BioCircuits:

A Systems Approach to Synthetic Biology

Eric Klavins

Department of Electrical Engineering
University of Washington
Seattle, WA

Last compiled April 17, 2015

Contents

Chapter 1. Introduction 1

1.1. What is Synthetic Biology? 1

1.2. The Applications of Synthetic Biology 5

1.3. The Challenges of Synthetic Biology 8

1.4. The Ethics and Risks of Synthetic Biology 12

1.5. Overview 14

1.6. Problems 15

Chapter 2. Basic Biochemistry 17

2.1. *E. coli*: The Model Procaryote 17

2.2. Molecules in the Cell 18

2.3. Processes 23

2.4. Re-programming *E. coli* 30

2.5. How we See What's Going On 30

2.6. Problems 30

Chapter 3. Differential Equations 31

3.1. Definitions and Examples 31

3.2. Simulation 33

3.3. Equlibria and Stability 34

3.4. Linearization and Local Stability 36

3.5. Non-Dimensionalization 38

3.6. Problems 38

Chapter 4. Mass Action Kinetics 41

4.1. Definitions 41

4.2. Example Networks 46

4.3. Modeling Regulatory Networks and Metabolism 49

4.4. Problems 51

Chapter 5. Enzyme Kinetics 53

5.1. Fast and Slow Dynamics 53

5.2. Enzyme Reactions 53

5.3. Hill Functions 56

5.4. Problems 59

Chapter 6. Synthetic Regulatory Networks 61
6.1. Re-programming *E. coli* 61
6.2. Switches and Oscillators 61
6.3. RNA Regulatory Mechanisms 64
6.4. Problems 64

Chapter 7. *In vitro* Synthetic Biology 65
7.1. Enzyme Free DNA Circuits 65
7.2. Transcriptional Circuits 67
7.3. Problems 67

Chapter 8. Stochastic Chemcal Kinetics 69
8.1. Stochasticity in Cell Dynamics 69
8.2. Foundations of Stochastic Chemical Reactions 69
8.3. Simple Examples 71
8.4. Moment Dynamics 75
8.5. Approximate Methods 77
8.6. Simulation 78
8.7. Computing with Molecules 80
8.8. Problems 83

Chapter 9. Composition and Modularity 87
9.1. Signals 87
9.2. Modules 87
9.3. Composition 87
9.4. Retroactivity 87

Chapter 10. Robustness and Sensitivity 89

Chapter 11. Parameter Estimation and System Identification 91

Chapter A. Mathematica 93

Chapter 1

Introduction

1.1. What is Synthetic Biology?

Synthetic biology is the design and study of artificial living organisms, re-engineered organisms, and engineered parts for existing organisms. The idea is to bring engineering tools and methodologies to the subject of biology, much as electrical engineering brings these tools to physics, with the goal of designing novel and useful biological systems. These novel systems, new life forms really, may someday afford humanity a new level control over the processes in our bodies, in our crops, in landfills, compost heaps, and the ocean waters: Instead of controlling the living world with machines and harsh chemicals, synthetic biology promises control at the level of individual molecules using engineered living cells as tiny chemical processing plants.

Of course, biology is more than just chemical processing. It is information processing. The instructions for how to grow and regulate a human being are encoded by the DNA inside each of the cells in your body. Growth and development follows precise, distributed, robust algorithms. The immune system can adapt itself to pathogens it has never seen. A predator can track its prey and outsmart it. Our brains can do research and write books. Life *is* computation. The molecules and structures inside cells are the tools that nature uses to implement life's computations, just as the silicon and wires inside your computer implement computations. Thus, to some extent, biology (synthetic or otherwise) ought to be seen more abstractly than our high-school biology books would have us see it. Our job as living system engineers is to define the abstractions, the composable modules, the signal carriers, the simplifications, the programming languages, and the debugging tools for living systems.

We are far from being able to engineer entirely new life forms, although remarkable advances are being made every day. Many of these advances are technological, such as rebooting cells so that they become stem cells [**49**], putting the genome of one cell into another [**32**], synthesizing large pieces of DNA [**20**], and improved methods for tracking molecules inside cells [**30**]. Other advances are more conceptual, such as new understandings of stochastic chemical kinetics [**37, 40**], system identification feedback control inside cells [**35**], directed evolution [**17**], and so on. Perhaps most telling, some advances look like hacking: artificial oscillators inside cells [**19**], engineered cell-cell communication [**11**], digital logic inside cells [**53**]. The fact that researchers in well-equipped labs and others in their garages [**34**] can now make their own widgets for living systems suggests that the technologies for and the conceptual understandings of living systems are now sufficiently rich to allow engineers to to tinker. And tinkering leads to engineering research, which leads to engineering practice.

It is fun to argue about what life *is* or what it would mean to actually have built a living organism or enough of one to call it a truely new species. However, in this book we will take a more pragmatic approach and look, instead, at what living systems seem to know how to do well: sense, actuate, control, process information, and evolve. These five capabilities – present in all the living systems we can think of – are probably necessary for all life forms, and in this book we focus mainly on them as goals for what to build. Furthermore, these capabilities, except for the last one, are usually thought of by engineers as the important aspects of embedded control systems (such as automobiles or cell phones). We hope to use our experience engineering embedded systems for engineering living systems, an idea that leads to the question: What do we mean by sensing, actuation, control, and information processing in the context of living systems?

Sensing: A cell can sense all sorts of quantities, inside itself and outside. For example, the bacterium *E. coli* makes a protein called *LacI* (for *lactose inhibitor*) that changes shape in the presence of the sugar lactose. This has downstream effects that change the cell's metabolic machinery so that it can efficiently digest lactose [**36**]. Essentially, *LacI senses* the presence of lactose. In fact, all cells have molecular mechanisms that can sense the small molecules normally found in nature, as well they should. Some of these molecules are nutrients, some are toxins, some represent signals from other cells, and so on. Any organism that has a good estimate of what is happening in its environment has an advantage over less "aware" organisms. There are also cells that can sense temperature [**18**], acidity [**?**], light [**?**], magnetic fields [**41**], and on and on. Synthetic biologists use these capabilities, often by putting the gene for a type of sensor found in one organism into a genetic regulator circuit being engineered in another, thereby giving their system new capabilities. For example, synthetic biologists have put arsenic sensors in *E. coli* to make a device that can warn users of tainted water [**10**].

Actuation: A cell can alter its environment, or itself, in a variety of ways. It can emit a small molecule as a signal to other cells. Many cells, such as *E. coli*, can move by swirling their flagella while other cells, such as amoebae, move by growing and digesting their cytoskeletons. Internally, cells can synthesize a wide variety of molecules that can have huge effects on the cell and its environment. In fact, *making something* is a form of actuation. A newly synthesized molecule, such as a protein, can effect structure, metabolism, signaling within and outside of the cell, and so on. Once again, synthetic biologists borrow the actuators from other organisms (or make their own) and hook them up to sensors in novel ways to create new behaviors. For example, a widely used actuator is *green fluorescent protein* (GFP) [**13**], which is a molecule usually found in jellyfish that happens to fluoresce. Synthetic biologists use GFP like embedded systems engineers use light emitting diodes, as indicators that something has changed inside the cell. For

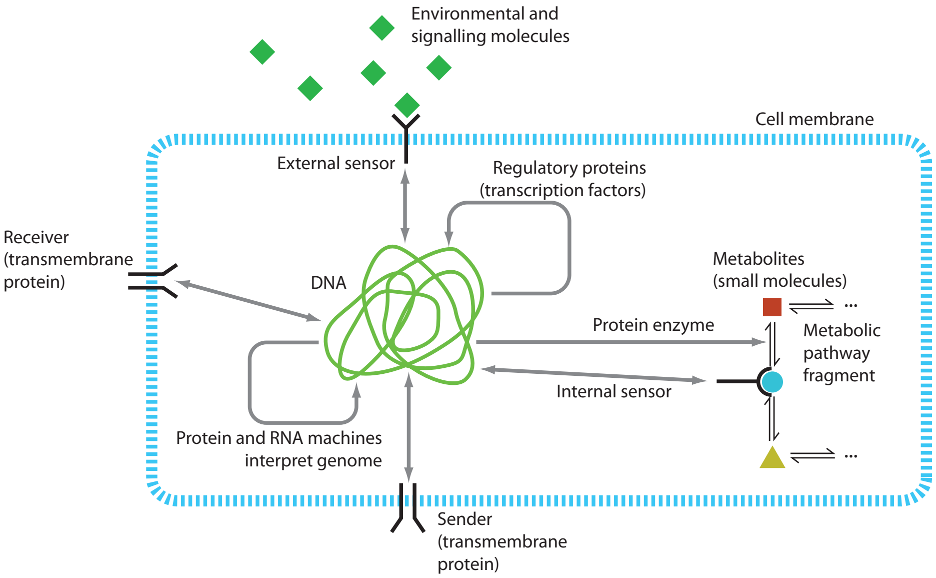


Figure 1. The general structure of a sensing, control, actuation and information in the cell. Small molecules can be sensed with transmembrane proteins. Internal small molecules can be sensed with other sensing proteins. Other controllable transmembrane proteins can shuttle molecules in or out of the cell. Enzymes control the rates of reactions among metabolites. All of this activity is orchestrated by protein and RNA machines that interpret the a program stored in the DNA.

example, the arsenic sensor mentioned above is coupled to GFP production so that cells that detect arsenic glow green under ultraviolet (dark) light.

Control: Many aspects of a cell's behavior need to be regulated. Biologists usually call the process by which the concentrations of various important molecules are regulated *homeostasis*. Good control requires feedback, which requires sensing and actuation. For example, yeast cells have control circuits for *osmotic pressure* [**35**], which work by (1) sensing the osmotic pressure difference across the cell membrane using a very elegant molecular sensor; and (2) changing the rate at which they pump glycerol out of the cell, by activating more "pumping" proteins. The system involves many types of molecules interacting in complex ways, but the net effect is that the osmotic pressure system behaves like a second order, integral-feedback controller. Organisms regulate many other things as well. Your body regulates blood-pressure, blood-glucose concentration, heart-rate, fat, balance, and many other important variables. Control of these quantities is important for living systems primarily because the environment is highly uncertain and, thus, cannot be relied upon to supply steady levels of any important variables; cells must regulate these things themselves. Synthetic biologists need to do essentially the same thing with their engineered systems – otherwise their engineered systems will be highly sensitive to the environment and, more pervasive, to their uncertainty about exactly how the molecules in the system interact.

It turns out that the main difficulty in engineering control systems inside cells is in implementing a given mathematical control law using only the biochemical parts available, which are few and poorly characterized at best. With electronics, and especially computers, control engineers have become accustomed to easy implementations of complex mathematical control algorithms. But with synthetic biology, a whole new *controller synthesis* technology must be developed. This *new synthesis* is a topic of intense research, is by no means complete, and is the main subject of this book.

Information: Information is stored and processed on many levels in a cell. For example, the DNA in a cell stores the instructions for how to build and regulate all of the systems comprising the cell. It very carefully maintains this information and passes it to its offspring. It is a remarkable fact that almost any cell in your body can, in principle, be used to grow a new you. Also, at any given time, a cell has a *state* (for example, dividing or not, growing or not, etc.) that is carefully maintained so that all subsystems "know" the state of the cell. For example, your liver cells are in different cellular states than your brain cells (hopefully). Furthermore, information is communicated from cell to cell via signals. Neurons use electrochemical signals and pituitary gland cells use protein hormones to send powerful signals all over your body. A most remarkable information processing system is the immune system (of, for example, humans) which can tell self from non-self based on, essentially, the information content of the molecules the immune system cells encounter. It does this both through innate and learned responses to information.

Management (and mismanagement) of information inside cells is essential for modern living organisms. Synthetic biologists have yet to really tap into information processing in synthetic systems. Yet having recently learned to engineer information systems (for example, the Internet), we are eager to engineer information processing systems for cells: almost every disease imaginable is essentially a problem of information (stored in bits of foreign DNA) getting into the wrong places and co-opting existing information processing systems.

Evolution: Engineers have not had to deal with technologies that self-replicate. We simply have never built anything so complicated. However, synthetic biologists are now tweaking existing organisms by inserting new functionalities, creating essentially new strains of the organism. These organisms *do* self-replicate and, therefore, populations of them do *evolve*. That living systems are capable of evolution is what makes them so incredibly robust to changing environments and competition from other organisms. In fact, the architecture of life seems even to have evolved the ability to evolve [**38**]! For example, some genes are copied with more errors than others during replication and some genes are less sensitive to mutations than others [**6**]. This quality directs genetic experimentation to less essential functions, producing distributions of offspring some of which are possibly more fit while all are still viable.

Synthetic biologists have to think about evolution for several reasons. First, it is usually the case that a synthetically introduced "feature" in a cell (such as an arsenic detector-indicator circuit) exerts a huge metabolic load on the host cell. Any cell that happens to drop the circuit, therefore, has a considerable selective advantage. How to build circuits that don't simple evolve away in a few generations is a huge problem [**12**]. Second, introducing a new functionality into a cell and putting a population of said cells into a new environment creates a new "starting point" for evolution [**7**]. Due to our limited understanding of evolutionary dynamics and ecology, we really have no idea what will happen to subsequent generations of engineered cells and are therefore subjecting ourselves to unknown risks. Third, synthetic biologists may be able to *use* evolution to tune engineered systems, if they can tie the performance of the system to some metabolic benefit. This opens up the possibility of a design paradigm wherein we build a circuit that behaves in essentially the right manner, and then tune it into place using evolution.

1.2. The Applications of Synthetic Biology

Engineers often reply to claims that we can make linear amplifiers or Boolean logic circuits out of molecules with: *Why would you want to do that? We can already make very good amplifiers out of silicon and metal.* While that is true, it isn't very easy to integrate electronics with living systems. Synthetic biology is not about using engineered cells in situations for which computers, cell phones, pacemakers, and microchips are ideally suited. Rather, it is about finding new applications that interface with living systems using the technology of living systems.

In fact, we already apply engineered living systems to all sorts of problems, if you count domesticated animals and bred and hybrid plants. Furthermore, herbicide-resistant soybeans, tomatoes that don't rot, rice with extra vitamins [**24**], and a host of other genetically engineered foods can be found in supermarkets today. Even more, genetic engineers have harnessed the protein-production capabilities of micro-organisms to make medicines for diseases from arthritis [**50**] to malaria [**?**]. For example, insulin is a protein hormone used to treat diabetes. It signals the body's cells to take up glucose from the blood. In the past, insulin was laboriously extracted from pig pancreases. Several decades ago, however, the gene for insulin was successfully inserted into bacteria or yeast, which can be grown in fermentors to produce cheap, high-quality insulin [**22, 33**]. Many other molecules can now be made in bacteria or yeast as well – from anti-malarial drugs [**39**] to artificial sweeteners [**?**].

Most of these examples are of *genetic engineering*, from which synthetic biologists (attempt to) distinguish themselves: While genetic engineering concerns swapping genes among organisms, *synthetic biology is the construction of entirely new systems*[1]. To explore the potential applications of synthetic biology, we need to find situations in which a system consisting of a molecular sensor, an actuator, and a control mechanism would be of use. In this sense, genetic engineering might be considered first-generation synthetic biology – in that it usually deals only with actuators, typically molecule production and secretion. Control engineers call such systems *open loop*. If we start closing the loop, we then ask: what would be possible if we could put a bit of computation in genetically engineered systems? The following is a very short discussion of possible applications, many of which are already under scrutiny by somebody somewhere. We have attempted to speculate on what might be possible with more control engineering in these applications. Many, many other applications have yet to be imagined.

[1] Genetic engineers and metabolic engineers often take umbrage at such distinctions and the very term *synthetic biology* is oft-criticized as nothing more than a re-branding of existing work.

Gene Therapy: Pharmaceuticals are typically substances that diffuse throughout the body, hopefully interacting with desired target cells, but also interacting with everything else. For example, the drug AZT is intended to interfere with a certain step in the replication of the HIV virus [**?**]. Unfortunately, it also interferes with a number of other natural processes in the body, and therefore causes all sorts of side effects. The idea of gene therapy, in contrast, is to deliver molecules only to the cells and tissues that need it. For example, a gene that produces an *interfering RNA* can be inserted into immune system cells so that HIV growth is inhibited [**?**]. Several semi-successful human trials of this basic idea have been tried and show great promise [**?**]. A similar idea is the *DNA vaccine*, wherein a small bit of DNA encoding for a protein antigen is inserted into the patient. The potential benefit here is that DNA can be easily synthesized and inserted into the patient, whereas the antigen itself must be produced *en masse* in fermentors and kept from spoiling. Both of these technologies, however, are *open loop*, uncontrolled deliveries, and any specificity and locality of the medicine are a result of the choice of tissue into which the genetic material is inserted.

Imagine now a genetically encoded program that is inserted into the genome of the patient and that is activated by the molecular signature of a pathogen (sensing) and, only when activated (control), produces (actuation) a regulated (control again) dose of an interfering molecule. Such a technology could deliver a drug with incredible specificity and could do so more robustly. A regulated gene-therapy may even help prevent rapid in-host evolution of the infecting pathogen by safely exposing the pathogen only to lethal doses at specific times.

Biofuels and other Useful Molecules: Ethanol, butanol, propanol and even hydrogen can be made from sugar cane, corn, yeast, algae, bacteria and other organisms that have been genetically modified to improve biofuel production (by increasing yields or by making it easier to extract). However, it is not clear whether an economically or environmentally viable approach has been or can be developed. The main issue seems to be that current approaches typically involve using or hacking organisms that have no real use for making, for example, large amounts of some alcohol-based fuel (or pharmaceutical, etc.). Thus, efficient biofuel production in the future, if any, will likely come from a variant of algae [**?**] (a guess) that has been engineered to the point of being almost unrecognizable to a modern-day biologist. The difficulties that arise are daunting: up-regulating the production of a biofuel steals critical resources needed for cell growth; biofuels are typically toxic to most cells at useful concentrations; the nutrients used to feed biofuel-producing organisms have to be literally dirt-cheap; the carbon-footprint of biofuel production should ideally be lower than traditional energy sources; and so on [**?**].

These issues are largely issues of chemical engineering and metabolic pathway[2] engineering[**44**]. However, as new molecular sensors are invented that can sense metabolites in a metabolic pathway, one can begin to imagine a systems engineering for metabolism: The state of a synthetic cell's metabolism is internally monitored and enzymes that regulate parts of the pathway are themselves regulated to produce desired effects. Perhaps, some day, we will invent a single organism that

In this text, we will try not to make get involved in this argument. We could have easily called the book *a systems approach to genetic engineering.* The emphasis is on systems engineering and engineering living systems. Call it what you like.

is programmed with every metabolic pathway ever discovered or devised encoded in its genome. To turn on a particular pathway or set of pathways, we might send a start signal, encoded in a sequence of light pulses. The engineered cell interprets this start signal and turns on the appropriate pathway. Controllers for the pathway can be similarly engineered to use whatever nutrient source is available, and to optimize the relative strengths of pathways, optimizing the production of everything from biofuel to pharmaceuticals.

Bioremediation: Most of what we throw and flush away is degraded by naturally occuring bacteria. Landfills, sewage plants, and compost heaps are filled with with beneficial microorganisms eating away at bits of paper, food, sewage, and almost anything else with energy left in it. Many of these organisms produce methane, which is sometimes used to run power plants (although the practice is not yet widespread). The waste-treatment industry has learned to use the right organisms in the right place. However, there are a host of substances that we routinely dumped or spilled that are not easily digested by known organisms: PCBs, oil (from oil spills), dioxins, hexavalent chromium, etc. It is tempting to suggest that the reason that no micro-organism that digests these substrates exists is because *the substrates are toxic!*. But another reason could be that none of these substances have existed in any substantial concentration until recently. Therefore, the pathways and enzymes for processing them have not had a chance to evolve – which suggests that an engineering approach might be useful.

Note that *carbon sequestration* using microorganmsism is a kind of bioremediation. It would be wonderful to design a microogranism that can use the carbon in carbon dioxide as a carbon source in an application specific manner (e.g. near fossil fuel burning factories), possibly by harnessing the carbon sequestration capabilities of plants.

Whatever the application, to find or build micro-organisms that digest a given substrate, several approaches are being explored [**48**]. First, bacteria can be evolved in a chemostat [**43**] with increasing levels of the substrate of interest. This approach has the effect of tweaking existing pathways to act on new substrates [**?**]. Second, when no single species of microorganism can be found to act on the substrate of interest, it may be possible to find a consortioum of microorganisms where one species catabolizes the first step in a pathway and a second finishes off the process. However, it may be, for most novel substrates, that no pathway exists in any organism or consortia to handle it [**?**]. In this case, entirely new pathways need to be developed, which is probably one of the most difficult engineering problems of modern times. And the problem is not unique to bioremediation: Any processing of small molecules (toxins, biofuels, drugs) in a microorganism requires pathway engineering.

[2] In this chapter we talk about *pathways* without carefully defining them. In Chapter **??**, we talk about pathways more formally. For now, think of a pathway as a sequence of chemical reactions, each step of which is mediated by a particular protein catalyst, or enzyme. For example, recall the citric acid cycle that uses oxygen to burn carbohydrates to obtain chemical energy. To *engineer a pathway* means to design the enzymes that go into the pathway and to make sure that increasing the activity along the pathway does not steal resources from other pathways or create toxic intermediates. Finally, to *borrow* a pathway from another organism means to copy the genes respoible for producing the enzymes for that pathway into a new organism.

New pathways are typically constructed by piecing together bits of pathways from existing organisms and tuning them into place [**44, ?**]. Often, an enzyme for a particular step simply cannot be found, in which case protein engineers can sometimes design a new enzyme using new protein design algorithms [**29**] – although this approach is quite new and relatively untested. However, when constructing a new pathway from such bits and pieces, the whole typically does not behave as expected, at least not until after considerable tuning. Synthetic biology could someday offer a design theory for metabolic pathways: composable sub-pathways, tunable architectures, and feedback.

Tools and Parts: All of the above applications involve genetic engineering. In the more complicated systems, many genes need to be put together into a *system* wherein some genes might encode enzymes, some might encode regulatory proteins, and so on. Such a system may have sophiscticated molecular sensors that involve several types of molecules, chemical actuators, and multiple leves of control and feedback. Ideally, synthetic biologists would like to put together found and engineered "parts" into whole systems in predictable ways much as electrical, computer and mechanical engineers do. Unfortunately, two systems that behave correctly in isolation do not always behave the same way when put toegher, or *composed*. Thus, although it is not an "end user" application, the design of robust, reliable, and composeable parts or subsystems, is a major intermediate application of the concepts of synthetic biology. A good parts library is an enabling technology for the above applications, like good op-amps are and enabling technology for audio preamplifiers.

Making parts means many things: It could mean designing easy to use genes with standard interfaces [**42**] or it could mean designing molecular mechanisms that are easy to *impedance match* [**?**] (so that they do not interfere with each other). In general, synthetic biologists are looking for a programming language or hardware description language for biology that can be used to program any behavior. In the next section we describe the main challenges so far encountered in persuing this goal.

1.3. The Challenges of Synthetic Biology

Composition: The primary challenge in synthetic biology is the design, implementation, and analysis of *composable molecular systems*. Composition means taking two systems putting them together. For example, we might compose an arsenic sensor with an activator for GFP. This requires putting together three subsystems – one for the detector, one for the activator, and one for GFP – in *series*. One might also imaging other compositions: parellel, feedback, etc.

The signals coming from each of these sub-systems are basically molecular concentrations. The problems arise in several ways: First, there are no wires inside cells to protect signals, even ones internal to a sub-system. Thus, the molecules occuring in a given subsystem at a given time *can potentially interact with everything else in the cell!* These interactions can make a sub-system that works in one context not work in another. Imagine the difficulty now of making a *datasheet* for such a molecular device: The engineer would have to characterize the behavior of the device for every possible context in which the device might be used! Second, two sub-systems connected in parallel can suffer from *retroactivity* [**?**], meaning that

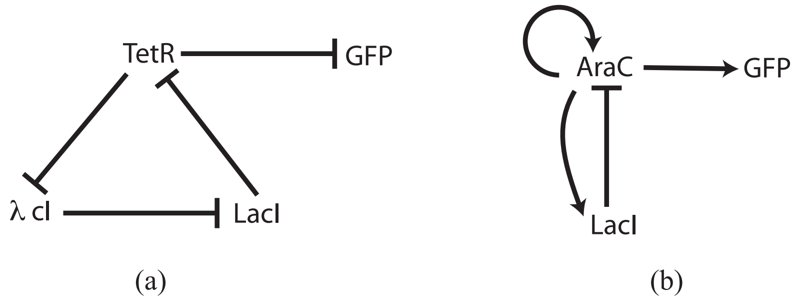


FIGURE 2. Two different architectures for a genetic oscillator. Nodes represent proteins (see Chapter **??**). Normal arrows mean "activates the production of". Blunt arrows mean "represses the production of". (a) A ring oscillator [**19**] called the *repressilator* consisting of three *repressors* connected in a ring. (b) A relaxation oscillator consisting of repressors and activators [**46**].

not only do changes in the first module in a series composition affect the second, but changes in the second module also affect the first. Electrical engineers see this when the hook up a device to a power supply: The voltage of the supply can drop (unless is is regulated by a voltage regulator). Similarly, electrical engineers have built for example op-amps that require incredibly low currents at their inputs, so that whatever device is supplying the input is not substantially affected.

In synthetic biology, we do not yet know how to build systems that solve these problems, but we have begun to explore different architectures for solving specific problems. For example, Figure 2 shows two different architecures for producing oscillations (these are covered in more detail later in this book). The first is a familar ring oscillator, and the scond is a relaxation oscillator. Why one would be better than the other in a given situation will hopefully become clear as we delve into the details of how they work.

Modeling: Systems biologists are tempted to write down equations describing every single reaction in the systems they study. For example, a recent paper describes a system involving 828 biochemical reactions with 499 ordinary differential equations (ODEs) with 229 (unknown) parameters [**15**]. This model is fitted to typically scant data obtained in the lab to obtain estimates of what the parameters are. The authors also note that the behavior of the system is insensitive to changes in a number of parameters (taken one at a time) and infer that as a result they "know" some paramters better than others.

Several fundamental problems with using enormous models are immediately clear. First, we don't really know good models of even single reactions inside cells – their structure is always an approximate (e.g. is it bimolecular, cooperative, etc.?) and the reaction rate constants are pretty much impossible to measure. In fact, the structure of these reactions typically come from what studies in which a gene is turned off aritifically and the behavior of other genes is observed to determine causality. When there is causailty, a reaction is included in the model, whether the interaction is direct or indirect, fast or slow, etc. Thus, a collction of even three such "reactions", let alone hundreds, is suspicious. Second, once a system of ODEs is large enough, it is usually possible to fit almost any finite data set. Therefore, fitting does not validate or invalidate the model. One may as well fit a neural network to the data! Finally, modeling and then fitting is merely *descriptive*, not *predictive*. A model is really only useful if it can help you do something (like $F = ma$ helps design spacecraft trajectories) or makes a scientific prediction that suggests further experiments.

Many systems biologists are now taking a more pragmatic view of modeling. One idea, roughly, is to pose the simplist model that both explains that data at hand and that is useful for predicting data in new situations[**3**]. This is an engineering approach called *system identification* and has been used successfully for decades outside of biology. Within biology, the approach is viewed with suspicion: what good is a simple model if it does not account for every moving part in the system? Another idea is to ask *qualitative* questions about networks[**?, ?**]. For example, one can show that networks without feedback loops are incapable of oscillating or that there exist a broad set of parameters that do make another architecture oscillate. This information could be used by the synthetic biologist to design an oscillator for which there is some hope of tuning it into place. Yet another approach is to actually automate the scientific method by designing experiments that are guaranteed to eliminate as many candidate models as possible [**?**]. Such an approach could be coupled with the design process much as a debugger is coupled with a compiler.

Robustness: Living systems are incredibly robust to variable nutirent supplies, temperature changes, toxins, mutations, new competitors, and on and on. Human technologies can be robust too, or not. Your typical compact car can take all sorts of abuse, but a the free laser pointer I just got as a freebie at a conference will probably last about a week before if falls apart. What exactly makes a system robust is an incredibly subtle question and is asked in all fields of biology and engineering (in fact, it is one of the primary questions that brings the two fields together). Control systems engineering is one of the only fields, however, to actually have devised a formal, mathematical notion of robustness. Roughly, a system is *robust* to a given *perturbation* if its behavior degrades gracefully as the intensity of the perturbation increases [**55**]. Robust behavior in the face of large discrete changes, however, is still a subject in need of attention.

For synthetic biologists, the most immediate challenge may not be designing robust molecular systems. Instead, the real challenge is to understand why and how biological systems are robust. Much of systems biology concerns this question [**18, ?, ?**] and successful explanations there promise to greatly inform the designs of synthetic biologists.

Stochasticity: Imagine a computer each of whose bits in RAM randomly flip from 0 to 1 and from 1 to 0 several times a day (meaning that the number of flips per second is huge). Such a computer would have to be programmed very differently than computers today. Unfortunately, randomness is all over the place in cells. It arises from several sources. First, because cells are small and molecules are even smaller, thermal fluctuations are actually quite large compared with signal intensities. A chemical reaction, for example, is just a description of the random event that the reactants happen to combine in just the right geometry to react, the randomness arising from the giggling and jostling of the reactants and all the other molecules in the cell. This kind of "noise" is called *extrinsic noise.* Second, the number of molecules of any given type (called the *copy number* of the molecule) inside a cell may be quite small, say from zero to 50 at any given time. The event that one of those molecules reacts with another is also a random process. This kind of randomness is called *intrinsic noise.*

Stochasticity can be readily seen in single cell observations [**?**]. Isogenic cells with essentially the same initial conditions can diverge in their behaviors within a relatively short period of time, due only to the random interactions of the molecules inside of them. Stochasticity is both compensated for and exploited by single cells. For example, the process of *mitosis* (cell-replication) has to have very little randomness in it to produce an essentially identical copy of a dividing cell, and much of the circuitry responsible for this behavior is presumed to minimize errors [**?**]. On the other hand, randomness may be used by populations of cells to diversify strategies. For example, soil bacteria deside to either become *competent* (take up DNA from the environment) or not using essentially a genetically encoded coin-flipping mechanism [**?**]. The result is that a precise percentage of a population of such bacteria will be competent, while others continue to grow. In a fluxuating environment, such a strategy might be the best bet to ensure survival. (THIS SHOULD BE A DIFFERENT EXAMPLE).

Synthetic biologists need to learn to compensate for and use stochasticity as well. For example, a fundamental question is how does noise propogate through biochemical reaction networks. If an intermediate signal is noisy, how is that noise suppressed in the ouput [**?**]? Also, engineers have found great uses for random number generators in computer codes (e.g. in security). How can we make and employ random number generators in cells?

Debugging: Electrical engineers use oscilloscpes and network analyzers to characterize and debug circuits. Computer scientists use debuggers and runtime analyzers to find bugs. What do synthetic biologists use? It turns out to be very difficult to determine what is actually happening is a cell. The information that is ideally required is a core-dump: we want to know the copy number of every kind of molecule in the cell as well as its state (e.g. phosphorylated or not, how it is folded, what it is attached to, etc). Unfortunately, we can't get this information. But there are several things we can do. We put GFP and similar reporters (there are about six colors available) that we can see. We can tag proteins with molecules that fluoresce in some situations but not in others [**?**]. We can take a population of cells and sort it by size, color, and fluorescence intesnsity [**?**]. We can kill the cell or, more commonly, a population of cells and measure the amount of RNA or protein to determine which genes were on just before the cell was killed. We can watch cells grow under the microscope and even see where certain proteins are being expressed. Every year more techniques for seeing what is happening inside cells are invented. But it is likely that we will be in the dark about most of what is going on inside cells for a good long time.

Synthetic biologist can respond to this rather fundamental limitation in a different ways. One idea is to focus on engineering debugger circuits that could be included in easy to use strains of bacteria. The idea would be that the cells have built into them standard debugging tools that can be turned on or off by external signalling. This is pretty far out, but gives a flavor of what is really required to do solid engineering with cells. A more down to earth idea is to carefully ask the question: Given that there are a limited number of reporter molecules (such as GFP) that can be used at any given time, where should they be put with respect to a new proposed design to yield the maximumn amount of information? And more generally, what information about the behavior of the cell can be infered from a dynamic trace of those reporter molecules? Recent work, for example, demonstrates what information can be gleaned from just the noise characteristics of gene expression [**?**, **?**].

Evolution Again: Recall the discussion above concerning evolution: It is a fundamental problem in synthetic biology that the systems we build will be deployed in evolving populations. We desire two things. One, the systems we build do not get evolved away; and two, the systems we build do not evolve into something we cannot control. For the time being, the main problem in synthetic biology is the former [**?**]. Primarily, dropping circuits is a fantastic strategy for a population since the circuits we so far know how to design put such an enormous metabolic load on the cell. As we get better at synthetic biology, expect more graceful circuits that do not interfere with normal processes. Even then, however, evolving-out will be a problem. A further improvement is to couple the behavior of the desired circuit with the very survival of the cell. For example, suppose that the cell is grown in antibiotics (a ver common way to test for successful incorporation of a synthetic gene is to also add a gene encoding for an antibiotic). Next suppose that part of the normal functioning of parts of the synthetic circuit is to build parts of an antibiotic protein and assemble it. If any part of the circuit drops out, the antibiotic is not made and the cell cannot survive. This idea is similar to the idea of protecting software from being copied or changed by having it occasionally compute a checksum of its own bits and aborting if the checksum does not add up.

1.4. The Ethics and Risks of Synthetic Biology

Synthetic biology has the potential to change everything, for the better or otherwise. New life forms or new variants on existing life forms could be dangerous, either accidentally or intentionally. Many people question whether synthetic biology should be studied at all, given its *dual use* nature. The question is, should we be allowed to engineer living systems? One answer is, I think, is that *we already do engineer living systems* – just not very carefully.

In fact, we use living systems in a number of ways to positively impact our health and economy. Bacteria help process nutrients in our guts and digest waste in landfills. Spiders control insect populations. Bees pollinate our crops. Algae help supply the world with oxygen. Exotic trees in the Amazon might produce molecules that help us treat diseases. It almost seems as if these organisms are in our service. On the other hand, you can get a tapeworm, bacterial meningitis, herpes, HIV, the flu, and a whole host of other nasty problems if you interact with the natural world in an unlucky way. Furthermore, locusts could eat your crops, termites can eat your home, or a goose could fly into the jet engine of your airplane to Boise.

In any case, some of the things we have done technologically and socially over the last several thousand years have resulted in vast changes to the natural world that, to date, are far more significant than changes due to direct genetic tweaking.

And the rate at which we are altering the environment by non-genetic means is increasing. When the environment is changed and a new niche emerges, some part of the natural world adapts or possibly collapses sometimes very quickly. Examples: The use of pesticides has resulted in new strains of more virulent pests. The use of antibiotics has resulted in more virulent strains of infectious bacteria. Our traveling the world in airplanes has allowed for completely new ways for infectious diseases to spread. Our use of artificial sweeteners may have made many people obese. It is very likely that many emerging infectious diseases and conditions will result directly from new niches that were accidentally created by for some totally innocuous sounding purpose.

The point is that our current understanding of everything from ecology to molecular biology is not sufficient for us to begin to evaluate the introduction of a new policy or technology (even if it does not seem to be directly related to genetic engineering) in terms of its impact on the well-being of our species. That is a frightening world to live in because we are not in control.

Furthermore, nature is designed to manipulate genetic information very well already. When a population of bacteria finds itself in a new niche (such as in a moist air-conditioning duct in an office building or the bilge water of a cargo ship) having strange new nutrients or surprising new stresses, the population evolves – often very quickly. The population of bacteria does this in what seems like a directed way, but it is really “just” evolution. Now, evolution is simple to describe, but it can move in surprising directions in seemingly deliberate ways as we discussed above. While still obeying the laws of evolution, bacteria have evolved techniques to evolve quickly and without merely randomizing their genomes. For example, a recent paper in Science showed that when a population of *Streptococcus pneumoniae* is subjected to antibiotics, the bacteria in the population begin to share DNA among each other [**?**]! This allows each bacterium in the population to try out new genes to fight antibiotics like you might try out a new piece of shareware from the Internet to combat adware.

We are surrounded by many other examples of rapidly evolving organisms that seem to have a greater facility with genetic engineering than we do (they are better at directed exploration to be more exact). HIV and avian flu evolve very quickly within each host they infect, thereby adapting to the host and possibly to a treatment regimen. Whats more, microorganisms evolve much more quickly than we do as a species because their life spans are shorter. In contrast to humans, who seem to be at the mercy of their biological environment, the natural world is actually very adept at surviving and even flourishing in the face of dramatic changes and attacks.

But our technology evolves for us. Computers evolved from massive, single purpose beasts the size of buildings to sleek personal digital assistants that inhabit our pockets (the PDA is just one family of computer among many flourishing today). Automobiles, agriculture, buildings, financial markets, power plants, warfare, chewing gum, and arthritis drugs have all undergone similar evolutions from crude beginnings. All of these technologies augment (or at lease change) our ability to survive in the natural world. Not much about us genetically speaking enables us to live in almost every part of the globe (except our brains). It is our technology and accumulated knowledge that enables us to survive mat present levels. Our technology is now allowing us to re-program living systems directly – which is a fundamental shift in the place of humanity in the universe.

If technology is truly subject to mutation and selection, then its development is essentially a random process. New technologies are developed, by engineers, based on old ones that work well. Customers use products that serve their current needs. Old technologies become obsolete and eventually extinct. Even better, our technology is co-evolving with us and with other life forms on our planet. However, just as biology is more likely to move in some directions than others, we ourselves can, to some extent, control the direction that our technologies evolve (with policy and planning). By better understanding the natural world and our place in it, we can develop technologies that make us safer, healthier, and happier. Synthetic biology is hopefully one of those technologies.

Of course, any sufficiently new and powerful technology has risks that need to be understood and managed. One of the startling aspects of synthetic biology (which is also what makes it fun to study) is that new potential applications (safe or otherwise) seem to pop up every day. Therefore, we simply do not know what we (or someone else) will be able to build tomorrow. Furthermore, the technology required to build something new in this field is suprisingly simple, making it available to anyone. How the uses of synthetic biology should be regulated is a difficult and open question. Two most-likely unworkable ideas present themselves. First, synthetic organisms could be regulated like toxic chemicals. Second, synthetic organisms could be regulated using the existing bio-safefty regulations for infectious agents (from *Salmonella* to the 1918 flu). Only laboratories capable of confining such organisms would be allowed to work on them. Both of these options are problematic because they view the organisms as the risk. In fact, it is the information – merely a sequence of A, T, C and Gs – that really defines a life form. Technologically, we are close to the day when almost anyone will be able to boot up a new organism starting only with the genetic code defining it.

Thus, what really needs to be considered, is how to keep track of the information, the code, and thre recipes of synthetic biology. A good analogy is with computer viruses. If you look up computer viruses on the Internet, you will find all sorts of descriptions on how to make them. Furthermore, you can find manuals on how to program in a variety of languages, and specifications of network protocols and operating system APIs to help you write very cool Internet applications, and also very nasty computer viruses, worms, trojan horses, and the like. The equivalent information for living systems is now being generated, compiled, and disseminated. But it is one thing to bring the Internet down, and another to bring the biosphere down.

Thus, among the goals of synthetic biology should probably be to (a) to understand in what ways synthetic biology is the same and different from existing bio-technologies so that it can be regulated; and (b) to understand how to recognize or test for the potential impact of a new technology on exiting life. The latter requires really a new, quantitative (and I would argue, engineering-based) understanding of living systems.

1.5. Overview

This text is an attempt to provide a foundation for the design, modeling, and analysis of synthetic biological systems. It focuses primarliy on aspects of dynamical systems and control. The goals are essentially: (1) introduce the theoretical issues of synthetic biology to students studying control engineering; (2) introduce control and dynamical systems to students studying synthetic biology; (3) to prepare students to understand the research in the area so that they can begin to contribute to the field. The text does not cover many other very important issues in synthetic biology, such as experimental techniques, or logic and computer science. We are preparing a laboratory manual the *does* introduce basic genetic engineering with bacteria, that is available online.

We assume a minimal background in dynamical systems (basic differential equations, linear algebra, and probability) and essentially no background in biology – although an undergraduate course in biochemistry would be very helpful.

The topics included in the text are geared toward understanding the behavior of systems that researchers have actually built or systems found in nature that have been well characterized. The goal is to focus on systems that have some hope of being implemented and the theoretical issues associated with getting them to work – as opposed to more speculative ideas and architetures that are (presently) only of abstract interest.

Therefore, the topics include the following. First, we give a very high level overview of the operation of procaryotic cells to ground the discussion of specific systems later. We give an overview of synthetic biology *in vitro*, as an alternative experimental reality. There is a breif review of differential equations, mostly to remind the reader and set notation. Then we discuss the main models used in systems and synthetic biology, namely mass action kinetics, enzyme kinetics and stochastic kinetics. The approach to enzyme kinetics is somewhat different than in other texts in that we first introduce the *singular value perturbation* method from nonlinear systems, and then show how to apply the method to mass action kinetics models to obtain enzyme kinetics models. We also discuss the potential for the misuse of enzyme kinetics models.

As discussed above, models in synthetic biology are to be viewed with suspcion, primarily because there are no well-established first principles from which to derive models. Typically the theory of chemical kinetics is applied to the reactions inside a cell, but the assumptions required by chemical kinetics are not really met (well-mixedness, point mass molecules, 3D difussion, etc.). Therefore, we describe how to evaluate how good a model is and how to obtain models from data. In particular, we discuss the composition of systems and what can go wrong; robustness and sensitivity; and parameter estimation and system identification.

1.6. Problems

1.1) Recall the five capabilities we associate with living systems: sensing, actuation, control, information storage, and evolution. Which of these capabilities does a car have? How is the capability instantiated? What other technologies have these capabilities?

1.2) Visit the *Kyoto Encyclopedia of Genes and Genomes* at

```
http://www.genome.ad.jp/kegg/
```

and search for “butanol”. Find where it says “Compound” and click on the first compound. Then click on the pathway for Butanoate metabolism. If you have a microorganism that synthesizes Butanoyl-CoA, enzyme could you use to get Butanoate? What organisms could you get the gene for this enzynes from (an exhaustive list is not required)?

1.3) Describe a technology with which you are familar. In what ways is it robust? In what ways is it fragile? How have designs of the technology evolved to become more robust?

1.4) Look up the *International Genetically Engineered Machine* (iGEM) competition using your favorite search engine. iGEM is a synthetic biology competition for undergraduate teams. Look over the team wikis from years past and find (a) a medical application; (b) a biofuel application; (c) a components-based (e.g. genetic logic gates, amplifiers, etc) project. Describe each project briefly and speculate what engineering principles might be brought to bear on the problem.

1.5) Look up the *Obligation of the Engineer* using your favorite search engine. It describes what responsibilities engineers have when practicing their art. How could this oath be changed into an *Obligation of the Synthetic Biologist*?

Chapter 2

Basic Biochemistry

2.1. *E. coli*: The Model Procaryote

Synthetic biologists work with a variety of organisms. The choice of which organism depends on the application in mind and also on the innate abilities of the organism. For example, if the goal is to produce butanol from sunlight, an organism that photosynthesizes and that normally produces some useful immediate precursor to butanol is an appropriate starting point (for example, the *cyanobacteria*). If the goal is to develop a medical treatment, a simple eukaryote (such as the yeast *S. cerevisiae*) would be appropriate. For our purposes, which are to design and debug basic, portable biochemical circuits and regulatory processes, the ideal organism is the humble bacterium *Escherichia coli*, or *E. coli* – shown in Figure 1.

E. coli is very well studied. Many of the most important genetic and molecular processes in biology were first discovered in *E. coli*, and at least eight Nobel prizes are associated with discoveries made with it (e.g. Monod, Lederberg, Taum and Beadle, ...). *E. coli* is, furthermore, the workhorse in almost every lab that uses DNA: It is used basically as a DNA factory – happily replicating any DNA you put into it. *E. coli* are used in industry as well. For example, the production of insulin, which used to be obtained from pig pancreases, is now done in genetically modified *E. coli* [**?**]. Because *E. coli* is so well understood, if *you* want to know how something in *E. coli works* so that you can hack it, you will more often than not be able to find someone to help you. In contrast, if you work on an obscure microorganism, and you will most likely be on your own.

E. coli is found naturally in the human gut, along with numerous other kinds of bacteria. Humans can't live without their bacteria, which help them digest their food, among other responsibilities. Most of the cells on and in your person, in fact, are (very small) bacteria cells and not (big) human cells. The strains of *E. coli* used in the lab are all descendants of *E coli* K-12, which was isolated at Stanford University 1922 from human feces. It was kept there for years as a stock strain and has, in fact, evolved to live alone in the comfort of the lab. Due to their relaxed lifestyle, K-12 can barely survive outside of the lab and is generally considered to be harmless (unlike the strains of *E. coli* that sometimes infect foodstocks [**?**]). You can now obtain from a variety of sources a huge range of strains related to K-12, each with its own "features". Some strains produce lactose digesting enzymes, some

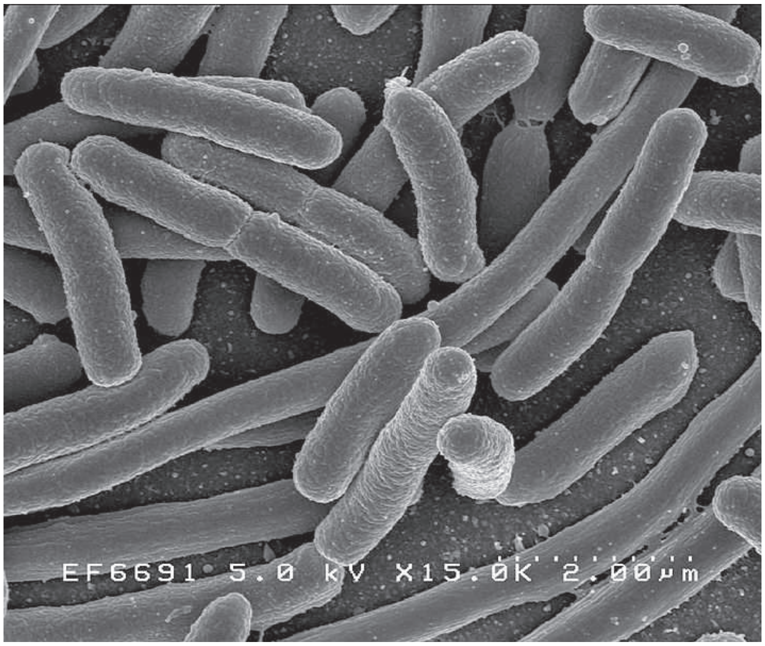


Figure 1. A scanning electron micrograph image of *E. coli* cells. Each cell is approximately 2 μm long and has a volume of about 1 μm^3. Several of the cells in this image have just divided end-to-end, producing genetically identical daughter cells. Credit: Rocky Mountain Laboratories, NIAID, NIH.

can conjugate with each other, some have had their DNA repair genes knocked out, and so on [**1**].

To some extent, if a synthetic biologist can get a device to work in *E. coli*, the principles of how its works should apply to other organisms as well and often the circuit can be transplanted as is into other organisms. As Jaques Monod said, "All that is true for the *Colon bacillus* [e.g. *E. coli*] is true for the elephant" [**?**]. Monod's observation is that all life forms on earth use essentially the same architecture – of which this chapter is an overview.

In particular, this chapter describes, at a very high level, the architecture of organisms like *E. coli* – that is *procaryotes* (cells not having a nucleus). The goal is to ground the ideas that appear later in the book in systems inside a well-understood model organism. Note that this chapter is not even remotely a good substitute for a textbook on biochemistry [**2**], a textbook on cell biology [**5**], or a text book on bacterial genetics [**45**].

2.2. Molecules in the Cell

Bacteria, including *E. coli*, are single celled *procaryotic* organisms. Among other things, this means that they do not have a nucleus containing their genomic DNA, and they do not have particularly sophisticated ways of processing their RNAs and proteins. Rather, the DNA and most of the other molecules that make up an *E. coli* cell's machinery, are located in the main interior of the cell called the *cytoplasm* or they are part of the cell *membrane* enclosing the cytoplasm. In this section we describe at a high level what most of the important molecules are called and what they do in the cell. It is difficult to know where to start with such a description as each type of molecular interacts with other types so that in describing RNA, we need to talk about DNA and proteins, for example. The order of presentation below is therefore somewhat arbitrary. The reader might consider reading this section twice to resolve references to one type of molecule while introducing another type of molecule.

DNA. Perhaps the most important kind of molecule in the cell is *deoxyribonucleic acid* or DNA. A big DNA molecule, called the genomic DNA, stores almost all of the information required to build and run an *E. coli.* DNA molecules are long, oriented polymers (long chains) of four different types of *nucleotides*, as seen in Figure 2. Each nucleotide consists of a *base*, adenine (or A), thymine (or T), guanine (or G) and cytosine (or T), attached to a deoxyribose sugar, and a phosphate group. The nucleotides in a DNA molecule are strung end to end to form a long chain or, more often, a circle. Since the nucleotides in a DNA molecule can appear repeatedly and in any order, there are in fact 4^N different DNA molecules of length N. For a standard strain of *E. coli*, $N \approx 4.6$ million. If we were to randomly make DNA molecules of this length and try to "boot them up" inside a cell, the chance we'd accidentally make a working organism like *E. coli* is approximately one in 4^{10^6}. Apparently, we need to understand how to put together (or program) DNA molecules more systematically.

DNA molecules are typically not found alone. Instead each DNA strand is attached, with *hydrogen bonds*, to a *complimentary strand* via *Watson-Crick* base pairing. It turns out that A sticks to T and C sticks to G, so the strand $5' - ATAGCA - 3'$ is complimentary to $3' - TATCGT - 5'$. In this notation, the strands are oriented from the $5'$ to the $3'$ ends (see Figure 2). It is this complimentary nature of DNA that makes it ideal for representing and copying information. If you (or the cell) unzips a double stranded piece of DNA, each makes a template for the construction of a new complimentary strand. This copying occurs every time a cell divides with incredible accuracy: the error rates are about one mismatch in every 10^9 bases copied [**?**].

In later sections of this chapter, we describe at a high level how DNA is interpreted by the machinery in the cell to direct growth, program how the cell responds to its environment, and so on.

RNA. Although we just said that DNA is the most important molecule in the cell, it is hard to overstate the importance of *ribonucleic acid* or RNA. RNA is structurally similar to DNA except that the bases are attached to a ribose sugar instead of a deoxyribose sugar and instead of thymine RNA contains uracil. The similarity ends there however. While DNA is a stable information carrier, RNA is less stable and to some extent defines the changing internal state of a cell. Furthermore, RNA is generally found single stranded, often modified by the addition of other small molecules, and is typically folded up on itself more like a protein, as shown in Figure 3. Finally, while DNA is a massive circular double-stranded beast, RNAs are typically short (from 10 to 3000 bases long).

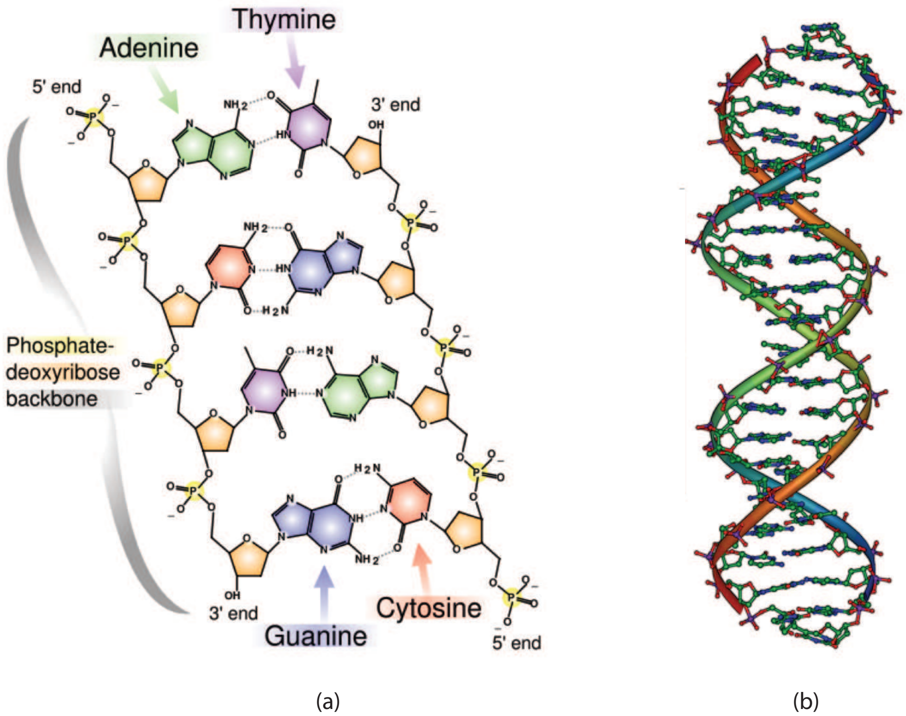
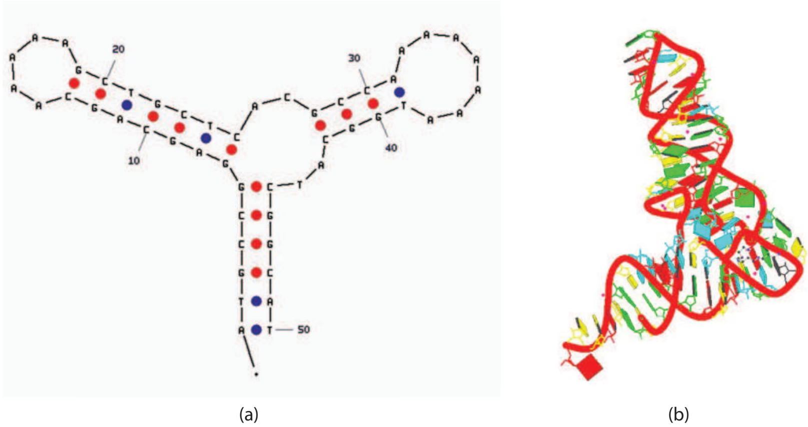


Figure 2. The structure of DNA. (a) Bases and base pairing. DNA consists of polymers of bases of four different types: adenine (or A), thymine (or T), guanine (or G) and cytosine (or T) each attached to a deoxyribose sugar. Two stands of DNA will form a duplex if they are complimentary: A sticks to T and C sticks to G. Note also that DNA has an orientation: the 5' end is different than the 3' end. (b) The double-helix. Double stranded DNA folds into a double-helix the 3D structure of which can form highly specific binding sites for regulator proteins.

All RNAs in the cell are *transcribed* (see below) from regions of the genome called genes. Once transcribed, an RNA is destined for one of a surprisingly many roles in the cell. *Messenger RNA* or mRNA is an intermediate representation of a protein. Its role is to carry the information for how to build a protein from the genome to the protein building machinery. The protein-building machinery itself is built primarily from two types of RNA called *transfer RNA* or tRNA and *ribosomal RNA* or rRNA. tRNAs are the interface between triples of bases called codons (found on on mRNAs) and the amino acids corresponding to the codons. Ribosomal RNAs glob together into molecular machines called ribosomes that translate mRNAs into proteins using tRNAs (this process is described below). Finally, there regulatory RNAs of many type. For example, *small interfering RNAs* or siRNAs are bits of RNA that are complimentary to regions of mRNAs that prevent their translation into protein. Another type of regulator RNA are the various *ribozymes*, which are RNAs that exhibit catalytic activity, for example, cleaving other RNAs in specific (and programmable) regions.



(a)

(b)

FIGURE 3. RNA. (a) The topology of a typical RNA molecule. The RNA strand folds up on itself in predictable ways with A sticking to U and C sticking to G. (b) The 3D structure of a transfer RNA.

Protein. Proteins are folded up polymers consisting of chains of sub-units called *amino acids*. There are 20 different amino acids (see Figure 7) that are found in normal living systems. Amino acids all have the same basic form consisting of an amino group (H_3N) and a carboxyl group (CO_2H) on either side of a $H - C - R$ group. The R is another group called the *residue* that determines the type of amino acid. Each amino acid is different: it may be hydrophobic or hydophyllic; it may be polar or not; etc.

Amino acids can form chains due to their modular nature. The amino group of one can stick to the carboxyl group of another forming a peptide bond and releasing a water molecule, as shown in Figure 4. By continuing this process, any chain of up to thousands of amino acids can be formed. Due to the different chemical natures of the amino acids, the chain will *fold up* into different shapes. The resulting protein will have pockets and grooves and bumps each with a function determined by the amino acids near that feature. Thus, proteins are programmed blobs of matter with programmed chemical functionality – and the possible functionality seems to be limitless. Almost every important function inside a cell is performed in part by a protein. They regulate the expression of genes; help transport molecules in and out of the cell; a particular protein *enzyme* can very specifically increase the activity of a particular chemical reaction; some proteins polymerize into long, rigid chains that give cells structure; protein motors can pull cargo around inside cells; some proteins are hormones that carry signals to other cells; and on and on. This programmability of function is unparalleled.

An example protein is GFP, discussed in the introduction. It consists of a sequence 264 amino acids with a mass[1]of just over 27 kDa. This sequence of amino acids, shown in Figure 5(a), folds up into a secondary structure due to the natural attractions and repulsions of its constituent amino acids, as shown in Figure 5(b). Stretches of a protein often form into recognizable substructures, either α-helices or β-sheets. The former are helical regions and the latter are regions that interact

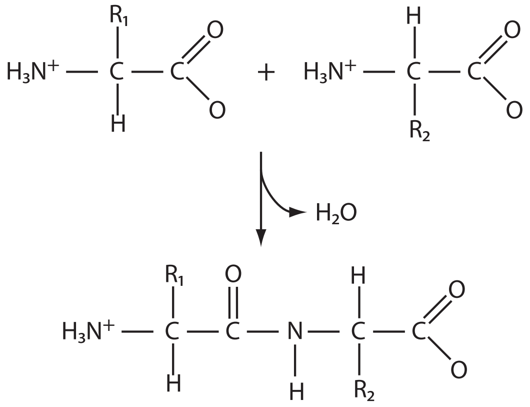


FIGURE 4. Two amino acids condense to form a peptide bond. Long chains of amino acids fold up to form proteins. The amino group of the chain is called the N-terminus and the carboxyl group is called the C-terminus.

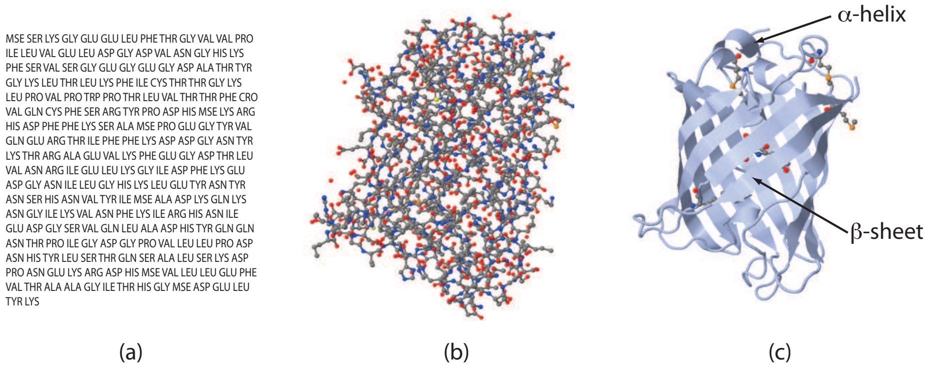

MSE SER LYS GLY GLU GLU LEU PHE THR GLY VAL VAL PRO
ILE LEU VAL GLU LEU ASP GLY ASP VAL ASN GLY HIS LYS
PHE SER VAL SER GLY GLU GLY GLU GLY ASP ALA THR TYR
GLY LYS LEU THR LEU LYS PHE ILE CYS THR THR GLY LYS
LEU PRO VAL PRO TRP PRO THR LEU VAL THR THR PHE CRO
VAL GLN CYS PHE SER ARG TYR PRO ASP HIS MSE LYS ARG
HIS ASP PHE PHE LYS SER ALA MSE PRO GLU GLY TYR VAL
GLN GLU ARG THR ILE PHE PHE LYS ASP ASP GLY ASN TYR
LYS THR ARG ALA GLU VAL LYS PHE GLU GLY ASP THR LEU
VAL ASN ARG ILE GLU LEU LYS GLY ILE ASP PHE LYS GLU
ASP GLY ASN ILE LEU GLY HIS LYS LEU GLU TYR ASN TYR
ASN SER HIS ASN VAL TYR ILE MSE ALA ASP LYS GLN LYS
ASN GLY ILE LYS VAL ASN PHE LYS ILE ARG HIS ASN ILE
GLU ASP GLY SER VAL GLN LEU ALA ASP HIS TYR GLN GLN
ASN THR PRO ILE GLY ASP GLY PRO VAL LEU LEU PRO ASP
ASN HIS TYR LEU SER THR GLN SER ALA LEU SER LYS ASP
PRO ASN GLU LYS ARG ASP HIS MSE VAL LEU LEU GLU PHE
VAL THR ALA ALA GLY ILE THR HIS GLY MSE ASP GLU LEU
TYR LYS
α-helix
β-sheet
(a)
(b)
(c)

FIGURE 5. The structure of green fluorescent protein (GFP), downloaded from the protein data bank. (a) The sequence of residues. Each three letter code corresponds to a different residue, according to the scheme in Figure 7. (b) The three dimensional or secondary structure of GFP. The relative coordinates of each residue are typically determined empirically using X-Ray crystallography. (c) The cartoon version of the protein showing more clearly its structure as a barrel-shaped molecule.

with other β regions to form sheet like structures. Figure **??**(c) shows a cartoon of GFP in which helix and sheet structures are highlighted. Sub-structures that are neither sheets nor helices are called random coils and are shown as simple lines are ropes.

Small Molecules. The are an enormous variety of small molecules in the cell (besides, of course, H_2O) that the cell is very adept at processing. Simple sugars,

[1] Masses of molecules are often measured in Daltons (Da). One Dalton is approximately the mass of one Hydrogen atom.

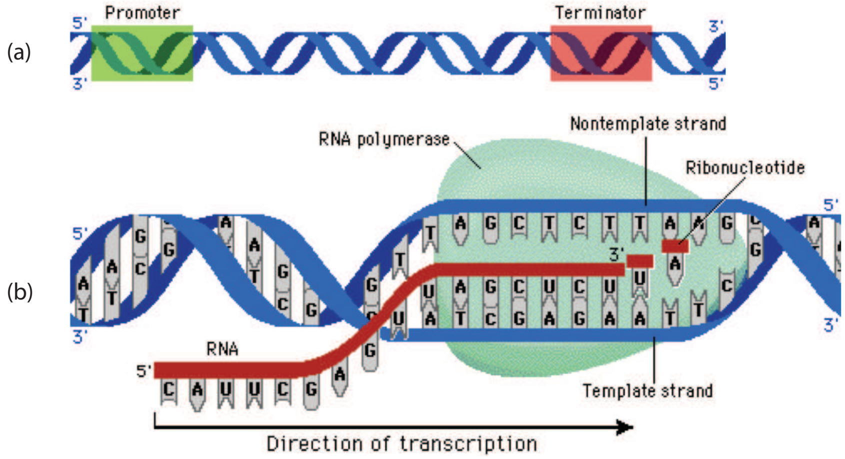


FIGURE 6. (a) A simple gene containing a promoter, a template, and a terminator region. (b) The process by which RNA polymerase (RNAP) transcribes the template DNA into an RNA transcript.

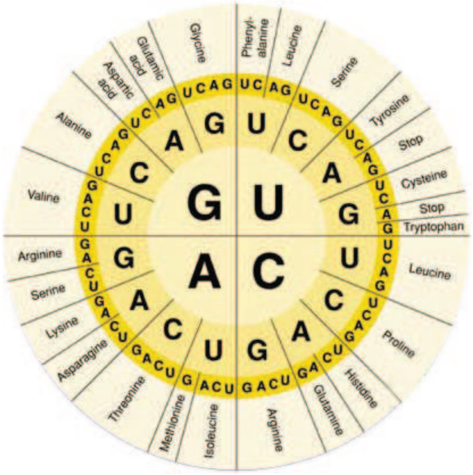
carbohydrates, lipids, alcohols, amino acids, nucleotides, signaling molecules and antibiotics are just some of the substances that cells know how to make. These molecules are of great interest to metabolic engineers who see cells as programmable chemical processing plants. By changing the expression of protein enzymes or by adding the genes for new protein enzymes into an organism, new substances can be made by old organisms.

2.3. Processes

2.3.1. Transcription. The long chain of genomic DNA inside a cell is dived up into segments called *genes*. These genes take on many forms. The most basic form consists of a short sequence of DNA called a *promoter*, after which follows a long sequence (up to thousands of base pairs) of DNA called a *template* containing the information represented by the gene, and terminated by a short sequence called a *terminator*. See Figure 6(a).

The promoter region of a gene has a specific three-dimensional structure due to the sequence of nucleotides in the region. This structure allows an important enzyme called *RNA Polymerase* (RNAP) to specifically bind to the promoter region (RNAP is actually a collection of proteins that act together as an enzyme). Once there, RNAP locally splits the DNA double helix in the template region and builds an RNA molecule complimentary to the template. The new piece of RNA is called the *transcript* from the template. In bacteria, this happens at an astounding rate of about 80 nucleotides per second [**?**]. The RNAP continues transcription of the template until it reaches the terminator, at which point the RNAP falls of the gene. See Figure 6(b).

A gene that is "on" (see regulation, below) may have many RNAP enzymes actively transcribing the gene at any given time and so there may be many transcripts from the gene present in the cell at any given time. Also, RNA is degraded fairly quickly. The signals and conditions represented by RNAs are therefore temporary.



Ala	Alanine
Val	Valine
Arg	Arginine
Ser	Serine
Lys	Lysine
Asn	Aspargine
Thr	Threonine
Met	Methionine
Ile	Isolucine
Gln	Glutamine
His	Histidine
Pro	Proline
Leu	Leucine
Trp	Tryptophan
Cys	Cycstine
Try	Tyrosine
Phe	Phenylalanine
Gly	Glycine
Glu	Glutamic Acid
Asp	Aspartic Acid

FIGURE 7. (a) The genetic code. The first letter in a codon is at the center. (b) Abbreviations.

The RNA transcripts in the cell have many possible destinies, summarized below:

Messenger RNA (mRNA) mRNA can be *translated* into one or several proteins (see Translation, below).

Transfer RNA (tRNA) tRNA is the interface between mRNA and protein. Each type of tRNA is modified to carry a specific amino acid on one end and has an *anticodon* on the other end

Ribosomal RNA (rRNA) The ribosome is a complex machine consisting of RNA and protein that translates mRNA into protein by stringing together amino acids presented by tRNAs

Other Other RNAs can fold into RNA enzymes called ribozymes or (esp. in Eucaryotes) form small antisense RNA or micro RNA

2.3.2. Translation. Some of the RNA transcribed has a chance to be *translated* into protein, as long as it doesn't get degraded first or isn't blocked by some antisense RNA. An mRNA contains a *ribosomal binding site* or RBS to which the enzyme responsible for translation, the *ribosome* binds. Once bound, the ribosome moves forward (in the 5' to 3' direction) on the mRNA until it encounters the sequence AUG – the *start codon* and also the codon for Methionine.

There are other three letter codons as well, after the start codon. Each codon corresponds to a specific amino acid according to the *genetic code* shown in Figure 7. This code is almost exactly the same in every organism ever studied – which is why a gene from one organism is likely to work when inserted into another organism. The ribosome steps through each codon starting with the start codon until it reaches a *stop codon*.

Specifically, when a ribosome parks at a codon, a tRNA having the complimentary sequence, the anti-codon, docs at the codon, held in place by the ribosome.

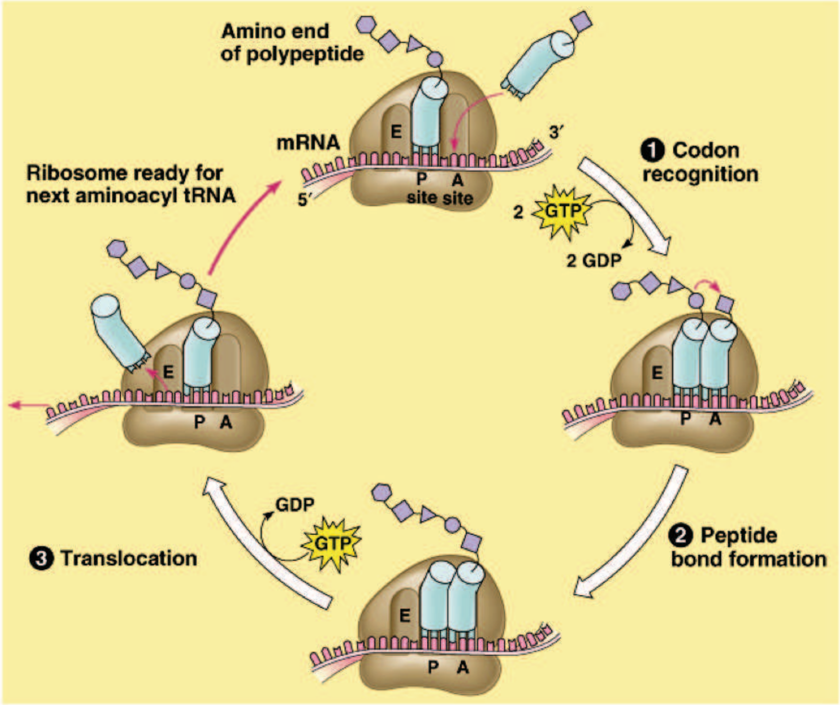


FIGURE 8. The process of translation. The Ribozyme binds onto the *ribosomal binding site* of an mRNA, moves forward to the start codon, and thne translates the codons in the mRNA into protein, ending at a stop codon.

Next, the amino acid at the other end of the tRNA is attached to a growing chain of amino acids by the ribosome. Then the ribosome steps forward to the next codon and discards the transfer RNA (without the amino acid). This process continues until the stop codon is reached at which point the sequence of codons in the mRNA has been completely translated into a sequence of amino acids. The amino acid chain folds up into a functional protein while it is being translated or very soon after translation ends.

The ribosome, which consists of several subunits of both rRNA and protein is arguably the most remarkable molecular machine in the natural world. Not only are they responsible for translation and protein biosynthesis, they do their job very well – correcting for errors and processing 12 to 21 amino acids *per second*.

In should be noted that transcription and translation, in procaryotes anyway, occur simultaneously. A given gene may have several RNAP molecules transcribing mRNAs off of it at any given time. Each of the mRNAs may have several ribosomes attached to it, translating protein.

2.3.3. Regulation. Some genes are on all the time, meaning that RNA transcripts are constantly generated by RNAP using the template in the gene. Such genes are called *constitutively expressed*. Most genes, however, are regulated in some way so that the RNA transcripts for the gene are only generated when they

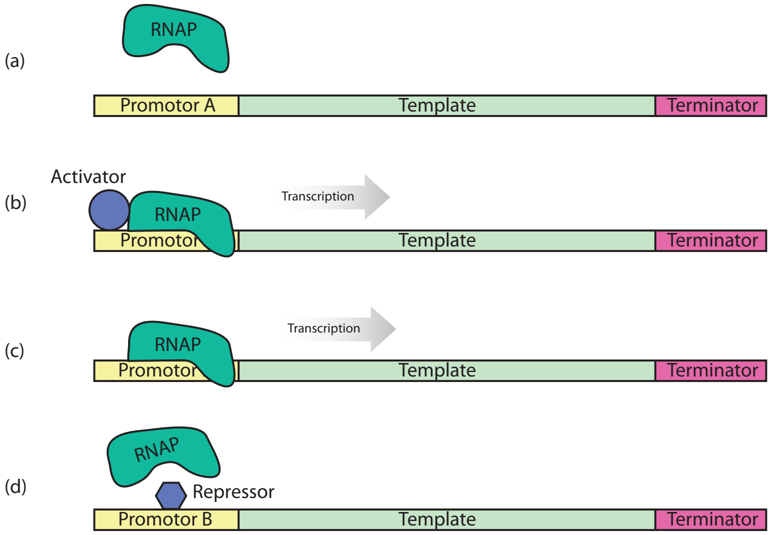


FIGURE 9. (a) RNAP has low affinity for promoter A. (b) The presence of a transcription factor specific to promoter A increases the affinity of RNAP for the promoter. (c) RNAP has high affinity for promoter B. The presence of a repressor specific to promoter B blocks RNAP from transcribing the gene.

are needed. The most common way for a gene to be regulated is via proteins called *transcription factors*.

As shown in Figure 9, transcription factors can either be *activators* or *repressors*. An activator A works by improving the affinity of RNAP for specific promoters p_A that are complimentary to A. In contrast, a repressor R works by blocking RNAP from docking to promoters of type p_R. For example, the *LacI* repressor (which is actually four identical proteins bound together into a tetramer) is found in *E. coli*. It sticks very selectively to the promoter p_{lac} due to the three dimensional structure of the DNA double helix comprising the promoter. When bound, it prevents RNAP from binding to the promoter. In general, promoters and transcription factors go together and the sequence of A, T, C and G in the promoter determines a three dimensional structure in the DNA helix that has some affinity for RNAP and some affinity for some transcription factors (either zero or more).

In addition to the above simple mechanisms, many transcription factors work in groups so that transcription occurs as some function

$$\text{rate of transcription} \;=\; f(T_1, ..., T_n)$$

of the concentrations of the transcription factors $T_1, ..., T_n$ associated with the gene.

Many transcription factors take on active and inactive forms based on the presence or absence of small molecules. Such transcription factors thus form sensors. For example, the repressor *LacI* is active when there is none of the sugar lactose around. In its active form, *LacI* represses the genes encoding the enzymes for incorporating and metabolizing lactose. In the presence of lactose, the repressor becomes inactive, and the lactose genes are transcribed (and translated). In general,

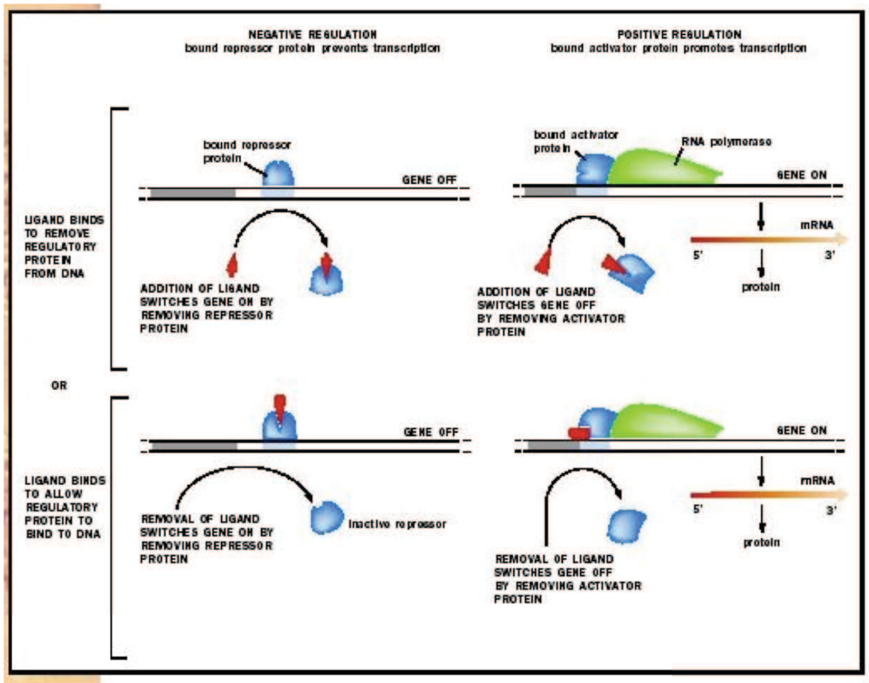

NEGATIVE REGULATION
bound repressor protein prevents transcription
POSITIVE REGULATION
bound activator protein promotes transcription
LIGAND BINDS TO REMOVE REGULATORY PROTEIN FROM DNA
bound repressor protein
GENE OFF
ADDITION OF LIGAND SWITCHES GENE ON BY REMOVING REPRESSOR PROTEIN
bound activator protein
RNA polymerase
GENE ON
mRNA
5'
3'
protein
ADDITION OF LIGAND SWITCHES GENE OFF BY REMOVING ACTIVATOR PROTEIN
OR
LIGAND BINDS TO ALLOW REGULATORY PROTEIN TO BIND TO DNA
GENE OFF
REMOVAL OF LIGAND SWITCHES GENE ON BY REMOVING REPRESSOR PROTEIN
inactive repressor
GENE ON
mRNA
5'
3'
protein
REMOVAL OF LIGAND SWITCHES GENE OFF BY REMOVING ACTIVATOR PROTEIN

Figure 10. Small molecules can activate or deactivate transcription factors in a variety of ways.

there are several ways in which a small molecule can interact with a transcription factor – these are summarized in Figure 10.

Now consider a set of protein-coding genes $g_1, ..., g_n$. Draw each gene as a circle. If a gene produces a transcription factor that activates or represses another gene, we write an arrow or a line with a bar respectively from one gene to the other. If a small molecule activates or deactivates a transcription factor, we draw similar arrows from nodes representing the molecule to the arrow representing an interaction. The result is a *genetic regulator network*, and example of which is shown in Figure 11.

There are a host of other ways in which genes are regulated, not always by proteins. For example, non-coding RNAs can interact with mRNAs in a variety of ways, either preventing the mRNAs from being transcribed, or even by allowing them to be transcribed. A good survey of these mechanisms can be found in [**28**].

2.3.4. Metabolism. A cell is a miniature chemical processing plant. Most cells can synthesize a broad variety of substances from the raw materials they take up from the environment. For example, *E. coli* can synthesize, among other things, amino acids for protein construction, nucleotides for RNA and DNA, and carbohydrates for energy storage. Furthermore, the substances that cells take up are highly variable, so many of the chemical reactions inside cells are designed to break down nutrients into key energy carrying molecules (such as ATP) and basic raw materials for later synthesis reactions (such as pyruvate). Figure 12 illustrates

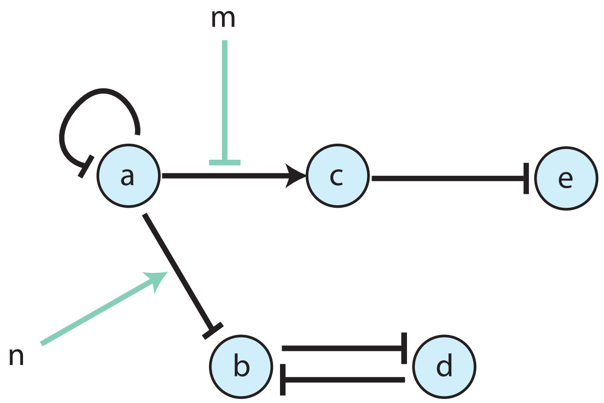


FIGURE 11. Small molecules can activate or deactivate transcription factors in a variety of ways.

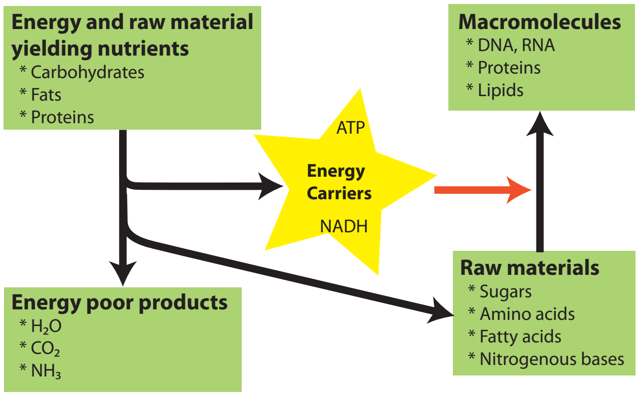


FIGURE 12. Metabolism ...

the process of *catabolism* (breaking down nutrients) and *anabolism* (synthesizing) useful molecules.

As described above, many proteins act as enzymes. An enzyme is a catalyst that increases the backward and forward rate of a chemical reaction (or family of reactions) by lowering the energy barrier between the reactants and the products. By expressing some protein enzymes and not others, the cell can steer metabolites through desired pathways. Furthermore, by sensing metabolites (for example with transcription factors that are activated by metabolites), a genetic regulatory network can *control* the metabolism.

A complete description of all of the metabolic pathways in even *E. coli* is beyond the scope of this (and most other) texts. There are an enormous number of metabolites and reactions. However, to give a flavor of what is involved, we briefly describe a key metabolic pathway that is fundamental to life, namely *glycolysis*. Glycolysis is found in every organism on the planet and likely evolved very early.

The glycolytic pathway (glycolysis) essentially is responsible for converting the sugar *glucose* into two molecules of *pyruvate* with a net gain in energy of two ATP molecules. Glucose is a basic nutrient that can either be taken up by the

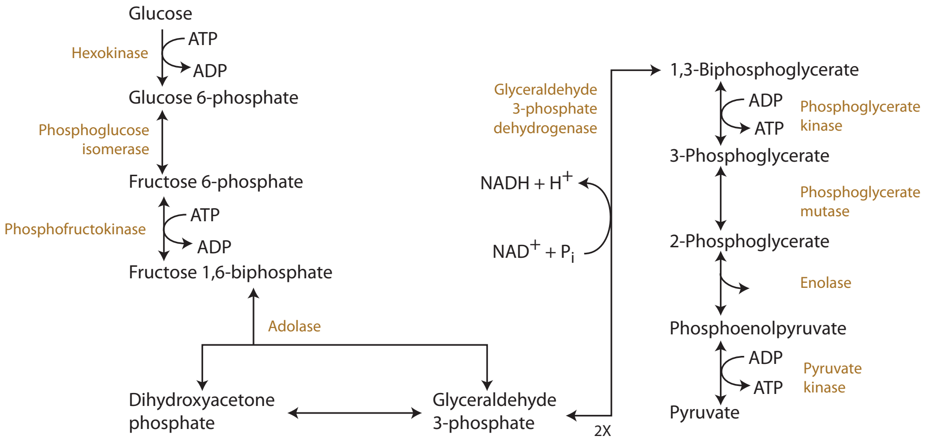


Figure 13. The glycolysis pathway.

cell from the environment or obtained from carbohydrate energy storage. It has three main destinies in the cell: (1) It can be put into storage; (2) it can be used in the synthesis of small molecules such as nucleotides and amino acids; (3) it's high-energy content can be used to make ATP. The first step of the third destiny is glycolysis. Pyruvate, the product of glycolysis, is still energetic and itself is a precursor for several pathways, the most important of which is the citric acid cycle, which milks the rest of the energy out of pyruvate by oxidizing it.

The glycolysis pathway is shown in Figure 13. It consists of a series of several steps. The names of the metabolites are shown. Each one consists of a minor change to the metabolite before it. In some steps ATP (adenosine triphosphate) is consumed to produce ADP (adenosine diphosphate), which is an energy consuming step. In other steps, ADP is consumed and ATP is produced. In the fifth step, energy is also produced via the *cofactor* NAD$^+$. This molecule is a cofactor of many steps in the metabolism of the cell and, after ATP, is one of the most common players in metabolic pathways.

Each step in the pathway is labeled by the name of the enzyme associated with the reaction. For example, the last step is catalyzed by the enzyme *pyruvate kinase*, which increases the rate of the reaction

$$\text{Phosphoenolpyruvate} + \text{ADP} \leftrightharpoons \text{Pyruvate} + \text{ATP}.$$

This step is essentially irreversible. We return to metabolism in Chapter **??**. At that point, the analysis and control of metabolic networks is discussed.

Metabolic engineers have achieved astounding results by introducing enzymes from one organism into others to rewrite their metabolic pathways. For example, an enzyme from *Artemisia annua* (sweet wormwood) that produces artemisinic acid was introduced into yeast [39]. Artemisinic acid is an expensive antimalarial drug that, if it could be produced cheaply in yeast, could be very useful in fighting malaria worldwide. However, presently it is still not cost effective to produce this drug in yeast, mainly because putting all of a yeast cell's resources into making artemisinic acid makes the yeast cell unhealthy. Increasing the yields of the artificial pathway over the past several years have involved incredibly meticulous tuning of every step of the pathway starting with Acetyl-CoA (one step after pyruvate) to artemisinic acid. It is unclear how this tuning process can be automated, but if it could, the synthesis of novel molecules in cells could become commonplace.

2.3.5. Signaling. e.g. G-protein

2.3.6. Growth and Division.

2.4. Re-programming *E. coli*

A nice feature of *E. coli*, and other organisms as well, is that an *E. coli* cell may contain one or more pieces of small, circular *plasmid* DNA. These independently replicating pieces of DNA are processed inside the cell much the same way as is genomic DNA. Furthermore, it turns out to be fairly easy to construct synthetic plasmids and stick them inside appropriately prepared cells. Once there, the

2.5. How we See What's Going On

2.6. Problems

2.1) An *E. coli* cell is an approximately 2 μm long cylinder with diameter 1 μm. What is its approximate volume? If there are about 3.6 million proteins inside a single cell, what is the concentration of protein in the cell in moles per liter?

2.2) The genetic code assigns an amino acid to each three letter sequence of nucleotides. How many genetic codes are possible? Why do you think all of the living systems (that we know of) use the same genetic code?

2.3) The molecular machinery for *copying* the DNA in a cell during cell-replication (not covered in this text) has very sophisticated error correction mechanisms. In contrast, the error correcting mechanisms for RNA polymerase (RNAP) are fairly limited – and RNAP therefore makes more mistakes. Why should this be?

2.4) Find two different promoters in the BioBricks registry at

`http://partsregistry.org/`.

Show the sequence of nucleotides in each and explain their similarities and differences.

Chapter 3

Differential Equations

Differential equations are the fundamental language of dynamical systems, where they describe how quantities change with time. In synthetic biology, the quantities of interest might be the concentration of a particular set of proteins or metabolites or the concentration of cells and nutrients in a solution of bacteria. When dealing with a large number of cells, concentrations make sense and we think of them as continuous. When dealing with single cells where there may be only 20 of a given protein at a given time, concentrations do not quite make sense. Nevertheless, many researchers assume that concentrations are continuous anyway and often arrive at useful mathematical models.

In this chapter we look at the basic definitions, examples and properties of ordinary differential equations. In Chapters 4 (Mass action Kinetics) and 5 (Time Scale Separation) we describe how to derive and analyze differential equations using modeling assumptions common in synthetic biology.

The material given in this chapter is meant as a reminder and means to fix notation. A much more complete introduction to differential equations and dynamical systems requires a textbook, such as the one by Hirsch and Smale [**27**] or [**?**].

3.1. Definitions and Examples

A system of differential equations has the form

$$\begin{aligned} \dot{x}_1 &= f_1(x_1, ..., x_n, t) \\ &\vdots \\ \dot{x}_n &= f_n(x_1, ..., x_n, t) \end{aligned}$$

where t is time, $x_1(t), ..., x_n(t)$ are time-dependent *state variables*, and $f_1, ..., f_n$ are functions. We usally write x_i instead of $x_i(t)$. The notation $\dot{x}_i$ means $\frac{d}{dt}x_i$. In vector notation, the above system is more compactly written

$$\dot{x} = f(x, t) \tag{3.1}$$

where $x(t) \in \mathbb{R}^n$ and $f : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$.

Example 3.1. Suppose that $v_X(t)$ represents the concentration of a constitutively expressed protein X. Let the rate at which the protein is produced be $k_1 > 0$

and the rate at which it is degraded be $k_2 > 0$. Using these two parameters, a (very) simple model of protein production is then

$$\dot{v}_X = k_1 - k_2 v_X. \tag{3.2}$$

Note the term $-k_2 v_X$ describes the fact that the higher the concentration of X, the more "degradation events" occur per minute. The production of X described by the k_1 term, on the other hand, is indpendent of how much X there already is in the system. ▲

Example 3.2. Consider a population of of bacteria growing in a solution of growth media. The two variables of interest are the amount of bacteria x_1 (in grams) and the amount of nutrient x_2 (in grams) in the solution. Monod described the dynamics of this situation as follows [**?**].

$$\begin{aligned} \dot{x}_1 &= \frac{v x_1 x_2}{k + x_2} \\ \dot{x}_2 &= -\gamma \frac{v x_1 x_2}{k + x_2} \end{aligned} \tag{3.3}$$

where v, k an γ are parameters. Note that if $x_1 = 0$ or $x_2 = 0$, then there is no growth since there are either no bacteria or no nutirents. Also notice that as the nutrients $x_2 \to \infty$, we get that the growth rate $\dot{x}_1 \to v x_1$. This models the fact that the population cannot grow infinitely quickly. The parameter γ describes the amount of nutrients needed to make one bacterium. Finally, the parameter k describes the *half-saturation* point, which is the amount of nutrient required to make $\dot{x}_1 = v x_1 / 2$. ▲

A *solution* to the differential equation (3.1) is an explicit function of time $x(t)$ that satisfies the equation. Explicit solutions are readily available for some types of differential equations. For example, equations in which each term appears linearly are easy to solve. However, for most differential equations, an explicit solution is difficult or (provably) impossible to find. In such cases, other analytical techniques must be used.

Example 3.3. The solution to linear differential equation (3.2) is

$$v_X(t) = e^{-k_2 t} \left(v_X(0) - \frac{k_1}{k_2} \right) + \frac{k_1}{k_2} \tag{3.4}$$

where $v_X(0)$ is the initial concentration of X. An explicit solution such as this tells us almost everything we might want to know about the behavior of the system. For example, it is easy to see that as $t \to \infty$, the concentration $v_X(t) \to k_1/k_2$.

In contrast, the differential equation 3.3 is nonlinear and has no explicit solution. ▲

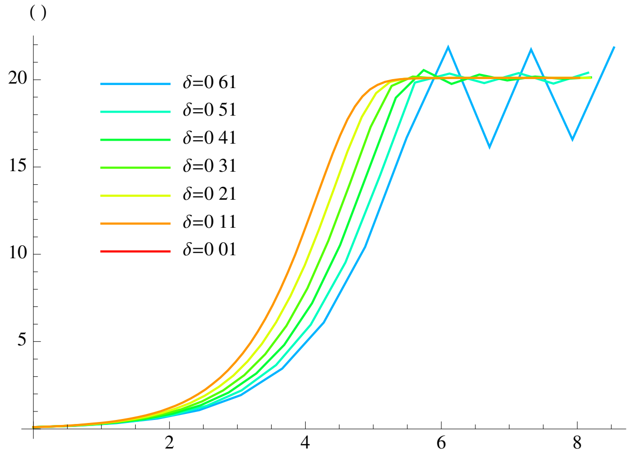


Figure 1. Simulations of bacterial growth using Euler integration with differing time steps. Large time steps give more results, but fewer steps. Smaller time give higher fidelity results.

3.2. Simulation

Probably the first thing most analysts do with a new set of differential equations is simulate them. There are a wide variety of simulation tools available. For example, one can use the `NDSolve` function in Mathematica (as described in Appendix A) or the `ode45` numerical integrator in MATLAB. All of these methods are based on essentially the same idea. One approximates the state $x(t+\delta)$ in some way using the state $x(t)$ and the vector field $f(x,t)$. In this section we breifly describe a simple method of numerically integrating a set of differential equations, called *Euler* integration. Once this method is understood, more advanced methods should make more sense.

In Euler integration, we use the fact that

$$\frac{dx}{dt} = f(x,t) = \lim_{h \to 0} \frac{x(t+h)-x(t)}{h} \approx \frac{x(t+\delta)-x(t)}{\delta}$$

when δ is small. Solving for $x(t+\delta)$ gives

$$x(t+\delta) \approx x(t) + \delta f(x(t),t). \tag{3.5}$$

We then have a simple algorithm.

```
(1) initialize $x_0$, $t_0 = 0$ and $k = 0$
(2) while $t_k < t_{max}$
(3)   $x_{k+1} = x_k + \delta f(x_k, t_k)$
(4)   $t_{k+1} = t_k + \delta$
(5) end while
```

The fidelity of this approximation, and the intensity of the compuation required, increase as δ decreases. The next example illustrates these observations.

Example 3.4. Figure 1 shows seven simulations of the bacterial growth system from Example 3.2 with $v = 2$, $k = 5$, $\gamma = 0.5$, $n(0) = 10$ and $x(0) = 0.1$. Note that the curves for $\delta = 0.11$ and $\delta = 0.01$ are essentially the same. Clearly the curves for $\delta \geq 0.21$ are of questionable fidelity. A large δ even introduces oscillations. ▲

3.3. Equlibria and Stability

A state x^* that is a root of f, so that $f(x^*, t) = 0$, is called an *equilibrium*. A system may have zero, one or many equilibria. Solutions to a system may tend toward x^*, meaning x^* is stable, or they may veer away from x^*, meaning x^* is unstable. More formally, we are usually concerned with the following notions of stability.

Definition 3.5. An equilibrium x^* is *locally asymptotically stable* if there exists a $\delta > 0$ such that

$$||x(0) - x^*|| < \delta \;\Rightarrow\; \lim_{t\to\infty} x(t) = x^*.$$

If δ can be anything in the above, then x^* is *globally asymptotically stable.*

A system is said to be (locally or globally) *exponentially stable* if the rate of convergence to x^* is exponential, that is, if there exist constants $a, b > 0$ such that

$$||x(t) - x^*|| \leq a||x(0) - x^*||e^{-bt}$$

for all $t \geq 0$.

Example 3.6. Suppose that $k_1 = 1$ and $k_2 = 1$ in Example 3.1. The dynamics of X are then

$$\dot{v}_X = 1 - v_X.$$

Setting the derivative equal to zero gives the equilibrium $v_X^* = 1$. As noted in Example 3.3, the solution is

$$v_X(t) = (v_X(0) - 1)e^{-t} + 1,$$

which tends toward 1 exponentially fast as $t \to \infty$. Thus, the equilibrium $v_X^* = 1$ is globally exponentially stable. ▲

Proving that an equilibrium of a system is stable can be difficult. If the system is linear, then it is easy. If the system in nonlinear, and one wants to show local stability, then the system can be linearized (see below), and the problem becomes easy. Proving that a point is globally stable is hard, however. Also difficult is characterizing the region (finding δ in Def. 3.5) around a locally stable x^* for which solutions tend toward x^* (i.e. estimating the *region of attraction*). Unfortunately, a point may be locally stable with a tiny region of attraction.

The main tool for proving global stability in nonlinear systems is to use a *Lyapunov function*, which is an energy-like function that decreases on all trajectories of the system as they tend toward the equilibrium. All stable systems have Lyapunov functions, and if a Lyapunov function can be found, the system is stable. However, find a Lyapunov function is difficult, and no efficient method for doing so can be found. Nevertheless, the method often proves to be useful.

Theorem 3.7 (Lyapunov's Direct Method). *Consider a system $\dot{x} = f(x)$ and an equilibrium x^*. The function $V : \mathbb{R}^n \to \mathbb{R}$ is a* Lyapunov Function *if*

i. $V(x) > 0$ *for all* $x \neq x^*$*;*
ii. $V(x^*) = 0$*; and*
iii. $\frac{d}{dt}V(x(t)) < 0$ *for all* $x \neq x^*$*.*

If V is a Lyapunov function, then the point x^* is globally asymptotically stable.

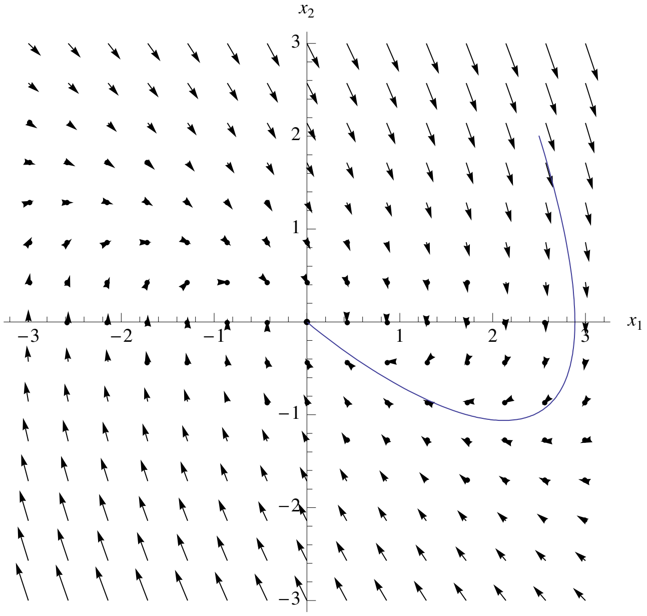


FIGURE 2. labelfig:lyapunov A phase portrait for the system in Example 3.8.

EXAMPLE 3.8. Consider the system

$$\begin{aligned}\dot{x}_1 &= x_2\\ \dot{x}_2 &= -x_1 - 2x_2.\end{aligned}$$

This system has an equilibrium when $x_1 = x_2 = 0$. Now define $V(x_1, x_2) = \frac{1}{2}(x_1^2 + x_2^2)$ to be a *candidate* Lyapunov Function. Clearly conditions (i) and (ii) in Theorem. 3.7 hold. For the third condition,

$$\begin{aligned}\frac{d}{dt}V &= \frac{d}{dt}(x_1^2 + x_2^2)/2\\ &= x_1\dot{x}_1 + x_2\dot{x}_2\\ &= x_1x_2 + x_2(-x_1 - 2x_2)\\ &= -x_2^2 < 0\end{aligned}$$

when $x \neq 0$. We can therefore safely assume that no matter what the initial condition, this system always converges to 0. ▲

One way to visualize what at least a two-dimensional system behaves like is to draw the vector field defined by the equations. The resulting plot is often called a *phase portrait*. In particular, we choose a number of points in the x_1, x_2 plane. For each point x, we draw a vector from the point x to $x + f(x)$. Since a solution curve in the plane must be tangent to each one of these vectors, the resulting image is quite useful.

EXAMPLE 3.9. The vector field for the dynamical system in Example 3.8 is shown in Figure **??**. The stability of the point 0 is evident. ▲

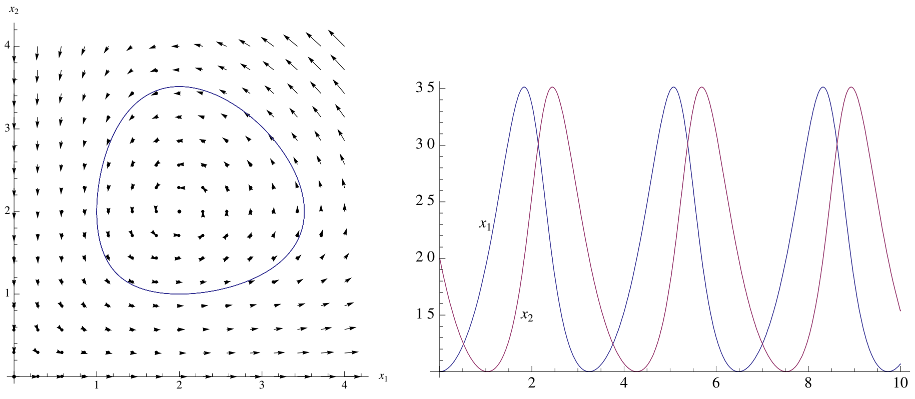


FIGURE 3. (a) Phase portrait for the system in Example 3.10. (b) The states versus time for the same system.

Besides converging to a stable equilibrium, many other limiting behaviors are possible. A system may oscillate or it may have multiple equilibria some of which are stable and others not.

EXAMPLE 3.10. A phase portrait for the system

$$\begin{aligned} \dot{x}_1 &= 2x_1 - x_1 x_2 \\ \dot{x}_2 &= x_1 x_2 - 2x_2 \end{aligned}$$

is shown in Figure 3. In this system, every solution in the right-upper quadrant is periodic.

We return to the analysis of oscillations later in the book.

3.4. Linearization and Local Stability

A powerful tool in analyzing a dynamical system is to *linearize* the system around an equilibrium. Using the Taylor series, we see that near an equilibrium x^*, the system

$$\dot{x} = f(x) \tag{3.6}$$

becomes

$$\dot{x} = f(x) = f(x^*) + \left.\frac{\partial f}{\partial x}\right|_{x=x^*} (x - x^*) + \text{higher order terms.}$$

The first term in the series is zero, since x^* is an equilibrium. The higher order terms are all quadratic or higher in x. The second term is the linear part of f near x^*, called the *Jacobian*, and is defined by the matrix

$$A = \left.\frac{\partial f}{\partial x}\right|_{x=x^*} = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{pmatrix}. \tag{3.7}$$

Thus, the dynamical system $\dot{x} = f(x,t)$ can be approximated, near x^* by

$$\dot{x} \approx A(x - x^*). \tag{3.8}$$

This is of great use in understanding local stability, due to the following two results.

THEOREM 3.11. *If x^* is an asymptotically stable equilibrium of (3.8) then it is a locally asymptotically stable equilibrium of (3.6).*

THEOREM 3.12 (Lyapunov's Indirect Method). *If the real parts of the eigenvalues of A in (3.8) are strictly negative, then the point x^* a stable equilibrium of (3.8) and a locally stable equilibrium of (3.6).*

Also, if the eigenvalues of A have positive real parts, you can conclude that the equilibrium is *not* stable. However, if any of the eigenvalues have zero real parts, then the method is inconclusive. See Exercise **??**.

EXAMPLE 3.13. Consider the system

$$\begin{aligned}\dot{x}_1 &= -x_1 - x_2 + x_1 x_2 \\ \dot{x}_2 &= -x_2 - x_1 x_2\end{aligned}$$

This system has two equilibra, one at 0 and the other at $(-1, \frac{1}{2})$. The Jacobian is

$$A = \left(\begin{array}{cc} -1 + x_2 & -1 + x_1 \\ -x_2 & -1 - x_1 \end{array} \right)\Bigg|_{x=x^*}$$

Evaluated at the first equilibrium, A is

$$A_{(0,0)} = \left(\begin{array}{cc} -1 & -1 \\ 0 & -1 \end{array} \right)$$

which has eigenvalues -1 and -1. Thus, we conclude that the point 0 is locally asymptotically stable. Evaluated at the second point, we get

$$A_{(-1,\frac{1}{2})} = \left(\begin{array}{cc} -\frac{1}{2} & -2 \\ -\frac{1}{2} & 0 \end{array} \right)$$

which has eigenvalues $(-1 \pm \sqrt{17})/2$: one negative and the other positive. Thus, the second equilibrium is not stable. ▲

EXAMPLE 3.14. A system that is related to the bacterial growth system is the *chemostat* in which nutrients are pumped into the chemostat and some bacteria and depleted nutrients are drained out. The new nutrient mixture effectively dilutes the bacteria. If we call the rate of dilution u (in liters per minute), then the equations are

$$\begin{aligned}\dot{x}_1 &= \frac{v x_1 x_2}{k + x_2} - u x_1 \\ \dot{x}_2 &= -\gamma \frac{v x_1 x_2}{k + x_2} + u(n_0 - x_2)\end{aligned} \tag{3.9}$$

where n_0 is the number of grams of nutrient in a liter of the input fluid. Solving for the steady state gives two equilibria. One of these equilibria where $x_1^(= 0$, meaning that the bacteria have all been diluted away and x_2^* is simply n_0. To examine the conditions under which this point is a stable equilibrium, we consider the Jacobian at this point:

$$A = \left(\begin{array}{cc} \frac{v}{n} k + n_0 - u & 0 \\ \frac{v \gamma n_0}{k + n_0} & -u \end{array} \right)$$

which has eigenvalues

$$\lambda_1 = -u \ \text{ and } \ \lambda_2 = \frac{v n_0}{k + n_0} - u.$$

Since $u > 0$, the first of these has a negative real part. For the second to have a negative real part, we require that

$$u < \frac{vn_0}{k + n_0} \triangleq u_{\max}.$$

Thus, as long as the dilution rate is less than $u_{\max}$, the bacteria will not get simply flushed away. ▲

DISCUSSION OF 2D LINEAR SYSTEMS AND EIGENVALUES GOES HERE

3.5. Non-Dimensionalization

3.6. Problems

3.1) Show that (3.4) is the solution to (3.2).

3.2) Write the equations for protein expression with RNA included in the model. That is suppose that some species R of RNA is generated by a constitutively expressed gene and that R is translated into protein X. Remember that both the RNA and the protein will degrade. Simulate your equations and compare to the similar model that only includes protein.

3.3) Find the equilibrium points of the system

$$\begin{aligned} \dot{x}_1 &= \sin x_2 \\ \dot{x}_2 &= \cos x_1. \end{aligned}$$

If you understand the idea, determine the local (asymptotic) stability of each by linearizing around each point and finding the eigenvalues.

3.4) (a) Show that there are an infinite number of equilibrium points of the bacterial growth equations in Example 3.2? (a) Which of the equilibria are locally stable? (c) For a given initial condition with $x_1(0)_0$ and $x_2(0) > 0$, which particular equilibrium point will the system go to?

3.5) Consider the system

$$\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = \begin{pmatrix} x_2(x_1 + a) \\ u + \sin x_1 - ax_2 \end{pmatrix}$$

Find the equilibrium points. For each equilibrium, determin conditions on the parameter a that result in stability.

3.6) Linearize the following system around the point 0 to obtain a linear approximation and determine the local stability of the point 0.

$$\begin{aligned} \dot{x}_1 &= \cos(x_2) - x_1 \\ \dot{x}_2 &= 2\sin(x_1 + x_2) + 3x_1 \end{aligned}$$

3.7) Determine the stability of the other equilibrium point in chemostat system of Example 3.14 when $u > u_{\max}$.

3.8) Consider the system

$$\begin{aligned}\dot{x}_1 &= x_2 - x_1(x_1^2 + x_2^2)\\ \dot{x}_2 &= -x_1 - x_2(x_1^2 + x_2^2).\end{aligned}$$

(a) Show that $V = x_1^2 + x_2^2$ is a Lyapunov function showing that 0 is globally asymtotically stable. (b) Show that Lyapunovs indirect method is inconclusive in this case.

3.9) Simulate the system

$$\begin{aligned}\dot{x} &= -y - z\\ \dot{y} &= x + 0.2y\\ \dot{z} &= 0.2 + z(x - 5.7)\end{aligned}$$

from a variety of initial conditions. Show the numerical solution curves you obtain in 3D, all in the same plot.

3.10) Consider the function $U(x) = cos(x)$. Find the equilibria of the system

$$\dot{x} = -\nabla U$$

Determine the stability of each equilibrium. Sketch the vector field and mark the basin of attraction of each equilibrium point.

3.11) Repeat the previous exercise, but with

$$\begin{aligned}\dot{x} &= v\\ \dot{v} &= -\nabla U - kv\end{aligned}$$

For what values of k does this system behave like the system in the previous exercise?

3.12) Consider the equations

$$\begin{aligned}\dot{x} &= \sigma(y - x)\\ \dot{y} &= x(\rho - z) - y\\ \dot{z} &= xy - \beta z\end{aligned}$$

where $\sigma = 10$, $\rho = 28$, and $\beta = \frac{8}{3}$. Use the software of your choice to draw the flow field of this system. Since it is three dimensional, you can either use your software to plot it in 3D, or project the vectors into a 2D plane (such as the $x - y$) plane.

Next, plot two trajectories of the system, starting at very nearby initial conditions. What happens?

3.13) Find chemical reactions whose dynamics match the equations in the previous exercise.

Chapter 4

Mass Action Kinetics

One way to describe what is going on inside a cell is to suppose that the cell is just a very small, well-mixed test tube containing all of the molecules (proteins, RNAs, DNAs, lipids, sugars, etc.) in a solution of H_2O. That is, we ignore the unseemly reality that most of these molecules are stuck to each other, to membranes, inside organelles, etc. If the cell really were a test tube, then the reactions between molecules would follow the laws of *mass action kinetics*, which we describe here. Essentially: molecules react with each other to form new molecules at rates proportional the concentration of the reactants.

Of course, the test tube idea is not even a remotely good assumption about what is really happening, but it does provide us with a starting point for modeling, and often the models obtained using mass action kinetics are fairly informative.

Chemical kinetics is a well-estabilished area in chemistry, mathematical modeling and non-linear systems. There are several good sources for this material, such as the notes by Martin Feinberg [**21**] from which the notes here draw heavily. Here, however, we present the mass action kinetics using a notation consistent with the other ideas in this book, and give examples that are related to molecular biology rather than chemistry.

4.1. Definitions

4.1.1. Reaction Networks. In general, we name the molecular *species* in a system X_1, X_2, ..., X_N. These might be the names of particular proteins, messenger RNAs or small molecules. In particular examples, the species are named with capital letters like B, M and E instead of the generic X_i. A *reaction* among the molecular species in question has the form

$$\underbrace{n_1X_1 + ... + n_NX_N}_{\text{reactants}} \rightharpoonup \underbrace{m_1X_1 + ... + m_NX_N}_{\text{products}}$$

where $n_1, ..., n_N, m_1, ..., m_N \in \mathbb{N} \cup \{0\}$. In this reaction, n_i molecules of species X_i are *consumed* and m_i molecules of species X_i are produced. A reaction can be written more conveniently as a vector

$$a = -\begin{pmatrix} n_1 \\ \vdots \\ n_N \end{pmatrix} + \begin{pmatrix} m_1 \\ \vdots \\ m_N \end{pmatrix} = -a_R + a_P \in \mathbb{Z}^N, \tag{4.1}$$

where a_R is the vector of reactant numbers and a_P is the vector of product numbers. The vector a is called the *stoichiometry* of the reaction.

Example 4.1. Suppose we have three species X, Y and Z. The single reaction

$$X + Y \rightharpoonup Z$$

can be written as a vector

$$a = \begin{pmatrix} -1 \\ -1 \\ 1 \end{pmatrix},$$

as long as we remember that the species are always ordered as we presented them above. ▲

A system inside the cell, consisting of a number of species and the reactions among them is modeled a *reaction network*, which is simply a set of reactions. The nodes of the network are *complexes* (e.g. $X+Y$ and Z in the previous example) and the edges are the reaction arrows. Furthermore, the reaction vectors associated with the reactions in a network can be grouped into a matrix A called the *stiochiometric matrix.*

Example 4.2. The reactions

$$X + Y \rightharpoonup Z \tag{4.2}$$

$$2Z \rightharpoonup X \tag{4.3}$$

form a reaction network with stiochiometric matrix

$$A = \begin{pmatrix} -1 & 1 \\ -1 & 0 \\ 1 & -2 \end{pmatrix}.$$

▲

4.1.2. Kinetics. In a system of reacting chemicals, the state at a particular time is described by the *concentrations* of each species, usually arranged into another vector. If X_i is a species, then we write $v_{X_i}(t)$ or just $v_i(t)$ to represent the number of molecules of species X_i per unit volume at time t. Then we group the concentrations into a vector

$$v = \begin{pmatrix} v_1 \\ \vdots \\ v_n \end{pmatrix} \in \mathbb{P}^N.$$

It is generally assumed that the number of molecules in the system is very large, so that stochastic effects do not come into play. For this reason, we may call the model described in this chapter as the *deterministic* model to contrast it with the *stochastic* model to be described in Chapter 8.

Associated with each reaction a is a *reaction rate*

$$K_a : \mathbb{P}^N \to \mathbb{P}$$

that depends on the concentrations. If $K_a(v) > 0$ then every reactant in a has nonzero concentration in the state v. Reaction rates, when noted, are usually written above the reaction arrows in a specification of the reaction. A *kinetics* for a network is such an assignment of a rate to each reaction a. These rates may be grouped into a vector $K(v)$ that depends on v:

$$K(v) = \begin{pmatrix} K_{a_1}(v) \\ \vdots \\ K_{a_M}(v) \end{pmatrix},$$

where we have numbered the reactions from 1 to M.

The usual assumption is that the kinetics obey the **law of mass action**, although they need not. In later chapters we will allow the rates to be defined in other ways. For now, however, we stick with the following definition of the rates.

Definition 4.3. Suppose that a is defined as in (4.1). Then the *mass action rate* for the reaction a os definted by

$$K_a(v) \triangleq k_a \prod_i v_i^{n_i} \tag{4.4}$$

where k_a is called the *rate constant* for the reaction a.

The idea the following. Suppose that an X molecule reacts with a Y molecule according to the reacton

$$X + Y \overset{k}{\rightharpoonup} Z$$

If there are v_X molecules of type X per liter and v_Y molecules of type Y per liter, then the number of possible reactions that can occur per liter is $v_X v_Y$. Since some types of reactions may be stronger or faster than others, we add a rate constaint k to our model to obtain $k v_X v_Y$. When the rate constant is important, we make a note of it by writing it, as above, near the arrow used to specify the reaction. The definition (4.4) is a generalization of the idea of having the rate of a reaction include a term that desribes the number of ways that a reaction can happen: the rate is proporional to the current concentrations, or masses, of the reactants.

Example 4.4. The reaction rates for the reactions in Example 4.2 are

$$K(v) = \begin{pmatrix} k_1 v_X v_Y \\ k_2 v_Z^2 \end{pmatrix}$$

where we suppose that k_1 and k_2 are the rate constants for the two reactions in the network respectively. ▲

4.1.3. The Determinstic Equations. The rate at which the concentrations in a reaction network change leads to a set of differential equations, called the deterministic equations (to contrast them with the stochastic equations we study later) that can be written succinctly as

$$\dot{v} = AK(v). \tag{4.5}$$

Along with an initial concentration vector $v(0)$ and actual values for the rate constants, this completely specifes the behavior of a reaction network.

Example 4.5. Applying (4.5) to the stoichiometric matrix A in Example 4.2 and the rate vector $K(v)$ in Example 4.4 gives

$$\begin{pmatrix} \dot{v}_X \\ \dot{v}_Y \\ \dot{v}_Z \end{pmatrix} = AK(v) = \begin{pmatrix} -k_1 v_X v_Y + k_2 v_Z^2 \\ -k_1 v_X v_Y \\ k_1 v_X v_Y - 2k_2 v_Z \end{pmatrix},$$

which is simply a list of three equations stating how the concentrations of X, Y and Z in the network change over time. ▲

4.1.4. Basic Properties.

The mass action kinetics equations for a given network are, in general, nonlinear. Explicit, closed-form solutions for the equations do not exist except in simple situations. However, systems of differential equations arising from mass action kinetics are not entirely arbitrary either, and all mass action systems obey some simple properties, which we now discuss.

The first property states that concentrations can never be negative as long as they start out positive. This property is certianly true of actual physical systems. Mathematically, the property holds because if a concentration ever becomes zero, its rate of change is necessarily non-negative.

Property 4.6. *If $v \in \mathbb{P}$ and $v_i = 0$, then $\dot{v}_i \geq 0$.*

Proof. Every reaction a in which X_i shows up as a reactant has $K_a(v) = 0$ whenever $K_a(v)$ is defined by Equation (4.5). Thus contributions to $\dot{v}_i$ are only from reactions in which X_i appears as a *product* and are therefore non-negative. □

The next property describes the set of all possible concentration vectors that can be reached starting from a given initial condition. We first require a definition:

Definition 4.7. The *stoichiometric subspace* of a system with matrix $A = (a_1, ..., a_N)$ is

$$\begin{aligned} S &= \text{span} A = \text{span}\{a_1, ..., a_N\} \\ &= \{r_1 a_1 + ... + r_N a_N \mid r_1, ..., r_N \in \mathbb{R}\}. \end{aligned}$$

The *rank* of a reaction network is the rank of S (i.e. the number of linearly independent columns or rows). If $v_1 - v_2 \in S$, then v_1 and v_2 are said to be *stoichiometrically equivalent.*

Example 4.8. Suppose that

$$A = \begin{pmatrix} -1 & 1 \\ -1 & 0 \\ 1 & -2 \end{pmatrix}.$$

as in Example 4.2. Then

$$S = \left\{ \begin{pmatrix} r_2 - r_1 \\ -r_1 \\ r_1 - 2r_2 \end{pmatrix} \mid r_1, r_2 \in \mathbb{R} \right\}.$$

In this case, S is a two dimensional subspace of $\mathbb{R}^3$, suggesting that the three differential equations we use to describe it are redundant in some way. ▲

The notion of *stiochiometrically equivalent* leads to equivalence classes of states and the simple property that all solutions of a mass acton kinetics system remain in an equivalence class defined by the initial condition. This is characterized in the following property.

PROPERTY 4.9. *All solutions $v(t)$ of*

$$\dot{v} = AK(v) \qquad \text{with } v(0) \text{ given}$$

lie in the stochiometric equivalence class of $v(0)$. *I.e.* $v(t)$ *is always stoichiometrically equivalent to* $v(0)$.

PROOF. The property follows from the definition of the derivative and of S:

$$\dot{v} = \lim_{h \to 0} \frac{v(t+h) - v(t)}{h} = AK(v) \in S.$$

□

These notions can be best seen in a two-dimensional example.

EXAMPLE 4.10. Consider the network

$$X \underset{k_2}{\overset{k_1}{\rightleftharpoons}} 2Y$$

with the initial condition defined by $v_X(0) = 1$ an $v_Y(0) = 0$. The stiochiometric matrix is

$$A = \begin{pmatrix} -1 & 1 \\ 2 & -2 \end{pmatrix}$$

and

$$S = \left\{ \begin{pmatrix} -r \\ 2r \end{pmatrix} \mid r \in \mathbb{R} \right\}.$$

By Property 4.9, for every t there exists an r such that

$$v(t) - v(0) = \begin{pmatrix} v_X(t) \\ v_Y(t) \end{pmatrix} - \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} -r \\ 2r \end{pmatrix}$$

from which it follows that all $v(t)$ have form

$$\begin{pmatrix} 1 - r \\ 2r \end{pmatrix}.$$

Furthermore, since all concentrations must be non-negative, $r \in [0, 1]$. We can further analyze this system be looking at the locus of equilibra. Setting

$$\dot{v} = AK(v) = \begin{pmatrix} -k_1 v_X + k_2 v_Y^2 \\ k_1 v_X - k_2 v_Y^2 \end{pmatrix} = 0,$$

we get $k_1 v_X = k_2 v_Y^2$ or $v_X = \frac{k_2}{k_1} v_Y^2$. The general picture is shown in Figure 1.

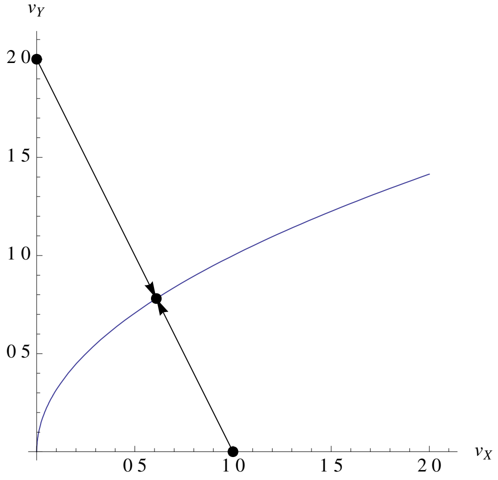


FIGURE 1. The stoichiometric subspace from Example 4.10 is a one-dimensional line connecting $(1, 0)$ to $(0, 2)$. The points on this line tend toward the point where stoichiometric subspace intersects the locus of equilibra

EXAMPLE 4.11. Continuing with the previous example, we note that the system is really one dimensional. We can write it in terms of r in fact as follows:

$$\begin{aligned} \dot{r} &= \frac{\dot{v}_Y}{2} \\ &= \frac{1}{2}\left(k_1 v_X - k_2 v_Y^2\right) \\ &= \frac{1}{2}\left(k_1(1-r) - k_2(2r)^2\right) \\ &= \frac{k_1}{2}(1-r) - 4k_2 r^2. \end{aligned}$$

Anything true abut this equation can be translated back to statements about the original set of equations. ▲

4.2. Example Networks

4.2.1. Sources and Sinks. If $a_r = 0$, then the species in a_p appear from nowhere. Thus, $a_r = 0$ models a *source*. We write, equivalently,

$$0 \rightharpoonup a_p$$

and

$$\varnothing \rightharpoonup m_1 X_1 + ... + m_N X_N.$$

Similarly, if $a_p = 0$, then reactants can disappear from the system and $a_p = 0$ is called a *sink*.

EXAMPLE 4.12. The reaction

$$\varnothing \overset{k}{\rightharpoonup} A$$

has the mass action kinetics

$$\dot{v}_A = k.$$

That is, A is added to the system at a constant rate. ▲

Example 4.13. The reaction

$$A \xrightharpoonup{k} \varnothing$$

has the mass action kinetics

$$\dot{v}_A = -kv_A.$$

That is, the more A is in the system, the faster its concentration decreases. ▲

4.2.2. Constant Concentrations. In the cell, many species are regulated using feedback so that they maintain essentially constant levels. For example, ATP – the basic unit of energy – is regulated tightly [**?**]. ATP is also involved in many reactions. We could include ATP as a reactant or product in all of the reactions in our network, but because it is regulated, we would also have to include the regulation machinery in our model as well. To get around this, we usually just pretend that $\dot{v}_{ATP} = 0$ and write, for example,

$$X \xrightharpoonup{kv_{ATP}} Y$$

to denote that the rate at which X is converted into Y depends on the concentration of ATP. The above reaction gives the kinetics

$$\dot{v}_X = -\dot{v}_Y = -kv_{ATP}v_X$$

where v_{ATP} does not depend on time (i.e. it is a constant parameter in our model).

4.2.3. Membranes. Say X and Y react according to the reaction

$$X \underset{k_2}{\overset{k_1}{\rightleftharpoons}} 2Y$$

in two compartments separated by a membrane with some active transport mechanism across it, as in Figure **??**. To model this system with mass action kinetics, we define new symbols X_1 and X_2 to refer to X molecules when they are in compartent one and compartment two, respectively. Similarly, we introduce the names Y_1 and Y_2. We can the write the reaction network

$$X_1 \underset{k_2}{\overset{k_1}{\rightleftharpoons}} 2Y_1 \qquad X_1 \underset{k_4}{\overset{k_3}{\rightleftharpoons}} X_2$$

$$X_2 \underset{k_2}{\overset{k_1}{\rightleftharpoons}} 2Y_2 \qquad Y_1 \underset{k_6}{\overset{k_5}{\rightleftharpoons}} Y_2$$

which describes a system with four “species” that model our two species reacting in a two-compartment system.

4.2.4. Conservative Systems. For the system

$$X \underset{k_2}{\overset{k_1}{\rightleftharpoons}} 2Y$$

we can define a vector m by

$$m = \begin{pmatrix} 2 \\ 1 \end{pmatrix}$$

and note that

$$\begin{aligned} \frac{d}{dt} m^T v &= m^T \dot{v} = m^T A K(v) \\ &= (2\ 1) \begin{pmatrix} -1 & 1 \\ 2 & -1 \end{pmatrix} K(v) \\ &= (0\ 0) K(v) = 0. \end{aligned}$$

The idea is that X molecules have twice the mass of Y molecules and m records this fact. The quantity $m^T v$ is the total mass of the system over all species. Its derivative is zero because mass in conserved in this system.

In contrast, the system

$$X \xrightarrow{k_1} 2Y$$

$$Y \xrightarrow{k_2} X$$

where Y decays back into a single X (forgetting that it seems to have been constructed out of two X molecules) does not have a meaningful mass vector. This is because

$$m^T A = (m_X\ m_Y) \begin{pmatrix} -1 & 1 \\ 2 & -1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

has no non-trivial solution. In this case, the network is not conservative.

Another possibility is that $m^T A = 0$ may have no positive solution. A negative solution would lead to a strange definition of mass. These ideas motivate the following definition. First, recall that if we are given a subspace S of a vector space V, the *orthogonal compliment* of S is

$$S^\perp = \{u \in V \mid u^T v = 0,\ \forall v \in S\}.$$

Definition 4.14. A reaction network with stoichiometric subspace S is conservative if there is a positive vector m contained in $S^\perp$.

To find a mass vector, then, one has to solve $m^T A$ for m and see if m can be positive. This amounts to solving a system of linear equations, which can be done with Gaussian elimnation, for example.

Example 4.15. Consider the network in Figure 2. In it, oligonucleotides form and unform complexes of one or more oligos. Thus each complex is a species, but it is also made up of individual oligos, suggesting that the number of oligos in a complex should be the mass of the complex. For this system, we have the following

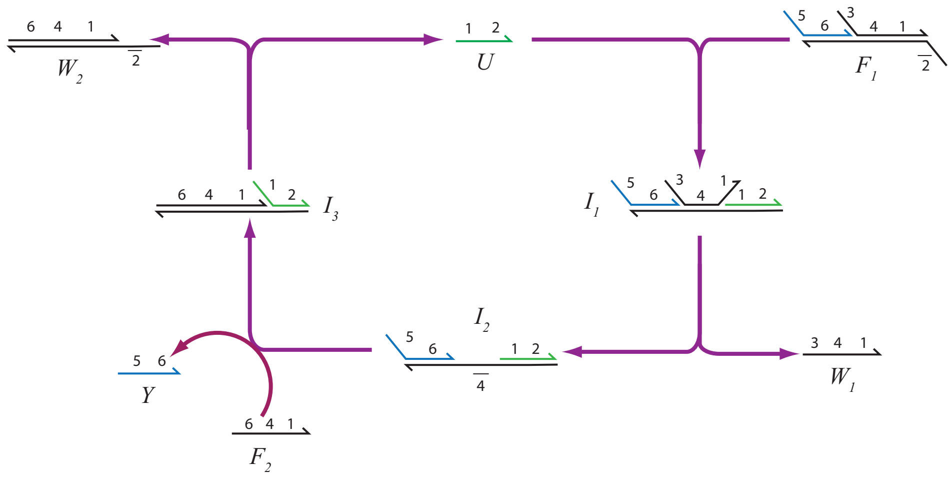


Figure 2. A DNA hybridization circuit from [**54**]. Each line represents a single stranded piece of DNA. Parallel lines correspond to double stranded DNA. Numbers denote binding domains, which are unique seqnences of A, T, C or G nucleotides. A number with a bar denotes a complimentary sequence.

reactions (ignoring the rates for now).

$$\begin{aligned} U + F_1 &\rightharpoonup I_1 \\ I_1 &\rightharpoonup W_1 + I_2 \\ I_2 + F_2 &\rightharpoonup I_3 + Y \\ I_3 &\rightharpoonup W_2 + U. \end{aligned}$$

If we define $v = (U\ F_1\ F_2\ I_1\ I_2\ I_3\ W_1\ W_2\ Y)^T$ then a suitable mass vector is

$$m = (1\ 3\ 1\ 4\ 3\ 3\ 1\ 2\ 1),$$

where the mass of each species is given by the number of oligos in it. This fact is confirmed in an exercise. ▲

4.3. Modeling Regulatory Networks and Metabolism

In this section we explore how mass action kinetics can be used to model genetic regulatory networks and metabolism. We start by modeling a single gene, and then simple combinations of genes. Finally, we show an example from metabolism. In the next chapter we show how the systems here can be simplified to some extent using a set of approximation called *enzyme kinetics*.

Example 4.16. Consider a single consitutively expressed gene producing a single protein X at a rate k_1. The protein is degraded and diluted due to cell growth at a rate k_2. Note that the concentration of RNAP, like ATP above, is regulated and, therefore, does not appear in these reactions. Instead it is absorbed into the rate constant k_1. Similarly, the concentration of proteins that degrade protein is absorbed into k_2. A simple model of our system is then

$$\varnothing \xrightarrow{k_1} X \xrightarrow{k_2} \varnothing$$

which gives the differential equation

$$\dot{v}_X = k_1 - k_2 v_X.$$

A more sophisticated model of this system includes the messenger RNA intermediate. Let's call it R. The gene produces R and R produces proteins before being degraded and diluted. Adding these complexities to our model gives

$$\varnothing \xrightarrow{\kappa_1} R \xrightarrow{\kappa_2} \varnothing$$

$$R \xrightarrow{\kappa_3} R + P$$

$$P \xrightarrow{\kappa_4} \varnothing$$

The second line, where R produces P shows that the mRNA is not consumed in the process of being transcribed into protein. Note also that the concentration of ribosomes is absorbed into κ_3. The above system gives rise to the equations

$$\begin{aligned} \dot{v}_R &= \kappa_1 - \kappa_2 v_R \\ \dot{v}_P &= \kappa_3 v_R - \kappa_4 v_P. \end{aligned}$$

One might ask: *What values of k_i and κ_i make these two descriptions similar?* To answer this, note that the rate of production and degradation of mRNA is faster than with protein. Thus, we can assume that the level of mRNA equilibrates quickly, meaning that

$$\dot{v}_R = \kappa_1 - \kappa_2 v_R = 0$$

which implies that $R = \kappa_1/\kappa_2$ fairly soon. Thus, the equation for protein production becomes

$$\dot{v}_P = \kappa_3 \frac{\kappa_1}{\kappa_2} v_R - \kappa_4 v_P.$$

This implies that

$$k_1 = \frac{\kappa_3 \kappa_1}{\kappa_2} \quad \text{and} \quad k_2 = \kappa_4.$$

A simulation of these two systems is shown if Figure 3. Note that the main difference in the two curves is that initially, $\dot{v}_P = 0$ in the more complex model. This may be an insignificant difference in some situations, but in others in is crucially important (as in Example 4.18). ▲

Example 4.17. Now consider a gene g that produces a protein product X that is repressed by a transcription factor T. In reactions, this can be written

$$g \xrightarrow{k_1} g + P$$

$$P \xrightarrow{k_2} \varnothing$$

$$T + g \underset{k_4}{\overset{k_3}{\rightleftharpoons}} g_{\text{off}}.$$

The first reaction models the fact that the gene g produces a protein product P at some rate, without itself being consumed. The second reaction models protein degradation. The third line models the interaction between the transcription factor

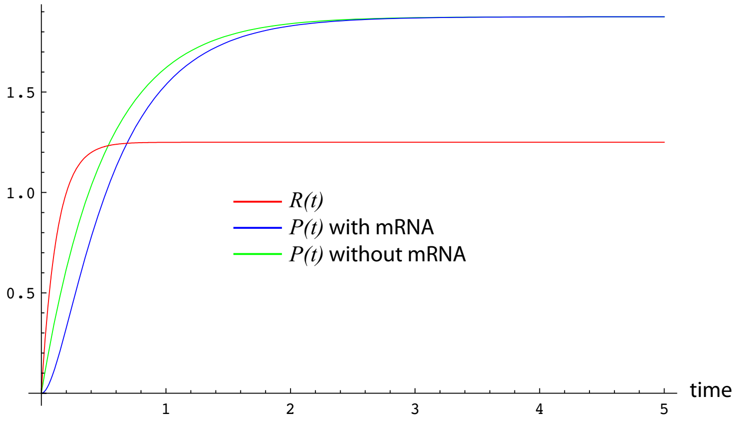


FIGURE 3. Two models of gene expression, one modeling mRNA explicitly and the other not.

T and the gene g. When T is bound to g, RNAP is unable to bind to g to produce P, thus the name g_{off}. Note that there is a conservation of mass with g and g_{off}. Namely, there is some constant C such that

$$v_g + v_{g_{\text{off}}} = C.$$

We now consider the steady state concentration of P for varying input concentrations of T. Since

$$\dot{v}_P = k_1 v_g - k_2 v_P,$$

we have $v_P^* = k_1 v_g^* / k_2$. The dynamics of g are given by

$$\dot{v}_g = -k_3 v_T v_g + k_4(C - v_g)$$

so that

$$v_g^* = \frac{k_4 C}{k_3 v_T + k_4}.$$

Thus, by substituting the above into the equation for the steady state concentration of P, we can see it is inversely related to the concentration of T:

$$v_P^* = \frac{k_1 k_4 C}{k_2 k_3 v_T + k_2 k_4}.$$

In the next chapter we study these kinds of relationships more extensively. ▲

EXAMPLE 4.18. The repressilator.

EXAMPLE 4.19. Relaxation oscillator.

4.4. Problems

4.1) Find the stable equilibrium in Example 4.8 assuming $k_1 = k_2 = 1$.

4.2) The system in Example 4.2 is really two-dimensional. Write the equations in terms of $\dot{r}_1$ and $\dot{r}_2$ to show this.

4.3) Confirm that m defined in Example 4.15 is indeed a mass vector by writing the stiochiometric matrix for the system and pre-multiplying it by m.

4.4) Consider ther system

$$2X + Y \overset{1}{\rightharpoonup} X + 2Y.$$

a) Find the stoichiometric matrix A and the kinetics vector $K(\vec{v})$. Use these to find the mass action equations.
b) Find the stoichiometric subspace and its dimension.
c) Pick a reasonable initial condition and describe its stoichiometric equivalence class.
d) If the system is conservative, determine a mass vector. Or state why the system is not conservative.
e) Determine the equilibrium points of the network, the Jacobians evaluated at these points, and the stability of the points.
f) Draw a vector flow field for the network.

4.5) Repeat the steps in for the following network

$$\begin{array}{rcl} X & \overset{1}{\rightharpoonup} & 2X \\ X + Y & \overset{1}{\rightharpoonup} & 2Y \\ Y & \overset{2}{\rightharpoonup} & \varnothing. \end{array}$$

4.6) Show that the two models of protein production from a single gene in Example 4.16 give the same steady state amount of protein.

4.7) Consider the system

$$\varnothing \underset{1}{\overset{1}{\rightleftharpoons}} X \overset{3}{\rightharpoonup} Y$$

$$2X + Y \overset{1}{\rightharpoonup} 3X.$$

Find the equilibria and determine their local stability. What do the imaginary parts of the eigenvalues mean? Simulate the system from a variey of initial conditions and describe the behavior obtained.

Chapter 5

Enzyme Kinetics

5.1. Fast and Slow Dynamics

Suppose we are modeling a system in which some states, call them z more very quickly to some equilibrium value. Meanwhile other states, call them x, change more slowly. In equations, we write

$$\dot{x} = f(x, z, t, \epsilon) \tag{5.1}$$

$$\epsilon z = g(x, z, t, \epsilon). \tag{5.2}$$

Here, ϵ denotes some system parameter (such as a reaction rate), that is very small compared with other parameters. You can see that if you divide (5.2) by ϵ then the rate of change of z is, thus, large.

Since ϵ is small, we might as well ask what if $\epsilon = 0$. In this case, (5.2) becomes

$$g(x, z, t, 0) = 0. \tag{5.3}$$

Suppose that $z = h(x, t)$ is a solution to (**??**). Then the "slow dynamics" of x are described by

$$\dot{x} = f(x, h(t, x), t, 0), \tag{5.4}$$

that is, solely in terms of x. Thus, by assuming that $\epsilon = 0$, we have come up with a reduced model of the original system. It turns out that how much smaller ϵ is compared to other parameters in the system determines the degree to which the full and reduced models are similar.

5.2. Enzyme Reactions

5.2.1. The Michaelis-Menten Model. As an example, consider the reaction

$$E + S \underset{k_2}{\overset{k_1}{\rightleftharpoons}} ES \xrightarrow{k_3} E + P$$

which describes how a species of enzyme E interacts with a substrate species S to produce a product species P. An enzyme molecule may repeatedly bind to substrate molecules to form enzyme-substrate complexes ES. Eventually the enzyme turns the a substrate molecule into a product. It is often the case that

$$k_1, k_2 \gg k_3,$$

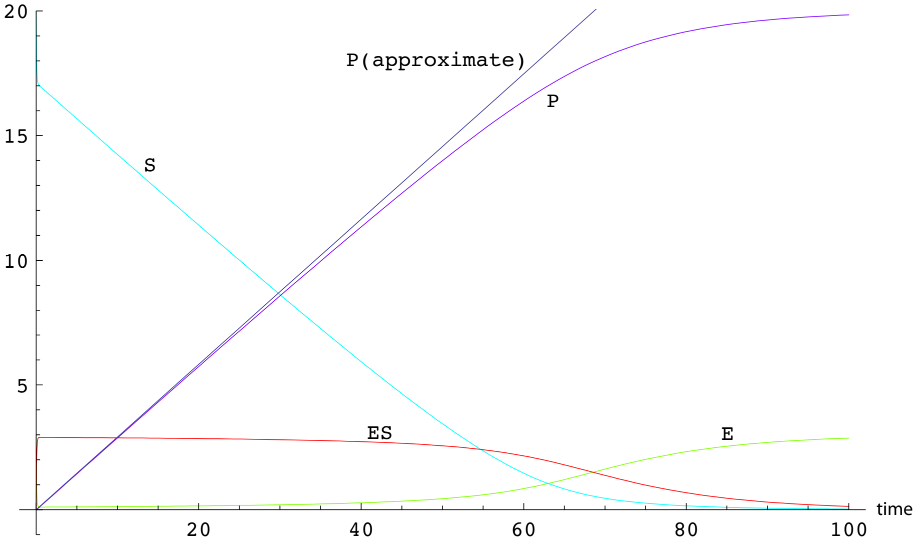


FIGURE 1. The simple enzyme substrate reaction and the Michaelis-Menton approximation. Here, $k_1 = 1.0$, $k_2 = 0.5$, $k_3 = 0.1$, $v_E(0) = 3$ and $v_S(0) = 20$.

suggesting that the population of ES stabilizes quickly compared to the production of P.

Now consider the analysis question: at approximately what rate is P produced, ignoring the brief period of time during which ES is equilibrating? Treating v_S as a constant (i.e. assuming there are a very large number of substrate molecules), and using the law of mass action gives

$$\dot{v}_P = k_3 v_{ES} \tag{5.5}$$
$$\dot{v}_{ES} = k_1 v_E v_s - k_2 v_{ES} - k_3 v_{ES}. \tag{5.6}$$

The total amount of enzyme does not change so that $v_E + v_{ES} = E_{tot}$. Thus, the equation for $\dot{v}_E$ is not needed and

$$\begin{aligned}\dot{v}_{ES} &= k_1(E_{tot} - v_{ES})v_s - k_2 v_{ES} - k_3 v_{ES} \\ &= k_1 e_{tot} v_s - (k_1 s + k_2 + k_3) v_{es}.\end{aligned}$$

Now define $\epsilon \triangleq (k_1 v_s + k_2 + k_3)^{-1}$. By our assumptions on the relative magnitudes of the k_i and that s is large, it follows that ϵ is small. We can now write:

$$\dot{v}_P = k_3 v_{ES} \tag{5.7}$$
$$\epsilon \dot{v}_{ES} = \frac{k_1 e_{tot} v_s}{k_1 v_s + k_2 + k_3} - v_{es}, \tag{5.8}$$

which matches (5.1) and (5.2). Setting the fast dynamics to zero gives the constraint that

$$v_{es} = \frac{k_1 e_{tot} v_s}{k_1 s + k_2 + k_3}$$

and substituting into (5.7) gives

$$\dot{v}_P = \frac{k_3 k_1 e_{tot} v_s}{k_1 v_s + k_2 + k_3} = \frac{k_3 e_{tot} v_s}{v_s + \frac{k_2+k_3}{k_1}}.$$

Chemists and biologists like to define

$$V_{max} \triangleq k_3 e_{tot} \qquad K_m = \frac{k_2+k_3}{k_1}$$

so that we arrive at the *Michaelis-Menton* model:

$$\dot{v}_P = \frac{V_{max} v_s}{v_s + K_m}. \tag{5.9}$$

The constant V_{max} is commonly referred to as the maximum production rate, which you get when $v_s \to \infty$. The constant K_m is called the half saturation constant, which is the concentration of S required to get a rate of $V_{max}/2$.

The situation is shown in Figure 1. Initially, while the amount of substrate is high, the rate of change of P is essentially constant with slope defined by Equation (5.9). Eventually, the substrate is depleted, and the assumption that $\dot{v}_S = 0$ no longer holds.

5.2.2. Competetive Inhibition. Now consider a system consisting of an enzyme in one of two states, either active E or bound to a deactivator I to form an inactive complex EI. The reactions are

$$E + S \underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} ES \xrightarrow{k_2} E + P$$

$$E + I \underset{k_{-3}}{\overset{k_3}{\rightleftharpoons}} EI.$$

In this case, we say that I and S are in competition for E. If ES and EI equilibrate quickly, then it seems reasonable to approximate the system by a simple one dimensional system in which the product P is produced at some rate that is a function of the concentrations of S (which we'll assume is large and relatively unchanging) and I (which is not produced or consumed).

First, note that there we have the conservation constraint that the enzyme is either free, bound to an S or bount to an I. That is,

$$v_{tot} = v_E + v_{ES} + v_{EI}.$$

Assuming that ES and EI equilibrate quickly gives

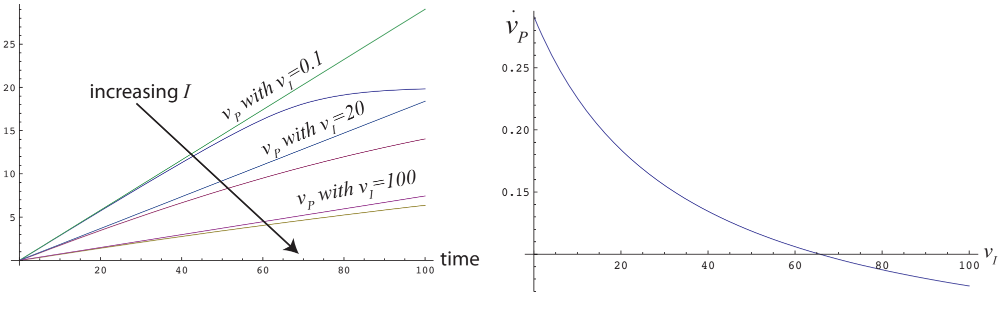
$$\dot{v}_{ES} = 0 = k_1 v_E v_S - (k_{-1} + k_2) v_{ES}$$

so that $v_E v_S = K_m v_{ES}$ where $K_m = (k_{-1} + k_2)/k_1$ and

$$\dot{v}_{EI} = 0 = k_3 v_E v_I - k_{-3} v_{EI}$$

which implies that $v_E v_I = K_3 v_{EI}$ where $K_3 = \frac{k_{-3}}{k_3}$. Putting these relationships into the constraint above gives

$$\begin{aligned} v_{tot} &= \frac{K_m v_{ES}}{v_S} + \frac{v_E v_I}{K_3} + v_{ES} \\ &= \frac{K_m v_{ES}}{v_S} + \frac{K_m v_{ES}}{v_S} \frac{v_I}{K_3} + v_{ES} \end{aligned}$$



(a) (b)

FIGURE 2. The competitive inhbition model. (a) Three different values for v_I and the resulting dynamics for v_P both the exact models (curves) and approximations (striaght lines). (b) As v_I increases, the rate of production of P decreases.

which implies that

$$v_{ES} = \frac{v_{tot}}{\frac{K_m}{v_S} + \frac{K_m v_I}{K_3 v_S} + 1} = \frac{v_{tot} v_S}{K_m + \frac{K_m}{K_3} v_I + v_S}.$$

From the above expression for v_{ES} it follows that

$$\dot{v}_P \approx \frac{V_{max} v_S}{K_m + \frac{K_m}{K_3} v_I + v_S} \tag{5.10}$$

where

$$V_{max} = k_2 v_{tot} \quad K_m = \frac{k_{-1} + k_2}{k_3} \quad \text{and} \quad K_3 = \frac{k_{-3}}{k_3}.$$

If there is no inhibitor I, we get the same results as with the Michaelis-Menton system. As the concentration of I increases, the rate of production of P goes to zero. Finally, if there is a high concentration of S, then $\dot{v}_P \approx V_{max}$. The situation is shown in Figure 2 ▲

5.3. Hill Functions

Models of genetic regulatory networks can get big fast, in terms of the number of species, the number of reactions, and the number of unknown parameters. One way to reduce this complexity slightly is to assume that regulatory interactions, such as the binding and unbinding of a repressor to the regulatory region of a gene, are much faster than the transcription and translation reactions. In fact, this is usually a pretty safe assumption, although one has to be careful. The result is to write rate of production of a gene's product in terms of the *fraction* of action versus total gene:

$$\dot{X} = \alpha \frac{g_{\text{active}}}{g_{\text{total}}} + \alpha_0 - \beta X \tag{5.11}$$

where α is the maximum rate of production, α_0 is the rate of "leaky" expression (sometimes used in these models, although not strickly legitimate), and β is the rate of protein degradation and dilution.

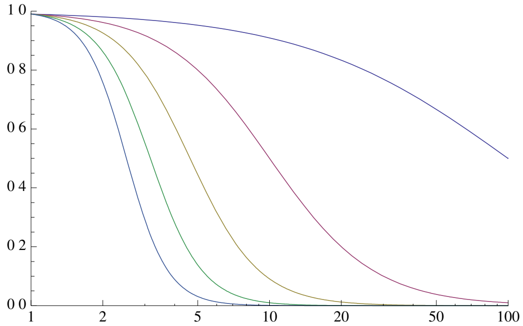


FIGURE 3. The Hill Function (Equation (**??**)) for repression plotted for various values of the cooperativity n. The function essentially computes the Boolean function $\neg R$. That is, when R is not present, the gene is expressed. Otherwise it is not expressed.

EXAMPLE 5.1. Consider a gene g repressed by a transcription factor R via the following reaction

$$g_{\text{on}} + nR \underset{k_r}{\overset{k_f}{\rightleftharpoons}} g_{\text{off}}.$$

The number n, called the *cooperativity*, is the number of molecules of R that must bind to g to turn it off. The reaction is a bit unrealistic, as the likelihood that $n+1$ molecules simultaneously interact and react is low for $n > 1$. The idea is that the reaction above is actually implemented by the sequential or parallel binding of individual molecules R, one after the other. The details of how different mechanisms affect the results in this example are explored in the exercises (and see also [**51**]).

If we assume that k_r and k_f are large, then, at equilibrium, we get

$$k_f g_{\text{on}} R^n = k_r g_{\text{off}}.$$

Thus,

$$\frac{g_{\text{on}}}{g_{\text{on}} + g_{\text{off}}} = \frac{g_{\text{on}}}{g_{\text{on}} + \frac{k_f g_{\text{on}} R^n}{k_r}} = \frac{1}{1 + \frac{R^n}{K_{\text{eq}}}} \tag{5.12}$$

where $K_{\text{eq}} = k_r/k_b$ is the equilibrium constant of the reaction. The result is a so called *Hill Function* [**?**] and is plotted in Figure 3 for various values of n. The Hill function for repression inserted into Equation (5.11) results in

$$\dot{X} = \frac{\alpha}{1 + \frac{R^n}{K_{\text{eq}}}} + \alpha_0 - \beta X,$$

which describes the rate of production of a protein X given the concentration of the repressor R. This system essentially computes the Boolean function $\neg R$. That is, when R is not present, the gene is expressed. Otherwise it is not expressed. ▲

EXAMPLE 5.2. A similar function for activation can be obtained by considering the reactions

$$g_{\text{off}} + nA \underset{k_r}{\overset{k_f}{\rightleftharpoons}} g_{\text{on}}.$$

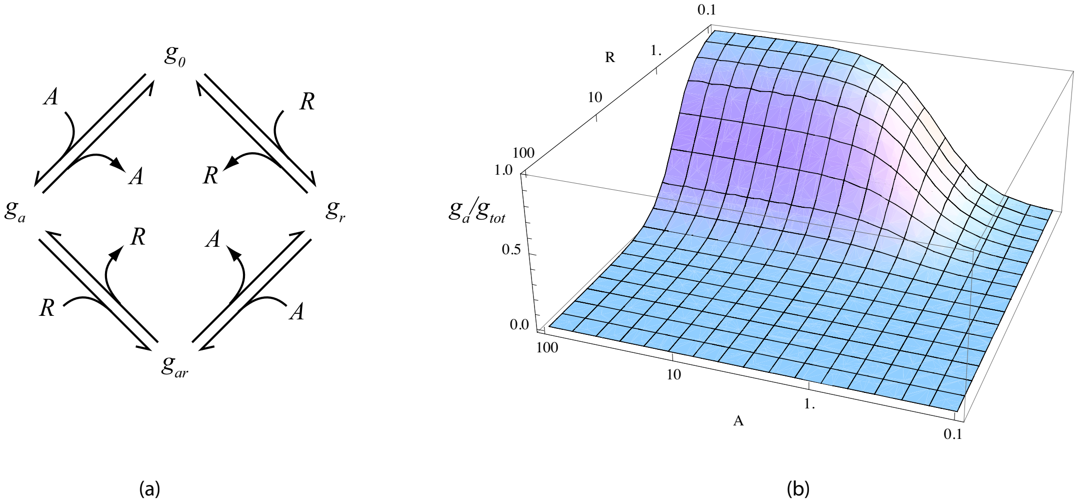


Figure 4. The Hill Function (Equation (**??**)) for a gene that is activated by a transcription factor A and repressed by a transcription factor R. The Hill function essentially computes the Boolean function $A \wedge \neg R$.

which results in the activation Hill Function

$$\frac{g_{\rm on}}{g_{\rm on} + g_{\rm off}} = \frac{K_{\rm eq} A^n}{A^n + K_{\rm eq}},$$

which can be substituted into Equation (5.11) to obtain a model for the production of a gene product as a function of the activator concentration. ▲

Example 5.3. (THERE ARE A BUNCH OF ERRORS IN THIS EXAMPLE TO BE FIXED) Many genes are regulated by several transcription factors, some that are activators, some that are inhibitors. Hill functions can be devised to model such situations as well. Here is an example. Suppose that a gene can be in one of four states: (1) g_0 with no transcriptions factors bound; (2) g_a with an activator bound; (3) g_r with a repressor bound; (4) and g_{ar} with both an activator and a repressor bound. The reaction network is shown in Figure 4(a). At equilibrium, we have

$$\begin{aligned} k_1 g_0 &= k_{-1} g_a \\ k_2 g_0 &= k_{-2} g_r \\ k_3 g_a &= k_{-3} g_{ar} \\ k_4 g_r &= k_{-4} g_{ar}. \end{aligned}$$

Thus, the total amount of gene in the system is

$$\begin{aligned} g_{\rm tot} &= g_0 + g_a + g_r + g_{ar} \\ &= g_0 + K_1 g_0 A^n + K_2 g_0 R^m + K_3 g_a R^m \\ &= g_0 + K_1 g_0 A^n + K_2 g_0 R^m + K_3 K_1 g_0 A^n R^m \end{aligned}$$

where $K_i = k_i/k_{-i}$. From the above expression we obtain the Hill Function

$$\frac{g_a}{g_{\text{tot}}} = \frac{K_1 A^n}{1 + K_1 A^n + K_2 R^m + K_3 K_1 A^n R^m}. \tag{5.13}$$

This function is plotted for $m = n = 2$ and $K_i = 1$ in Figure 4(b). Only when the concentration of A is high and the concentration of R is low is the gene active. In logical terms, the system implements the Boolean function $A \wedge \neg R$. ▲

Warning: The cooperativity n is a subject of some debate and misuse. It is not unusual to find a paper stating that, for their system, n is not an integer! The reason is that Hill Functions are used more to *fit* data than they are as first-principles models like we are presenting them above. The student is warned to treat the parameter n with a grain of salt: It does not necessarily correspond to anything physical. In fact, relieved of the need for a physical meaning for n, we are free to treat it as just another parameter.

5.4. Problems

5.1) Consider the system

$$EA + S \underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} EAS \xrightarrow{k_2} EA + P$$

$$E + A \underset{k_{-3}}{\overset{k_3}{\rightleftharpoons}} EA.$$

in which an enzyme has an inactive form E and an active form EA. The molecule A in this system is an activator. Determine an enzyme kinetics model for this system similar to the competive inhibition. That is, determine an approximate rate of product production as a function of v_S and v_A. Compare the mass action kinetics model of the system with the appoximate version in simulation to produce a plot similar to that in Figure 2(a).

5.2) Determine the Hill Function for the case of two activators where the active state of the gene is when both activators are bound to the gene.

5.3) Consider the bistable switch described by

$$\begin{aligned}
2A_1 &\underset{u_1}{\overset{k_1}{\rightleftharpoons}} B_1 \\
2A_2 &\underset{u_2}{\overset{k_2}{\rightleftharpoons}} B_2 \\
g_1 + B_2 &\underset{\gamma_2}{\overset{\gamma_1}{\rightleftharpoons}} g_{1,\mathit{off}} \\
g_2 + B_1 &\underset{\gamma_2}{\overset{\gamma_1}{\rightleftharpoons}} g_{2,\mathit{off}} \\
g_i &\xrightarrow{\alpha} g_i + A_i \\
A_i &\xrightarrow{\beta} \varnothing
\end{aligned}$$

Find equations for this system. Find two conservations and use them to reduce the number of equations by two. Assume that the first four reactions are fast and use the resulting equalities reduce the equations further.

Chapter 6

Synthetic Regulatory Networks

6.1. Re-programming *E. coli*

6.1.1. Basic Genetic Engineering. A nice feature of *E. coli*, and other organisms as well, is that an *E. coli* cell may contain one or more pieces of small, circular *plasmid* DNA. These independently replicating pieces of DNA are processed inside the cell much the same way as is genomic DNA. Furthermore, it turns out to be fairly easy to construct synthetic plasmids and stick them inside appropriately prepared cells. Once there, the ...

6.1.2. How we See What's Going On. how we see what is going on: fluorescence, micrsoscopy, gene arrays, RT-pcr, flow-cytometry, etc.
Regulation: Transcription factors, ribozymes and other mechanisms
Observables: GFP, FRET, populations, single cells

6.1.3. Inducing and Tuning Behaviors. Inputs: IPTG, UV, periodic inputs, etc.

6.2. Switches and Oscillators

As discussed in Chapter **??**, we can hook up genes that produce transcription factors to each other. Consider a set of genes g_1, ..., g_n. Draw each gene as a circle. If a gene produces a transcription factor that activates or represses another gene, we write an arrow or a line with a bar respectively from one gene to the other. If a small molecule, called an *inducer*, activates or deactivates a transcription factor, we draw similar arrows from nodes representing the molecule to the arrow representing an interaction as shown in Figure 11.

Synthetic biologists have built such networks by borrowing transcription factors and promotors the regulatory regions from various organisms. We describe some of these efforts below.

6.2.1. The Bistable Switch. In 2000, researchers from James Collins' lab published one of the first artificial genetic regulatory networks designed specifically to demonstrate the computational or information processing potential of genetic regulatory networks. The system was a bi-stable switch [**25**] designed from a pair of repressors that each repressed the expression of the other, as shown in Figure 1.

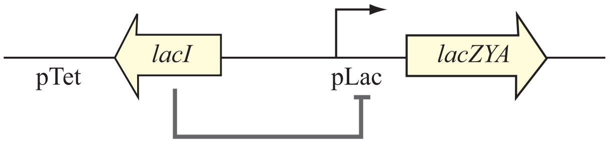


Figure 1. The genetic toggle switch. Two repressors and their promotors are encoded on a plasmid. The first repressor is *lacI*, which acts on the pLac promoter. The pLac promoter is placed before the gene for the second repressor, which was either *tetR* or λ, and *gfp*. *tetR* acts on the pTet promoter, which is placed before *lacI*.

The transcription factors used in the paper are the lactose inhibitor *lacI* from *E. coli* for the first repressor, and either tetracycline resistance *tetR* or λ from the bacteriophage λ for the second. As discussed above, *lacI* can be induced by IPTG. *tetR* can be induced by anhydrotetracycline, a derivative of tetracycline. Finally, λ can be induced by heat. The inducers allow experimentors to switch the toggle switch from one state to the other.

In the paper on the toggle switch, the authors propose the following model

$$\begin{aligned} X_1 &= \frac{\alpha_1}{1+X_2^n} - X_1 \\ X_2 &= \frac{\alpha_2}{1+X_1^n} - X_2. \end{aligned} \tag{6.1}$$

The authors note that bistability arises when the cooperativity exceeds unity: $n, m > 1$. For simplicity we assume that $\alpha_1 = \alpha_2 = \alpha > 0$.

The case when $n = m = 1$: When $n = 1$, there is only on positive equilibrium at

$$X_1^* = X_2^* = \frac{-1+\sqrt{1+4\alpha}}{2}.$$

The Jacobian at this point is

$$A = \left(\begin{array}{cc} -1 & -\frac{\alpha}{(1+X_2)^2} \\ -\frac{\alpha}{(1+X_2)^2} & -1 \end{array} \right) \Bigg|_{X=X^*} = \left(\begin{array}{cc} -1 & -\frac{4\alpha}{(1+X_2\sqrt{1+4\alpha})^2} \\ -\frac{4\alpha}{(1+X_2\sqrt{1+4\alpha})^2} & -1 \end{array} \right).$$

The eigenvalues of A are real and negative. Thus the single equilibrium is stable and the system does not describe a bistable switch!

The case when $n = m > 1$: In this case, it is difficult to obtain a closed form solution for X^*. One way to proceed is to analyze the phase portrait. To do this, we define the following notion.

Definition 6.1. Given a two-dimensional system of the form

$$\begin{aligned} \dot{x}_1 &= f_1(x_1, x_2) \\ \dot{x}_2 &= f_2(x_1, x_2), \end{aligned}$$

the first *nulcline* is the curve obtained by setting $f_1(x_1, x_2) = 0$, i.e. where the first state variable x_i is not changing. The second *nulcline* is obtained via $f_2(x_1, x_2) = 0$. Anywhere two nuclines intersect is an equilibrium point.

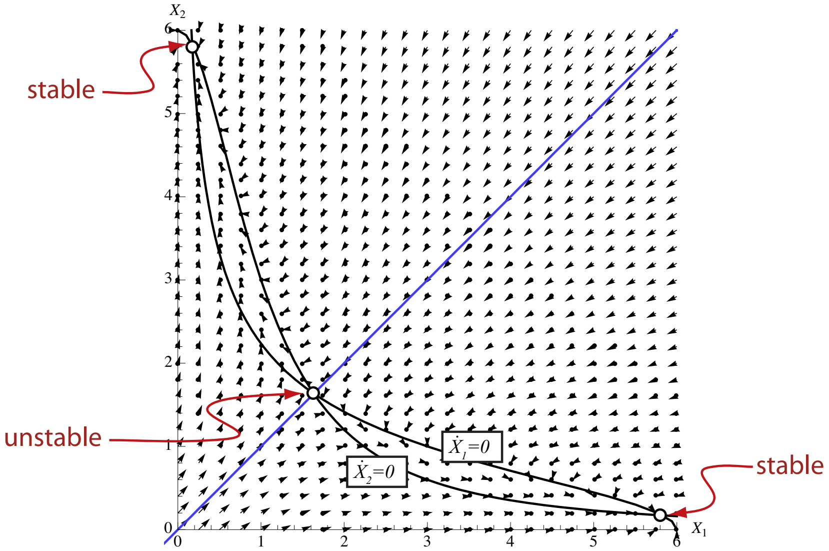


Figure 2. The nulclines and vector field associated with the bistable switch in Equation 6.1 with $n = 2$ and $\alpha = 6$. There are two stable equilibria and one unstable equilibrium.

The nulclines for the bistable swtich with $n = 2$ and $\alpha = 6$ are shown, with the vector field defined by Equation 6.1, in Figure 2. Note that the nulclines intersect in three points. Examination of various regions in the plot indicates which are stable and which are not. For example, above the two nulcines, all vectors point inward. Below, they point outward. Along the nulcline for $\dot{X}_1 = 0$ and between the separatrix (the blue line) and the top equilibrium point, $\dot{X}_2 > 0$. Along the $\dot{X}_2 = 0$ line and between the separatrix and the top equilibrium point, $\dot{X}_1 < 0$. From this we can conclude that the top equilibrium is stable. Similar reasoning shows the bottom equilibrium is stable as well.

No Cooperativity. There are papers stating that bistability is in fact impossible with non-cooperative system [**16**]. These papers use an extended Michaelis-Menton somewhat blindly without recalling the assumptions that go into the approximation. In particular, the assumption that there is a large amount of X_1 cannot hold when there is a large amount of X_2.

It turns out that a simple mass action model can produce bistability without cooperativity. Consider the network

$$\begin{array}{ccccc}
\varnothing & \xrightarrow{g_i \alpha_i} & X_i & \xrightarrow{\beta_i} & \varnothing \\
g_1 + X_2 & \underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} & g_{1,\mathit{off}} & \xrightarrow{\gamma_1} & g_1 \\
g_2 + X_1 & \underset{k_{-2}}{\overset{k_2}{\rightleftharpoons}} & g_{2,\mathit{off}} & \xrightarrow{\gamma_2} & g_2 .
\end{array}$$

Note that we have modeled the degradation of X_i in two places. One, X_i simply degrades. Two, X_i when bound to g_j to make $g_{j,\mathit{off}}$ degrades, turning $g_{j,\mathit{off}}$ into

g_j. We add this to the model, because when there is very little X_i, the degradation of even the small amount of it that is bound to g_j is significant.

It can be shown that the equation $\dot{v} = AK(v)$ has four equilibria and that parameters can be found so that one has $X_1 = X_2 < 0$ while the other three have $X_1 > 0$ and $X_2 > 0$. The three positive equilibra consist of two stable and one unstable equilibrium, similar to the cooperative version of bistability above.

6.2.2. The Repressilator. A construction similar to the bistable switch, and indeed appearing in the same issue of Nature in 2000, is the ring oscillator made out of three repressors [**19**], aptly named the *repressilator*.

6.2.3. Hysteresis Based Oscillators. Circadian rhythms are responsible for many phenomena, the most famous of which is the 17-year cycle of certain North American cicadas. Sleep cycles in humans are another. The repressilator is not a likely model for circadian rhythms, mostly because it is very sensitive to noise. A more frequently proposed model for the kind of structure that likely generates such oscillations is the *hysteresis based oscillator* [**9**], which was to some extent implemented in [**8**].

[**46**]

6.2.4. A Bandbass Filter. [**14**]

6.3. RNA Regulatory Mechanisms

Lock and key, ribozymes, etc. [**28**]

6.4. Problems

6.1) Look up the equations used in the paper about the Repressilator [**19**] and simulate them.

6.2) Look up the equations used in the paper about the hysteresis-based oscillator that uses *araC* and *lacI* [**46**] and simulate them.

Chapter 7

In vitro Synthetic Biology

In this chapter we explore biochemical networks that can be built *in vitro*, or in a test tube, out of simple molecules. We can use short strands of DNA that interact in ways similar to how RNA interacts inside cells, but without all of the other interactions seen inside cells [**23**]. We can also add enzymes, such as RNA polymerase, that allow us to explore transcription inside a test tube [**31**], once again without all of the other interactions seen inside cells.

7.1. Enzyme Free DNA Circuits

7.1.1. Mechanisms. Hybridization, strand invasion, displacement, FRET.

7.1.2. Design Constraints. Sequence independence, toeholds and kinetics, secondary structure.

7.1.3. Structure Prediction Software.

7.1.4. The Design of an OR Gate.

7.1.5. Three Different AND Gates. In this section, we describe three different mechanisms for making an AND gate. Each one uses the same reporter complex, which consists of two strands as shown in Figure 1(a). The top strand has the fluorophore, TAMRA, attached to it. The bottom strand has a quencher, BHQ or Black Hole Quencher, attached to it. The reporter does not fluoresce due to the proximity of the fluorophore and the quencher. To use the reporter, an input strand can be introduced that displaces the top strand. Once the top strand is displaced and moves away from the quencher, the top strand fluoresces.

The first AND gate design, shown in Figure 2, consists of a backbone strand (grey) and a toehold strand (magenta) and an intermediate strand (blue). The first input to the and gate, A, attaches to the toehold and, through strand invasion, removes the toehold strand, resulting in an intermediate complex and a waste complex. Removing the magenta strand exposes a second toehold that is complentary to part of input B, which attaches and displaces the blue intermediate. The blue intermediate then interacts with the reported to produce a fluorescent signal.

The second AND gate design, shown in Figure 3, is similar to the first and gate, expect that the first input A binds to the grey backbone strand. The other

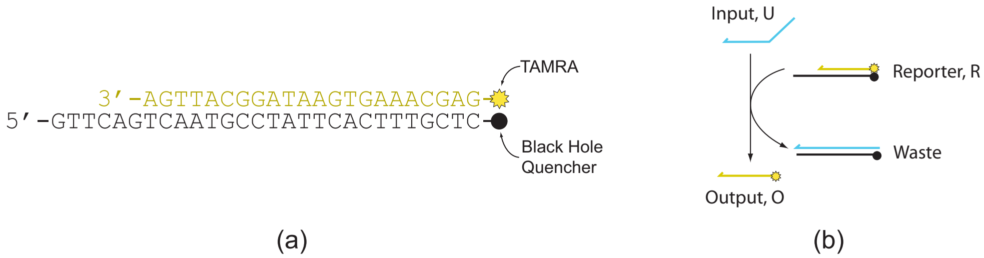


Figure 1. The reporter used for the logic gate design problem. (a) Sequence design. (b) Use of an input strand to displace the fluorophore-labeled strand.

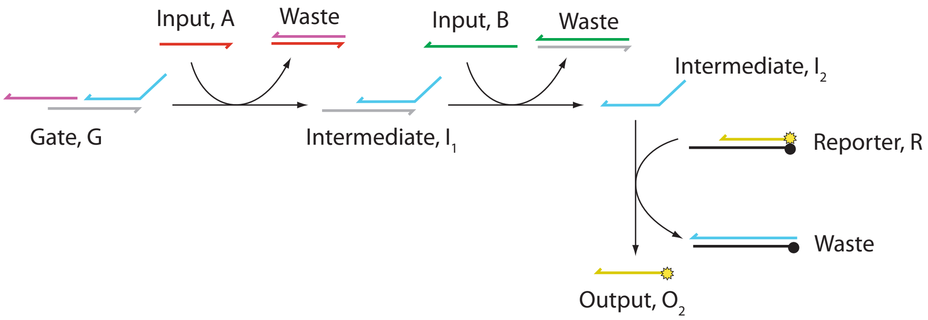


Figure 2. First design for an AND gate. Input A binds to the gate G, enabling input B to bind.

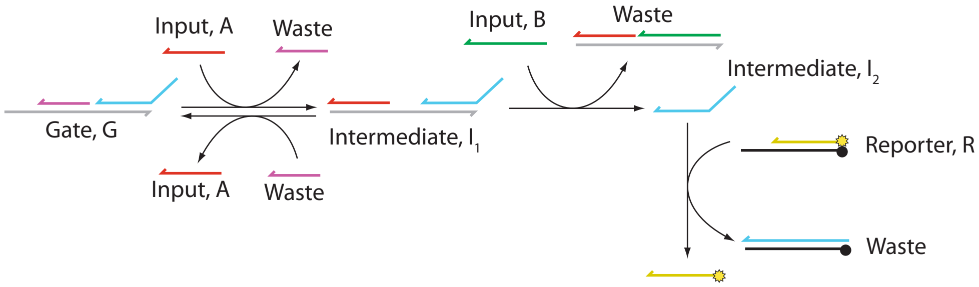


Figure 3. Second design for an AND gate. Input A binds to the gate G, enabling input B to bind. This design differs from the first design in that A can reversibly bind to the gate.

main difference is that the input A and the magenta strand essentially compete as equals for the grey backbone strand, making the first step of the process reversible.

In both of the first two AND gate designs, input A must bind to the gate before B can. The third AND gate design, shown in Figure 4, is symmetric in how the inputs attach to the gate. Either can bind, and unbind first. Only once the second input binds, the process cannot go back.

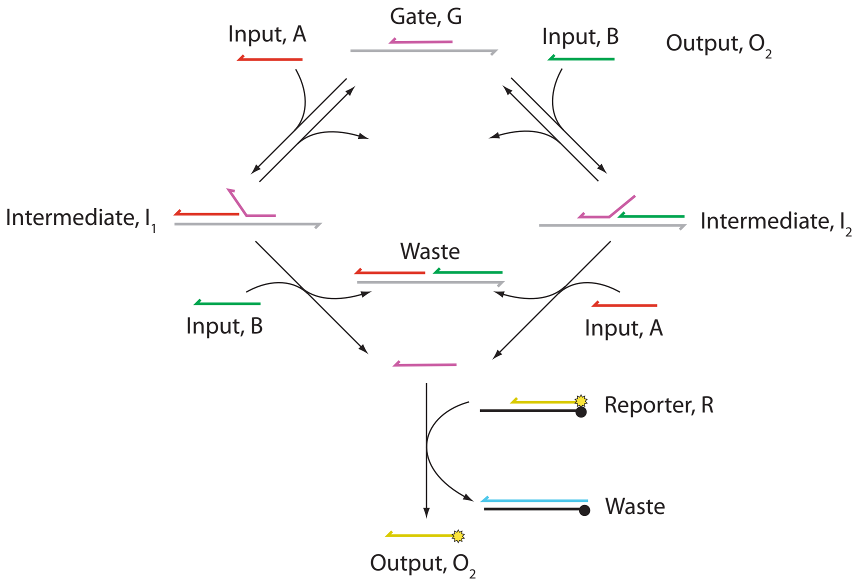


Figure 4. Third design for an AND gate. Input A and B both reversibly binds to the gate G to make an intermediate. The inputs can then bind irreversibly to complete the interaction. This allows the inputs to bind to the gate in either order.

7.2. Transcriptional Circuits

7.3. Problems

7.1) Describe each of the AND gates in Figure 2-4 in terms of mass action kinetics and simulate them using mass action kinetics using all combinations of the presense or absense of the input strands. Assume the rate constants are identical for all bimolecular reactions.

7.2) In this problem you will design sequences to build your own DNA logic gate.

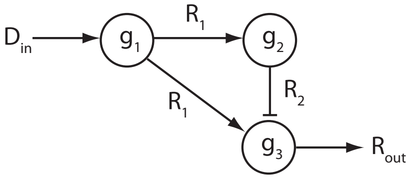
(a) Using the May 1, 2009 lecture as a guide, design sequences for an AND gate. Show the pairwise interactions between your strands predicted by NuPack, and also the results of each step in the reaction predicted by NuPack. You should name your sequences according to the scheme $GRPxSTDy$ where x is your group number and y is the strand number. Send your sequence designs to Kevin Oishi (koishi@u.washington.edu) by May 6 at 9:30 AM.

(b) Simulate your AND gate using mass action kinetics using all combinations of the presense or absense of the input strands. Assume that the reaction rate for each step is proportional to the number of bases in the toehold region that initiate the interaction. Use the following table to determine which AND gate design to use:

Group	Design
1	1
2	2
3	3
4	1
5	2
6	3
7	1
8	2
9	3
10	1

7.3) Design a system of interacting DNA strands (using no enzymes whatsoever) that implements a three input AND gate. You can design the topology of the strands and the domains, but do not need to design the actual sequences (unless you want to). Show simulations of your system using reasonable assumptions for every combination of the three inputs.

7.4) (A Transcriptional Pulse Generator) (a) Design a transcriptional system (using DNA, RNA, RNAP and RNase H as in [**31**]) that implements the following network:



where g_1, g_2 and g_3 are transcriptional swtiches, R_1, R_2 and R_{out} are RNA transcripts, and D_{in} is a DNA input signal. For this we use a DNA input signal because we don't want it to degrade via RNase H. Design the topology and the domains, but not the sequences. (b) Derive equations for this system based on regulatory network kinetics. (c) Simulate the system from an initial condition in which all RNAs have zero concentration and D_{in} is a step input (i.e. it goes from zero to a nonzero constant instantaneously). You should see a pulse in R_{out}.

Chapter 8

Stochastic Chemcal Kinetics

8.1. Stochasticity in Cell Dynamics

The formalism and example systems in the preceed chapters tacitly make the assumption that there are so many molecules in the system of interest that small fluctuations in the quantities, positions, energies, etc. of these molecules do not contribute significantly to the dynamics of the system. However, when we compare the behaviors of individual isogenic cells, we see substantial variation in their behaviors. For example, starting from isogenic and synchronized *E. coli* cells, the time between cell divisions is observed to be highly variable, ranging from 30 to 60 minutes [CHECK THIS] at a given temperature [**?**]. As another example, in bacterial chemotaxis, cells respond to the addition of uniform attractant by swimming smoothly, but relax back into tumbling at random times [**?**]. And as a third example, *E. coli* have been see to switch to and from presenting type I *fimbriae* at random time [**?**].

Two types of randomness have been proposed in the literature [**47**]: *Intrinsic* and *Extrinsic*. The former has to do with randomness that is part of a system of interest and that may be due to the fact that a given transcription factor may be present in integral quantities: 0, 1, 2, etc. In this case, one has to, for example, rethink what chemical equilibrium means: it may be that on average the number of molecules of each type is fixed, but reactions are still occuring and variations in the amounts of them can be significant. The latter has to do with ...

8.2. Foundations of Stochastic Chemical Reactions

The common assumption in modeling stochastic chemical systems is that the system is *well-mixed*. Mathematically, this means:

(1) The probability that a given pair of molecules reacts in the next dt seconds is independent of time;
(2) A given molecule is equally likely to interact with every other molecule in the system.

Physically, the assumptions 1 and 2 can arise if the rate of diffusion is much higher than the rate of reaction, so that the molecules are essentially uniformly distributed over the containing volume. In this case, every instant of time is essentially the same in terms of where the molecules are, and thus how the molecules arrived there is irrelevant.

Consider a vessel containing two molecules, A and B, that react to form a product molecule C according to

$$A + B \xrightarrow{k} C$$

where kdt is defined to be *the probability that molecules A and B react to form a C in the time interval* $[t, t + dt)$. Define by $f(t)$ the probability density function for when in time the reaction occurs when the system is started at time 0. Then

$$F(t) = \int_0^t f(\tau) d\tau$$

is the probability that the reaction occurs in time $[0, t]$. Suppose the reaction has not occured by time t. Because the system is well-mixed, we require that the probability density function $g(t)$ for when the reaction occur later, given that it has not occured at time t, is distributed identically to $f(t)$, except shifted in time. We can use this notion to determine an exact expression for $f(t)$.

First, put

$$G(t) = 1 - F(t)$$

to be the cumulative distribution for the event that the reaction occurs after time t. Define T be the random variable defining when the reaction occurs, so that $G(t) = P[T > t]$. Then the well-mixed assumption, essentially the same as the Markov assumption that future events only depend on the current state of the process, requires that

$$P[T > t + \tau | T > \tau] = P[T > \tau].$$

Using Bayes rule, we have

$$P[T > t + \tau] = P[T > t]P[T > \tau],$$

which amounts to

$$G(t + \tau) = G(t)G(\tau).$$

The only reasonable (integrable, analytic) function that satisfies the above equation is the exponential, so that $G(t) = e^{-\alpha t}$ for some $\alpha > 0$ (See Exercise 8.1). Since $F(t) = 1 - G(t)$ we have that

$$F(t) = 1 - e^{-\alpha t}$$

and $f(t) = \frac{d}{dt}F(t) = \alpha e^{-\alpha t}$. To determine what α is, note that $F(dt) = kdt$ so that $f(0) = \frac{d}{dt}F(dt)\big|_{t=0} = k$. Thus, $\alpha = k$ and

$$f(t) = ke^{-kt}.$$

Therefore, the statistics of the reaction time are distributed according to a Poisson waiting process with mean $1/k$.

The A, B, C system is a continuous time, discrete state Markov process. The above derivation allows us to write it down as such. First, note that the system has two states: state 1 in which the reaction has not occurred and state 2 in which it has occured. The rate at which we transition from state 1 to 2 is k. See Figure 1. Typically, when talking about Markov processes, we define the probability $p_i(t)$ of

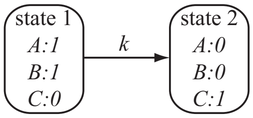


FIGURE 1. The Markov Process induced by the reaction $A + B \xrightarrow{k} C$.

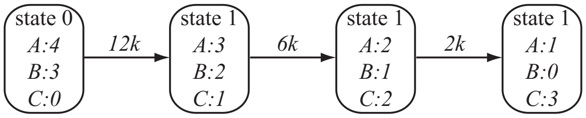


FIGURE 2. The Markov Process induced by the reaction $A + B \xrightarrow{k} C$ where initially there are 4 As, 3 Bs, and 0 Cs.

being in a particular state i. We have already worked these probabilites out. We have

$$\begin{aligned} p_1(t) &= G(t) = e^{-kt} \\ p_2(t) &= F(t) = 1 - e^{-kt}. \end{aligned}$$

If we differentiate $p_1(t)$ we get $\dot{p}_1 = -ke{-}kt = -kp_1$. Similarly, $\dot{p}_2 = kp_1$. We thus arrive a what is called Kolmogorov's equation or the *Chemical Master Equation* for this system:

$$\dot{p} = Qp \tag{8.1}$$

where p is a vector of probabilities and, in this case,

$$Q = \begin{pmatrix} -k & 0 \\ k & 0 \end{pmatrix}.$$

The matrix Q is called the infinitesimal generator or the *rate matrix* for the system.

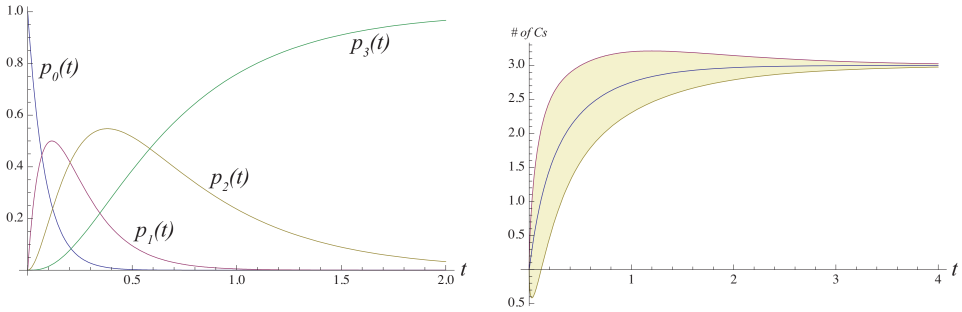
8.3. Simple Examples

In general, the rate matrix for a Markov process is constructed as follows. The $Q_{i,i}$ matrix is the negative sum of the rates leaving state i. The Qi, j matrix, for $i \neq j$, is the rate from state j to state i. Furthermore, when Q is small (say fewer than 1000×1000), the probability vector p can be solved for explicitly:

$$p(t) = e^{Qt}p(0).$$

All of the statistics, such as the mean copy number and variance of particular species, can be obtained from $p(t)$. Several examples are now given.

EXAMPLE 8.1. Suppose we look at the same reaction, $A + B \xrightarrow{k} C$, as above, except where there are initially 4 A molecules and 3 B molecules. This leads to a system with four states, which correspond to the number of C molecules in the system, either 0, 1, 2, or 3. Call these states 0 though 3. The rate at which the system progresses from state 0 to state 1 has to do with the probability that some A reacts with some B to produce a C. There are $4 \times 3 = 12$ possible such pairs, and so probability of transition from state 0 to 1 in time $[0, dt)$ is $12k$. The rest of the transitions are similar. The entire Markov process is shown in Figure 2.



(a) (b)

Figure 3. (a) The probabilities of being in one of the states of the Markov Process in Figure 2 versus time (for $k = 1$). (b) The mean number of Cs and the one-standard window.

Kolmogorov's equation for this system, $\dot{p} = Qp$, is

$$\begin{pmatrix} \dot{p}_0 \\ \dot{p}_1 \\ \dot{p}_2 \\ \dot{p}_3 \end{pmatrix} = \begin{pmatrix} -12k & 0 & 0 & 0 \\ 12k & 6k & 0 & 0 \\ 0 & -6k & -2k & 0 \\ 0 & 0 & 2k & 0 \end{pmatrix} \begin{pmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \end{pmatrix}.$$

This equation can be solved[1] to obtain:

$$\begin{aligned} p_0(t) &= e^{-12t} \\ p_1(t) &= 2e^{-12t}\left(e^{6kt}-1\right) \\ p_2(t) &= 2\frac{3}{5}e^{-12t}\left(e^{2kt}-1\right)^2\left(2+4e^{2kt}+6e^{4kt}+3e^{6kt}\right) \\ p_2(t) &= 2\frac{1}{5}e^{-12t}\left(e^{2kt}-1\right)^3\left(1+3e^{2kt}+6e^{4kt}+5e^{6kt}\right). \end{aligned}$$

From the above expressions, we can obtain the mean number of C molecules

$$\langle C \rangle = \sum_i ip_i = 3 - \frac{1}{5}e^{-12t} - e^{-6t} - \frac{9}{5}e^{-2t}$$

and the second moment

$$\langle C^2 \rangle = \sum_i i^2 p_i = 9 - e^{-12t} + e^{-6t} - 9e^{-2t}.$$

Figure 3(a) shows the probabilities versus time. The initial state contains no Cs. The probabilit of those states decays exponentially. The probability of the intermediate states increases and then decreases, and finally all the probability is absorbed in the final state. Figure 3(b) shows the mean number of Cs and the one standard deviation window (calculated from $\langle C \rangle$ and $\langle C^2 \rangle$) for the reaction. ▲

[1] To find the solution to $\dot{x} = Ax$ where A is a matrix, we use the result that $x(t) = e^{At}x(0)$ where $e^A = I + A + A^2/2! + A^3/3! + \ldots$. The matrix e^{At} can be found in MATLAB using the `expm` function or in *Mathematica* using the `MatrixExp` function.

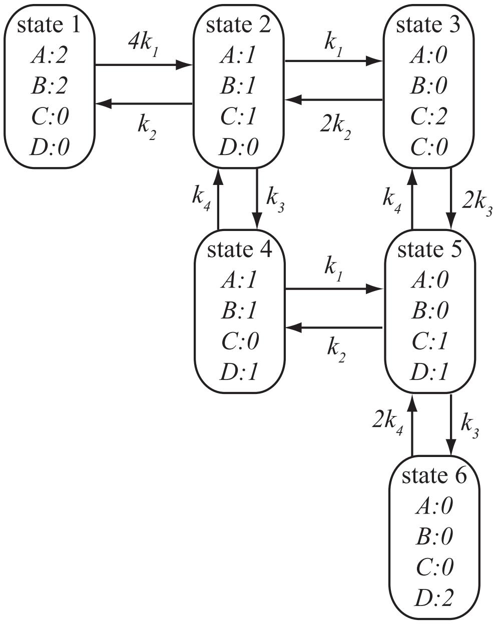


Figure 4. The Markov process for the chemical reaction in Example 8.3.

Example 8.2. Consider the reaction

$$A \underset{k_2}{\overset{k_1}{\rightleftharpoons}} B$$

and suppose we start with two As and two Bs for a total of 4 molecules and five states. Then we arrive at a five state Markov process with

$$Q = \begin{pmatrix} -4k_2 & k_1 & 0 & 0 & 0 \\ 4k_2 & -k_1 - 3k_2 & 2k_1 & 0 & 0 \\ 0 & 3k_2 & -2k_1 - 2k_2 & 3k_1 & 0 \\ 0 & 0 & 2k_2 & -3k_1 - k_2 & 4k_1 \\ 0 & 0 & 0 & k_2 & -4k_1 \end{pmatrix}$$

where index i corresponds to the state in which there are i A molecules and $4 - i$ B molecules.

If we start with some arbitrary number of molecules n, the above matrix could only be written schematically, and it would be difficult to obtain the moments of A and B, as we did in the previous example, in terms of n. In the next section, we describe a method that sometimes workds for dealing with this kind of problem. ▲

In general, if a system has a mass vector, so that new molecular mass is not created, then resulting Markov process has a (possibly very many) finite number of states. If a system can produce molecules from nowhere, even if they are degraded at some rate, then there can in principle be any number of such molecules in the system, resulting in an infinite number of states. The next two examples describe conservative and a non-conservative systems.

Example 8.3. The system

$$\begin{array}{c} A + B \quad \underset{k_2}{\overset{k_1}{\rightleftharpoons}} \quad C \\ C \underset{k_4}{\overset{k_3}{\rightleftharpoons}} D \end{array}$$

is conservative (show this). The Markov process it produces, given initially two As, two Bs and no Cs or Ds is shown in Figure 4 and the resulting rate matrix is

$$Q = \begin{pmatrix} -4k_1 & k_2 & 0 & 0 & 0 & 0 \\ 4k_1 & -k_1 - k_2 - k_3 & 2k_2 & k_4 & 0 & 0 \\ 0 & k_1 & -2k_2 - 2k_3 & 0 & 0 & 0 \\ 0 & k_3 & 0 & -k_1 - k_4 & k_2 & 0 \\ 0 & 0 & 2k_3 & k_1 & -k_2 - k_3 & 2k_4 \\ 0 & 0 & 0 & 0 & k_3 & -2k_4 \end{pmatrix}$$

▲

Example 8.4. Consider the open-loop expression and degradation of a gene X according to

$$\varnothing \xrightarrow{k_1} X \xrightarrow{k_2} \varnothing.$$

Probalistically speaking, this system can produce any number of X molecules. The result is a Markov process with an infinite number of states 0, 1, 2, ... describing the number of X molecules in the system. We can describe the rate of change of the probabilities of these states by the inifinite set of differential equations

$$\begin{array}{rcllllll} \dot{p}_0 & = & -k_1p_0 & +k_2p_1 & & & & \\ \dot{p}_1 & = & k_1p_0 & -(k_1+k_2)k_2p_1 & +2k_2p_2 & & & \\ \dot{p}_2 & = & & k_1p_1 & -(k_1+2k_2)p_2 & 3k_2p_3 & & \\ \dot{p}_3 & = & & & k_1p_2 & -(k_1+3k_2)p_3 & 4k_2p_3 & \\ & \vdots & & & & & & \end{array}$$

Some useful information can be gained from this set of equations. For example, the steady state distribution can be found as follows. First, set the above equations to zero. The first equation gives

$$p_1^* = \frac{k_1}{k_2}p_0^*.$$

Adding the first two equations gives

$$0 = -k_1p_1^* + 2k_2p_2*$$

from which we obtain

$$p_2^* = \frac{k_1}{2k_2}p_1^* = \frac{k_1^2}{2k_2^2}p_0^*.$$

In general, adding the first n equations gives

$$p_n^* = \frac{\alpha^n}{n!}p_0^*$$

where $\alpha = k_1/k_2$. Thus, we have all of the probabilities at steady state in terms of p_0^*, which we can determine by noting that the sum of the probabilities should be 1. That is

$$\sum_{n=0}^{\infty} p_n^* = \sum_{n=0}^{\infty} \frac{\alpha^n}{n!} p_0^* = 1 \Leftrightarrow \sum_{n=0}^{\infty} \frac{\alpha^n}{n!} = \frac{1}{p_0^*}.$$

The infinite sum evaluates to e^α so that $p_0^* = e^{-\alpha}$ and in general,

$$p_n^* = \frac{\alpha^n}{n!} e^{-\alpha}.$$

From this result, we can obtain, for example, the mean and second moment:

$$\langle X \rangle^* = \sum n p_n^* = \alpha = \frac{k_1}{k_2}$$

and

$$\langle X^2 \rangle^* = \sum n^2 p_n^* = \alpha + \alpha^2.$$

Using the fact that the standard deviation of X is $\sqrt{\langle X^2 \rangle - \langle X \rangle^2}$ be get that the simple result that the standard deviation in X is $\sqrt{\alpha} = \sqrt{k_1/k_2}$. ▲

8.4. Moment Dynamics

When there are an arbitrary number of molecules present in the system, explicity enumerating the Markov process doesn't really work. For example, consider Example 8.2 in which we enumerated the states for $A \rightleftharpoons B$ for two molecules. We might like to know the statistics of how many molecules of A we have at equilibrium, independent of the initial number of As and Bs. The other problem is that the system may have an infinite number of states, in which case it may be difficult to reason about what is going on, although we were successul with the example $\varnothing \rightharpoonup X \rightharpoonup \varnothing$.

Another way to proceed is to use the *extended generator* to produce the moment dynamics of a random variable of interest. Suppose we have a reaction network with state space S and having transitions $1, 2, 3, ..., r$ with rates k_1, k_2, ..., $'k_r$. Define a *test function* over the state space by ψ. For example, ψ might take a state s and return the number of A molecules in that state. The rate of change of the expected value of ψ follows the simple formula

$$\frac{d}{dt}\langle \psi \rangle = \langle L\psi \rangle \tag{8.2}$$

where L is the extended generator, defined by

$$L\psi = \sum_{i=1}^{r} (\psi(s') - \psi(s)) k_i. \tag{8.3}$$

In the equation for L, we sum over all reactions i the rate k_i times the difference of the value of ψ after the reaction and the valueof ψ before the reaction. We can put any polynomial we like into the equation for ψ, such as A or A^2 to get the rate of change of the first and second moments.

Example 8.5. Again, consider the reaction

$$A \overset{k_1}{\underset{k_2}{\rightleftharpoons}} B$$

that we first encountered in example 8.2. This system has two reactions, the rate of the first is $k_1 A$ and the rate of the second is $k_2 B$. In general, the generator is

$$L\psi = \left(\psi\begin{pmatrix} A-1 \\ B+1 \end{pmatrix} - \psi\begin{pmatrix} A \\ B \end{pmatrix}\right) k_1 A + \left(\psi\begin{pmatrix} A+1 \\ B+-1 \end{pmatrix} - \psi\begin{pmatrix} A \\ B \end{pmatrix}\right) k_2 B.$$

We can use the generator to determine the rate of change of the expected number of A and B molecules. We have

$$\begin{aligned} LA &= (A-1-A)k_1 A + (A+1-A)k_2 B = -k_1 A + k_2 B \\ LB &= (B+1-B)k_1 A + (B-1-B)k_2 B = k_1 A - k_2 B \end{aligned}$$

Therefore, using Equation (8.2) we have

$$\begin{aligned} \frac{d}{dt}\langle A\rangle &= -k_1\langle A\rangle + k_2\langle B\rangle \\ \frac{d}{dt}\langle B\rangle &= k_1\langle A\rangle - k_2\langle B\rangle. \end{aligned}$$

We can also determine the second moment dynamics. These are given by

$$\begin{aligned} LA^2 &= ((A-1)^2 - A^2)k_1 A + ((A+1)^2 - A^2)k_2 B \\ &= (-2A+1)k_1 A + (2A+1)k_2 B \\ &= k_1 A - 2k_1 A^2 + k_2 B + 2k_2 AB. \end{aligned}$$

Notice that if we put the above into the Equation (8.2), we would get the rate of change of $\langle A^2\rangle$ in terms of the moments $\langle A\rangle$, $\langle B\rangle$, and $\langle A^2\rangle$ (which we already know), but also $\langle AB\rangle$. We could expand this as well, but note that $A+B=n$ is constant. So instead, we get

$$\begin{aligned} \frac{d}{dt}\langle A^2\rangle &= k_1\langle A\rangle - 2k_1\langle A^2\rangle + k_2 n - k_2\langle A\rangle + 2k_2 n\langle A\rangle - 2k_2\langle A^2\rangle \\ &= k_2 n + (k_1 - k_2 + 2k_2 n)\langle A\rangle - 2(k_2 + k_1)\langle A^2\rangle. \end{aligned}$$

Note: we could also have eliminated the equation for $\langle B\rangle$ and the result would be the rates of change of $\langle A\rangle$ and $\langle A^2\rangle$ in terms of $\langle A\rangle$ and $\langle A^2\rangle$. ▲

EXAMPLE 8.6. Consider again the gene expression example

$$\varnothing \xrightarrow{k_1} X \xrightarrow{k_2} \varnothing.$$

We have for the first moment

$$LX = k_1 - k_2 X$$

so that

$$\langle X\rangle = k_1 - k_2\langle X\rangle$$

and $\langle X\rangle^* = \frac{k_1}{k_2}$. For the second moment

$$\begin{aligned} LX^2 &= ((X+1)^2 - X^2)k_1 + ((X-1)^2 - X^2)k_2 X \\ &= (2X+1)k_1 + (-2X+1)k_2 X \\ &= k_1 + (2k_1 + k_2)X - 2k_2 X^2 \end{aligned}$$

so that

$$\langle X^2\rangle = k_1 + (k_1 + k_2)\langle X\rangle - k_2\langle X^2\rangle.$$

At equilibrium, we have

$$\begin{aligned}\langle X^2\rangle &= \frac{k_1+(2k_1+k_2)\frac{k_1}{k_2}}{2k_2} \\ &= \frac{k_1}{k_2}+\frac{k_1^2}{k_2^2}\end{aligned}$$

which is the same result as in Example 8.4. ▲

The extended generator is a powerful tool for accessing the statistics of a stochastic chemical reaction, but it does have limitations, which we now describe. Suppose a system has molecules A, B, C. The moments of interest are usually the first moments $\langle A\rangle$, $\langle B\rangle$, $\langle C\rangle$ and the second moments $\langle A^2\rangle$, $\langle AB\rangle$, $\langle AC\rangle$, $\langle B^2\rangle$, $\langle BC\rangle$, etc. There are higher order moments too, such as $\langle A^2BC^3\rangle$, but one typically is not concerned with these. One promising looking result is that the dynamics of each moment are linear in the rest of the moments. That is, if we stack all the moments in an infinite dimensional vector μ, then $\dot\mu = A\mu + B$ describes the rate of change of all the moments, where A is an infinite dimensional matrix and B is an infinite dimensional vector.

However, it may be that lower order moments depend on high order ones. This is not the case in Examples 8.5 and 8.6 where the rate of change of the first moments depend on only the first moments; the rate of change of the second moments depend on the rate of change of the first and second; and so on. We call this property *moment closure*: The rates of change of all moments depend only on themselves and lower order moments. This property, when present, makes the moment equations easy to solve. A general result, that the above two examples share, is as follows.

THEOREM 8.7. *The moment dynamics of a system of reactions in which each reaction is no more than unimolecular (involving one or fewer reactants) are closed.*

For bimolecular reactions, which includes many systems of interest, the moments are typically not closed, as the following example shows.

EXAMPLE 8.8. Considera again the reaction

$$A + B \overset{k}{\rightharpoonup} C.$$

Note that if there are initially n_A As, n_B Bs and zero Cs, we have the conservation laws $A = n_A - C$ and $B = n_B - C$. Therefore, we can simply look at the moments of C. We have

$$LC^n = ((C+1)^n - C^n)\,ABk = ((C+1)^n - C^n)\,(n_A - C)(n_B - C)k,$$

which is an $n{+}1$ order polynomial. Thus $\langle C^n\rangle$ depends on $\langle C^{n+1}\rangle$ and the moments are not closed. ▲

8.5. Approximate Methods

A discussion of the finite state projection algorithm goes here.

8.6. Simulation

Analytical methods for solving stochasti processes are hard to come by, and are not always easy to analyze. A common recourse is to

Simulating stochastic processes can be difficult. A naive approach, and one that is really the best you can do for arbitrary stochasti processes, is to choose what will happen in the next *delta* seconds, where $\delta > 0$ is a small number. For example, if there are two reactions that can occur, one with rate k_1 and one with rate k_2, then one would roll a three sided die with weights

$$p_1 = \delta k_1, \;\; p_2 = \delta k_2, \;\; p_3 = 1 - \delta(k_1 + k_2).$$

If the die comes up with option 1 or 2, the state is updated. If the die comes up with option three, the state is not update. Then the time is updated by δ seconds and the process continues.

There are several problems with this approach. First, it is approximate: The trajectories obtained in this manner are not exactly drawn from the correct distribution, although they sampling becomes more correct as $\delta \to 0$. Second, if δ is small, then the only thing that happens in most steps is that the time is updated. This is because the probability that something happens in the next δ seconds approaches zero as δ approaches zero! So the more accurate the simulation, the longer it takes to simulate.

Continuous time Markov processes are actually much easier to simulate. The observation that one can "exactly" simulate the Markov process arising from a stochastic chemical reaction is due to Gillespie [**26**]. The algorithm is called the *Stochastic Simulation Algorithm* or SSA, but is oftern referred to as the *Gillespie Algorithm.*

To explain how it works, consider a Markov process with states $1, 2, 3, ...$ Suppose the current state is i at time 0. We would like to know the joint probability $P[t, j]$ that the next transition is from i to j and occurs in time $[t, t + dt]$. We can use the rate $k_{i,j}$, which is the probability that an i to j transition occurs in the next dt seconds, to get

$$P[t, j] = P[T > t]k_{i,j}dt \tag{8.4}$$

where $P[T > t]$ is the probability that the transition does not occur in time $[0, t]$.

We can derive a similar expression for $P[T > t + dt]$. First note that

$$\left(1 - \sum_{j=1^N} k_{i,j}\right) dt = (1 - K_i)$$

is equivalent to the probability that no transition occurs in the next dt seconds. Here, we have defined $K_i = \sum_{j=1}^N k_{i,j}$ to be the total rate out of state i. Thus,

$$P[T > t + dt] = P[T > t](1 - K_i)dt.$$

Rearranging gives

$$\frac{P[T > t + dt] - P[T > t]}{dt} = -P[T > t]K_i.$$

The left hand side of this is $\frac{d}{dt}P[T > t]$. Solving this differential equation for $P[T > t]$ and substituting into Equation (8.4) gives

$$P[t, j] = k_{i,j}e^{-K_i t}. \tag{8.5}$$

We can now determine how to choose the next state and the next reaction time.

Next Reaction: Integrating Equation (8.5) over all time gives the probability that the next reaction is j, no matter what time it occurs.

$$p_{i,j} = \int_0^\infty k_{i,j} e^{-K_i t} dt = \frac{k_{i,j}}{K_i}. \tag{8.6}$$

Thus, to choose the next reaction, we roll an N sided die, with weights $p_{i,j}$ for $j = 1, N$.

Reaction Time: To choose the time, we sum over all reactions to get that the proabilility densitiy function for the next reaction time is

$$\sum_{j=1}^{N} k_{i,j} e^{-K_i t} = K_i e^{-K_i t}.$$

The cumulative distribution function corresponding to this function is

$$\int_0^t K_i e^{-K_i t \tau} d\tau = 1 - e^{-K_i t} = r \in [0,1],$$

which relates t distributed according to the Poisson distribution $K_i e^{-K_i t}$ and r distributed uniformly in $[0,1]$. Since computers can pick uniform random numbers easily, we solve for t to get

$$t = \frac{1}{K_i} \ln \frac{1}{1-r}. \tag{8.7}$$

Therefore, to choose the next time in a simulation, we choose r uniformly in $[0,1]$ and then compute the time according to the above.

The SSA:. Now, using the above results, we describe the Stochastic Simulation Algorithm.

1. Choose an initial condition v equal to some vector of the copy numbers of the species in the reaction network.
2. Set $t = 0$.
3. For each reaction ρ applicable in v, determine the rate k_v.
4. Choose the next reaction via Equation 8.6.
5. Choose the Δt via Equation 8.7 and set t to $t + \Delta t$.
6. Goto 3.

In practice, one typically sets a maximum time and a maximum number of steps to force a termination condition. Also, a single trajectory is usually not sufficient to characterize the behavior, so multiple trajectories are sampled (say thousands) and combined to obtain sample averages.

Example 8.9. Consider the reaction network

$$E + S \underset{k_2}{\overset{k_1}{\rightleftharpoons}} ES \xrightarrow{k_3} E + P$$

where $k_1 = 1$, $k_2 = 0.5$ and $k_3 = 0.1$. Starting in a state $v_1 = (E, S, ES, P) = (2, 10, 0, 0)$ at time $t = 0$, the only reaction applicable is the first. Thus the probability that the next state is $(1, 9, 1, 0)$ is 1. The reaction has rate $20k_1 = 20$. The

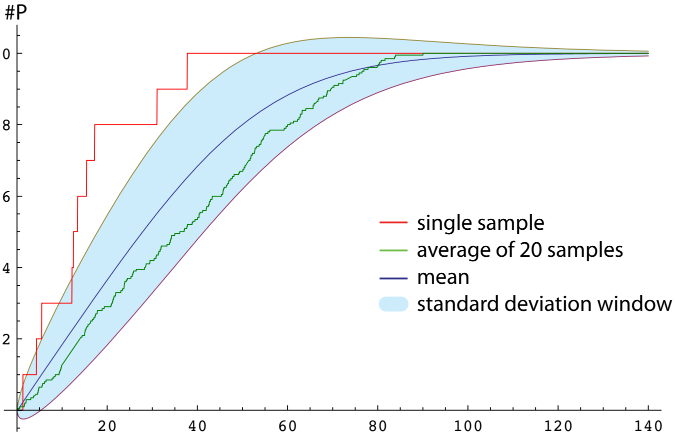


FIGURE 5. The Michaelis-Menton system from Example 8.9. Shown are a sample trajectory (red), 20 samples averaged together (blue) and the mean and standard deviation window computed from the Kolmogorov Equation.

time the reaction occurs is distributed according to the p.d.f. $20e^{-20t}$. From the state $(1, 9, 1, 0)$ there are three possible reactions with rates

$$9, \ 0.5, \ \text{and } 0.9.$$

Thus, the probabilities of each of these reactions are

$$\frac{9}{10.4}, \ \frac{0.5}{10.4}, \ \text{and } \frac{0.9}{10.4}$$

and the p.d.f. of when the reaction occurs is distributed according to $10.4e^{-10.4t}$. Figure 5 shows a typical sample trajectory along with the average of 20 samples, and the mean and variance computed from the Kolmogorov Equation in which all 30 states are enumerated.

8.7. Computing with Molecules

One of the central questions of synthetic biology is whether and how computations can be carried out with molecular systems. Many researchers have tried to answer this question and computations have been done with oligos [**4**] and DNA self-assembly [**52**] for example. Furthermore, many models of molecular computation have been proposed [**?**]. Here, we examn one such model, which is an implementation of regiester machines [**?**] by stochastic chemical reaction networks [**?**].

A *register machine* is a kind of abstract computational machine. It is equivalent in expressibility to the Turning Machine model, and thus any algorithm can in principle be implemented as a register machine. Register machines consist a finite set of states S_0, S_1, ..., S_n and a set of registers R_0, R_1, ..., R_m. Register machine transitions have of two types. The first is an increment:

$$\text{inc}(i, r, j)$$

which means *if the state is i, then increase register r and go to state j.* The second is a decrement:

$$\text{dec}(i, r, j, k)$$

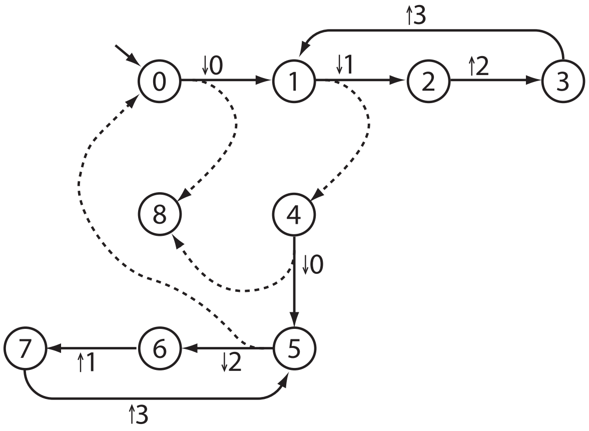


Figure 6. A register machine that multiplies the initial contents of registers 0 and 1. Register 3 holds the final value and register 2 is swap space.

which means *if the state is i and register r is greater than zero, then decrease register r and go to state j; otherwise go to state k.*.

Register machines can be written as diagrams, as in Figure 6, which describes a register machine that can multiply. The increment transitions are simply arrows from i to j with the register r being incremented shown by the arrow. For example, the arrow from state 2 to 3 labeled ↑ 2 describes the command inc$(2, 2, 3)$. The decrement transitions are similar, except that a dashed arrow describes what state to go to if the register being decremented is zero.

To implement a register machine with stochastic chemical reactions, we do the following. First, introduce a new chemical species S_i for each sate. We will require that at any given time, the total number of state molecules of any kind is 1. Also, we introduce new chemical species R_i for every register. The number of R_i molecules in the system at a given time corresponds to the value of that register at that time. Next, for each increment transition inc(i, r, j) in the system we introduce the reaction

$$S_i \xrightharpoonup{k} S_j + M_r.$$

Finally, for each decrement transition dec(i, r, j, k) we introduce the two reactions

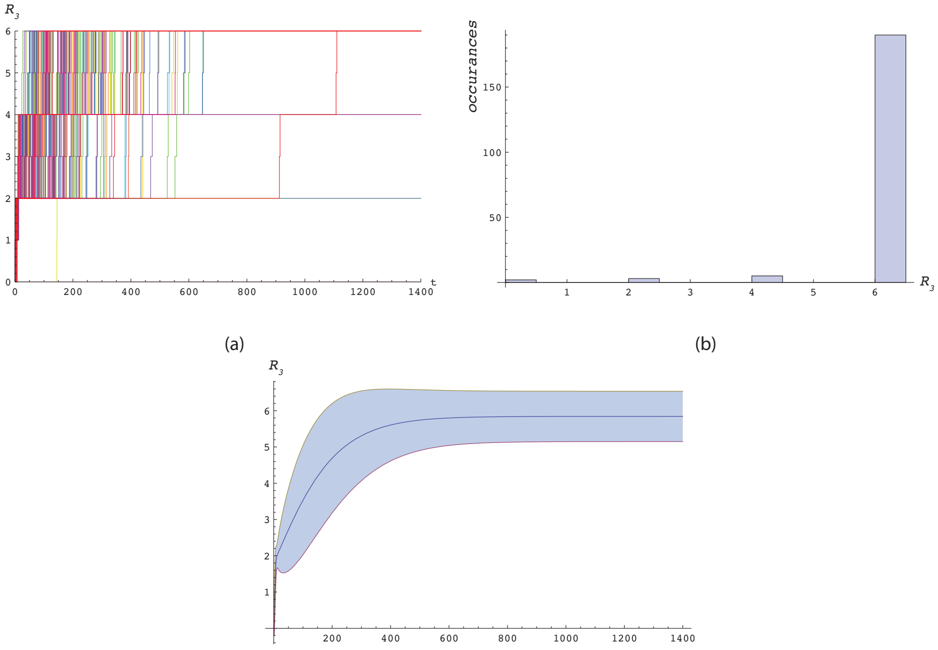
$$S_i + M_r \xrightharpoonup{k} S_j$$

and

$$S_i \xrightharpoonup{\varepsilon} S_k.$$

The first decrement reaction describes the interaction of an S_i molecule with an M_r molecule. This reaction has rate $M_r k$ so that if there are no M_r molecules, the reaction rate is zero. The second reaction is the "else" part of the decrement rule. If there are no M_r molecules, then eventually this section reaction should occur. However, there is some small probability

$$\frac{\varepsilon}{\varepsilon + M_r k}$$



(a) (b)

(c)

Figure 7. (a) Several trajectories of the chemical implementation of the multiplication register machine computing 2×3. (b) The histogram of the computed results. (c) The mean and standard deviation behavior computed from the Kolmogorov equation.

that the second decrement reaction might occur even if there are M_r molecules around. Thus, we would like ϵ to be small. On the other hand, when it is not an error to take the second decrement reaction, we sould like the rate to be large. This tradeoff is fundamental to this formulation: *accurate computation takes longer.*

Example 8.10. The register machine[2]in Figure 6 multiplies the contents of registers R_0 and R_1 and puts the result in register R_3. This can be implemented with twelve reactions: two each for the decrement reactions and one each for the increment reactions. Figures 7(a) and (b) shows the resulting behavior when computing $2 \times 3 = 6$ with $k = 1$ and $\epsilon = 0.01$. Stochastic simulations show the system can end up with the result 0, 2, 4 and 6 with the most likely answer being the correct one. Figure 7(c) shows the exact solution. Using the methods in this chapter, the entire state space (of the Markov process) was enumerated to yield 78 states and a 78×78 dimensional rate matrix. Shown are the average number of R_3 molecules and the standard deviation window. ▲

[2] The author thanks Kevin Oishi for finding a way to multiply with register machines.

8.8. Problems

8.1) Show that if $G(t+\tau) = G(t)G(\tau)$ is analytical and integrable, then $G(t) = e^{-\alpha t}$ for some $\alpha > 0$.

8.2) Consider the reaction network

$$\begin{aligned} A &\xrightarrow{k_1} B + C \\ C + D &\underset{k_3}{\overset{k_2}{\rightleftharpoons}} E \end{aligned}$$

starting with two As and two Ds and no Bs, Cs or Es. (a) Enumerate the states and draw the Markov process the results from this system. (b) Determine the rate matrix. (c) Find the probabilities of being in each state as a function of time by finding $e^{Qt}p(0)$. (d) Find the mean and variance for the number of Es as a function of time and plot the mean and the one standard deviation window versus time (as in Figure 3(c)) assuming all the rates are unity. Note: To simiplfy things, you may set $k_1 = 1$, $k_2 = 1$, and $k_3 = 1$. Also, if you use MATLAB or Mathematica to compute the solution, you do not need to show the output if it is large, just show the code.

8.3) Show that if a system of reactions admits a mass vector, then the Markov process describing its stocahstic behavior has a finite number of states. Give an example of a non-conservative system that still produces a finite number of states.

8.4) For the gene production and degradation system described in Examples 8.4 and **??**, determine the number n such that the probability of there being more than n molecules of X at steady state is less than 1%.

8.5) Determine the time varying probabilities of being in the various states of Example 8.2. Plot the probabilities versus time for the case when both rates are 1 and we start with no B molecules.

8.6) Verify that the moment equations for $\langle A \rangle$ and $\langle A^2 \rangle$ obtained in Example 8.5 match those obtained using the solutions from Exercise 8.5.

8.7) Consider the system

$$\varnothing \xrightarrow{k_1} X \xrightarrow{k_2} Y \xrightarrow{k_3} \varnothing.$$

Using equations (8.2) and 8.3, determine equations for $\langle X \rangle$, $\langle Y \rangle$, $\langle X^2 \rangle$, and $\langle Y^2 \rangle$ versus time. Determine the steady state values for the means and standard deviations. Starting from no molecules, plot the expected number of X and Y molecules and standard deviation windows (as in Figure 3(c)) assuming the rates are all unity.

8.8)
Consider the network

$$EA + S \underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} EAS \xrightarrow{k_2} EA + P$$

$$E + A \underset{k_{-3}}{\overset{k_3}{\rightleftharpoons}} EA$$

with rates

k_1	$9.3\ s^{-1}$
k_{-1}	$1.2\ s^{-1}$
k_2	$0.5\ s^{-1}$
k_3	$5.0\ s^{-1}$
k_{-3}	$1.5\ s^{-1}$.

a) Suppose the initial number of E, A and S molecules is 2, 1 and 4 respectively and the initial number of the remaining molecule types is 0. Draw the Markov process that results.

b) Plot the mean and standard deviation window for the number of P molecules as a function of time, as in Figure 5. You should obtain these either by solving the Kolmogorov Equation via e^{Qt} or simply simulating the differential equations. You should find $\langle P \rangle$ and $\langle P^2 \rangle$ as functions of time, and use these to find the standard deviation as a function of time.

c) Simulate the system 20 times using the stochastic simulation Make a plot of one of the trajectories and of the average of all the trajectories showing the number of P molecules, as in Figure 5.

8.9) Consider the system

$$\varnothing \xrightarrow{k_1} X \xrightarrow{k_2} \varnothing,$$

which produces an infinite dimensional Markov process. Kolmogorov's equation for this system has an infinite dimensional Q matrix. In this exercise, you will approximate this matrix wirth a series of finite matrices. Define $Q(n)$ to be the rate matrix for the following approximation to the above system

$$0 \underset{k_2}{\overset{k_1}{\rightleftharpoons}} 1 \underset{2k_2}{\overset{k_1}{\rightleftharpoons}} 2 \underset{3k_2}{\overset{k_1}{\rightleftharpoons}} \ldots \underset{nk_2}{\overset{k_1}{\rightleftharpoons}} n.$$

Consider the case when $k_1 = 5$ and $k_2 = 1$ and there are initially no X molecules. Using $\dot{p} = e^{Q(n)t}p(0)$, plot the mean number of X molecules (plus standard deviation windows) versus time for $n = 5,\ 10,\ 20,\ 30,$ and 40. How could this idea be used to approximate the behavior any reaction network, with a finite number of states or not?

8.10) Write down the chemical reactions for the multiplication register machine shown in Figure 6.

8.11) What is the probability that the chemical implementation of the multiplication register machine computes 1×1 incorrectly (in terms of k and ε?).

8.12) Find a register machine that determines whether the initial value of of register R_0 is even or odd. If it is even, the machine should terminate with a 0 in register R_1. Otherwise, it should terminate with a 1 in register R_1.

8.13) Find the stochastic chemical reaction implementation of the register machine you found in Exercise 8.12. Using Gillespie's algorithm, simulate the reaction network 100 times for 5 initial R_0 molecules and plot the results. Do it again for 6 initial R_0 molecules. Find Kolmogorov's equations for 5 initial R_0 molecules and plot the mean and standard deviation windows the expected number of R_1 molecules. Use $k = 1$ and $\varepsilon = 0.1$.

8.14) Propose an idea for how the chemical implementation of register machines could be implemented inside cells. Where else would errors come from in your implementation? Give an example of a computation that would be useful in a synthetic biology setting.

8.15) (Extra Credit) Find a register machine that computes $n!$ for $n \geq 0$. Simulate the chemical implementation of it using Gillespie's algorithm.

Chapter 9

Composition and Modularity

9.1. Signals

9.2. Modules

9.3. Composition

9.4. Retroactivity

Chapter 10

Robustness and Sensitivity

Chapter 11

Parameter Estimation and System Identification

Appendix A

Mathematica

Many tools are available for ...

Bibliography

1. *The Coli genetic stock center*, http://cgsc.biology.yale.edu/.
2. *Principles of biochemistry*, 4 ed., W.H. Freeman and Company, 2005.
3. M. Acar, J.T. Mettetal, and A. van Oudenaarden, *Stochastic switching as a survival strategy in fluctuating environments*, Nature Genetics **40** (2008), 471–475.
4. L. Adleman, *Molecular computation of solutions to combinatorial problems*, Science **266** (1194), 1021–1024, http://www.sciencemag.org/.
5. B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts, and P. Walter, *The molecular biology of the cell*, 5 ed., Garland Science, 2002.
6. D.I. Andersson, E.S. Slechta, and J.R. Roth, *Evidence that gene amplification underlies adaptive mutability of the bacterial lac operon*, Science **282** (1998), no. 5391, 1133–1135.
7. A.P. Arkin and D.A. Fletcher, *Fast, cheap and somewhat in control*, Genome Biology **7** (2006), no. 114.
8. M.R. Atkinson, M.A. Savageau, J.T. Myers, and A.J. Ninfa, *Development of genetic circuitry exhibiting toggle switch or oscillatory behavior in* escherichia coli, Cell **113** (2003), no. 5, 597–607.
9. N. Barkai and S. Leibler, *Biological rhythms: Circadian clocks limited by noise*, Nature **403** (2000), 267–268.
10. J.M. Barnes and D.F. Bruhn, *Evaluation of a GFP reporter gene construct for environmental arsenic detection*, Talanta **58** (2002), no. 1.
11. S. Basu, Y. Gerchman, C.H. Collins, F.H. Arnold, and R. Weiss, *A synthetic multicellular system for programmed pattern formation*, Nature **434** (2005), 1130–1134.
12. B. Canton, A. Labno, and D. Endy, *Refinement and standardization of synthetic biological parts and devices*, Nature Biotechnology **26** (2008), 787–793.
13. M. Chalfie, Y. Tu, G. Euskirchen, W.W. Ward, and D.C. Prasher, *Green fluorescent protein as a marker for gene expression*, Science **263** (1994), 802–805.
14. M.T. Chen and R. Weiss, *Artificial cell-cell communication in yeast* saccharomyces cerevisiae *using signaling elements from arabidopsis thaliana*, Nature Biotechnology **23** (2005), 1551–1555.
15. W.W. Chen1, P.J. Jasper B. Schoeber and, M. Niepel, U.B. Nielsen, D.A. Lauffenburger, and P.K. Sorger, *Inputoutput behavior of ErbB signaling pathways as revealed by a mass action model trained against dynamic data*, Molecular Systems Biology **5** (2009), no. 239.
16. J.L. Cherry and F.R. Adler, *How to make a biological switch*, Journal of Theoretical Biology (2000), no. 203, 117–133.
17. E. Dekel and U. Alon, *Optimality and evolutionary tuning of the expression level of a protein*, Nature **436** (2005), no. 7050, 588–922.
18. H. El-Samad, H. Kurata, J.C. Doyle, C.A. Gross, and M. Khammash, *Surviving heat shock: Control strategies for robustness and performance*, Proceedings of the National Academy of Science **102** (2005), no. 8.
19. M.B. Elowitz and S. Leibler, *A synthetic oscillatory network of transcriptional regulators*, Nature **403** (2000), 335–338.

20. D.G. Gibson et al., *Complete chemical synthesis, assembly, and cloning of a mycoplasma genitalium genome*, Science **319** (2008), no. 5867, 1215–1220.
21. Martin Feinberg, *Lectures on chemical reaction networks*, University of Wisconsin-Madison, 1980, http://www.che.eng.ohio-state.edu/~feinberg/LecturesOnReactionNetworks/.
22. B.H. Frank, J.M. Pettee, R.E. Zimmermann, and P.J. Burck, *The production of human proinsulin and its transformation to human insulin and c-peptide*, Peptides: Synthesis-structurefunction. Proceedings of the Seventh American Peptide Symposium (D.H. Rich and E. Gross, eds.), 1981, pp. 729–738.
23. D.Y. Zhang G. Seelig, D. Soloveichik and E. Winfree, *Enzyme-free nucleic acid logic circuits*, Science **314** (2006), 1585–1588.
24. E. Galun, A. Breiman, and J. Barton, *Transgenic plants*, Imperial College Press, 1997.
25. T.S. Gardner, C.R. Cantor, and J.J. Collins, *Construction of a genetic toggle switch in Escherichia coli*, Nature **403** (2000), 339–342.
26. D. T. Gillespie, *Exact stochastic simulation of coupled chemical reactions*, Journal of Physical Chemistry **81** (1977), 2340–2361.
27. M. W. Hirsch and S. Smale, *Differential equations, dynamical systems, and linear algebra*, Academic Press, 1974.
28. F.J. Isaacs, D.J. Dwyer, and J.J. Collins, *RNA synthetic biology*, Nature Biotechnology **24** (2005), 545–554.
29. L. Jiang, E.A. Althoff, F.R. Clemente, L. Doyle, D. Röthlisberger, A. Zanghellini, J.L. Gallaher, J.L. Betker, F. Tanaka, C.F. Barbas, D. Hilvert, K.N. Houk, B.L. Stoddard, and D. Baker, *De novo computational design of retro-aldol enzymes*, Science **319** (2008), no. 5868, 1387–1391.
30. D. Kentner and V. Sourjik, *Dynamic map of protein interactions in the Escherichia coli chemotaxis pathwa*, Molecular Systems Biology **5** (2009), no. 238.
31. J. Kim, K.S. White, and E. Winfree, *Construction of an in vitro bistable circuit from synthetic transcriptional switches*, Molecular Systems Biology **2** (2006), no. 68.
32. C. Lartigue, J. Glass, N. Alperovich, R. Pieper, P. Parmar, C.A. Hutchison, H.O. Smith, and J.C. Venter, *Genome transplantation in bacteria: Changing one species to another*, Science **317** (2007), no. 5838, 632–638.
33. J. Markussen, U. Damgaard, I. Diers, N. Fiil, M.T. Hansen, P. Lassen, F. Norris, K. Norris, O. Schou, L. Snel, L. Thim, and H.O Voigt, *Biosynthesis of human insulin in yeast via single chain precursors*, Diabetologia **29** (1986), 586A–569A.
34. P. Mckenna, *Amateur biologists play with DNA*, The New Scientist **201** (2008), no. 2688, 20–21.
35. J. T. Mettetal, D. Muzzey, C. Gomez-Uribe, and A. van Oudenaarden, *The frequency dependence of osmo-adaptation in Saccharomyces cerevisiae*, Science **319** (2008), no. 482, 482–484.
36. J.F. Monod, *Genetic regulatory mechanisms in the synthesis of proteins*, Journal of Molecular Biology **3** (1961), 318–356.
37. Brian Munsky and Mustafa Khammash, *The finite state projection algorithm for the solution of the chemical master equation*, Journal of Chemical Physics **124** (2006), 044104–1–044104–13.
38. M. Pigliucci, *Is evolvability evolvable?*, Nature Review Genetics **9** (2008), no. 1, 75–82.
39. D.K. Ro, E.M. Paradise, M. Ouellet, K.J. Fisher, K.L. Newman, J.M. Ndungu, K.A. Ho, R.A. Eachus, T.S. Ham, J. Kirby, M.C.Y. Chang, S.T. Withers, Y. Shiba, R. Sarpong, and J.D. Keasling, *Production of the antimalarial drug precursor artemisinic acid in engineered yeast*, Nature **440** (2007), 940–943.
40. H. El Samad, M. Khammash, L. Petzold, and D. Gillespie, *Stochastic modelling of gene regulatory networks*, International Journal of Robust and Nonlinear Control **15** (2005), no. 15, 691–712.
41. D. Schüler, *Formation of magnetosomes in magnetotactic bacteria*, Journal Molecular Microbiolgy Biotechnology **1** (1999), no. 1, 79–86.
42. R. Shetty, D. Endy, and T.F. Knight, *Engineering biobrick vectors from biobrick parts*, Journal of Biological Engineering **2** (2008), no. 5, doi: 10.1186/1754-1611-2-5.
43. H.L. Smith and P. Waltman, *The theory of the chemostat: Dynamics of microbial competition*, Cambridge University Press, 2008.
44. C.D. Smolke, *The metabolic pathway engineering handbook*, CRC Press, 2009.
45. L. Snyder and W. Champness, *Molecular genetics of bacteria*, 3 ed., ASM Press, 2007.

46. J. Stricker, S. Cookson, M.R. Bennett, W.H. Mather, L.S. Tsimring, and J. Hasty, *A fast, robust and tunable synthetic gene oscillator*, Nature **465** (2008), no. 27, 516–519.
47. P.S. Swain, M.B. Elowitz, and E.D. Siggia, *Intrinsic and extrinsic contributions to stochasticity in gene expression*, Proceedings of the National Academy of Science **99** (2002), no. 20, 12795–800.
48. K.N. Timmis and D.H. Pieper, *Bacteria designed for bioremediation*, Trends in Biotechnology **17** (1999), no. 1, 200–204.
49. G. Vogel, *Breakthrough of the year: Reprogramming cells*, Science **322** (2008), no. 5909, 1766–1767.
50. M.E. Weinblatt, *A Trial of Etanercept, a Recombinant Tumor Necrosis Factor Receptor:Fc Fusion Protein, in Patients with Rheumatoid Arthritis Receiving Methotrexate*, New England Journal of Medicine **340** (1999), no. 4, 253–259.
51. J.N. Weiss, *The hill equation revisited: uses and misuses*, The FASEB Journal **11** (1997), no. 11, 835–841.
52. E. Winfree, *Algorithmic self-assembly of DNA: Theoretical motivations and 2D assembly experiments*, Journal of Biomolecular Structure and Dynamics **11** (2000), no. 2, 263–270.
53. Y., R. Weiss, and F.H. Arnold, *Directed evolution of a genetic circuit*, Proceedings of the National Academy of Science **99** (2002), no. 26, 16587–16591.
54. David Y Zhang, Andrew J Turberfield, and Bernard Yurke abd Erik Winfree, *Engineering entropy-driven reactions and networks catalyzed by DNA*, Science (2007), no. 318, 1121–1125.
55. K. Zhou and J.C. Doyle, *Essentials of robust control*, Prentice Hall, 1996.