Functor of Points on Schemes & Yoneda Lemma

G. Chiusole

Contents

1 Category Theory

Established material for Category Theory includes [Mac Lane und Eilenberg, 2013], [nlab], [stacksproject].

Definition 1. A **category**[1] is a triple, consisting of a family of objects $\mathrm{Obj}(\mathbf{C})$, a family of morphisms $\mathrm{Mor}(\mathbf{C})$, and a binary operation $\circ : \mathrm{Mor}(\mathbf{C}) \times \mathrm{Mor}(\mathbf{C}) \to \mathrm{Mor}(\mathbf{C})$ s.t.

(CAT1) $\circ$ is associative: $\forall f, g, h \in \mathrm{Mor}(\mathbf{C}) : (f \circ g) \circ h = f \circ (g \circ h)$.

(CAT2) $\circ$ has identities: $\forall f : A \to B, \exists 1_A, 1_B \in \mathrm{Mor}(\mathbf{C}) : 1_B \circ f = f = f \circ 1_A$.

Note that as opposed to a monoid, a category has a (possibly different) identity for every morphism. Not even left and right identity need to be the same - they usually are not.
One can think of these definitions in a purely syntactic way, and not use any set theory what so ever (i.e. not use $\in$ etc.). We will, however, for the sake of convenience, adapt set theory and think of the structures as one would expect.

[1] Here, we will denote a category by using bold font (\mathbf{$\cdot$}). Görtz & Wedhorn mostly use regular font.

In this spirit, by abuse of notation, we will often write $A \in \mathbf{A}$ to mean, that A is an object of the category $\mathbf{A}$ and think of an $f \in \mathrm{Mor}(\mathbf{C})$ as a function $A \in \mathrm{Obj}(\mathbf{C}) \to B \in \mathrm{Obj}(\mathbf{C})$.
This set theoretical thinking is often the motivation or provides a primary example for many categorical concepts - but beware!
Consider the following definition for example:

Definition 2. A *monomorphism* $m : B \longrightarrow C$ is a morphism s.t. for any two parallel morphisms $a_1, a_2 : A \longrightarrow B$ we have that $m \circ a_1 = m \circ a_2$ implies $a_1 = a_2$.
In other words: it is left cancelable[2].

One would be inclined to think that this is just a translations of the definition for injections into the language of category theory. But that is not the case. In fact, there are categories (e.g. **Hot**) where a monomorphism need not be injective. Maybe even more notably, there are categories, for which the intuitive idea of injection and surjection we are familiar with from set theory crumbles.

1.1 Examples of Categories

We will now consider some examples of categories. Many appear naturally in algebra, topology, and geometry.

Category	Objects	Morphisms
Set	Sets	(Set Theoretic) Functions
Grp	Groups	Group Homomorphisms
Ab	Abelian Groups	Group Homomorphisms
$\mathbf{CRing}_1$	Com. Rings with 1	1-Preserving-Ring Homomorphisms
Top	Topological Spaces	Continuous Functions

In all of the examples above, the composition of two "structure-preserving" functions, was a "structure-preserving" function. For example, the composition of continuous functions is continuous, the compositions of group homomorphisms is homomorphic. Hence, if one wants to define a category, it is crucial to check whether the category is closed under $\circ$. Until now we have only considered very intuitive, well behaved categories[3]

Other, less obvious examples include

Category	Objects	Morphisms
Hot	CW-Complexes	Homotopy Class of Continuous Maps
Pos	Elements	$(\exists! f : A \to B) \Leftrightarrow (A \leq B)$
X	Open Subsets of a topological space X	Inclusion Maps $U \hookrightarrow V$
$\mathbf{Sh}(\mathbf{X})$	Sheaves on a top. space X	Natural Transformations between Sheaves on X

1.2 Side Note: CT as a one-sorted theory

As we have seen, and is natural to think, category theory (CT) is a two-sorted theory: i.e. there are two types: objects, and morphisms. However, we can interpret the axioms of CT with a one

[2]An epimorphism, which is right cancelable, is defined similarly.
[3]In fact, we have considered concrete categories. The fact that the category of schemes is not concrete will prove to be an obstacle later on.

sorted model as well, consisting only of the morphisms.

For example, Zermelo-Fraenkel Set Theory is a one sorted theory - the single, primitive type is the set. Graph theory, on the other hand, is two-sorted - in a nutshell, we work with two underlying sets with empty intersection. Considering CT as two-sorted we can make a lot of arguments for categories by thinking of graphs.
Interpreting CT as one sorted means identifying every object A with the identity morphism 1_A on it. Then, of course, a morphism is no longer defined as

$$f \in \mathrm{Mor}(\mathbf{C}) : \mathrm{Obj}(\mathbf{C}) \times \mathrm{Obj}(\mathbf{C}) \to \mathrm{Obj}(\mathbf{C}) \tag{1}$$
$$A \mapsto f(A) = B \tag{2}$$

but rather

$$f \in \mathrm{Mor}(\mathbf{C}) : \mathrm{Mor}(\mathbf{C}) \times \mathrm{Mor}(\mathbf{C}) \to \mathrm{Mor}(\mathbf{C}) \tag{3}$$
$$1_A \mapsto f(1_A) = f \circ 1_A = 1_B \quad . \tag{4}$$

For application in algebraic geometry, that is not all that relevant, but it already gives a hint that one should study morphisms rather than objects.

1.3 Functors

Finally, we want to consider the category of categories. Its elements are categories, as examples are given above. The morphisms between them are called **functors**. Concretely, this means that a (covariant) functor[4] $\mathfrak{f}$ is a morphism

$$\begin{tikzcd}
\mathbf{A} \arrow[d, "\mathfrak{f}"'] & \ni & A \arrow[r, "a"] \arrow[d, mapsto, "\mathfrak{f}"'] & A' \arrow[d, mapsto, "\mathfrak{f}"'] \\
\mathbf{B} & \ni & \mathfrak{f}(A) \arrow[r, "\mathfrak{f}(a)"] & \mathfrak{f}(A')
\end{tikzcd}$$

s.t.

(F1) $\forall A \in \mathbf{A} : \mathfrak{f}_{1_A} = 1_{\mathfrak{f}(A)}$

(F2) $\forall f, g \in \mathbf{A} : \mathfrak{f}(g \circ f) = \mathfrak{f}(g) \circ \mathfrak{f}(f)$

These two conditions ensure that the composition of functors is again a functor, as they preserve the structure of a category.
A contravariant functor is a functor that reverses the order of composition. Hence, for a contravariant functor $\mathfrak{f}'$ the second functor axiom becomes

$$(\mathrm{F2})^{op} : \qquad \forall f, g \in \mathbf{A} : \mathfrak{f}'(g \circ f) = \mathfrak{f}'(f) \circ \mathfrak{f}'(g) \tag{5}$$

We denote $\mathfrak{f}' : \mathbf{A}^{op} \to \mathbf{B}$.

[4]Here, we will denote a functor by using fraktur font (\mathfrak{·}). Görtz & Wedhorn use calligraphy (\mathcal{·}) or regular font.

Proposition 1. *The mapping that assigns to group $G \in \mathbf{Grp}$ its center $Z(G) \in \mathbf{Ab}$ is not a functor.*

Proof. Consider the following diagram:

$$\begin{tikzcd}
\mathbf{Grp} \arrow[d, "\mathfrak{Z}"'] & C_2 \arrow[r, hook, "\iota"] \arrow[d, mapsto, "\mathfrak{Z}"'] & S_3 \arrow[r, mapsto, two heads, "\pi"] \arrow[d, mapsto, "\mathfrak{Z}"'] & S_3/A_3 = C_2 \arrow[d, mapsto, "\mathfrak{Z}"'] & e' \arrow[r, mapsto, "\iota"] \arrow[d, mapsto, dotted, "\mathfrak{Z}"'] & (1\ 2) \arrow[r, mapsto, "\pi"] \arrow[d, mapsto, dotted, "\mathfrak{Z}"'] & e' \arrow[d, mapsto, dotted, "\mathfrak{Z}"'] \\
\mathbf{Ab} & C_2 \arrow[r, mapsto, "1_e"] & \{e\} \arrow[r, mapsto, "1_e"] & C_2 & e' \arrow[r, mapsto, "1_e"] & e \arrow[r, mapsto, "1_e"] & ?
\end{tikzcd}$$

Note that a dotted mapping arrow signalizes abuse of notation since a functor maps objects and morphisms - not elements in those objects.

We see that

$$\mathfrak{Z}(\pi \circ i(e')) = e' \neq e = \mathfrak{Z}(\pi) \circ \mathfrak{Z}(i)(e') \quad . \tag{6}$$

For some more comments see the Mathematics Stack Exchange Question with ID 158438. □

In essence, this is because group homomorphisms do not necessarily map centers to centers.

1.4 Natural Transformations

Now consider the category $[\mathbf{A}, \mathbf{B}]$ where the objects are functors $\mathfrak{f} : \mathbf{A} \to \mathbf{B}$. Morphisms in this category are called **natural transformations**[5].

Definition 3. A natural transformation λ is a family of morphisms $\{\lambda_A\}_{A \in \mathbf{A}} : \mathfrak{f}(A) \to \mathfrak{g}(A)$ s.t. for objects $A, A' \in \mathbf{A}$ and $a : A \to A'$ the square on the right commutes

$$\begin{tikzcd}
\mathfrak{f} \arrow[dd, "\lambda"'] & & & \mathfrak{f}(A) \arrow[r, "\mathfrak{f}(a)"] \arrow[dd, "\lambda_A"'] & \mathfrak{f}(A') \arrow[dd, "\lambda_{A'}"'] \\
& & A \arrow[ur, mapsto, "\mathfrak{f}"] \arrow[dr, mapsto, "\mathfrak{g}"'] & & \\
\mathfrak{g} & & & \mathfrak{g}(A) \arrow[r, "\mathfrak{g}(a)"] & \mathfrak{g}(A')
\end{tikzcd}$$

That is that

(NAT1) $\forall A, A' \in \mathbf{A} : \mathfrak{g}(a) \circ \lambda_A = \lambda_{A'} \circ \mathfrak{f}(a)$

Similarly to functors, the conditions imposed on natural transformations (namely naturality) ensures that natural transformations preserve the structure of functors, and that compositions of natural transformations are, again, natural transformations.

[5]Here, we will denote a natural transformation by using lower case Greek letters.

1.5 Hom-Set and Hom-Functor

CT was originally developed by S. Eilenberg and S. Mac Lane in their study of algebraic topology. Functors arise naturally in the study of homology and cohomology theories[6]. One should note that their original goal was to understand natural transformations and functors and that "the whole concept of a category is essentially an auxiliary one" [**?**]. The idea of studying structures through the structure preserving maps between them - the morphisms - was already pursued by E. Noether in her study of rings in the beginning of the 20th Century.
It thus seems natural to consider the following kinds of sets:

Definition 4. The **hom-set** $\hom_{\mathbf{A}}(A, B)$ of objects A, B is a set, the elements of which are the morphisms $f_i \in \mathbf{A} : A \to B$.

Other notations include $\hom(A, B)$, $\mathrm{Mor}(A, B)$, $[A, B]_{\mathbf{A}}$, $\mathbf{A}(A, B)$, $\mathfrak{h}^A(B)$, and $\mathfrak{h}^B(A)$. If it is unclear which morphisms we want to consider, the specification as subscript can be useful. If the context is clear however, we will suppress this subscript.
For example, it makes a difference whether we consider the set $\hom_{\Bbbk-\mathbf{Sch}}(A, B)$ or $\hom_{\mathbf{Sch}}(A, B)$. A priori it is not clear that $\hom(A, B)$ is a set and not a proper class - and in general it is not[7]. However, for most relevant categories it is. Categories $\mathbf{C}$ s.t. for each $A, B \in \mathbf{C}$ we have $\hom(A, B)$ a set are called **locally small**. For the rest of the talk we will only consider such categories.
The family of objects of the functor category $[\mathbf{A}, \mathbf{B}]$ may be viewed as a hom set in the category **Cat**.
Consider the assignment $(A, B) \mapsto \hom_{\mathbf{A}}(A, B)$ for objects in a category $\mathbf{A}$. It can be extended to a set valued functor which entails information about any two objects in $\mathbf{A}$. So in order to study an object X in $\mathbf{A}$, we can consider the (contra-variant[8]) **hom-functor**

$$\mathfrak{h}_X : \mathbf{C}^{op} \to \mathbf{Set} \tag{7}$$
$$A \mapsto \mathfrak{h}_X(A) = \hom_{\mathbf{A}}(A, X) \tag{8}$$

On morphisms $f : A \to B$ in the category $\mathbf{A}$ this gives

$$\begin{tikzcd}
A \arrow[r, "f"] \arrow[d, "\mathfrak{h}_X"'] & B \arrow[d, "\mathfrak{h}_X"'] \\
\hom(A, X) \arrow[r, "\mathfrak{h}_X(f)"] & \hom(B, X)
\end{tikzcd}
\qquad
\begin{tikzcd}
A \arrow[r, "g"] \arrow[dr, "f"'] & X \\
 & B \arrow[u, "\mathfrak{h}_X(f)(g)"']
\end{tikzcd}$$

So $\mathfrak{h}^X(f) : g \mapsto g \circ f$.

Definition 5. Let $\mathfrak{f} : \mathbf{C} \to \mathbf{D}$ be a functor. For any two objects $X, Y \in \mathbf{C}$ this functor induces the functions

$$f_{X,Y} : \hom_{\mathbf{C}}(X, Y) \to \hom_{\mathbf{D}}(\mathfrak{f}(X), \mathfrak{f}(Y)) \quad . \tag{9}$$

Then the functor $\mathfrak{f}$ is called

[6] In the sense of Eilenberg-Steenrod, an ordinary/generalized homology/cohomology theory is a functor satisfying some axioms.
[7] This is relevant in the study of localizations of categories and model categories.
[8] A covariant version of the hom-functor exists of course. It is obtained by reversing all the arrows in the definition. We will later need the contra-variant version though.

- faithful, if $f_{X,Y}$ is injective
- full, if $f_{X,Y}$ is surjective
- fully faithful, if $f_{X,Y}$ is bijective

Definition 6. A **concrete category** is a pair $(\mathbf{C}, \mathcal{F})$ of a category $\mathbf{C}$ and a faithful functor $\mathcal{F} : \mathbf{C} \to \mathbf{Set}$.

Most well behaved categories are concrete. Examples include **Set**, **Grp**, **Top** and **Par**.
We can think of this functor as assigning to an object its underlying set. It is often referred to as *a* forgetful functor, as it forgets part of the structure. Note that "forgetful functor" is an informal notion. For example, both the functor $\mathbf{CRing} \to \mathbf{Ab}$ and $\mathbf{CRing} \to \mathbf{Set}$ are forgetful. They assign to a ring its additive group and underlying set, respectively.

2 Points on Schemes

Arguments, similar to the ones in this section can be found in [Mumford, 1999], and [Görtz und Wedhorn, 2010].

2.1 Motivation

Consider the following examples:

Example 1. Consider the category of sets **Set**, and let $z \in \mathbf{Set}$ be a singleton (which is of course unique up to isomorphism). Then for any $S \in \mathbf{Set}$

$$\mathfrak{h}^z(S) = \hom(z, S) \simeq \mathcal{F}(S) \tag{10}$$

where $\mathcal{F}$ is the forgetful functor from before.

Example 2. Consider the category of differentiable manifolds **Man**, and let $z \in \mathbf{Man}$ be the manifold consisting of only one point. Then for any $M \in \mathbf{Man}$

$$\mathfrak{h}^z(M) = \hom(z, M) \simeq \mathcal{F}(M) \tag{11}$$

The iso on the right assigns to a morphisms its image.

Example 3. Consider the category of groups **Grp**, and let $\mathbb{Z} \in \mathbf{Grp}$ be the additive group of the integers. Then for any free group $G \in \mathbf{Grp}$

$$\mathfrak{h}^{\mathbb{Z}}(G) = \hom(\mathbb{Z}, G) \simeq \mathcal{F}(G) \tag{12}$$

The iso on the right assigns to a homomorphism the image of $1 \in \mathbb{Z}$.

Example 4. Consider the category of commutative rings with identity $\mathbf{CRing}_1$, and let $\mathbb{Z}[x] \in \mathbf{CRing}_1$ be the polynomial ring in one variable x. Then for any $R \in \mathbf{CRing}_1$

$$\mathfrak{h}^{\mathbb{Z}[x]}(R) = \hom(\mathbb{Z}[x], R) \simeq \mathcal{F}(R) \tag{13}$$

The iso on the right assigns to a homomorphism the image of $x \in \mathbb{Z}[x]$.

In all of the above cases, the action of the forgetful functor $\mathcal{F}$ was equal to that of the hom-functor $\mathfrak{h}^z$ for some $z \in \mathbf{C}$. This is what is called a **representation of the forgetful functor**.
They satisfy our condition of injectivity. Arguably, this is most clearly illustrated by example 1. For some $z \in \mathbf{C}$ we want a morphism $f : X \to Y$ to uniquely determine a morphism $\overline{f} : \mathfrak{r}(X) \to \mathfrak{r}(Y)$. In other words, we want the functor $\mathfrak{r} : \mathbf{C} \to \mathbf{Set}$ to be faithful - i.e. we require

$$\forall X, Y : \hom_{\mathbf{C}}(X, Y) \rightarrowtail \hom_{\mathbf{Set}}(\mathfrak{r}(X), \mathfrak{r}(Y)) \tag{14}$$

to be an injection.
In general, such a transition of data by an $\mathfrak{h}^z$ with an $z \in \mathbf{C}$ is not possible for schemes.

2.2 Definition

So in conclusion, characterizing schemes by a set of morphisms from/to *a single* scheme is not enough information.
It was Grothendieck's idea, to not consider a single $z \in \mathbf{Sch}$, and hence $\hom(z, X)$ as a characterization of X, but to consider

$$\{\hom(z, X)\}_{z \in \mathbf{Sch}} \; . \tag{15}$$

This results in the definition to give a scheme X as a functor

$$\mathfrak{h}_X : \mathbf{Sch}^{op} \to \mathbf{Set} \tag{16}$$

$$A \mapsto \hom(A, X) \tag{17}$$

$$(f : A \to B) \mapsto (\hom(f, X) : \hom(A, X) \to \hom(B, X); g \mapsto g \circ f) \tag{18}$$

2.3 Reservations about the Definition

We have just added huge amounts of information to the previous notion of points (as morphisms into the object). It is (at least) reasonable to ask whether that is sensible. Usually the problem with such an approach becomes apparent in one of two ways[9]

- Loss of efficiency. For example, brute force approaches in algorithmic are intuitive, but they use "the maximum" of available data. As a trade-off they are very inefficient. Take for example the simplex algorithm: the set of solutions to a system of linear inequalities forms a (possibly unbounded) convex polytope, where it is our goal to find the optimal solution among these points. The brute force approach would be to just check every point (using "all" the data). However, in order to find the optimal solution we only need to consider vertexes of the polygon. Furthermore, with the algorithm, we choose one vertex and then move from one to another. This could mean the difference between checking less than 10 points to uncountably many.

  Slogan: "We don't want data that is known to be irrelevant."

- Too much data to make meaningful statements. The prime example for this is the discrete topological space $(X, \tau = \wp(X))$. Here, every function into any other topological space is continuous, defeating the very purpose of defining topological spaces in the first place.

[9] There may be better analogies to what is going on. The general idea, however, is that there *are some* reservations about this.

Another, very similar example is a measurable space with the σ-algebra being equal to $\wp(X)$.

Slogan: "We want little enough data to make distinctions."

In the following part, we will see that our choice of data gives just the right amount. The proof for this is the Yoneda Lemma.

3 Yoneda Lemma

This section is largely based on section 30.6 [Herrlich und Strecker, 1973]. Görtz & Wedhorn cover the Yoneda Lemma (less thoroughly) in section (4.2).

As always in CT, there are two versions of this lemma: a co-variant and a contra-variant version. For our application on schemes we will need the contra-variant version, which is why we will now consider this version.

Theorem 1. *Let $\mathfrak{h}_X, \mathfrak{f} : (\mathbf{C})^{op} \to \mathbf{Set}$ be contra-variant functors, where $\mathfrak{h}^X$ is representable. Then there exists a bijection*

$$\Gamma : \mathit{Nat}_{[(\mathbf{C})^{op},\mathbf{Set}]}(\mathfrak{h}_X, \mathfrak{f}) \xrightarrow{\sim} \mathfrak{f}(X) \quad . \tag{19}$$

that is natural in both X and $\mathfrak{f}$. Its inverse is given by Γ'.

The proof will be done by showing that the mappings

$$\begin{aligned} \Gamma : \lambda &\mapsto \lambda_X(1_X) && (20) \\ \{\rho_A\}_{A \in \mathbf{C}} = \rho &\leftarrow\!\shortmid x : \Gamma' && (21) \end{aligned}$$

with the property that $\rho_A(f) = (\mathfrak{f}(f))(x)$ are inverse to each other.
Firstly though, we will make two checks of well-definition.
Consider therefore the following two remarks:

Remark 1. $\lambda_X(1_X) \in \mathfrak{f}(X)$.

Proof. λ_X is a mapping $\hom(X, X) \to \mathfrak{f}(X)$. Hence $\lambda_X(1_X)$ is an element of $\mathfrak{f}(X)$. □

Remark 2. ρ is a natural transformation: $\mathfrak{h}_X \to \mathfrak{f}$.

Proof. Hence, we need to show that that for every $g : A \to X$ the following diagram commutes

$$\begin{tikzcd} A \arrow[d, "f"'] & \hom(A,X) \arrow[r, "\hom(f,X)"] \arrow[d, "\rho_A"'] & \hom(B,X) \arrow[d, "\rho_B"'] \\ B & \mathfrak{f}(A) \arrow[r, "\mathfrak{f}(f)"] & \mathfrak{f}(B) \end{tikzcd}$$

That is:

$$
\begin{aligned}
(\rho_B \circ \hom(f, X))(g) &= \rho_B(g \circ f) && (22)\\
&= \mathfrak{f}(g \circ f)(x) && (23)\\
&= (\mathfrak{f}(f) \circ \mathfrak{f}(g))(x) && (24)\\
&= \mathfrak{f}(f)(\rho_A(g)) && (25)\\
&= (\mathfrak{f}(f) \circ \rho_A)(g) && (26)
\end{aligned}
$$

Hence, ρ is natural. □

3.1 Isomorphism

Lemma 1. $\Gamma \circ \Gamma' = 1_{\mathfrak{f}(X)}$.

Proof. For an arbitrary $x \in \mathfrak{f}(X)$, we have

$$
\begin{aligned}
(\Gamma \circ \Gamma')(x) &= \Gamma(\rho) && (27)\\
&= \rho_X(1_X) && (28)\\
&= \mathfrak{f}(1_X)(x) && (29)\\
&= 1_{\mathfrak{f}(X)}(x) && (30)
\end{aligned}
$$

Since we chose x arbitrarily, $\Gamma \circ \Gamma' = 1_{\mathfrak{f}(X)}$. □

Lemma 2. $\Gamma' \circ \Gamma = 1_{\mathtt{Nat}(\mathfrak{h}_X, \mathfrak{f})}$.

Proof. Let λ be a natural transformation $\mathfrak{h}_X \to \mathfrak{f}(X)$. Then we have

$$\Gamma(\lambda) = \lambda_X(1_X) \qquad (31)$$

Now let $\delta \in \mathtt{Nat}(\mathfrak{h}_X, \mathfrak{f})$ and define $\delta := \Gamma'(\lambda_X(1_X))$. We now need to show that $\delta = \lambda$ i.e. that for an arbitrary element $A \in \mathbf{C} : \delta_A = \lambda_A$.

Firstly we will show this for $A = X$, as we will need it in the proof of the general case. By definition $\rho_A(f) = (\mathfrak{f}(f))(x)$ we have that

$$\lambda_X(1_X) = 1_{\mathfrak{f}(X)}(\lambda_X(1_X)) = \mathfrak{f}(1_X)(\lambda_X(1_X)) = \rho_X(1_X) \quad . \qquad (32)$$

Now for the general case:

Let $A \in \mathbf{C}$ and $f : A \to X$. Then

$$\rho_A(f) = \rho_A(1_X \circ f) = (\rho_A \circ \hom(f, X))(1_X) \qquad (33)$$

Now by apply the naturality of ρ (and λ) and consider the special case from before to obtain

$$(\rho_A \circ \hom(f, X))(1_X) = (\mathfrak{f}(f) \circ \rho_X)(1_X) = (\mathfrak{f}(f) \circ \lambda_X)(1_X) = (\lambda_A \circ \hom(f, X))(1_X) \qquad (34)$$

Now since λ is also a natural transformation we can go the way back (read right to left)

$$\lambda_A(f) = \lambda_A(1_X \circ f) = (\lambda_A \circ \hom(f, X))(1_X) \quad . \qquad (35)$$

This is what we wanted to show. Hence $\rho = \lambda$ and $\Gamma' \circ \Gamma = 1_{\mathtt{Nat}(\mathfrak{h}_X, \mathfrak{f})}$.

□

3.2 Naturality

Finally, we need to show that if we consider $\mathtt{Nat}_{[(\mathbf{Sch})^{op},\mathbf{Set}]}(\mathfrak{h}_X, \mathfrak{f})$ and $\mathfrak{f}$ as functors: $\mathbf{Set}^{op} \times \mathbf{C} \to \mathbf{Set}$, the natural transformation Ξ needs to be natural in both X and $\mathfrak{f}$. This means that the following two diagrams have to commute:

$$\begin{tikzcd}
\mathtt{Nat}(\mathfrak{h}_X, \mathfrak{f}) \arrow[r, "\mathtt{Nat}(\mathfrak{h}^f, \mathfrak{f})"] \arrow[d, "\Gamma_X"'] & \mathtt{Nat}(\mathfrak{h}^{X'}, \mathfrak{f}) \arrow[d, "\Gamma_{X'}"'] \\
\mathfrak{f}(X) \arrow[r, "\mathfrak{f}(f)"] & \mathfrak{f}(X')
\end{tikzcd}$$

Naturality in X

$$\begin{tikzcd}
\mathtt{Nat}(\mathfrak{h}_X, \mathfrak{f}) \arrow[r, "\mathtt{Nat}(\mathfrak{h}_X, \mu)"] \arrow[d, "\Gamma_{\mathfrak{f}}"'] & \mathtt{Nat}(\mathfrak{h}_X, \mathfrak{g}) \arrow[d, "\Gamma_{\mathfrak{g}}"'] \\
\mathfrak{f}(X) \arrow[r, "\mu_X"] & \mathfrak{g}(X)
\end{tikzcd}$$

Naturality in $\mathfrak{g}$

We will not show this here, but rather reference the (extensive) proof given by Drew Armstrong.

3.3 Conclusion

This is precisely what we wanted:
The injectivity ensures that we can meaningfully transition between schemes and no data was lost; the surjectivity ensures that no new data is introduced.
Consider the case where $Y \in \mathbf{Sch}$ and $\mathfrak{f} = \mathfrak{h}_Y$. Then by the Yoneda Lemma we have the bijection

$$\mathtt{Nat}(\mathfrak{h}_X, \mathfrak{h}_Y) \cong \hom(Y, X) \tag{36}$$

meaning the functor $\mathfrak{h}_X$ is faithful (and full) by definition 5. It is precisely this faithfulness we desired for the characterization to be useful. The surjectivity on the other hand entails that no information is lost in the process of expanding the characterization to an entire functor.

4 Fiber Products (Pullbacks)

We start this section by the simple observation that the set theoretic product does not make sense in the setting of schemes. Consider affine schemes: Naively, one may define the product of a locally ringed space as the cartesian product $A_1 \times A_2$ of the underlying sets of the underlying topological spaces (A_1, τ_1), (A_2, τ_2), endowed with the product topology and let that product, then, induce the product on sheaves. However, this approach fails, since the Zariski topology on $A_1 \times A_2$ is different than the product topology of τ_1, τ_2. Hence, we will have to consider different notions of product for schemes.

Before we define the fiber product, let us briefly consider the regular, categorical product in a category $\mathbf{C}$.

Definition 7. A product $A_1 \times A_2$ (or $A_1 \coprod A_2$) of two objects $A_1, A_2 \in \mathbf{C}$ is a triple $(A_1 \times A_2, \pi :$ $1, \pi_2)$ s.t. the following diagram commutes and $A_1 \times A_2$ has the universal property

$$\begin{tikzcd}
P' \arrow[dr, dashed, "\exists!"] \arrow[drr, "v"] \arrow[ddr, "u"'] & & \\
& A_1 \coprod A_2 \arrow[r, "\pi_1"'] \arrow[d, "\pi_1"] & A_1 \\
& A_2 &
\end{tikzcd}
\qquad
\begin{tikzcd}
& A_1 \arrow[d, "\iota_1"'] \arrow[ddr, "v"] & \\
A_2 \arrow[r, "\iota_2"] \arrow[drr, "u"'] & A_1 \coprod A_2 \arrow[dr, dashed, "\exists!"'] & \\
& & P'
\end{tikzcd}$$

Product Co-Product

The dual notion: the coproduct is given in the right diagram

Definition 8. The *fiber product* $P = A_1 \times_{A_0} A_2$ of $A_1, A_2 \in \mathbf{C}$ with $f_1 : A_1 \to A_0$, $f_2 : A_2 \to A_0$ is an object in $\mathbf{C}$ together with two maps $p_1 : P \to A_1$, $p_2 : P \to A_2$ s.t. the following diagram commutes

$$\begin{tikzcd}
P' \arrow[dr, dashed, "\exists!"] \arrow[drr, "v"] \arrow[ddr, "u"'] & & \\
& P \arrow[r, "p_1"'] \arrow[d, "p_1"] & A_1 \arrow[d, "f_1"] \\
& A_2 \arrow[r, "f_2"'] & A_0
\end{tikzcd}
\qquad
\begin{tikzcd}
A_0 \arrow[r, "f_1"] \arrow[d, "f_1"'] & A_1 \arrow[d, "p_1"'] \arrow[ddr, "v"] & \\
A_2 \arrow[r, "p_2"] \arrow[drr, "u"'] & P \arrow[dr, dashed, "\exists!"] & \\
& & P'
\end{tikzcd}$$

Pullback Pushout

hence s.t. $f_1 \circ p_1 = p_2 \circ f_2$ and s.t. for every P', u, v factor uniquely through P. The diagram given on the right describes the dual notion: the pushout.

From their universal property it follows that pullbacks and pushouts are unqiue up to unique isomorphism, given that they exist - they do not exist in an arbitrary category though.
Now consider this in conjunction with the definition of a relative scheme:

Definition 9. A *relative scheme* (or S-scheme) is an object in the category $\mathbf{Sch/S}$ s.t.

$$\mathtt{Obj}(\mathbf{Sch/S}) := \{f : X \to S | X \in \mathbf{Sch}\} \tag{37}$$
$$\mathtt{Mor}(\mathbf{Sch/S}) := \{g : X \to X' | f = f' \circ g\} \tag{38}$$

In other words, s.t. the following diagram commutes:

$$\begin{tikzcd} X \arrow[rr, "g"] \arrow[dr, "f"'] & & X' \arrow[dl, "f'"] \\ & S & \end{tikzcd}$$

From this definition it is clear that the fiber product is the categorical product in the category $\mathbf{C}/A_0$. And it is also clear that if we form the slice category $\mathbf{C}/Z$ over a terminal object $Z \in \mathbf{C}$, we have the fiber product in that category equal to the product in the category $\mathbf{C}$. This is because by the definition of the terminal object we can identify the morphisms $X \to z$ with the "elements" of X.

We say that p_1 is the pullback of f_2 along f_1 and that p_2 is the pullback of f_1 along f_2.
Consider the situation for **Set** to gain some intuition for how products and co-products, and pullbacks and pushouts are different:

$$\begin{tikzcd} A_1 \prod A_2 \arrow[r, "\pi_1"'] \arrow[d, "\pi_1"] & A_1 \\ A_2 & \end{tikzcd} \qquad \begin{tikzcd} & A_1 \arrow[d, "\iota_1"'] \\ A_2 \arrow[r, "\iota_2"] & A_1 \coprod A_2 \end{tikzcd}$$

$$A_1 \prod A_2 := \{(a_1, a_2) \mid a_1 \in A_1, a_2 \in A_2\} \qquad A_1 \coprod A_2 := \{(a, i) \mid a \in A_i\}$$

$$\begin{tikzcd} A_1 \times_{A_0} A_2 \arrow[r, "p_1"'] \arrow[d, "p_2"] & A_1 \arrow[d, "f_1"] \\ A_2 \arrow[r, "f_2"'] & A_0 \end{tikzcd} \qquad \begin{tikzcd} A_0 \arrow[r, "f_1"] \arrow[d, "f_1"'] & A_1 \arrow[d, "p_1"'] \\ A_2 \arrow[r, "p_2"] & A_1 +_{A_0} A_2 \end{tikzcd}$$

$$A_1 \times_{A_0} A_2 := \{x \in A_1 \prod A_2 \mid f_1 \circ g_1(x) = f_2 \circ g_2(x)\}$$

$$A_1 +_{A_0} A_2 := (A_1 \coprod A_2)/\sim \quad \text{s.t.} \quad (a, 1) \sim (a, 2) :\Leftrightarrow f_1^{-1}((a, 1)) = f_2^{-1}((a, 2))$$

With some slight changes, the examples above can be considered in **Set**, **Grp**, or **Top**, too.
Sometimes, we denote the pullback of f_1 along p_2 by $p_2{}^* f_1$ and the pushout by $p_{2*} f_1$. This notation should look familiar, as we have defined the direct image $\mathcal{F}' = f_* \mathcal{F}$ of a sheave $\mathcal{F}$ on a topological space X as

$$\mathcal{F}' = f_* \mathcal{F} = \mathcal{F} \circ f^{-1} \tag{39}$$

i.e. the pushout of $\mathcal{F}$ along f^{-1}. With this spirit, we may consider the image of $\mathcal{F} \in \mathbf{Sh}(X)$ as $\mathbf{Set} \times_X Y$. The inverse of a sheave, on the other hand, is given as the sheafification of the pullback.

References

[nlab] : *nLab.* `https://ncatlab.org/nlab/show/HomePage`

[stacksproject] : *Stacks Project, Category Theory.* `https://stacks.math.columbia.edu/tag/0011`

[Görtz und Wedhorn 2010] Görtz, Ulrich ; Wedhorn, Torsten: *Algebraic Geometry.* Springer, 2010

[Herrlich und Strecker 1973] Herrlich, Horst ; Strecker, George: *Category Theory.* Allyn and Bacon, 1973

[Mac Lane und Eilenberg 2013] Mac Lane, Saunders ; Eilenberg, Samuel: *Categories for the Working Mathematician.* Bd. 5. Springer Science & Business Media, 2013

[Mumford 1999] Mumford, David: *The Red Book of Varieties and Schemes: includes the Michigan lectures (1974) on curves and their Jacobians.* Bd. 1358. Springer Science & Business Media, 1999