The following content is provided by MIT OpenCourseWare under a Creative Commons license. Additional information about our license and MIT OpenCourseWare in general is available at ocu.mit.edu.

GEORGE CHURCH: OK. We're getting ready. And let's go. OK. So welcome to the first class of BIOE101 or biophysics 101 or HST.508 or Genetics 224. You can see that this is a truly interdisciplinary class, and I hope that's going to be one of its strengths. It has been in the past.

I hope that you get to know some of your colleagues in this room or maybe in the other half of this class, which is taught on the medical school campus, because they could be some of your greatest assets in years to come. This course-- basically, if you cannot, for some reason, on a particular day, make this meeting at 5:30, you have the option of going at noon over in the Canon room, or you have the option of tuning in to the video, which is intended mainly for the distance education students but also serves as a backup in case one of you gets sick or is out of town or something like that. So that's what's going on right there, is that we put this on our internet site. It's synchronized to the PowerPoint slides. You should have PowerPoint handouts right now.

This course, as stated on the website and on this first slide, is based, essentially, entirely on six problem sets and a course project. In the past, the students and I have had a great time in the course projects. We see, really, the cutting edge of computational biology. And the six problem sets are really intended to get you up to speed so that you are as close as possible to a publication-grade computational biology project, which, generally speaking, is collaborative, involving two or more people. But you can do it solo as well. There'll be more details on the website and also in your sections.

You're also free to work with other people to use any resources you want for the problem sets. I would just ask that you-- however you answer the question, whether it comes just purely from your head or from collaboration or from some website, that you just say for each problem, just briefly, where you got the answer. This is just a good academic discipline for acknowledging your sources.

This course is intended to be an introductory course, introductory to almost every subject that it combines. It has minor prerequisites. And those who feel that they are a little weak in either molecular biology, statistics, or computing, there will be classes in addition to the regular sections which will address these intensively, the especially at the beginning of the course.

Then the sections themselves will be crafted so that they are directed at people who feel strong in biology and slightly weaker in computing. So the focus will be on getting the computing up from-- more or less from scratch. And those that are stronger, say, in computing, then there'll be sections on advanced topics which will fit in with the course.

By the middle of the course, almost all the sections will be very close to identical. But there still will be that slight difference. We will try to provide mechanisms by which you can interact with colleagues that have different strengths but similar interests so that you can get together on problem sets and the projects. You shouldn't feel it was essential. It's an opportunity, not an obligation.

It's important you hand in your questionnaire immediately after class so that we can assign you the sections. The sections will-- the sections will be the way that you get information and the way that you interact with the teaching fellow who will be responsible for grading your problem sets. So that teaching fellow will interact with you on all six problem sets in your project, and so you should get to know that person very well.

And I'm very indebted to this crew, by the way, which is up here at the top. Suzanne [? Camilli ?] is the head teaching fellow this year. Many of them-- she was a teaching fellow last year. And many of these-- almost all of these teaching fellows have taken this course last year.

That's most of the bureaucratic aspects of it unless there are some questions. Please feel free to interrupt at any time. This can be interactive if you want it to be. I can set aside enough time for that. You're certainly also welcome to come to me before class, after class, and during the break, and we can even set up additional times. Questions? Yes.

AUDIENCE: The extra sections, those are more filling in the background.

GEORGE CHURCH: That's right. There will be this--

AUDIENCE: There seems to be only one, which is Thursday evening.

GEORGE CHURCH: So the question is about extra sessions. There will be a schedule of all the sections evolving on the website within the next day or two. Depending on what your questionnaires show up, there will probably be three or four extra sections within the first couple of weeks.

AUDIENCE: And how do we [INAUDIBLE]?

GEORGE CHURCH: Through your section head. If you put your email address, it will allow the section head teaching fellow to contact you, or you can contact any of them if no one contacts you. OK.

So the overview for the course-- this is constrained to fit into the calendar of both Harvard and MIT. It also tries to achieve the goal of keeping the Division of Continuing Education students so that they only have to come in for one day a week, which is a considerable plus section. Some of them choose to have section the same day. And some of them have considerable commutes. So that determines the timing of this.

The topics-- we'll have two introductory lectures. We'll actually cover pretty interesting topics for introductory thematically, one on computing followed by one on biology, although both of them and the whole course is on computational biology and systems biology. Then as another way of focusing, we'll have a series of six lectures, two on DNA, two on RNA, and two on proteins or proteomics, which reflects the central dogma of molecular biology where DNA encodes RNA which encodes proteins.

But much more so, it's telling us about the flow of information that we need to establish experimentally and in terms of modeling that allows us to do functional genomics, and hence, systems biology, and to mine the datasets which are on everyone's attention right now. Then finally, three topics which really focus on the overarching theme of the whole course, which is on networks and systems biology.

And these really cover the gamut of network analysis on cellular all the way through ecological scale modeling in order to see how we integrate various data types. Then comes the very interesting section of the course, three of these two-hour sessions where you get to present to me the topics that you're excited about, how you have taken the course material problem sets and so forth and incorporated into your view of what's exciting and then present to all of us that information as team or individual presentations.

OK. That's the overview for the whole course. Today's story-- I try to make this somewhat a narrative and with themes. So we're going to toggle back and forth throughout the lecture on living systems and computational systems-- similarities, differences, how we can use one to tell us something about the other.

In particular, we'll focus in on the aspect of living systems which is fairly unique, which is self-assembly and replication. These are put in the context of another theme for today, which is discrete versus continuous data, discrete versus continuous modeling of data. Since this is an introductory lecture and we want to get you warmed up, I'm going to try to illustrate as much as possible with minimal examples, small examples, so you can see it all in one page or all in one line, even, examples of minimal life-- things that illustrate the minimal aspects of life replication, mainly, and minimal programs, which allow us to analyze some key aspects of modeling living systems.

Some of the key aspects are catalysis and replication. And here we'll use differential equations. This will be quite, I think, an interesting approach to it, painless, and exciting the way it connects to biology. And then after talking about that replication in the context of differential equations, we'll introduce [? directed ?] graphs, which indicate how growth occurs in a pedigree, not just growth in exponential mode. And then finally, we'll connect this to the issues that surround single molecules, which are actually very significant in biological systems and involve a different type of analysis than the continuous functions that we will use in ordinary differential equations.

When we analyze errors, either in data or variance in biological populations, we use statistics, broadly speaking, that fall under bell curve statistics-- actually, more than that. And overall, the entire-- the theme that unites all of biology and most of the course, and this lecture in particular, is the idea that many of these functions in biological systems can be under selection, experimental selection in the laboratory in order to obtain an optimal growth rate or some other aspect of optionality. You generally will do well embedding the biological system either as optimal or can be made optimal for a particular task.

OK. Now, I'm just poking fun at the number for this course. It is also an intrinsically important symbol for the 0's and 1's that occur in all of our computers and under underlie the way we either deal with discrete data or make continuous data appear to be discrete. We'll start with the biological entities most similar to the 0's and 1's that make your computers hum. And these are the nucleotides of DNA and RNA, A, C, G, and T, or A, C, G, and U in the case of RNA.

And of course, all of this is symbols. The A stands for adenine. And adenine, of course, doesn't look like an A. It looks like some electron density. And we often represent it with chemical formula you'll see in subsequent slides. But here in slide 5, we're talking about, schematically, A, C, G, and T being-- A being represented by the two-digit binary number 00, and C 01 and 10 and 11. So you can see that two digits isn't quite enough to encode the four nucleotides which are strung together where you might have 3 billion such nucleotides making up one of your two human genomes.

Now, things get a little bit more complicated once this digital 3 billion nucleotides in your genome starts being turned into molecules that actually do the work of the cell, the work of your body, which is RNA and proteins. This is an example of one of the primary transcripts common to all living organisms that we know of. This is a transfer RNA.

I actually participated in solving this complicated structure in the '70s. And this is color coded, the DNA sequence that encodes this particular RNA. And when the RNA is made, it folds up into this three-dimensional structure which you see rotating here. This is actually a stereo image, which, if you cross your eyes in the right way, will appear to be even more three-dimensional than it is there. You will learn more about this later in the course if you don't know already.

But the point is that this goes from a 5-prime end at blue all the way out to the 3-prime end in red. And it has to fold in this manner. It reproducibly folds in this manner, and it has to stay in order to perform its function, which is actually translating from DNA-- sorry-- from RNA sequences into protein. But what we're illustrating here is the relationship between this discrete binary digital code and DNA and the more continuous code that you have, which is the x, y, and z-coordinates of the atoms-- the continuous nature of the probability distribution of electrons around each of those atoms and the continuous nature of its position in space and its various binding constants, affinities for other molecules in the cell.

Let me just take those two examples, the DNA sequence and the three-dimensional structure for the transfer RNA that it encodes, and expand upon them a little bit or give other examples. We have a sequence. In the previous slide, I showed a sequence of 76 nucleotides. On the left-hand side are examples of discrete concepts, concepts which are very naturally encoded in a discrete digital way, and then as close as we can get to the continuous version of that.

So a continuous version of a sequence might be a probability of a sequence. In other words, at the first position, if you look at a population of sequences-- the number of different tRNAs, the number of different people in this room-- it may not be that there's always an A at position 1. It could be that it's got a different probability in different people. So you represent that as a probability A, C, G, and T. It's a vector with four numbers in it. That's more continuous.

Similarly, we have digital analog devices in the instruments that collect the data that you'll be working with in this course. Integration can be represented, practically speaking, as a sum of small steps. When we're talking about, say, a neural network that's responsible for some of your thoughts or a regulatory network that causes homeostasis in your body, biologists casually often reflect on these as being on and off. It's a good approximation. It allows you to make very simple diagrams.

But you must remember that many of these are actually composed of gradients or graded responses. Not necessarily all are [? on/off. ?] Some of these gradients or graded responses have this sigmoid shape that I've drawn as an icon down the middle of this, separating the discrete from the continuous. And so for all intents and purposes, things tend to hang out either off here at the bottom or on at the top of the concentration limits or the signal limits if the signal is electrical or so forth.

Similarly, we'll have examples where some cells will be-- we'll have a field of cells which are either on or off, effectively very extreme ends of this. But because there's a mixture of them, if you were to mush them up and measure some property of them, it would appear to be somewhere intermediate. So this is an alarm that should be going off every time you see mixtures that they may be composed of-- you may need-- to model it, you may have to model it as a population, each individual behaving as more extreme than the average with very few in the middle. Similarly with mutations, you might say that certain mutations are essential for life. Others are neutral. They have no effect. Others, more gray zone, are conditional.

OK. Just a point of orientation for how we can describe not only the bits, the discrete or digital components of this course. But also, many of these prefixes are useful for describing the continuous as well. But you can see this-- many of you will be familiar with these. All of you, I'm sure, have access to computers, and so you've used terms like kilobyte and megabyte and gigabyte.

Technically, 2 to the 10th power is 1,024, not 1,000, and so it shouldn't be referred to as kilo. But for most intents and purposes, these are so close to 1,000 or a million or a billion that they're used interchangeably. But this is the official standard here.

And certainly for the continuous numbers that we'll be talking about, the order we'll have is 10 to the plus 3 will [? be kilo. ?] 10 to the minus 3 will be milliliters. Mega, micro, giga, nano, and so on, all the way up. [? For the ?] numbers, atom mole is getting-- zeptomole is getting close to the limit where you're getting close to single molecules. Petabyte is getting close to the limit of where your computers can go right now.

Why is it important to have defined quantitative measures? Why can't we just casually say, oh, yeah, it's a long time rather than seconds, or it's really long distance to Harvard Square in meters? Why do we have to use meters, kilograms, moles, degrees, Kelvin, candelas, amperes? These are the seven basic international system units from which most of the other units that you use in science are derived. In a certain sense, there is some inter-convertibility even within these.

We'll be talking about precision a little bit at the end of today's class and throughout the course. In fact, a theme today is that even things that are represented digitally are not-- in order to represent things digitally, there are approximations that are made. And you should always question what those assumptions are when you make an approximation. The precision for some of these units of measure-- for example, the time scale measured in seconds-- can be as precise as 14 significant decimal digits.

Now, you may wonder from time to time in the course why biology doesn't have 14 significant figures. But I will leave that up as an exercise for one of your projects, maybe, to figure out how to get that. But for all practical purposes, most of biology is three or four significant figures. There are quantum limits to time and length, which are so short that we don't need to concern ourselves. But the quantum unit for the mole is actually of great importance to this course.

A mole is 6 times 10 to the 23rd entities. These entities can be photons or molecules, or we even have over 6 times 10 to the 23rd of the bacteria in an ocean. So what's important here is that the quantum of this is the molecule, and many of the things we'll be dealing with are single molecules. And that'll be the last topic of today's lecture, will be how one deals with single molecules as opposed to large buckets of molecules.

Now, we have all those great quantitative definitions for all the standard units used in physics and chemistry, and even in biology, but what about the definition of biology itself? Most biology books start with this. There are entire books dedicated to this question of what is life. I think that rather than, in this session, ask what is alive or not-- rather than make it a dichotomy, I'd prefer to say how alive is something. What is the probability that a given entity will replicate?

We have, here on slide 10, the probability of replication, and not just how likely is it for the whole thing to replicate or some part of it to replicate, but is it doing so using simple parts, simple environmental components, and creating great complexity from that. How faithful is the replication?

Now, this probability of replication from simplicity to complexity can be defined for a specific environment. Sometimes, the environment required for replication is, of necessity, very specific. Certain organisms do not do well except in their native environments. Zoos discover this, and we keep killing off species for a variety of reasons, one of them being this specificity.

So when we talk about-- so another aspect of life-- which each of these, in principle, can be quantitated-- is how robust is it. How many environments can it handle? Inevitably, there are environments that cannot be handled. But the more robust and adaptable it is, in a certain sense, the more alive it is or the more alive it will be, its descendants will be, millions of years from now, probably.

Now, some very challenging examples I list here. I'm not going to walk through all of them. But things that have challenged people's definitions of life before are mules-- that is to say, sterile hybrids which will not leave behind progeny but seem quite as alive as their parents were but will not replicate, generally.

So the probability of replication is low for the entire organism, but maybe for individual cells. Fires replicate quite well, but most people would like to exclude them from this definition. Or perhaps if we're in the mode of not excluding things from life, the probability of replication complexity versus simplicity might impinge upon fires.

Crystals-- if you nucleate a supersaturated solution with a crystal, it will make, in a certain sense, copies of itself. How [? painful ?] is that? How simple is it? Flowers, viruses, predators-- these require very complex environments. They require environments often more complex than themselves in order to replicate. Does that make them less alive? I'll show an example of molecular ligation as the simplest case just to get us warmed up for this introductory class.

Since we're in the topic here, briefly, of general biology, not just the historical terrestrial biology from which all of us feel particular affinity-- if we had visitors from some other planet or if we started making our own self-assembling machines-- and to some extent, we already are making self-assembling machines in factories. They require a very complex environment which includes humans. But how do we define these things?

Now, in order to define-- to get at that complexity versus simplicity issue and faithfulness of replication, we need to define both replication and complexity. Replication is not perfectly faithful in many of the things that you would consider alive. So simple bacterium, even though it will make a copy that does many of the same things-- we know it when we see it-- it looks like it's the same thing. If you actually counted the molecules, no two bacteria would be alike. No two humans, certainly, are alike. But even supposedly genetically identical bacteria will have different numbers of proteins and small molecules.

Complexity has at least four definitions we will use in this course. There is computational complexity, which is of practical significance in computational biology, as the computer scientists in the room will know, in that this tells us the speed and memory tradeoffs we have in scaling up any problem. Does a problem scale up as a simple linear function of the number of inputs that you give to the computer?

Does it scale up as a simple polynomial, or is it worse than that? Is it exponential in behavior? Is it something that you can prove you got the right answer in polynomial time, and so on? We'll get to this later on, but I just want to introduce it as something where the word "complexity" is used.

Number two sounds similar but is actually quite a bit different. It's algorithmic complexity or algorithmic randomness. And this is basically any string-- not just a computer program, but a computer program can be represented as a string-- can be reduced down where you get rid of obvious redundancy, and what's left is the randomness. And the number of bits it takes to encode that algorithm is a reflection of the complexity. That doesn't necessarily give you any predictions about how long it will run or how much memory it will require during computation.

Entropy and information, number 3, is related to item number 2 in that the more complex the string-- a string is just a series of symbols like this, and that was what we were talking about with the randomness-- the more complex the string, or an image can be turned into a string-- these are three images here-- then the more bits you need to encode it in order to, say, make a file. You might compress the data, but ultimately, after it's fully compressed, that's the amount of information you need.

Entropy is a chemical term. This information definition was championed by Shannon-- we'll come back to it-- and entropy by Boltzmann and others. And the fusion of these two into a unified theme is extremely important in both chemistry and information theory.

Nevertheless, none of these above reflect our intuitive feeling for when we look at these three panels. We have a highly ordered, which would be a very low entropy, array on the left-hand side. We have, on the far right-hand side, something highly disordered, essentially random. This may have been generated by a coin toss where, as you fill up the array, you just say black or white, toss a coin.

The one in the middle-- and so the low entropy on the left could be easily represented as 010101 in a simple array-- very little information, very little entropy, algorithmically, not very random. At the other end, though, it has high entropy, high information content. It takes a lot of bits to represent it even though you know you got it just by a coin toss, and it has high algorithmic randomness.

But if you allow a coin toss to be part of your description of the physical complexity of this pattern on the far right, say a gas or something like this generated by coin toss, then you can represent it as a particular kind of random description, which this is almost as-- this is about as easy to describe as a highly ordered system. So even though it's high entropy, it has low complexity. And complexity lies somewhere in between where you have lots of different kinds of symmetries, lots of different scales of structures, and it's very hard to represent it either as a random coin toss or as a highly ordered system.

How do we quantitate this? We're really trying to move from the vaguer definitions to something that really encompasses what we intuitively feel about complexity. Here's an example I happen to like. I would say that this is not something that's broadly adapted, but I want to expose you to it.

Here we have entropy or the randomness, the information-- Shannon information on the horizontal axis of slide 12 where the low entropy that's highly ordered is on the left-hand side. It's 0. And the highly disordered, random high entropy of 1 on this type of scale is on the right-hand side.

And complexity, as we said, we expect does not correlate perfectly with information or entropy in that this highly disordered structure on the right-hand side actually has very low complexity. And complexity is not even a single-valued function of entropy or information because you can have multiple different structures that have the same entropy but have different complexities. That's what we see here when you take a slice or a line up from the horizontal axis up through the complexity. You can have multiple different values in here.

OK. That was an example of a model. Yeah.

AUDIENCE: [INAUDIBLE] despite how complexity is measured or [INAUDIBLE] definition--

GEORGE CHURCH: You'll really have to refer to this article. It's a fairly complicated one. But the idea here was you take a complicated set of data generated by a logistic map that we'll come to in just a few slides, and you ask-- as you increase the complexity of the map in a way that appeals to the intuitive definition of complexity you had in the previous one, you calculate the number of symmetries, the physical symmetry that it would take to represent this, allowing coin tosses as part of the algorithm for simplifying it.

It's just barred from all the previous definitions. Crutchfield and coworkers have championed this. And I would say it's not widely accepted, but it's certainly not rejected either. And I find it appealing.

So why do we model? That was an example of modeling complexity. I pointed out earlier the models that we had for a three-dimensional structure. This course is mainly about measurements. So why model? I would argue that when we measure, we actually must model. And we need to do that not merely to understand the biological, chemical data that we are collecting.

And if I understand, we can have various tests for our understanding. Tests might include, but not be limited to, designing useful modifications of a system. The course is mainly about systems. And by using our models to design a new variation on the system and then obtaining additional data and modeling that new data, we get a better successful approximation to what the underlying data means. And we prove it, often with useful modifications that can have impact on society or impact on other research agenda.

Another reason why we model is in order to share data. Typically, when you look in a database-- those of you who have looked at biological or chemical databases, that is not data in the database, generally speaking. There are models in there. And the things that you download are models.

So for example, the sequence that I showed at the beginning-- that's 76 nucleotides of A's, C's, G's and P's-- that's a model that represents our interpretation of some kind of fluorescent patterns that we see when we run our instruments on amplified DNA taken from a variety of organisms. That's the model.

Similarly, the three-dimensional rotating transfer RNA that we saw was probably more obviously a model. There, we integrate the chemistry of molecular mechanics and the physics of the diffraction data where the atoms-- the electron density in a crystal lattice diffract and make a pattern. We integrate those models, and that's what shared in databases.

Not only does it allow us to share data, but the way we share it is by searching. Those of you who have searched either biological databases or the internet with Google or something like that know how powerful a search can be. Typically, it is easier to search with a model and more powerful and more accurate to search with a model than to search through raw data.

When we merge datasets, we will align them. We will find redundancies. We will integrate different concepts. That requires modeling. And finally, checking data-- one of the themes of this course will be to embrace your outliers. When you model your raw data and you find data that are very far away from expectation, this is a good thing. It allows you to find errors in your model or errors in your data or discoveries. So checking is another.

Finally, integrating-- as I said, this course is about measures and models. And it's not just a survey of all the things that you can do with computation in biology and computational biology, but most importantly, it's about integration. We need more than one data type to make progress in biology and medicine, agriculture. And integration is one of the biggest challenges that we have right now. It's relatively easy to collect a single homogeneous dataset, but then to connect that to the rest of the world is the big challenge that all of you will have.

A theme that will go throughout the course will be this business about errors, two types of errors, random and systematic. And every time that somebody hands you some data or you collect your own data or you're dealing with something from the literature, you should immediately assume there are both random and systematic errors. You should know which is which and how much there is. You should not accept anything as being true without qualification.

Random errors mean that if you repeat the experiment again and again, you will get slightly different variations on the error types. And to a certain extent, they will average out over enough determination. Systematic errors, on the other hand-- you have a high probability of getting almost exactly the same error again and again, or a very small subset of a certain class of errors, which means that just doing it over and over will not improve your statistics or improve your accuracy. You need to change paradigms altogether, collecting data by more than one method.

So those are the types of reasons that we model. And now we should go through the kinds of models that we'll do. This is a course that's basically on sequence, how sequence leads to three-dimensional or even four-dimensional structures with time, how three-dimensional structures lead to function, how function is embedded in complicated systems, so which models we will be searching, merging, and checking, as we said in the previous slide.

Which models will we sequence? We will be making the assumption in dynamic programming, which is just a fancy way of saying searching and aligning-- dynamic programming will make the assumption that sequences are related to one another. They make the further assumption they are related by ancestry. That is to say, you mutated them in the laboratory, or perhaps you were mutated before you came to the laboratory. That's dynamic programming that can align sequences or three-dimensional structures which are quite different from one another.

A few slides back when we talk about replication, if two things are replicated-- showing common ancestry, say-- the faithfulness of that replication is key to-- the dynamic programming is key to your accepting that something is actually replicated. If it changes into something completely new in the process of replication, then it's unlikely it will be able to maintain that.

Three-dimensional structure-- we will be talking about motifs, catalysis, complementary surfaces. I'll give you a beautiful example of complimentary surface in a few slides. And all of this is dealing with the continuous functions of energy and kinetics where different complementary surfaces will bind to one another with very specific rate constants covering many orders of magnitude.

These energetic and kinetic phenomenon is what underlies the functional genomics. When we functional genomic data, one of the ways we model it is we ask whether phenomena that we're studying inside our bodies or inside of microorganisms, so forth-- the protein levels, the RNA levels, and so forth, where they go up and down together means that they're a cluster. They have common properties. Does this common properties reflect yet another aspect of them, like their common functions or their common mechanisms, for being coregulated?

In systems biology, we have these qualitative diagrams that tell us about the all or none, or Boolean, which means logical 0's and 1's. Or we can treat them as continuous differential equations or stochastic where you have some of the power of the differential equations, but you deal with individual molecules or individual organisms and populations. Organisms can be stochastic. Molecules can be stochastic.

And another thing-- again, this theme of optimization-- we can ask whether a network or network component is optimal. Is it optimal due to the past history of the organism, or is it something we can set a goal of a biotech process to optimize it for a new function? Linear programming is the mathematical tool that one can use for studying this. It's common in economic algorithms.

So we were talking about modeling in this whole realm, going from minimal life sequences through the catalysis that involves interactions of three-dimensional structures, functional genomics, and the optimality that we get that's required, as we will see momentarily, for getting single molecules to work.

What's our parts list? We're going to try to start simple, but put the simplicity in the context of the big picture. The big picture for the atoms is the periodic table of the more than 100 elements. We have a very short list here-- sodium, potassium, ion, chlorine, calcium, magnesium, molybdenum, manganese, sulfur, selenium, copper, nickel, cobalt, and silicon, which are useful in many species.

If you have to pick something that's related to us as a minimal of biochemical system that shows some of the replicative properties and evolutionary properties of life, you would say RNA-based life. One of the breakthroughs in experimental science over the last couple of decades is recognizing that RNA, like proteins, has catalytic capabilities. It has the potential to have been one of the early replicating units.

Just for the sake of describing a simple system, let's look at something made up of five elements. It's not necessarily the simplest, but it's just something to think about. These five elements can be in the presence of an environment which can be composed of the same five elements. The environment would be very simple, that complexity versus simplicity.

It would be water, ammonium, positively charged ions, nucleotide triphosphate, negatively charged ions, which are precursors to making polymers, and then possibly lipids, the fatty substances that bound membranes. An example of catalyzed RNA polymerization is in this article, and then I'll deal with another one that, rather than using nucleotide triphosphates, uses slightly larger RNA precursors.

Now let's start doing the toggling back and forth between living systems and computational systems so that, whether you're from a computational background or biological or both, you'll see some interesting relationships. Here we talked about a minimal biological system with five elements. Here we're going to talk about some minimal problems with a very small number of elements. Basically, they're limited to a single line of code each.

And they do something which is related to the topic here. Replication is an exponential process, exponentially growing. If something is autocatalytic is another way of describing it. So we've used that as the theme for our minimal programs. What is this exponential function? We give it a very specific argument. The argument is 1. So this is e to the 1 power, e being this number here that's represented the number of times, about 2.7.

The four languages that we're demonstrating here is Perl, Excel, Fortran 77, and Mathematica. In this course, we'll mainly use Perl and Mathematica for reasons that will be evident in the next couple of slides. But I just want to show you these different ones.

And in the theme of accuracy of replication, how faithfully is a particular string handled, either the string of nucleotides in a simple life form or the string of digits in a number-- and you can see that the internal representation of the math that goes on inside the computer has to be done digitally even though these are transcendental numbers which have an arbitrary number of digits. And you can see that some programs are typically limited.

And you can even guess the number of bits representing it internally. Here down at the bottom of slide 17, you can see some of the nitty-gritty detail of how these things are represented as 0's and 1's representing the [INAUDIBLE], the exponent, and so on. But you can see some of these programs, when they print, when they actually print, the program is not aware of or the programmer was not aware of exactly what the internal representation was. And so all these trailing 0's are incorrect.

The winner in this, of course, is Mathematica, here to ask for the e to the 1 power. You say, let's give it-- this n bracket says let's start giving arbitrary precision where the numeric, the n numeric, is 100 digits. And this seems like a stunt in this particular case. But actually, when you start doing a series of calculations where errors can accumulate, the ability to go into arbitrary precision will allow you to prevent a catastrophic imprecision.

And what happens is, in a mathematical calculation, if it needs a little more precision than you initially anticipated, it will go out and get it on its own, which is quite remarkable. Try to do that in any of these other languages. They will typically give you what's called an underflow or an overflow error and either stop or just make mistakes.

OK. That's my first advertisement for Mathematica. You can see all of these are very simple programs that belie and hide underneath the very complicated electronics and algorithms that are built in to accomplish this. I'll give you a feeling for what that is in just a couple of slides. So back to self-replication. We're toggling back and forth between simple biological and computational systems.

We have-- here we've represented tri and hexanucleotides, three nucleotides and six. Again, remember, these letters are crude-- or I should say simple representations of electron density involving dozens of atoms in the shape of a cytosine, or a C, or guanine, or a G. It's important that you know that there are two strands in DNA, typically, or even many RNAs-- in this particular example, whether this is RNA or DNA.

And in order to indicate the orientation of the strand, it has a directionality. We indicate 5-prime, which is the name of an atom in the ribose, but the details are not important from a computational standpoint. It's just a way of indicating this is one end of the RNA. And a CCG is not the same as a GCC. The 5-prime indicates the 5-prime end of that.

We take two of these, which are basically identical, and they will ligate together spontaneously if you set up the chemistry right to make these two trinucleotides into a hexanucleotide, CCG CCG. In the presence of a complement here, which is CGG CGG, a different sequence-- I've tried to emphasize that by making it capital and green-- that would bind to-- these two trinucleotides would bind to this, aligned by Watson-Crick base pairs.

The rules for Watson-Crick base pairs, many of you may know. A pairs with T, and C pairs with G. And that will catalyze this process. It will speed up the kinetics in which these trinucleotides [? turn ?] [? into ?] hexanucleotides. Now, here it gets interesting. This hexanucleotide now drops down here and speeds up the process of this different trinucleotide [INAUDIBLE] and forming the original one that catalyzes the first reaction.

So this catalyzes, speeds up the first reaction, which produces a product that speeds up the second, which produces the first catalyst, and so on. This is called a hypercycle, championed by Eigen and Schuster. And this is probably one of the simplest examples of it. And I think it gives you a feeling for how a variety of interdependent biochemical processes can result in autocatalysis where you get these exponential cycles which we recognize as something very similar to replication.

This would have a very-- this could have a very high probability of replication. It could be very faithful, but its complexity is low. The input complexity is low, and the output complexity is low. That's how we would qualify it.

Toggling back to computing and simple examples, I've given some simple examples already of Perl in Mathematica. But in order to give a few more here on slide 19, why did we choose these two for this course? It's been my experience, both in our laboratory and research environment, and also in this class, that these are two of the easiest languages to learn.

That doesn't mean that they're absolutely easy, but the learning curve is very simple. It's very fast. And you can, by example, change programs that are working into things that do what you want very quickly.

It's a high-level language in a sense-- the higher the level of the language, the closer you are to English conversation. The lower the level of language you are, the closer you are to bits, 0's and 1's, or the actual electronics inside of computers. So in the hierarchy, Perl and Mathematica are very high up there. Yeah.

AUDIENCE: Where do you get the password to download Mathematica?

GEORGE CHURCH: You will get that from your section teaching fellow. Yeah. So the question is where do you get the password for Mathematica for this course, and the teaching fellows are responsible for that. And that's been tested. I think that works.

So Perl is freely-- it's open source. It's interesting in that regard. I think many of you will find the concept of open source where anyone can-- when software breaks or needs to be expanded, or it needs to be understood, it's there for your inspection rather than hidden behind some corporate doors. Very interesting aspect of Perl.

Mathematica is not open source as far as I know. It has some interesting redeeming features, though. It's very strong on math. No surprise given its name. In particular, it's both symbolic and numeric and in graphics. And we'll have some examples of that in just a moment. It can do things that are hard to do in Perl and other languages.

Perl is also very strong for web applications. Many of the most amazing things which are done on the web have Perl behind the scenes. Now, when we-- a further comparison of-- this is the dark side of computation and biology, showing another analogy, which is parasites. We have parasitic computer and viral-- biological viruses.

This one-- in past years, I had this little bit of a computer virus-- this is not the whole thing. This is just a little piece of it. It's fairly short code. And this was a very nasty one at the time, quite a while ago. And I had it in the PowerPoint presentation. When people would download it, they would then send me an email. I'm sorry to inform you you have a virus in your PowerPoint.

And so this year, I've upgraded the PowerPoint. So this is actually an image rather than the actual text. And so your viral detectors will not detect this unless they're a lot smarter than I think they are. This, on the other hand, is the costs of these viruses. This is not a laughing matter.

This is 4 times the cost of the entire Human Genome Project per year. The Genome Project took us about 20 years to get going. Every year, we spend four times that amount on computer viruses, which are, as far as I can tell, completely frivolous. Even more serious is this. Now, this is real text. This is not an image. It looked like an image to you.

It's very serious. And I sincerely hope that some of the people in this class make a contribution to the intellectual avenues which will ultimately lead to the defeat of the AIDS virus and various other viruses and bacteria like them that cause so much suffering in the world. 20 million have died. This is worse than the Black Plague and the 1918 influenza epidemic.

And the analogy here between this part of the virus-- this is a little piece of the viral code in symbols of the 20 amino acids single-letter code. This is a little piece of the computer virus. And I've highlighted here the command copy in the computer virus. This is just part of what it does to get this particular VBS script, the thing you're seeing here, into some other part of your directory.

And this, essentially, is part of the copy command in the AIDS virus. This is the polymerase. The protein is responsible for making copies of the code for the virus. And highlighted in red here are some of the mutations that make this particular AIDS virus resistant to the drugs which have been instrumental, in some cases, in taming AIDS temporarily, and at great cost. Clearly, vaccines, and hopefully other public health measures, will be the answer.

Here are some other conceptual connections. These are not meant to be dogma or to limit you in any way but hopefully to expand the way you think about these things. Obviously, as with other analogies, they will break down. But let's just follow for a moment.

What we're calling the instruction set, the concept of an instruction set, in computers is a program, in organisms is a genome. We've already said bits, so 0's and 1's, A, C, G and T. The stable memory, the thing that you can depend on but is a little bit slow for access, are disks and tapes in computers, and they're DNA, or in some cases RNA genomes.

That then-- you take it out of the slow memory, stable memory, and move it into something that's more active, more volatile, which is random-access memory in computers and the RNA in organisms. The environment for computers tends to be complex-- it's the internet sockets and people banging on the keyboard-- while organisms can be very simple. I gave some examples earlier where it's complex, but it can be as simple as water and salts, in which they can replicate very complicated structures.

The input and output can be-- the input can be analog and converted to digital in this process, digital analog for output-- the analog, say, of your screens. And the input-output is governed by these proteins at the end of the central dogma in organisms.

When we make these complicated systems from simple things, they go from so-called monomers, which really combine into polymers, which need not be linear polymers, although they often are in biological systems-- basically, you go from minerals to chips in computers. These are replicated in factories. The factories or cells can be as small as femtolitre.

Remember, this prefix, this femto, is 10 to the -15. That's about 1 cubic micron. Very small factories, very amazing productivity and complexity. Again, input-output as above. And communication is extremely fast in computers, slower but very rich in organisms.

After a very short break-- we can stretch-- we'll come back and talk in more detail about how computers actually are made and how biological systems are made. Thank you.