

Introduction

On June 26th, 2000 the President of the United States and the British Prime Minister held simultaneous press conferences to announce that a rough draft of the human genome was ready for inspection. For the first time, humans knew (roughly) the complete sequence of their own DNA. The achievement was widely heralded in the press, vividly describing our species “blueprint,” our “recipe,” our “instruction book.” But how accurately does the string of 3.1 billion Gs, As, Ts and Cs describe the nearly 100 trillion cells in our body? Beyond documented sequence variations, accumulated mutations and all the extrachromosomal genetic information known to exist, what does the human genome sequence (or any sequence data for that matter) say?

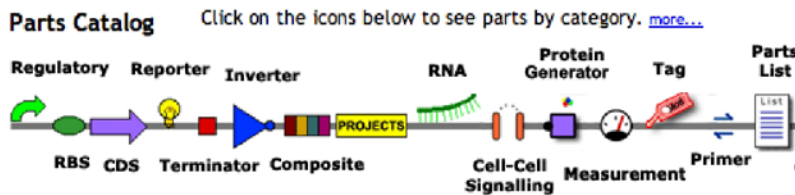
Consider the often-used analogy of DNA as a “blueprint” for a cell. Engineers and architects use blueprints to guide their construction projects, but could we, as biological engineers, use DNA sequence to guide us in the fabrication of a cell? We can’t yet make a lowly bacterial cell let alone a human cell even though we know the genome sequence. The limit arises not from our new status as a biological engineers but from the limited information in the DNA “blueprint” itself. Perhaps a better analogy, then, is DNA as a “recipe.” Recipes specify lists of ingredients, number of servings, cooking times and occasionally variations to try. Sequence data, in contrast, is far more difficult for us to interpret than a cooking recipe. We can make something to eat using a recipe, but what can we make knowing “GCACCACCACTAGAACCAATAGAA”? A cell can do something with that, but can we?

DNA as our “instruction book” is perhaps the least satisfying analogy of all. Beyond the fact that cells change expression patterns over time in ways that are not evident from the DNA sequence, there is also a multimillion-dollar industry selling books and promoting workshops to help us understand each other. No one yet has suggested we start understanding people by scanning each others’ basepair sequences, though the XY balance is considered important (see for example: [Men Are From Mars, Women Are From Venus](#)).

A better analogy, and the one we will explore, is to think of DNA as a low-level programming language and a genome as a particular program. Like software, DNA has an alphabet but with only four letters in the genetic code. Since there are proof-reading mechanisms in the cell (or hardware) implementing the program, syntax errors are less likely to arise than in Python or Perl or C++. The code for cellular programs is messy but so are computer programs. Subroutines are often dependent on one another (the cell cycle and DNA replication for example) and parts of the program get reused in useful, but complicated and unpredictable ways (seen as cross-talk in signaling pathways for example). Genetic code and computer code are both susceptible to viruses that hijack normally benign functions. The analogy of the DNA as computer code is not perfect. We have to set aside the presumption of an intelligent agent responsible for writing the initial

program as well as natural events that change the code over time (evolution leading to mutations). And no good tools exist for systematically debugging the genetic code.

What would make genetic code easier to write? One idea is to make it a more “object oriented” language, defining units of known function that could be combined in standard and predictable ways. One effort to facilitate genetic programming can be found at [The Registry for Standard Biological Parts \(http://parts.mit.edu\)](http://parts.mit.edu). There you will see a catalog of parts that describe basic biological functions. For example BBa_B0010 is the part number for a transcriptional terminator. Interested in a promoter? There are lots. A perennial favorite is BBa_R0010, the promoter from the bacterial lac operon, and you’re already familiar with BBa_R0085 from your protein engineering work since this part is the promoter recognized by the T7 RNA polymerase. The Registry of Standard Biological Parts makes its parts freely available to interested researchers and engineers, and allows registered users to add and annotate parts, as you will do later in this experimental module.



In the first part of today’s lab you’ll browse the catalog at the Registry of Standard Biological Parts then you’ll use parts described there to design and specify a protein generating device. Finally, you’ll begin working with a bacterial strain engineered to sense and respond to light using parts from the registry.