Reprogramming Cells

Our Friends, the Microorganisms

Biologists have come up with a lot of clever ways to sort and analyze cellular processes, but even gene chips and mass spectroscopy are basically still just smashing the computer with a hammer and picking through the fragments. You *can* learn a lot that way, but wouldn't you rather have some tools that were actually made for the job?

Well, you can—if you're willing to outsource. Living cells are jam-packed with all sorts of molecular machinery, and over the years we've found many ways to make it work for us.

Restriction Enzymes and Restriction-Methylase Systems

Viruses, as we mentioned in an earlier chapter, hijack a cell by inserting a copy of their own DNA into the host's genome—essentially "tricking" it into making more viruses. Given that cells would much rather that *not* happen (if nothing else, it's a huge waste of resources), evolution has equipped them with a few ways to resist.

One of their most effective types of antiviral weaponry is the **restriction-methylase (R-M) system**. And despite the scientific-sounding name, it's the same basic idea as hiring a bunch of big guys with axes to kill anyone who wanders in without a badge.

To identify its own DNA, cells attach some sort of molecular marker directly to the strands. Typically, this "badge" is a methyl group (one carbon

W. W. Cohen, C. K. Cohen, *A Computer Scientist's Guide to Cell Biology*,
https://doi.org/10.1007/978-3-031-55907-5_6

and three hydrogen molecules); the protein that actually attaches it to the nucleotides, then, is called a **methylase.**

To kill off intruders, cells also have protein complexes called **restriction endonucleases (REs)**. Like a guard checking for ID cards, when these enzymes stumble across a piece of DNA that doesn't have the proper methyl "badge," they act quickly to destroy it.[1] More specifically, they look for a specific sequence of nucleotides, such as "GAATTC" (along with the complementary "CTTAAG" on the other strand) where they can attach themselves to the DNA strand and chop through the sugar-phosphate "backbone" to create a pair of newly shortened (but still double-stranded) nucleic acids. (Remarkably, both the initial binding and the resulting cleavage require no external energy sources.)

As with everything in biology, REs are a lot more complicated in practice.

Luckily, we don't need to understand everything about them to make use of them. You can think of a RE as a sort of black box, a complicated software module that you can drop into a program wholesale as long as you understand its "interface." DNA fingerprinting, for example, doesn't actually care *how* the RE chops up the sample, only that it does so in a consistent way.

And with just a little more knowledge of that "interface" (i.e., what sequence the RE acts on), you can begin synthesizing entirely new DNA molecules by cutting and pasting together strands of existing DNA.

A note on nomenclature: "restriction endonuclease" may be quite a mouthful, but like many terms in biology, there's a twisted sort of logic to it.

- The suffix "-ase" indicates that you're talking about an enzyme.
- The rest of an enzyme's name is usually the substrate it interacts with. REs make changes to nucleic acids—hence, "nuclease."
- "Endo-" and "exo-" are suffixes that mean "within" and "without," respectively. REs attack the middles of nucleic acids—hence, "endonuclease."

You can probably guess where "restriction" comes from.

By convention, specific endonucleases are identified by four letters—the first three identifying the organism where it originated, and the fourth the exact "strain"—and a Roman numeral. The second RE found in strain **d** of the **Haemophilus influenzae** bacterium, for example, is known as HindII (pronounced "hin dee two").

[1] The DNA, not the poor employee that forgot their badge.

Constructing Recombinant DNA with REs and DNA Ligase

Strangely, it turns out that the second half of that operation—the "pasting"—also depends on the actions of restriction endonucleases. Specifically, it hinges on *how* the enzymes make their cuts.

REs, you see, don't make perfectly clean cuts and leave behind perfectly symmetrical, double-stranded DNA strands. They actually make diagonal cuts, bisecting a handful of base pairs in the process and leaving each of the newly divided strands with a little tail of single-stranded DNA—the **sticky end**.

It's probably easier to imagine this visually. Take, for example, the *E. coli*–derived RE EcoRI shown in Table 1. The gray areas show one of the two fragments after the cut; the white areas the other.

Table 1 Small fragment of DNA before being cut by EcoRI

GATTACA	G	AATT	C	CATATTAC
CTAATGT	C	TTAA	G	GTATAATG

Our two sticky ends, then, are AATT and TTAA. You might notice that the two sequences are complementary. They match; surely the fragments will quickly "stick" and glue themselves back together, right?

Well…sort of. Using a restriction endonuclease will leave you with a *lot* of fragments, all of them with the same sticky ends; those ends might still be able to match, but who's to say what nucleotides came *before* the target site (see Table 2). If the original strand has multiple target sites, there's absolutely no guarantee that the fragments will glue themselves back together in the right order—or even wind up the same length.

Take a closer look at the white fragment in Table 2, and its specific sticky ends. They're the same on both sides, meaning that you could totally remove that particular fragment and still get a stable reconstruction. This kind of randomness means that letting a sample reassemble doesn't just give you the original strand—you're left with *all possible* versions of the reassembly process.

Consider two strands of DNA, *xSy* and *wSz*, where "S" is the sequence recognized by a restriction endonuclease and *w, x, y,* and *z* are all different

Table 2 A longer DNA fragment, showing how it is cut by an RE

GATTACA	G	AATT	C	ATTACCAT	G	AATT	C	CATATTAC
CTAATGT	C	TTAA	G	TAATGGTA	C	TTAA	G	GTATAATG

DNA sequences. If you treat the two strands with the RE and allow them to reassemble, you'll still have strands *xSy* and *wSz* in the final mixture, but you'll also have the brand-new (or **recombinant**) strands *xSz* and *wSy*.

CRISPR/Cas9

Useful as they are, restriction enzymes are still limited—there are only so many known examples, targeting only so many specific DNA sequences. Trying to make a cut in exactly the right place can be more like a logic puzzle than anything else, as frustrated biologists try to mix-and-match enzymes to isolate the sequences they're interested in.

What would really be useful, thousands of researchers have grumbled, would be some sort of "programmable" restriction enzyme, one that you can tell exactly when and where to cut.

And as luck would have it, such a thing actually exists. In one of the most important discoveries of the last 20 years, scientists discovered a sort of "modular" restriction enzyme known as **Cas9**. Unlike most such proteins, Cas9 doesn't bind directly to DNA. Instead, it delegates the attachment process to a strand of **guide RNA (gRNA).** The gRNA is responsible for determining where the greater Cas9 complex grabs and cuts, not the intricacies of protein structure—and it's much, *much* easier to synthesize custom gRNA than it is to design entirely new restriction enzymes.

Cas9 doesn't exist solely for the convenience of biologists, of course. Under normal conditions, it serves as part of a bacterial immune system, of sorts—a collection of restriction enzymes that break down viral RNA before it can infect the cell's own DNA. And just like the eukaryotic immune system, this system is capable of "learning" from past experience.

How does this "learning" process work when a viral DNA (or any foreign DNA) is inserted in a cell? Certain restriction enzymes chop up some of the invasive nucleic acid and produce short fragments known as **protospacers**. Proteins in the cell then take those fragments and add a short sequence called a **protospacer adjacent motif (PAM)** to each end—thus "labeling" the fragment as foreign, and creating a longer sequence called (you guessed it) a **spacer**. These spacers are then saved in "long-term memory" by being spliced into the genome in a particular region known as **CRISPR,** short for "**clustered regularly interspaced short palindromic repeats."** CRISPR is a long series of repeating patterns occasionally interrupted by one of these captured spacer sequences. The repetitive structure is recognizable enough that a single set of enzymes can "read" and transcribe any spacer, recognizing it as some

sort of foreign DNA—even though the exact contents of different spaces are different.

These spacer-containing sequences ultimately become the guide RNA we mentioned earlier. Once transcribed, they're trimmed down to shorter fragments and the rest of the Cas9 protein complex forms around them, like a pearl building up around a speck of dirt. Now the cell has a defense mechanism tuned to cut a particular kind of DNA—DNA from the original virus that was chopped up to form these spacers.

From the cell's point of view, this mechanism is a great defense against reinfection. And because the memory of CRISPR sequences are part of the bacterial genome, they'll be passed down to any daughter cells, so as long as some distant ancestor of a cell stumbled across a virus before the cell is protected.

But this mechanism is also highly convenient for biologists. Like we just saw, normal restriction enzymes are highly specific—they attach to one RNA sequence, and one sequence only, and modifying one to work with something else would be a hideously complicated process. Cas9 is a complex that does a similar thing, and it has already evolved to be modular, as it can accept many RNA sequences as gRNAs. So cutting DNA exactly where you like is as simple building a new Cas9 complex, which—in turn—is as simple as synthesizing a bit of RNA and dropping it in a test tube full of proteins.

From a computer scientist's perspective, this kind of ability is fascinating. If the broader CRISPR/Cas9 machinery is a sort of cellular function, the viral fragment from the spacer is essentially an argument to that function. You can change the DNA sequence—call a different argument—and the process will still function.

Inserting Foreign DNA into a Cell

Let's take a moment to think like computer scientists again. DNA is, basically, the language with which the cell is programmed—that's a common enough analogy, and it generally does work fairly well. By that logic, the genome is a set of rock-bottom commands that, when properly arranged, give rise to vastly more complicated systems.

Following the metaphor a little farther, we could think of each individual gene as its own little program; each codon, as a single specific task ("get protein A, get protein B, get protein C…").

Restriction enzymes—and CRISPR in particular—let us open up that source code and make our own changes, essentially "hacking" the cell. It

opens the door to all sorts of interesting new experiments. What happens when you remove a particular protein from the genome? What about adding one? Or maybe you need a lot of a protein normally produced by a slow-growing bacteria—can you transfer the gene to some nice prolific HeLa cells and harvest the protein from them?

Plasmids as Insertion Vectors

Modifying DNA can be done **in vitro**—in a lab, rather than a living cell (which would be **in vivo**). But if you want to actually run your newly hacked program—sorry, synthesize your set of proteins using the recombinant DNA—in a live cell, to see what it will really do, you have to find a way to upload the code to a living organism, and make the program executable. How do biologists do that?

Plasmids, as we discussed earlier, are little loops of DNA that are commonly absorbed by prokaryotes. Since prokaryotes are already used to picking up new genes from plasmids, inserting a recombinant DNA sequence into, say, a culture of *E. coli* is simple. All you need to do is add your hacked gene to a plasmid, give it a few runs through a PCR machine to create lots of copies, and squirt it into your cell culture.

That said, if you want to get a prokaryote to actually *use* the DNA you'll need to take an extra two steps, both of which also require modifying the DNA sequence. We'll discuss this more below, but for now, what you need to know is that there is a molecular machine that binds to DNA to initiate transcription. To get that machine started, you'll need to include a **promoter**, a special DNA sequence that gives DNA or RNA synthase an attachment point, and an **origin of replication** sequence, to show the machine where to actually begin transcription. Adding these sequences is all that's needed for the natural machinery of the cell to run your "uploaded code."

DNA insertion with plasmids is a little bit more complicated with eukaryotic cells, which are smarter about not running random bits of code.[2] You'll have to give them a helping hand—for example, placing your cells in a salty solution can make their membranes leaky enough for plasmids to slip inside.

[2] Perhaps they know not to open suspicious email attachments?

Markers

Whatever cells you're working with, however you're adding plasmids, the process isn't going to be 100% effective—at the end of the day there are still going to be a bunch of cells left that did not incorporate your plasmids. (In fact, it's worse than that, because the process of recombination you just used to make your plasmids isn't perfect either. You'll actually end up with a mix of altered and unaltered plasmids, and cells that have absorbed either plasmid, both plasmids, or neither). Or, in other words, if you follow the recipe above you'll end up with a mixture of modified and unmodified (or **wild-type**) cells.

If you want to see what your DNA does, you'll have a rather noisy signal—and if you wanted to use some bacteria to synthesize a protein for you, you're going to be culturing a bunch of free-loading wild-type cells that are doing nothing useful. To fix this, you need some way of killing off everything that didn't get a plasmid, or got the wrong kind of plasmid.

Let's imagine that our plasmid originally contained the sequence (xy)(z), where (xy) is a gene that makes the host resistant to a particular antibiotic A, and (z) does the same thing for antibiotic B. By picking the right restriction enzymes (or using CRISPR), you can make your "cut" right in the middle of gene xy—and make sure that (w), your strand of recombinant DNA, was "cut" at the same sites, so that the sticky ends will match. If you did it right, you'll wind up with the sequence (x(w)y)(z),[3] a functional gene (w), and a nonfunctioning gene (xy). You'll also still have a bunch of (xy)(z) plasmids that closed back up without any changes.

Once you've finished introducing your plasmids to your cells, you'll wind up with three intermixed populations.

- Cells that haven't picked up a plasmid at all, and are vulnerable to antibiotics A and B.
- Cells that picked up only nonaltered plasmids, and are thus resistant to both A and B.
- Cells that picked up altered plasmids, which—because (xy) no longer functions—are only resistant to antibiotic A.

Getting rid of the first group is easy—all you need to do is add antibiotic A, and that will kill them. Removing the second group is harder, since the

[3] Actually, you'll also wind up with (x(w)(w)y)(z). And (x(w)(w)(w)y)(z), and (x(w)(w)(w)(w)y)(z), and on and on. This isn't a problem; it just means that some plasmids will tell the cell to make more of protein W than others. For the purposes of this explanation (and to keep the page count down), I'll keep using a single (w).

population you want to kill is the one that *resists* the drug. The simplest way to do so, **replica plating**, takes advantage of bacteria's rapid reproduction. If you seed a petri dish with a relatively small number of cells, the resulting colonies will be widely spaced and **monoclonal**—each one grown from a single cell. That means you can take a tiny bit from a colony, transfer it to a new plate, and be confident that anything that grows on that plate is the same as the original colony.

To break it down then, in replica plating, you first take a single petri dish D1 that contains many cell colonies, and copy the colonies, in parallel. (One way to do this is to touch a blotter to the surface of the dish, and then touch the blotter to a new dish.) After some cell growth, the result is a copy D2 of the colonies in D1, where all colonies have the same relative position in D1 and D2. You then treat D2 with antibody A and see which colonies die off: these are sensitive to A. Finally, you go back to the original disk D1 and pick out the colonies that were the sources of the A-sensitive colonies in D2.

Promoters and Regulating Recombinant Organisms

If you wanted to use some bacteria to synthesize a protein, you might still be in for a disappointment: it might be that the next morning, all your carefully selected cultures are dead. One likely reason is that your poor bacteria drove themselves into exhaustion trying to keep up the kind of protein expression your plasmid demanded.

To keep this kind of thing from happening,[4] you need to make sure that whatever gene or genes you're injecting can be easily **regulated.** That means picking a promoter that only functions under certain conditions, such as a high temperature or different food source. That way, you can give your cells the time they need to grow and reproduce before starting your experiment.

Phages as Insertion Vectors

Plasmids aren't the only **insertion vectors** that can be used to inject recombinant DNA into other cells. They were the first that biologists learned to use, but they're limited to short strands of DNA, around 8000–20,000 base-pairs long. If you want to transfer more, you'll need to use viruses. After all, they've evolved to perform exactly this task—inserting foreign DNA into a living

[4] To say nothing of your creation breaking out of your lab and wreaking havoc on the unsuspecting population.

cell—and nothing else. They can handle much longer DNA sequences, and come pre-armed with the tools to get it inside a cell. The process is basically identical; the only difference is that you replace your custom plasmids with bespoke viruses.

For safety, you'll want to use viruses that only infect single-celled organisms, known as **phages.**

Using Genomic DNA Libraries

The experiments above are feasible for many labs to do in-house. But they can also be scaled up and parallelized by specialist labs. Like most of the techniques we've talked about, you can deploy insertion vectors (or other tools for injecting foreign DNA) in a massively parallel way and create a **genomic DNA library** for other scientists to use.

A genomic DNA library is a large collection of cells that have had another organism's DNA inserted—one gene's worth for each colony of cells. Because those colonies can then continue to grow indefinitely, it's easy to split off a new population to use for a particular experiment without depleting the original stock. In other words, it is never necessary to "return" anything to this library—one can withdraw a copy of every piece, and the entire library will still be available for the next researcher.

Genomic libraries are typically created by randomly fragmenting the subject's genome, inserting *those* pieces into plasmids, and adding multiple plasmids to the same colony of cells. By carefully balancing the number of cells to the number of plasmids, you can make it just hard enough for a single cell to grab multiple plasmids so you can be reasonably confident that cells have only a single plasmid, if they survive the initial screenings.

Genomic DNA libraries are particularly useful finding the DNA code that gave rise to a particular mRNA molecule[5].

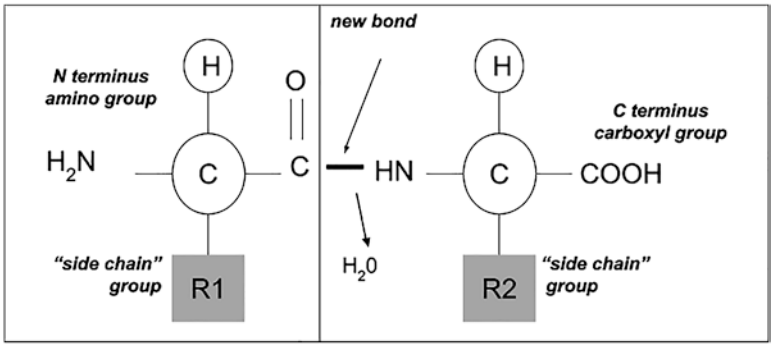
[5] In eukaryotes, mRNA is often spliced and otherwise altered before it's translated; the sequence of the final molecule can be quite different from the original gene—meaning you can't just search the genome for the exact mRNA sequence.

Creating Novel Proteins: Tagging and Phage Display

Adding new genes isn't the only thing that recombinant DNA is good for. It can also be used to *modify* already-existing genes. Sometimes that means messing with a protein to see what happens if you swap out *this* amino acid for *that* one—If something goes dramatically wrong, that was probably important to the protein's function. And sometimes it means **tagging** particular proteins by adding some kind of marker that makes them easier to find later on.

The classic example of this is probably **green fluorescent protein (GFP)**, which we briefly mentioned a few sections back. (Other colors exist, but green was the first to be discovered.) One way to track a protein in a fluorescent microscope is not to use a dye, but to modify the DNA of the protein so that it contains, in addition to the usual amino acid sequence, a sequence for GFP. Originally found in jellyfish, GFP has two properties that make it a popular choice for this:

- It's small—made up of less than 238 amino acids—and thus can be added to the end (the n- or c-terminal; see Fig. 1) of another protein, usually without interfering with its functions. (Exceptions exist, but they're rare.)
- It's very stable and remains fluorescent for a long time before photobleaching. Once produced, GFP creates a new set of covalent bonds that let them absorb and reemit light, then curls up into a narrow tube around those precious bonds, offering extra protection.



A protein, which is a chain of amino acids, has an **N-terminus** (where there is an unlinked nitrogen-containing **amino group**) and a **C-terminus** (with an unlinked carbon-containing **carboxyl group**).

Fig. 1 Structure and nomenclature of proteins

Once altered, the resulting protein is known as a **fusion protein** or **chimeric protein.**

Besides microscopy, another good use for fusion molecules is to mess with a virus' protein "coat." Adding the gene for a particular protein to the right section of the virus' genome results in it becoming part of the coat—the virus "displays" the protein on the outside, where it's free to bind with whatever other proteins it typically interacts with. This is another of those techniques that can be deployed in parallel. By creating **phage displays**—large collections of viruses that have each been altered to "display" a different protein—you can test millions of potential protein partners at the same time.

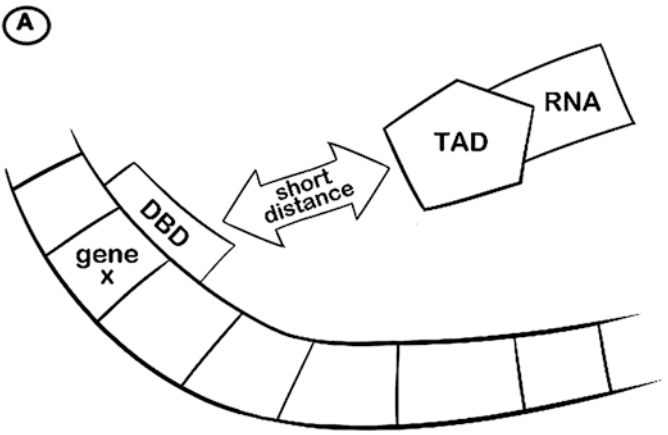
You can test your altered viruses against a target protein or molecule *q* that's been anchored to the bottom of a petri dish. All you need to do is add your phages, each for a different protein *p* to the plate, and give them some time to find partners. Then you can wash away the excess so that only phages "displaying" proteins *p* that bind strongly to the target *q* remain in the plate. This lets you know which proteins *p* interact with *q.* The viruses might not be fluorescent, but they do carry the gene for whatever protein just interacted with the target, and extracting and sequencing the DNA from those leftover viruses lets you precisely identify those proteins.

Yeast Two-Hybrid Assays Using Fusion Proteins

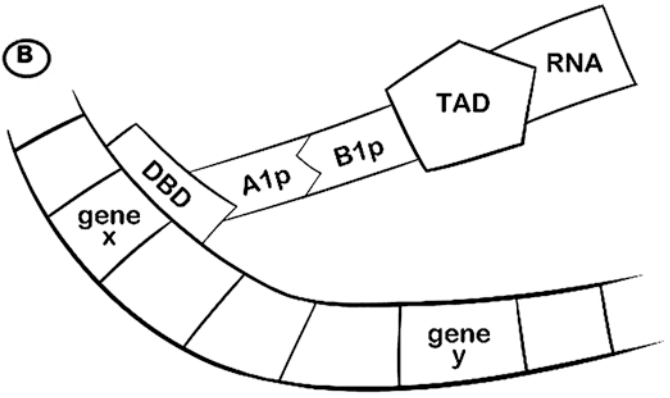
It's even possible to test protein-protein reactions at the genetic level by taking advantage of an interesting quirk. Most eukaryotic **transcription factors**—the proteins that help RNA synthase attach itself to the DNA—are made up of two subunits. The **DNA-binding domain (DBD)** handles the connection with the strand of DNA; the **transcription activation domain (TAD)** calls in transcriptional machinery like RNA synthase.

And the two subunits *don't need to bind together* to act as a functional transcription factor—mere proximity is enough (see Fig. 2A). Conversely, if you stop the two subunits from getting close to each other, they won't encourage any transcription. You might have already guessed where this is going: to test if two proteins (let's call them *p* and *q*) interact, you can combine one with each half of the transcription factor for one of those reporter genes. If *p* and *q* interact with one another, the transcription factor will come together and work normally (Fig. 2B).[6]

[6] For the most part, the two subunits would be physically connected in this case, which may or may not affect things.



(A) In wild yeast, *A* binds *B*, which activates gene *x*. Only the DNA binding domain (DBD) is needed for *A* to find the promoter site, and only the transcription activation domain (TAD) is needed for *B* to activate transcription.



(B) In hybrid yeast, the DNA has a promoter for x near a reporter gene y. *A1p* can bind to the promoter site using the DBD of *A*, and *B1q* will activate transcription—of gene *y*—using the TAD of *x*. But *A1p* will only recruit *B1q* if proteins *p* and *q* bind. So, *y* is expressed iff *p* and *q* bind.

Fig. 2 The yeast two-hybrid system

To perform such an experiment, you'll need to find or introduce is a **reporter gene** that has an obvious effect when expressed, and arrange the DNA so that a reporter gene can only be expressed when activated by *p* plus the DBD and *q* plus the TAD. It's possible to use something like GFP as a reporter gene, but it's usually easier to cut out the middleman and use a vital

metabolic gene—one that the cell has been altered to lack. If the two proteins *p* and *q* interact, then the DBD and TAD domains will be put in proximity, the reporter gene will be expressed, and the cell will live. If they *don't* interact, the subunits will not get close enough to do their job, the vital protein won't get made, and the cell will die.

What makes this process especially useful is how easily it scales up. All you need to do here is assemble two separate cell libraries—one with a wide range of options for protein *p*, and the other with lots of variants on protein *q*. Once prepared, cross-breeding the two libraries results in a massive number of different hybrids, each with a different *p* and *q* pairing. At that point, the mechanics above kick in—cells whose *p* and *q* variants interact will survive the screening, and cells with nonmatching variants will die.

Since such experiments are normally done using yeast cells, the process is known as the **yeast two-hybrid system**.