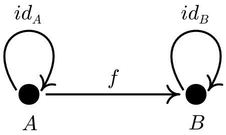


Figure 7: Category $\mathbf{2}$.

Category Theory Brief Introduction

The category **3** has three morphisms besides the identities. The morphisms are f, g and their composition $g \circ f$. The figure below illustrates the category with all its morphisms.

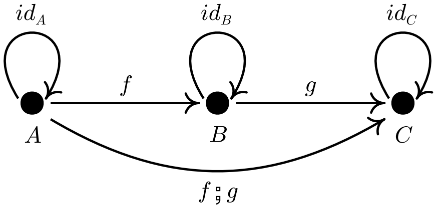


Figure 8: Category **3** showing all morphisms.

Category Theory Brief Introduction

Here are some more interesting categories:

1. **Set** which is the category of sets, where the objects are sets and the morphisms are functions between sets.
2. **Top** is the category where topological spaces are the objects and continuous functions are the morphisms.
3. $\mathbf{Vec}_{\mathbb{F}}$ is the category where vector spaces over field $\mathbb{F}$ are the objects, and linear transformations are the morphisms.
4. **Gr** is the category of directed graphs, where $\mathrm{Ob}_{\mathbf{Gr}} := \{\mathrm{Vertex}, \mathrm{Arrow}\}$, and the morphisms are

$$\mathrm{Mor}_{\mathbf{Gr}} := \{src, tgt, id_{\mathrm{Vertex}}, id_{\mathrm{Arrow}}\}$$

   where $src : \mathrm{Arrow} \to \mathrm{Vertex}$ returns the source vertex for each arrow and $tgt : \mathrm{Arrow} \to \mathrm{Vertex}$ returns the target vertex.

Category Theory Brief Introduction

Objects defined in terms of existence and uniqueness of morphisms are known as **universal constructions**.

Definition (Zero, Initial and Terminal)

Let $\mathcal{C}$ be a category.

1. An object $I \in \mathrm{Ob}_{\mathcal{C}}$ is *initial* if for every $A \in \mathrm{Ob}_{\mathcal{C}}$, there is exactly one morphism from I to A. Thus, from I to I there is only the identity.
2. An object $T \in \mathrm{Ob}_{\mathcal{C}}$ is *terminal* if for every $A \in \mathrm{Ob}_{\mathcal{C}}$, there is exactly one morphism from A to T. Thus, from I to I there is only the identity.
3. An object is *zero* if it is both terminal and initial.

Definition (Functor)

Let $\mathcal{C}$ and $\mathcal{D}$ be two categories. A functor $F : \mathcal{C} \to \mathcal{D}$ is a pair of mappings with the following properties:

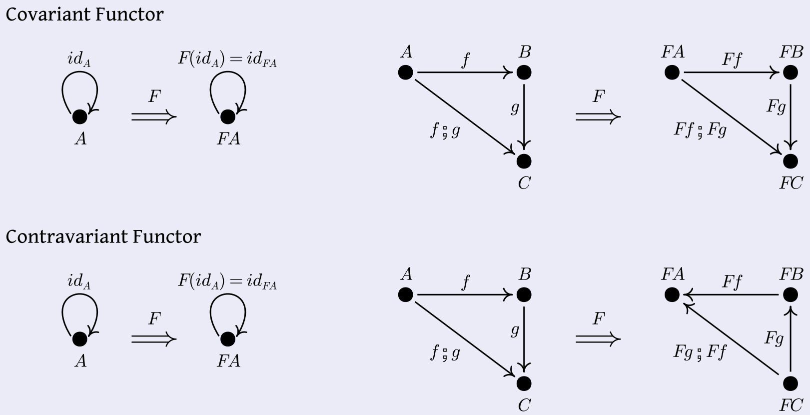


Figure 9: Diagrams showcasing the properties of functors.

Definition (Natural Transformations)

Let $\mathcal{C}$ and $\mathcal{D}$ be categories, and let $F, G : \mathcal{C} \to \mathcal{D}$ be functors. A natural transformation $\alpha : F \to G$ is such that the following diagram commutes:

$$\begin{tikzcd}
FA \arrow[r, "\alpha_A"] \arrow[d, "F(f)"'] & GA \arrow[d, "G(f)"] \\
FB \arrow[r, "\alpha_B"'] & GB
\end{tikzcd}$$

$$F(f) \mathbin{;} \alpha_B = \alpha_A \mathbin{;} G(f)$$

Figure 10: Commutative diagram of a natural transformation highlighting the commutative property of the definition.

Monoids and **Monads** are two ubiquitous constructions both in Category Theory and Functional Programming. These two concepts will be used when talking about data visualization. Therefore, it is required of us to introduce these constructions.
Let's start with the definition of a monoid in the context of Set Theory.

Definition (Monoid - Set Theory)

A monoid is a triple $(M, \otimes, e_M)$ where M is a set, $\otimes : M \times M \to M$ is a binary operation and e_M the neutral element, such that:

1. $a \otimes (b \otimes c) = (a \otimes b) \otimes c$
2. $a \otimes e_M = e_M \otimes a = a$.

An example of a monoid is $(\mathbb{N} \cup \{0\}, +, 0)$. It is easy to check that the summation operator satisfies the associativity neutrality properties.

Category Theory Brief Introduction

Definition (Monoid in the category **Set**)

A monoid in **Set** is a triple (M, μ, η), where $M \in \mathrm{Ob}_{\mathbf{Set}}$, $\mu : M \times M \to M$ and $\eta : 1 \to M$ are two morphisms in **Set** satisfying the commutative diagrams below. Note that 1 is the terminal object in **Set**, i.e. the singleton set (which is unique up to an isomorphism).

$$
\begin{tikzcd}
M \times M \times M \arrow[r, "id_M \times \mu"] \arrow[d, "\mu \times id_M"'] & M \times M \arrow[d, "\mu"] \\
M \times M \arrow[r, "\mu"'] & M
\end{tikzcd}
\qquad
\begin{tikzcd}
M \arrow[r, "\eta \times id_M"] \arrow[dr, "id_M"'] & M \times M \arrow[d, "\mu"] & M \arrow[l, "id_M \times \eta"'] \arrow[dl, "id_M"] \\
 & M &
\end{tikzcd}
$$

Figure 11: Commutative diagram for monoid.

Category Theory Brief Introduction

Definition (Monad)

A monad is a monoid in $\mathbf{End}_{\mathcal{C}}$, which is the triple (T, μ, η), where $T : \mathcal{C} \to \mathcal{C}$ is a functor, $\mu : T \circ T \to T$ and $\eta : 1 \to T$ are natural transformations in $\mathbf{End}_{\mathcal{C}}$ satisfying the commutative diagrams 12. Note that 1 is the identity functor in $\mathcal{C}$.

$$
\begin{tikzcd}
T \circ T \circ T \arrow[r, "id_T * \mu"] \arrow[d, "\mu * id_T"'] & T \circ T \arrow[d, "\mu"] \\
T \circ T \arrow[r, "\mu"'] & T
\end{tikzcd}
\qquad
\begin{tikzcd}
T \arrow[r, "\eta * id_T"] \arrow[dr, "id_T"'] & T \circ T \arrow[d, "\mu"] & T \arrow[l, "id_T * \eta"'] \arrow[dl, "id_T"] \\
 & T &
\end{tikzcd}
$$

Figure 12: Commutative diagram for monad.

Programming

One of the most common applications is in Programming, where Category Theory can presents a strong connection with the Functional Programming paradigm.

Definition (Category $\mathbf{Prog}(L)$)

Category $\mathbf{Prog}(L)$ is a subcategory of **Set** where programming types are the objects and correspond to a set. The morphisms are pure referentially transparent and terminating functions, which correspond to functions in **Set**. If two functions in L are denotationally the same, i.e. for every $\mathrm{x} :: \mathrm{T}$ we have $\mathrm{f(x)} = \mathrm{g(x)}$, then they correspond to the same function in $\mathbf{Prog}(L)$.

An Application in Data Visualization - Diagrams

Data Visualization Pipeline

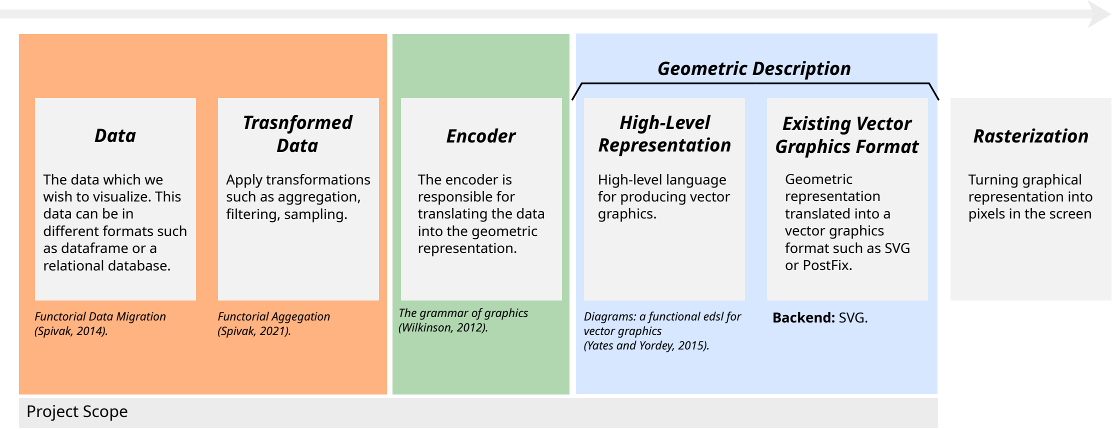


Figure 13: Visualization pipeline.

Diagrams as Monoids

?] proposed a Domain-Specific Language which models the process of generic diagram creation. The paper explains some of the ideas that went behind the development of the framework *Diagrams* [**?**], which is a Haskell library for drawing diagrams.

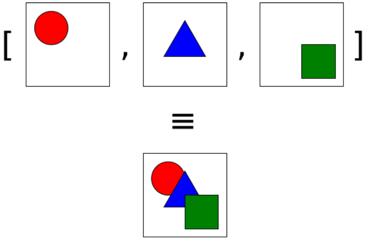


Figure 14: Example of how a diagram works. Image from **?**].

References I