Categories and Quantum Informatics: Adjoint functors

Chris Heunen

Spring 2017

In this section we consider *adjoint functors.* Adjoint functors stand to functors between categories as dual objects stand to objects in monoidal categories. But adjoint functors are more general, and don't need any tensor product. As we will see, adjoint functors arise everywhere. They are very effective means of translating structure between categories. They provide an essential tool in your kit of spotting patterns when working with mathematical objects you are not familiar with (in terms of objects that are easier to handle). Whenever you see a functor, one of the first questions that should arise in your mind is whether it has an adjoint, because the answer nearly always brings important insight into your functor.

3.4 Adjoint functors

We will see define adjoint functors in two ways. The first definition is often easier to work with, and the second relates to dual objects more clearly. We start with the first, and later prove that it is equivalent to the second.

Definition 3.1 (Adjunction)**.** Let $\mathbf{C} \xrightarrow{F} \mathbf{D}$ and $\mathbf{D} \xrightarrow{G} \mathbf{C}$ be functors. An adjunction between F and G is a natural bijection

$$\mathbf{D}(F(A), B) \simeq \mathbf{C}(A, G(B)) \tag{3.38}$$

for objects A in $\mathbf{C}$ and B in $\mathbf{D}$. We say F is *left adjoint* to G, and G is *right adjoint* to F, and write $F \dashv G$.

Think of categories as countries for a minute, objects as citizens, and morphisms as people talking to each other in the country's language. We may then intuitively regard functors as translations from one language to another that preserve meaning. In this perspective, F and G are adjoint when it doesn't matter if person A travels to their friend B across the border and speaks in B's language, or whether they meet in A's home country and speak the local language there.

Examples of adjoint functors are numerous. We will now see a long list of them.

Example 3.2 (Free vector space)**.** Write $U\colon \mathbf{FVect} \to \mathbf{Set}$ for the functor that takes a vector space to its underlying set, forgetting that it had operations like addition, and that takes a linear map to its underlying function, forgetting that was linear. Write $F\colon \mathbf{Set} \to \mathbf{FVect}$ for the functor that takes a set B to the vector space of all formal linear combinations of elements of B:

$$F(B) = \{\varphi\colon B \to \mathbb{C} \mid \varphi(b) \neq 0 \text{ for only finitely many } b \in B\}.$$

Addition in $F(B)$ is given by $(\varphi + \psi)(b) = \varphi(b) + \psi(b)$, scalar multiplication by $(z \cdot \varphi(b)) = z \cdot \varphi(b)$, and the zero vector is $b \mapsto 0$. In other words, $F(B)$ is a vector space with basis B. Think about $\varphi \in F(B)$ as a 'formal' linear combination $\sum_{b \in B} \varphi(b) \cdot b$. For example,

$$F(\{1, \ldots, n\}) \simeq \mathbb{C}^n.$$

On functions $f\colon B \to C$, the functor acts as $F(f)\colon F(B) \to F(C)$ by $\varphi \mapsto \varphi \circ f$.

We will show that $F \dashv U$. To do this, we have to establish a one-to-one correspondence:

$$\frac{\text{linear functions } g\colon F(B) \to V}{\text{functions } f\colon B \to U(V)}$$

Given f, define g_f by $\varphi \mapsto \sum_{b \in B} \varphi(b) \cdot f(b)$. Thus $f \mapsto f_g$ evaluates a formal linear combination into an actual linear combination. Conversely, given g, define f_g by sending $b \in B$ to $g(\delta_b)$, where $\delta_b \in F(B)$ is the characteristic function $c \mapsto \delta_{b,c}$. Then $f_{g_f} = f$ and $g_{f_g} = g$, so these two constructions are each other's inverse. Finally, these assignments are natural.

Example 3.3 (Free monoid)**.** Write **Monoid** for the category of monoids and homomorphisms, and $U\colon \mathbf{Monoid} \to \mathbf{Set}$ for the functor that takes a monoid to its underlying set and a homomorphism to its underlying vector space. Write $F\colon \mathbf{Set} \to \mathbf{Monoid}$ for the functor that takes a set A to the set of all finite words made up of zero or more letters from A; this forms a monoid under concatenation of words, with the empty word $\emptyset$ being the identity. For example,

$$F(\{\bullet\}) \simeq \mathbb{N}, \qquad F(\{0,1\}) = \{\emptyset, 0, 1, 00, 01, 10, 11, 000, 001, \ldots\}.$$

On functions $f\colon A \to B$, the functor acts as $F(f)\colon F(A) \to F(B)$ by applying f to each letter of the word individually.

We will show that $F \dashv U$. To do this, we have to establish a one-to-one correspondence:

$$\frac{\text{monoid homomorphisms } g\colon F(A) \to M}{\text{functions } f\colon A \to U(M)}$$

Given f, define g_f by sending a word $a_1 \cdots a_n$ to $f(a_1) \cdot f(a_2) \cdots f(a_n)$, and the empty word to 1. Conversely, given g, define f_g by applying g to the one-letter word $a \in A$. Then $f_{g_f} = f$ and $g_{f_g} = g$, so these two constructions are each other's inverse. Finally, these assignments are natural.

Example 3.4 (Free category)**.** Write **Graph** for the category of directed graphs, whose morphisms are functions between vertex sets and between edge sets such that source and target of edges are preserved. Write **Cat** for the category of categories and functors, and $U\colon \mathbf{Cat} \to \mathbf{Graph}$ for the functor that sends a category to its underlying directed graph. Write $F\colon \mathbf{Graph} \to \mathbf{Cat}$ for the functor that takes a graph with vertex set V and edges $E \subseteq V^2$ to the following category: objects are vertices $v \in V$; morphisms as paths $v \xrightarrow{e_1} \cdots \xrightarrow{e_n} w$ of edges $e_i \in E$; composition is concatenation of paths; and the identity on v is the path $v \to v$ of length 0. On a graph morphism $f\colon (V,E) \to (V',E')$, the functor F acts as $F(f)\colon F(V,E) \to F(V',E')$ by sending objects v of $F(V,E)$ to $f(v)$, and sending a morphism $v \xrightarrow{e_1} \cdots \xrightarrow{e_n} w$ of $F(V,E)$ to the morphism $f(v) \xrightarrow{f(e_n)} \cdots \xrightarrow{f(e_1)} f(w)$.

We will show that $F \dashv U$. To do this, we have to establish a one-to-one correspondence:

$$\frac{\text{functors } G\colon F(V,E) \to \mathbf{C}}{\text{graph morphisms } f\colon (V,E) \to U(\mathbf{C})}$$

Given f, define G_f as follows: send an object v to $f(v)$; and send a morphism $v \xrightarrow{e_1} \cdots \xrightarrow{e_n} w$ to the composition $f(e_n) \circ \cdots \circ f(e_1)$. Conversely, given G, define f_G as follows: send a vertex v to $G(v)$; and send an edge e to $G(e)$. Then $f_{G_f} = f$ and $G_{f_G} = G$, so these two constructions are each other's inverse. Finally, these assignments are natural.

Remark 3.5 (Free algebraic structures)**.** The above three examples are part of a large family of constructions of *free* structures of various kinds. Let **C** be some category of sets with some extra structure, and structure-preserving functions. Then there is always a *forgetful functor* $U\colon \mathbf{C} \to \mathbf{Set}$. If that functor has a left adjoint $F\colon \mathbf{Set} \to \mathbf{C}$, it is called the *free* functor. In many algebraic settings, this is important, because every object is a quotient of a free one.

For example, we could alter Example 3.3 to give *free groups*. Instead of taking words whose letters are elements of the given set A, take words whose letters are either $a \in A$ or a formal symbol a^{-1} for $a \in A$, and identify consecutive letters aa^{-1} and $a^{-1}a$ with the empty word. Any group G can be given by generators $A \subseteq G$ and relations, so any group is a quotient of the free group A.

Similarly, the *free abelian* group on a set A consists of words as in the free group, but now we additionally identify words if a permutation of their letters turns one into the other. In other words, the free abelian group consists of functions $\varphi\colon A \to \mathbb{Z}$ that are nonzero on only finitely many elements of A, almost as in Example 3.2.

However, in general this only works in for algebraic structures. There is no such thing as a free field on a set, because division is only defined for nonzero elements. Similarly, the forgetful functor $\mathbf{Hilb} \to \mathbf{Set}$ does not have a left adjoint.

Above we concentrated mostly on left adjoints. In general, these make an object of their domain into an object of their codomain by adding the least amount of structure necessary to do so. We will next focus on right adjoints, that take away the least amount of structure. But first we consider some mixed cases.

Example 3.6 (Adjoint functors versus adjoint matrices)**.** Let H be finite-dimensional Hilbert spaces. Write $\mathrm{Sub}(H)$ for the set of subspaces, which is partially ordered by inclusion. The inner product provides the additional structure of *orthocomplementation*: if $U \subseteq H$ is a subspace, then so is

$$U^\perp = \{v \in H \mid \forall u \in U\colon \langle u|v\rangle = 0\}.$$

Moreover, if $U \leq V$ in $\mathrm{Sub}(H)$, then $V^\perp \leq U^\perp$. If $f\colon H \to K$ is a linear function, then there is a monotone function $\mathrm{Sub}(f)\colon \mathrm{Sub}(H) \to \mathrm{Sub}(K)^{\mathrm{op}}$ given by $U \mapsto f(U)^\perp$.

Now $\mathrm{Sub}(f) \dashv \mathrm{Sub}(f^\dagger)$. That is, adjoint morphisms in $\mathbf{FHilb}$ induce adjoint functors. To show this, let $U \in \mathrm{Sub}(H)$ and $V \in \mathrm{Sub}(K)$, and observe

$$\begin{aligned} U \to \mathrm{Sub}(f^\dagger)(V) \text{ in } \mathrm{Sub}(H) &\iff U \subseteq f^\dagger(V)^\perp \\ &\iff \forall u \in U, v \in V\colon \langle u|f^\dagger(v)\rangle = 0 \\ &\iff \forall u \in U, v \in V\colon \langle f(u)|v\rangle = 0 \\ &\iff V \subseteq \mathrm{Sub}(f)(U) \\ &\iff \mathrm{Sub}(f)(U) \to V \text{ in } \mathrm{Sub}(K)^{\mathrm{op}}. \end{aligned}$$

Conversely, one can show that if $\mathrm{Sub}(f)$ and $\mathrm{Sub}(g)$ are adjoint functors then if $zf = g^\dagger$ for a scalar $z \in \mathbb{C}$.

Example 3.7 (Quantifiers)**.** Let A and B be sets, and write $p\colon A \times B \to A$ for the projection $(a,b) \mapsto a$. Write $\mathcal{P}(A)$ for the powerset of A, partially ordered by inclusion, regarded as a category. Define a functor $p^*\colon \mathcal{P}(A) \to \mathcal{P}(A \times B)$ by sending $V \subseteq B$ to $p^{-1}(V) = \{(a,b) \in A \times B \mid a \in V\}$. This functor has a left adjoint

$$\begin{aligned} \exists\colon \mathcal{P}(A \times B) &\to \mathcal{P}(A) \\ U &\mapsto \{a \in A \mid \exists b \in B\colon (a,b) \in U\} \end{aligned}$$

because for $U \subseteq A \times B$ and $V \subseteq A$,

$$\exists(U) \subseteq V \iff \forall a \in A\big(\exists b \in B((a,b) \in U) \Rightarrow a \in V\big) \iff \forall (a,b) \in U\colon a \in V \iff U \subseteq p^*(V).$$

The functor p^* also has a right adjoint

$$\begin{aligned} \forall\colon \mathcal{P}(A \times B) &\to \mathcal{P}(A) \\ U &\mapsto \{a \in A \mid \forall b \in B\colon (a,b) \in U\} \end{aligned}$$

for similar reasons.

Example 3.8 (Terminal object)**.** Let $\mathbf{C}$ be any category. Write $\mathbf{1}$ for the category with one morphism (the identity object on one object). There is a unique functor $\mathbf{C} \to \mathbf{1}$. In the other direction, a functor $\mathbf{1} \to \mathbf{C}$ is just a choice of object of $\mathbf{C}$. Right adjoint functors to $\mathbf{C} \to \mathbf{1}$ are precisely terminal objects in $\mathbf{C}$.

Lemma 3.9 (Adjoint functors compose)**.** *If* $\mathbf{C} \xrightarrow{F} \mathbf{D} \xrightarrow{F'} \mathbf{E}$ *and* $\mathbf{E} \xrightarrow{G'} \mathbf{D} \xrightarrow{G} \mathbf{C}$ *are functors, and* $F \dashv G$ *and* $F' \dashv G'$*, then* $F' \circ F \dashv G' \circ G'$*.*

Proof. Write $\alpha_{C,D}$ for the natural bijection $\mathbf{D}(F(C), D) \to \mathbf{C}(C, G(D))$, and $\beta_{D,E}$ for the natural bijection $\mathbf{E}(F'(D), E) \to \mathbf{D}(D, G'(E))$. Then $\alpha_{C,G'(E)} \circ \beta_{F(C),E}$ is a natural bijection $\mathbf{E}(F' \circ F(C), E) \to \mathbf{C}(C, G \circ G'(E))$. $\square$

Definition 3.10 (Adjunctions via units)**.** Let $\mathbf{C} \xrightarrow{F} \mathbf{D}$ and $\mathbf{D} \xrightarrow{G} \mathbf{C}$ be functors. An adjunction $F \dashv G$ consists of natural transformations

$$\eta_C \colon C \to G(F(C)) \qquad\qquad \varepsilon_D \colon F(G(D)) \to D$$

that make the following diagrams commute:

$$\begin{tikzcd}
F(C) \arrow[r, "F(\eta_C)"] \arrow[dr, "\mathrm{id}_{F(C)}"'] & F(G(F(C))) \arrow[d, "\varepsilon_{F(C)}"] \\
& F(C)
\end{tikzcd}
\qquad
\begin{tikzcd}
G(D) \arrow[r, "\eta_{G(D)}"] \arrow[dr, "\mathrm{id}_{G(D)}"'] & G(F(G(D))) \arrow[d, "G(\varepsilon_D)"] \\
& G(D)
\end{tikzcd}
\tag{3.39}$$

We call η the *unit* of the adjunction, and ε the *counit*. Compare this to the snake equations (3.5).

Proposition 3.11. *Functors* $\mathbf{C} \xrightarrow{F} \mathbf{D}$ *and* $\mathbf{D} \xrightarrow{G} \mathbf{C}$ *are adjoint according to Definition 3.1 if and only if they are adjoint according to Definition 3.10.*

Proof. First let $\alpha_{C,D} \colon \mathbf{D}(F(C), D) \to \mathbf{C}(C, G(D))$ be a natural bijection. Set

$$\eta_C = \alpha_{C,F(C)}(\mathrm{id}_{F(C)}), \qquad \varepsilon_D = \alpha^{-1}_{G(D),D}(\mathrm{id}_{G(D)}).$$

The equations (3.39) follow directly from naturality of α.

Conversely, let $\eta_C \colon C \to G(F(C))$ and $\varepsilon_D \colon F(G(D)) \to D$ be natural transformations satisfying (3.39). Define $\alpha_{C,D}(f) = G(f) \circ \eta_C$. Its inverse is $\alpha^{-1}_{C,D}(g) = \varepsilon_D \circ F(f)$, and both are natural. $\square$

Remark 3.12. In this case of forgetful functors as in Remark 3.5, another way to say that there is a left adjoint is by the following universal property. For every set A there is a function $\eta \colon A \to F(A)$ such that, if $f \colon A \to C$ is any other function to an object of $\mathbf{C}$, then there is a unique morphism $\hat{f} \colon F(A) \to C$ with $f = \hat{f} \circ \eta$.

$$\begin{tikzcd}
A \arrow[r, "\text{function } \eta"] \arrow[dr, "\text{function } f"'] & F(A) \arrow[d, dashed, "\text{morphism } \hat{f}"] \\
& C
\end{tikzcd}$$

In general, $F(A)$ is *much* larger than A. For example, the free vector space on a finite set of Example 3.2 is uncountable; and the free monoid on a finite set of Example 3.3 is countably infinite.

Example 3.13 (Function spaces)**.** Let B be a set, and consider the functor $L \colon \mathbf{Set} \to \mathbf{Set}$ given by $L(A) = A \times B$. Write $R \colon \mathbf{Set} \to \mathbf{Set}$ for the functor $R(C) = C^B$, the set of functions from B to C.

On morphisms, $R(f)$ sends $g\colon B \to C$ to $f \circ g$. We will show that $L \dashv R$. To do this, we have to establish a one-to-one-correspondence:

$$\frac{\text{functions } f\colon A \times B \to C}{\text{functions } g\colon A \to C^B}$$

Given g, define f_g by $(a,b) \mapsto g(a)(b)$. Conversely, given f, define g_f by $g_f(a)(b) = g(a,b)$. These two assignments are natural and each other's inverse. Passing from f to g_f is called *currying*, and passing from g to f_g is called *uncurrying*. It is a staple of functional programming.

The previous example makes sense not just for sets, but in any category with products.

Example 3.14 (Heyting algebra)**.** Let $(P, \leq)$ be a partially ordered set. Suppose that P, regarded as a category, has products. That is, P is a *meet-semilattice*: every two elements $p, q \in P$ have a greatest lower bound $p \wedge q$. An exponential $q \multimap r$ in P would satisfy

$$p \wedge q \leq r \qquad \Longleftrightarrow \qquad p \leq q \multimap r.$$

If P also had coproducts, i.e. least upper bounds, it is called a lattice. Lattices that are closed as categories are called *Heyting algebras*. They play a major role in logic. Think about the elements of P as logical propositions, and of the order $p \leq q$ as saying that p implies q. Then the adjunction formula above says that

deriving conclusion r from hypotheses p and q

is the same as

deriving from hypothesis p that q implies r.

Even for monoidal categories Example 3.13 is useful.

Definition 3.15 (Exponentials)**.** A monoidal category $\mathbf{C}$ is *closed* when the functor

$$\begin{aligned} - \otimes B\colon \mathbf{C} &\to \mathbf{C} \\ A &\mapsto A \otimes B \\ f &\mapsto f \otimes \mathrm{id}_B \end{aligned}$$

has a right adjoint $R_B\colon \mathbf{C} \to \mathbf{C}$ for each object B. The object $R_B(C)$ is called an *exponential*, and sometimes written as $B \multimap C$ or C^B.

This can equivalently be stated as a universal property: for all objects B and C there is an object $B \multimap C$ and a natural transformation $\varepsilon_{X,Y}\colon X \otimes (X \multimap Y) \to Y$, such that any morphism $f\colon X \otimes Z \to Y$ factors through $\varepsilon_{X,Y}$ via a unique morphism $\hat{f}\colon Z \to (X \multimap Y)$.

$$\begin{tikzcd}
X \otimes Z \arrow[d, dashed, "\mathrm{id}_X \otimes \hat{f}"'] \arrow[dr, "f"] & \\
X \otimes (X \multimap Y) \arrow[r, "\varepsilon_{X,Y}"'] & Y
\end{tikzcd}$$

We also call $\varepsilon_{X,Y}$ the *evaluation* maps.

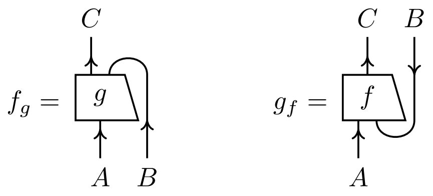
Remark 3.16 (Recursion)**.** Let $\mathbf{C}$ be a closed monoidal category. Think of objects as data types, and morphisms as functions in a programming language. The fact that the category is closed means that the programming language supports *higher order* functions: we may pass functions as arguments to other functions. Thus we may think of the specification of a recursive function $X \to Y$ as a morphism $s\colon (X \multimap Y) \to (X \multimap Y)$. A state $f\colon I \to (X \multimap Y)$ satisfying $s \circ f = f$ is called a *fixed point* of this specification. It corresponds to a morphism $X \to Y$ that meets the specification. So if we could somehow pick a smallest fixed point systematically, our category would provide semantics for general *recursion*.

Lemma 3.17. *Any compact category is closed.*

Proof. Define $B \multimap C$ to be $C \otimes B^*$. We have to establish a natural bijection:

$$\frac{f\colon A \otimes B \to C}{\overline{g\colon A \to C \otimes B^*}}$$

Take



Then $f_{g_f} = f$ and $g_{f_g} = g$ by the snake equations, and these assignments are natural. $\square$