PROFESSOR: Welcome to Foundations of Computational and Systems Biology. This course has many numbers. We'll explain all the differences and similarities. But briefly, there are three undergrad course numbers, which are all similar in content, 7.36, 20.390, 6.802. And then, there are four graduate versions.

The 7.91, 20.490, and HST versions are all very similar, basically identical. But the 6.874 has some additional AI content that we'll discuss in a moment. So make sure that you are registered for the appropriate version of this course.

And please interrupt with questions at any time. The main goal today is to give an overview of the course, both the content as well as the mechanics of how the course will be taught. And we want to make sure that everything is clear.

So this course is taught by myself and Chris Burge from Biology, Professor Fraenkel from BE, and Professor Gifford from EECS. We have three TAs, Peter Freese and Collette Picard, from Computational and Systems Biology, and Tahin, from EECS. All the TAs have expertise in computational biology as well as other quantitative areas like math, statistics, computer science.

So in addition to the lectures by the regular instructors, we will also have guest lectures by George Church, from Harvard, toward the end of the semester. Doug Lauffenburger will give a lecture in the regulatory network section of the course. And we'll have a guest lecture from Ron Weiss on synthetic biology. And just a note that today's lecture and all the lectures this semester are being recorded by AMPS, by MIT's OpenCourseWare. So the videos, after a little bit of editing, will eventually end up on OpenCourseWare.

What are these courses? So these course numbers are the graduate level versions,

which are survey courses in computational biology. Our target audience is graduate students who have a solid background and comfort-- or, a solid background in biology, and also, a comfort with quantitative approaches.

We don't assume that you've programmed before, but there will be some programming content on the homeworks. And you will therefore need to learn some Python programming. And the TAs will help with that component of the course. We also have some online tutorials on Python programming that are available.

The undergrad course numbers-- this is an upper level undergraduate survey course in computational biology. And our target audience are upper level undergraduates with solid biology background and comfort with quantitative approaches. So there's one key difference between the graduate and undergraduate versions, which I'll come to in a moment.

So the goal of this course is to develop understanding of foundational methods in computational biology that will enable you to contextualize and understand a good portion of research literature in a growing field. So if you pick up *Science,* or *Nature,* or *PLOS Computational Biology* and you want to read those papers and understand them, after this course, you will have a better chance. We're not guaranteeing you'll be able to understand all of them. But you'll be able to recognize, perhaps, the category of paper, the class, and perhaps, some of the algorithms that are involved.

And for the graduate version, another goal is to help you gain exposure to research in this field. So it's actually possible to do a smaller scale computational biology research project on your own, on your laptop-- perhaps, on ATHENA-- with relatively limited computational resources and potentially even discover something new. And so we want to give you that experience. And that's through the project component that we'll say more about in a moment.

So just to make sure that everyone's in the right class-- this is not a systems biology class. There are some more focused systems biology classes offered on campus. But we will cover some topics that are important for analyzing complex systems. This is also certainly not a synthetic biology course. Some of the systems methods

are also used in synthetic biology. And there will be this guest lecturer I mentioned, Ron Weiss, which will cover synthetic biology.

It's also not an algorithms class. We don't assume that you have experience in designing or analyzing algorithms. We'll discuss various bioinformatics algorithms. And you'll have the opportunity to implement at least one bioinformatics algorithm on your homework. But algorithms and not really the center of the course.

And there's one exception to this, which is, those of you who are taking 6.874 will go to a special recitation that will cover more advanced algorithm content. And there will be special homework problems for you as well. So that course really does have more algorithm content.

So the plan for today is that I will just do a brief, anecdotal history of computational and systems biology. This is to set the stage and the context for the class. And then we'll spend a significant amount of time reviewing the course mechanics, organization, and content. Because as you'll see, it's a little bit complicated. But it'll make sense once we go over it, hopefully.

So this is my take on computational and systems biology. Again, it's not a scholarly overview. It doesn't hit everything important that happened. It just gives you a flavor of what was happening in computational biology decade by decade.

So first of all, where does this field fall in the academic scheme of things? So I consider computational biology to be actually part of biology. So in the way that genetics or biochemistry are disciplines that have strategies for understanding biological questions, so does computational biology. You can use it to understand a variety of computational questions in gene regulation, many other areas.

There is also, some people make a distinction that bioinformatics is more about building tools whereas computational biology is more about using tools, for example. Although many people don't-- it's very blurry-- and don't try to-- people use them in various ways. And then, you could think of bioinformatics as being embedded in the larger field of informatics, where you include tools for

management and analysis of data in general.

And it's certainly true that many of the core concepts and algorithms in bioinformatics come from the field, come from computer science, come from other branches of engineering, from statistics, mathematics, and so forth. So it's really a cross disciplinary field. And then, synthetic biology cuts across in the sense that it's really an engineering discipline. Because you're designing and engineering synthetic molecular cellular systems. But you can also use synthetic biology to help understand natural biological systems, of course.

All right, so what was happening, decade by decade? So in the '70s, there were not genome sequences available or large sequence databases of any sort. Except there were starting to be some protein sequences. And early computational biologists focused on comparing proteins, understanding their function, structure, and evolution. And so in order to compare proteins, you need a protein-- an amino acid substitution matrix-- a matrix that describes how often one amino acid is substituted for another.

And Margaret Dayhoff was a pioneer in developing these sorts of matrices. And some of the matrices she developed, the PAM series, are still used today. And we'll discuss those matrices early next week.

So in terms of asking evolutionary questions, two big thinkers were Russ Doolittle and Carl Woese, analyzing both ribosomal RNA sequences to study evolution. And Carl Woese realized, looking at these RNA alignments, that actually, the prokaryotes, which had been-- there was this big split between prokaryotes and eukaryotes was sort of a false split-- that actually, there was a subgroup of single-celled anuclear organisms that were closer to the eukaryotes-- and named them the Archaea. So a whole kingdom of life was recognized, really, by sequence analysis.

And Russ Doolittle also did a lot of analysis approaching sequences and came up with this molecular clock idea, or contributed to that idea, to actually build-- instead of systematics being based on phenotypic characteristics, do it on a molecular level.

So in the '80s, the databases started to expand. Sequence alignment and search became more important. And various people developed fast algorithms to compare protein and DNA sequences and align them. So the FASTA program was widely used. BLAST-- several of the authors of BLAST are shown here-- David Lipman, Pearson, Webb Miller, Stephen Altschul.

The statistics for knowing when a BLAST search result is significant were developed by Karlin and Altschul. And there was also progress in gapped alignment, in particularly, Smith-Waterman, shown above. Also progress in RNA secondary structure prediction from Nusinov and Zuker. We'll talk about all of these algorithms during the course.

And there was also development of literature databases. I always liked this picture. Many of you probably used PubMed. But Al Gore was well coached here, by these experts, in how to use it.

And then, in the '90s, computational biology really started to expand. It was driven partly by the development of a microarrays, the first genome sequences, and questions like how to identify domains in a protein, how to identify genes in the genome. It was recognized that this family of models from electrical engineering, the hidden Markov model, were quite useful for these sort of sequence labeling problems. That was really pioneered by Anders Krogh here, and David Haussler. And a variety of algorithms were developed that performed these useful tasks.

There was also important progress in the earliest comparative genomic approaches, since you have-- the first genomes were sequenced in the mid '90s, of free living organisms. And so you could then start to compare these genomes and learn a lot. We'll talk a little bit about comparative genomics.

And there was important progress on predicting protein structure from primary sequence. Particularly, David Baker made notable progress on this Rosetta algorithm. So it's a biophysics field, but it's very much part of computational biology as well.

So in the 2000s, definitely, genome sequencing became very fashionable, as you can see here. And the genomes of now larger organisms, including human, it became possible to sequence them. And then, this introduced a huge host of computational challenges in assembling the genomes, annotating the genomes, and so forth. And we'll hear from Professor Gifford about some genome assembly topics. And annotation will come up throughout.

Actually, let's just mention, this is Jim Kent, who's the guru who did the first human genome assembly-- at least, that was widely used-- and also was involved in UCSC. And here, Ewan Birney has started Ensembl and continues to run it today. You know who these other people are, probably.

OK. All right. So in another phase of the last decade, I would say that much of biological research became more high throughput than it was before. So molecular biology had traditionally, in the '80s and '90s, mostly focused on analysis of individual gene or protein products. But now it became possible, and in widespread use, that you could measure the expression of all the genes, in theory-- using microarrays, for example-- and you could start to profile all of the transcripts in the cell, all of the proteins in the cell, and so forth.

And then, a variety of groups started to use some of these high throughput data to study various challenges in gene expression to understand how transcription works, how splicing works, how microRNAs work, translation, epigenetics, and so forth. And you'll hear updates on some of that work in this course. Bioimage informatics, particularly for developmental biology, became popular. It continues to be a new emerging area.

Systems biology was also really born around 2000, roughly. A very prominent example would be the development of the first gene regulatory network models that describe sea urchin development, here, my Eric Davidson, as well as a whole variety of models of other gene networks in the cell that control things like cell proliferation, apoptosis, et cetera.

At the same time, a new field of synthetic biology was born with the development of

some of the first completely artificial gene networks that would then program cells to perform desired behavior. So an example would be this so-called repressilator, where you have a network of three transcription factors. Each represses the other. And then, one of them represses GFP.

And you put these into bacteria. And it causes oscillations in GFP, expressions that are described by these differential equations here. And some of the modeling approaches used in synthetic biology will be covered by Professor Fraenkel and Lauffenburger later.

All right. So late 2000s, early 2010s, it's still too early to say, for sure, what the most important developments will be. But certainly, in the late 2000s, next gen sequencing-- which now probably should be called second generation sequencing, since there may be future generations-- really started to transform a whole wide variety of applications in biology, from making genome sequencing-- instead of having to be done in the genome center, now an individual lab can easily do microbial genome sequencing. And when needed, it's possible, also, to do genome sequencing of larger organisms.

Transcriptome sequencing is now routine. We'll hear about that. There are applications for mapping protein-DNA interactions genome wide, including both sequence specific transcription factors as well as more general factors like histones, protein-RNA interactions-- a method called CLIP-Seq-- methods for mapping all the translated messages, the methylated sites in the genome, open chromatin, and so forth.

So many people contributed to this, obviously. I'm just mentioning, Barbara Wold was a pioneer in both RNA-Seq as well as ChIP-Seq. And some of the sequencing technologies that came out here are shown here. And we will discuss those at the beginning of lecture on Thursday.

So I encourage you to read this review here, by Metzger, which covers many of the newer sequencing technologies. And they're pretty interesting. As you'll see, there's some interesting tricks, interesting chemistry and image analysis tricks.

All right. So that was not very scholarly. But if you want a proper history, then this guy, Hallam Stevens, who was a History of Science PhD student at Harvard and recently graduated, wrote this history of bioinformatics.

OK. So let's look at the syllabus. So also posted on the [INAUDIBLE] site is a syllabus. It looks like this. This is quite an information-rich document. It has all the lecture titles, all the due dates of all the problem sets, and so forth. So please print yourself a copy and familiarize yourself with it. So we'll just try to look at a high level first, and then zoom in to the details.

So at a high level, if you look in this column here, we've broken the course into six different topics. OK? So there's Genomic Analysis I, that I'll be teaching, which is more classical computational biology, you could say-- local alignment, global alignment, and so forth. Then, Genomic Analysis II, which Professor Gifford will be teaching, covers some newer methods that are required when you're doing a lot of second generation sequencing-- the standard algorithms are not fast enough, you need better algorithms, and so forth.

And then, I will come back and give a few lectures on modeling biological function. This will have to do with sequence motifs, hidden Markov models, and RNA secondary structure. Professor Fraenkel will then do a unit on proteomics and protein structure. And then, there will be an extended unit on regulatory networks. Different types of regulatory networks will be covered, with most of the lectures by Ernest, one by David, and one by Doug.

And then, we'll finish up with computational genetics, by David. And there will also be some guest lecturers, one of them interspersed in regulatory networks, and then two at the end, from Ron Weiss and George Church.

So I just wanted to point out that in all of these topics, we will include some discussion of motivating question. So, what are the biological questions that we're seeking to address with these approaches. And there will also be some discussion of the experimental method.

So for example, in the first unit, it's heavy on sequence analysis. So we'll talk about how sequencing is done, and then, quite a bit about the interaction between the experimental technology and the computational analysis, which often involves statistical methods for estimating the error rate of the experimental method, and things like that. So the emphasis is on the computational part, but we'll have some discussion about experiments.

Everyone with me so far? Any questions? OK.

All right. So what are some of these motivating questions that we'll be talking about? So what are the instructions encoded in our genomes? You can think of the genome as a book. But it's in this very strange language. And we need to understand the rules, the code that underlies a lot of research in gene expression.

How are chromosomes organized? What genes are present-- so tools for annotating genomes. What regulatory circuitry is encoded? You'd like to be able to eventually look at a genome, understand all the regulatory elements, and be able to predict that there's some feedback circuit there that responds to-- a particular stimulation that responds to light, or nutrient deprivation, or whatever it might be.

Can the transcriptome be predicted from the genome? This is a longstanding question. The translatome, if you will-- well, let's say, the proteome can be predicted from the transcriptome in the sense that we have a genetic code and we can look up those triplets. So there's a dream that we would be able to model other steps in gene expression with the precision with which the genetic code predicts translation-- that we'd be able to predict where the polymerase would start transcribing, where it will finish transcribing, how a transcript will be spliced, et cetera-- all the other steps in gene expression. And that motivates a lot of work in the field.

Can protein function be predicted from sequence? So this is a very classical problem. But there are a number of new and interesting developments as resolved from a lot of this high throughput data generation, both in nucleic acid sequencing as well as in proteomics.

Can evolutionary history be reconstructed from sequence? Again, this has been a longstanding goal of the field. And a lot of progress has been made here. And now most evolutionary classifications are actually based on molecular sequence at some level. And new species are often defined based on sequence.

OK. Other motivating questions. So what would you need to measure if you wanted to discover the causes of a disease, the mechanisms of existing drugs, metabolic pathways in a micro-organism? So this is a systems biology question.

You've got a new bug. It causes some disease. What should you measure? Should you sequences its genome? Should you sequence its transcriptome? Should you do proteomics? What type of proteomics? Should you perturb the system in some way and do a time series? What are the most efficient ways? What information should be gathered, and in what quantities? And how should that information be integrated in order to come up with an understanding of the physiology of that organisms so that, then, you can know where to intervene, what would be suitable drug targets?

Yeah. What kind of modeling would help you to use the data to design new therapies, or even, in a synthetic biology context, to re-engineer organisms for new purposes? So microbes to generate-- to produce fuel, for example, or other useful products. What can we currently measure?

What does each type of data mean individually? What are the strengths and weaknesses of each of the types of high throughput approaches that we have? And how do we integrate all the data we have on a system to understand the functioning of that system? So these are some of the questions that motivate the latter topics on regulatory networks. OK.

So let's now zoom in and look more closely at the course syllabus. I've just broken it into two halves, just so it's more readable. So today we're going over, obviously, course mechanics mostly. On Thursday, we'll cover both some DNA sequencing technologies and we'll talk about local alignment on BLAST. More on that in a bit.

The 6.8047 recitation-- 6.874, thank you-- recitation will be on Friday. The other

recitations will start next week. And then, as you can see, we'll move through the other topics. So each of the instructors is going to briefly review their topic. So I won't go through all the titles here.

But please note, on the left side here, that their assignment due dates are marked. OK? And they're all due at noon on the indicated day. And so some of these are problem sets. So for example, problem set 1 will be due on Thursday, February 20 at noon.

And some of the other assignments relate to the project component of the course, which we're going to talk more about in a moment. In particular, we're going to ask you to submit a brief statement of your background and your research interests related to forming teams. So the projects are going to be done in teams of one to five students.

And in order to facilitate especially cross disciplinary teams-- we'd love if you interact with, maybe, students in a different grad program, or whatever-- you'll post your background. You know, I'm a first year BE student and I have a background in Perl programming, but never done Python, or whatever-- something like that. And then, I'm interested in doing systems biology modeling in microbial systems, or something like that.

And then you can match up your interests with others and form teams. And then you'll come up with your own project ideas so that the team and initial idea will be due here, February 25th. Then you'll need to do some aims and so forth. So the project components here, these are only for those taking the grad version of the course. We'll make that clear later. OK.

So after the first three topics here, taught by myself and David, there will be an exam. More on that later. And then there will be three more topics, mostly taught by Ernest. And notice there are additional assignments here related to the project-- so, to research strategy-- and the final written report, additional problems sets.

We'll have a guest lecturer here. This will be Ron Weiss. Then, there will be the

second exam. Exams are non-cumulative, so the second exam will just cover these three topics here predominantly. And then there will be another guest lecturer. This would be George Church here.

And then notice, here, presentation. So those who are doing the project component, those teams will be given-- assigned a time to present, to the class, the results of their research. And you'll be graded-- the presentation will be part of the overall project grade assigned by the instructors. But you'll also-- we'll also ask all the students in the class to send comments on the presentations.

So you may find that you get helpful suggestions about interpreting your data from other people and so forth. So that'll be a required component of the course for all students, to attend the presentations and comment on them. And we hope that will be a lot of fun.

OK. So is this the right course for me? So I just wanted to let you know you're fortunate to have a rich selection of courses in computational systems, synthetic biology here at MIT. I've listed many of them. Probably not all, but the ones that I'm aware of that are available on-campus.

757 is really only for biology grad students. But the other courses listed here are generally open. Some are more geared for graduate students, some more undergrads. Some are more specialized. So for example, Jeff Gore's systems biology course, it's more focused on systems biology whereas our course covers both computational and systems. So keep that in mind. Make sure you're in the right place, that this is what you want.

OK. A few notes on the textbook-- so there is a textbook. It's not required. It's called *Understanding Bioinformatics* by Zvelebil and Baum. It's quite good on certain topics. But it really only covers about, maybe, a third of what we cover in the course. So there is good content on local alignment, global alignment, scoring matrices-- the topics of the next couple lectures. And I'll point you to those chapters.

But it's very important to emphasize that the content of the course is really what

happens in lecture, and on the homeworks, and to some extent, what happens in recitation. And the textbook is just there as a backup, if you will, or for those who would like to get more background on the topic or want to read a different description of that topic. So you decide whether you want to purchase the textbook or not.

It's available at the Coop or through Amazon. Shop around. You can find it. It's paperback. Pretty good general reference on a variety of topics, but it doesn't really have much on systems biology.

All right. Another important reference that was developed specifically for this course a few years ago is the probability and statistics primer. So you'll notice that some of the homeworks, particularly in the earlier parts of the course, will have significant probability and statistics. And we assume that you have some background in this area. Many of you do. If you don't, you'll need to pick that up. And this primer was written to provide those topics, in probability especially, that are foundational and most relevant to computational biology.

So for example, there are some concepts like p-value, probability density function, probability mass function, cumulative distribution function, and then, common distributions, exponential distribution, Poisson distribution, extreme value distribution. If those are mostly sounding familiar to you, that's good. If they're familiar, but you couldn't-- you really don't-- you get binomial and Poisson confused or something, then, definitely, you want to consult this primer.

So I think, looking at the lectures and the homeworks, it should probably become pretty clear which aspects are going to be relevant. And I'll try to point those out when possible. And you can also consult your TAs if you're having trouble with the probability and statistics content.

So we are going to focus, here, on, really, the computational biology, bioinformatics content. And we might briefly review a concept from probability, like, maybe, conditional probability when we talk about Markov chains. But we're not going to spend a lot of time. So if that's the first time you've seen conditional probability, you

might be a little bit lost. So you'd be better off reading about it in advance. OK?

Questions? No questions? All right. Maybe it's that the video is intimidating people. OK. All right. The TAs know a lot about probability and statistics and will be able to help you.

OK. So homework-- so I apologize, the font is a little bit small here. So I'll try to state it clearly. So there are going to be five problem sets that are roughly one per topic. Except you'll see p set two covers topics two and three. So it might be a little bit longer.

The way we handle students who have to travel-- so many of you might be seniors. You might be interviewing for graduate schools. Or you might have other conflicts with the course. So rather than doing that on a case by case basis, which, we've found, gets very complicated and is not necessarily fair, the way we've set it up is that the total number of points available on the five homeworks is 120. OK? But the maximum score that you can get is 100.

So if you, for example, were to get 90% on all five of the homeworks, that would be 90% of 120, which would be 108 points. You would get the full 100-- you'd get 100% on your homework. That would be a perfect score on the homework. OK? But because of that-- because there's more points available than you need-- we don't allow you to drop homeworks, or to do an alternate assignment, or something like that.

So the way it works is you can basically miss-- as long as you do well on, say, four of the homeworks-- you could actually miss one without much of a penalty. For example, if each of the homeworks were worth 24 points and you got a perfect score on four of them, that would be 96 points. You would have an almost perfect score on your homework and you could miss that fifth homework.

Now of course, we don't encourage you to skip that homework. We think the homeworks are useful and are a good way to solidify the information you've gotten from lecture, and reading, and so forth. So It's good to do them. And doing the

homeworks will help you and perhaps prepare you for the exams. But that's the way we handle the homework policy.

Now I should also mention that not all the homeworks will be the same number of points. We'll apportion the points in proportion to the difficulty and length of the homework assignment. So for example, the first homework assignment is a little bit easier than the others. So it's going to have somewhat fewer points.

So late assignments-- so all the homeworks are due at noon on the indicated day. And if it's within 24 hours after that, you'll be eligible for 50% credit. And beyond that, you don't get any points, in part because the TAs will be posting the answers to the homeworks. OK? And we want to be able to post them promptly so that you'll get the answers while those problems are fresh in your mind. Questions about homeworks? OK. Good.

So collaboration on problem sets-- so we want you to do the problem sets. You can do them independently. You can work with a friend on them, or even in a group, discuss them together. But write up your solutions independently. You don't learn anything by copying someone else's solution.

And if the TAs see duplicate or near identical solutions, both of those homeworks will get a 0. OK? And this occasionally happens. We don't want this to happen to you. So just avoid that. So discuss together, but write up your solutions separately.

Similarly, with programming, if you have a friend who's a more experienced programmer than you are, by all means, ask them for advice, general things, how should I structure my program, do you know of a function that generates a loop, or whatever it is that you need. But don't share code with anyone else. OK? That would be a no-no. So write up your code independently.

And again, the graders will be looking for identical code. And that will be thrown out. And so we don't want to have any misconduct of that type occurring.

All right. So recitations-- there are three recitation sessions offered each week, Wednesday at 4:00 by Peter, Thursday at 4:00 by Colette, Friday at 4:00 by Tahin.

And that is a special recitation that's required for the 6.874 students-- David? Yes-- and has additional AI content. So anyone is welcome to go to that recitation. But those who are taking 6.874 must go.

For students registered for the other versions of the course, going to recitation is optional but strongly recommended. Because the TAs will go over material from the lectures, material that's helpful for the homeworks or for studying for exams-- in the first weeks, Python, probability as well. So go to the recitations, particularly if you're having trouble in the course. So Tahin's recitation starts this week. And Peter and Colette's will start next week.

Question. Yes.

AUDIENCE: Are they covering the same material, Peter and Colette?

PROFESSOR: Peter and Colette will cover similar material and Tahin will cover different material.

Python instruction-- so the first problem set, which will be posted this evening, doesn't have any programming on it. But if you don't have programming, you need to start learning it very soon. And so there will be a significant programming problem on p set two. And we'll be posting that problem soon-- this week, some time. And you'll want to look at that and gauge, how much Python do I need to learn to at least do that problem.

So what is this project component that we've been hearing about? So here's a more concrete description. So again, this is only for the graduate versions of the course. So students will-- basically, we've structured it so you work incrementally toward the final research project and so that we can offer feedback and help along the way if needed.

So the first assignment will be next week. I think it's on Tuesday. I'll have more instructions on Thursday's lecture. But all the students registered for the grad version will submit their background and interests for posting on the course website. And then you can look at those and try to find other students who have, ideally, similar interests but perhaps somewhat different backgrounds. It's particularly

helpful to have a strong biologist on the team. And perhaps, a strong programmer would help as well.

So then, you will choose your teams and submit a project title and one-paragraph summary, the basic idea of your project. Now we are not providing a menu of research projects. It's your choice-- whatever you want to do, as long as it's related to computational and systems biology.

So it could be analysis of some publicly available data. Could be analysis of some data that you got during your rotation. Or for those of you who are already in labs that you're actually working on, it's totally fine and encouraged that the project be something that's related to your main PhD work if you've started on that. And it could also be more in the modeling, some modeling with MATLAB or something, if you're familiar with that.

But, a variety of possibilities. We'll have more information on this later. But we want you to form teams. The teams can work independently or with up to four friends in teams of five. And if people want to have a giant team to do some really challenging project, then you can come and discuss with us. And we'll see if that would work.

So then there's the initial title and one-paragraph summary. We'll give you a little feedback on that. And then you'll submit an actual specific aims document-- so with actual, NIH-style, specific aims-- the goal is to understand whether this organism has operons or not, or some actual scientific question-- and a bit about how you will undertake that.

And then you'll submit a longer two-page research strategy, which will include, specifically, we will use these data. We will use this software, these statistical approaches-- that sort of thing. And then, toward the end of the semester, a final written report will be due that'll be five pages.

You'll work on it together, but it'll need to be clear who did what. You'll need an author contribution statement. So and so did this analysis. So and so wrote this section-- that sort of thing. And then, as I mentioned before, there will be oral

presentations, by each team, on the last two course sessions. OK? Questions about the projects? There's more information on the course info document online. OK? Good.

Yes, so for those taking 6.874, in addition to the project, there will also be additional AI problems on both the p sets-- and the exam? David? Yes. OK. And they're optional for others. All right. Good.

OK. So how are we going to do the exam? So as I mentioned, there's two 80-minute exams. They're non-cumulative. So the first exam covers, basically, the first three topics. The second exam is on the last three topics. They're 80 minutes. They're during normal class time. There's no final exam.

The grading-- so for those taking the undergrad version, the homeworks will count 36% out of the maximum 100 points. And the exams will count 62%. And then, this peer review, where there's two days where you go, and you listen to presentations, and you submit comments online counts 2%. For the graduate Bio BE HST versions, it's 30% homeworks, 48% exams, 20% project, 2% peer review. For the EECS version, 6.874, 25% homework, 48% exams, 20% project, and then, 5% for these extra AI related problems, and 2% peer review.

And those should add up to 100. And then, in addition, we will reward 1% extra credit for outstanding class participation-- so questions, comments during class. OK. All right.

So a few announcements about topic one. And then, each of us will review the topics that are coming up. So p set one will be posted tonight. It's due February 20th, at noon. It involves basic microbiology, probability, and statistics. It'll give you some experience with BLAST and some of the statistics associated.

P set two will be posted later this week. So you don't need to, obviously, start on p set two yet. It's not due for several weeks. But definitely, look at the programming problem to give you an idea of what's involved and what to focus on when you're reviewing your Python. Mentioned the probability/stats primer.

Sequencing technology-- so for Thursday's lecture, it will be very helpful if you read the review, by Metzger, on next gen sequencing technologies. It's pretty well written. Covers Illumina, 454, PACBIO, and a few other interesting sequence technologies.

Other background reading-- so we'll be talking about local alignment, global alignment, statistics, and similarly matrices for the next two lectures. And chapters four and five of the textbook provide a pretty good background on these topics. I encourage you to take a look.

OK. So I'm just going to briefly review my lectures. And then we'll have David and Ernest do the same. So sequencing technologies will be the beginning of lecture two. And then we'll talk about local ungapped sequence alignment-- in particular, BLAST.

So BLAST is something like the Google search engine of bioinformatics, if you will. It's one of the most widely used tools. And it's important to understand something about how it works, and in particular, how to evaluate the significance of BLAST hits, which are described by this extreme value distribution here.

And then, lecture three, we'll talk about global alignment and introducing gaps into sequence alignments. We'll talk about some dynamic programming algorithms-- Needleman-Wunsch, Smith-Waterman. And in lecture four, we'll talk about comparative genomic analysis of gene regulation-- so using sequence similarity across genomes two infer location of regulatory elements such as microRNA target sites, other things like that.

All right. So I think this is now-- oh sorry, a few more lectures. Then, in the next unit, modeling biological function, I'll talk about the problem of motif finding-- so searching a set of sequences for a common subsequence, or similar subsequences, that possess a particular biological function, like binding to a protein. It's often a complex search space. We'll talk about the Gibbs sampling algorithm and some alternatives.

And then, in lecture 10, I'll talk about Markov and hidden Markov models, which

have been called the Legos of bioinformatics, which can be used to model a variety of linear sequence labeling problems. And then, in the last lecture of that unit, I'll talk a little bit about protein-- I'm sorry, about RNA-- secondary structure-- so the base pairing of RNAs-- predicting it from thermodynamic tools, as well as comparative genomic approaches.

And you'll learn about the mfold tool and how you can use a diagram like that to infer that this RNA may have different possible structures that I can fold into, like those shown. All right. So I'm going to pass it off to David here.

PROFESSOR: Thanks very much, Chris. All Right. So I'm David Gifford. And I'm delighted to be here.

It's really a wonderfully exciting time in computational biology. And one of the reasons it's so exciting is shown on this slide, which is the production of DNA base sequence per instrument over time. And as you can see, it's just amazingly more efficient as time goes forward.

And if you think about the reciprocal of this curve, the cost per base is basically becoming extraordinarily low. And this kind of instrument allows us to produce hundreds of millions of sequence reads for a single experiment. And thus do we not only need new computational methods to handle this kind of a big data problem, but we need computational methods to represent the results in computational models.

And so we have multiple challenges computationally. Because modern biology really can't be done outside of a computational framework. And to summarise the way people have adapted these high throughput DNA sequencing instruments, I built this small figure for you.

And you can see that-- Professor Burge will be talking about DNA sequencing next time. And obviously, you can use DNA sequencing to sequence your own genomes, or the genomes of your favorite pet, or whatever you like. And so we'll talk about, in lecture six, how to actually do genome sequencing. And one of the challenges in doing genome sequencing is how to actually find what you have sequenced. And

we'll be talking about how to map sequence reads as well.

Another way to use DNA sequencing is to take the RNA species present in a single cell, or in a population of cells, and convert them into DNA using reverse transcriptase. Then we can sequence the DNA and understand the RNA component of the cell, which, of course, either can be used as messenger RNA to code for protein, or for structural RNAs, or for non-coding RNAs that have other kinds of functions associated with chromatin.

In lecture seven, we'll be talking about protein/DNA interactions, which Professor Burge already mentioned-- the idea that we can actually locate all the regulatory factors associated with the genome using a single high throughput experiment. We do this by isolating the proteins and their associated DNA fragments and sequencing the DNA fragments using this DNA sequencing technology.

So briefly, the first thing that we'll look at in lecture five is, given a reference genome sequence and a basket of DNA sequence reads, how do we build an efficient index so that we can either map or align those reads back to the reference genome. That's a very important and fundamental problem. Because if I give you a basket of 200 million reads, we need to build its alignment very, very rapidly, and quickly, and accurately, especially in the context of repetitive elements. Because a genome obviously has many repeats in it. We need to consider how our indexing and searching algorithms are going to handle those sorts of elements.

In the next lecture, we'll talk about how to actually sequence a genome and assemble it. So the fundamental way that we approach genome sequencing given today's sequencing instruments is that we take intact chromosomes at the top, which, of course, are hundreds of millions of bases long, and we shatter them into pieces. And then we sequence size selected pieces in a sequencing instrument. And then we need to put the jigsaw puzzle back together with a computational assembler.

So we'll be talking about assembly algorithms, how they work, and furthermore, how resolve ambiguities as we put that puzzle together, which often arise in the context

of repetitive sequence. And in the next lecture, in lecture seven, we'll be looking at how to actually take those little DNA molecules that are associated with proteins and analyze them to figure out where particular proteins are bound to the genome and how they might regulate target genes. Here we see two different occurrences of OCT4 binding events binding proximally to the SOX2 gene, which they are regulating.

So that's the beginning of our analysis of DNA sequencing. And then we'll look at RNA sequencing-- once again, with going through a DNA intermediate-- and ask the question, how can we look at the expression of particular genes by mapping RNA sequence reads back onto the genome. And there are two fundamental questions we can address here, which is, what is the level of expression of a given gene, and secondarily, what isoforms are being expressed.

The second sets of reads, you see up on the screen, are split reads that cross splice junctions. And so by looking at how reads align to the genome, we can figure out which particular axons are included or excluded from a particular transcript. So that's the beginning of the high throughput biology genomic analysis module. And I'll be returning to talk, later in the term, about computational genetics, which, really, is a way to summarize everything we're learning in the course into an applicable way to ask fundamental questions about genome function, which Professor Burge talked about earlier.

Now we all have about 3 billion bases in our genomes. And as you know, you differ from the individual sitting next to you in about one in every 1,000 base pairs, on average. And so one question is, how do we actually interpret these differences between genomes. And how can we build accurate computational models that allow us to infer function from genome sequence? And that's a pretty big challenge.

So we'll start by asking questions about, what parts of the genome are active and how could we annotate them. So we can use, once again, different kinds of sequencing based assays to identify the regions of the genome that are active in any given cellular state. And furthermore, if we look at different cells, we can tell

which parts of the genome are differentially active.

Here you can see the active chromatin during the differentiation of an ESL into a terminal type over a 50 kilobase window. And the regions of the genome that are shaded in yellow represent regions that are differentially active. And so, using this kind of DNA seq. data and other data, we can automatically annotate the genome with where the regulatory elements are and begin to understand what the regulatory code of the genome is.

So once we understand what parts of the genome are active, we can ask questions about, how do they contribute to some overall phenotype. And our next lecture, lecture 19, we'll be looking at how we can build a model of a quantitative trait based upon multiple loci and the particular alleles that are present at that loci. So here you see an example of a bunch of different quantitative trait loci that are contributing to the growth rate of yeast in a given condition.

So part of our exploration during this term will be to develop computational methods to automatically identify regions of the genome that control such traits and to assign them significance. And finally, we'd like to put all this together and ask a very fundamental question, which is, how do we assign variations in the human genome to differential risk for human disease. And associated questions are, how could we assign those variants to what best therapy would be applied to the disease-- what therapeutics might be used, for example.

And here we have a bunch of results from genome wide association studies, starting at the top with bipolar disorder. And at the bottom is type 2 diabetes. And looking along the chromosomes, we're asking which locations along the genome have variants that are highly associated with these particular diseases in these so-called Manhattan plots. Because the things that stick up look like buildings.

And so these sorts of studies are yielding very interesting insights into variants that are associated with human disease. And the next step, of course, is to figure out how to actually prove that these variants are causal, and also, to look at mechanisms where we might be able to address what kinds of therapeutics might

be applied to deal with these diseases.

So those are my two units. Once again, high throughput genomic analysis, and secondarily, computational genetics. And finally, if you have any questions about 6.874, I'll be here after lecture. Feel free to ask me. Ernest.

PROFESSOR: Thank you very much, Dave. All right. So in the preceding lectures, you'll have heard a lot about the amazing things we can learn from nucleic acid sequencing. And what we're going to do in the latter part of the course is look at other modalities in the cell. Obviously, there's a lot that goes on inside of cells that's not taking place at the level of DNA/RNA. And so we're going to start to look at proteins, protein interactions, and ultimately, protein interaction networks.

So we'll start with the small scale, looking at intermolecular interactions of the biophysics-- the fundamental biophysics of a protein structure. Then we'll start to look at protein-protein interactions. And as I said, the final level will be networks.

There's been an amazing advance in our ability to predict protein structure. So it's always been the dream of computational biology to be able to go from the sequence of a gene to the structure of the corresponding protein. And ever since Anfinsen, we know that, at least, that should be theoretically possible for a lot of proteins. But it's been computationally virtually intractable until quite recently. And a number of different approaches have allowed us to predict protein structure.

So this slide shows, for example-- I don't know which one's which-- but in blue, perhaps, prediction, and in red, the true structure, or the other way around. But you can see, it doesn't matter very much. We're getting extremely accurate predictions of small protein structure.

There are a variety of approaches here that live on a spectrum. On the one hand, there's the computational approach that tries to make special purpose hardware to carry out the calculations for protein structure. And that's been wildly successful. On the other end of the extreme, there's been the crowd sourcing approach to have gamers try to predict protein structure. And that turns out to be successful. And

there's a lot of interesting computational approaches in the middle as well. So we'll explore some of these different strategies.

Then, the ability to go from just seeing a single protein structure, how these proteins interact. So now there are pretty good algorithms predicting protein-protein interactions as well. And that allows us, then, to figure out not only how these proteins function individually, but how they begin to function as a network.

Now one of the things that we've already-- going to be touched on in the early parts of the course are protein-DNA interactions through sequencing approaches. We want to now look at them at a regulatory level. And could you reconstruct the regulation of a genome by predicting the protein-DNA? And there's been a lot of work here.

We talked earlier about Eric Davidson's pioneering approaches, a lot of interesting computational approaches as well, that go from the relatively simple models we saw on those earlier slides to these very complicated networks that you see here. We'll look at what these networks actually tell us, how much information is really encoded in them. They are certainly pretty, whatever they are.

We'll look at other kinds of interactions networks. We'll look at genetic interaction networks as well, and perhaps, some other kinds. And finally, we'll look at computable models. And what do we mean by that? We mean model that make some kind of very specific prediction, whether it's a Boolean prediction-- this gene will be on or off-- or perhaps, an even more quantitative prediction. So we'll look at logic based modeling, and probably, Bayesian networks as well.

So that's been a very whirlwind tour of the course. Just to go back to the mechanics, make sure you're signed up for the right course. The undergraduate versions of the course do not have a project. And they certainly do not have the artificial intelligence problems.

Then, there are the graduate versions that have the project, but do not have the AI, and finally, the 6.874, which has both. So please be sure you're in the right class so

you get credit for the right assignments. And finally, if you've got any questions about the course mechanics, we have a few minutes. We can talk about it here, in front of everybody. And then, we'll all be available after class, a little bit, to answer questions.

So please, any questions? Yes.

AUDIENCE: Can undergrads do a project?

PROFESSOR: Can undergrads do a project? Well, they can sign up for the graduate version, absolutely. Other questions? Yes.

AUDIENCE: Yeah, I actually can't make either of those sessions.

PROFESSOR: Well, send an email to the staff list and we'll see what we can do. Other questions? Yes.

AUDIENCE: 6.877 doesn't exist anymore-- Computational Evolutionary Biology, the class.

PROFESSOR: Oh, the thing we listed as alternative classes? Sorry.

PROFESSOR: We'll fix that.

AUDIENCE: Really? I would be so happy.

[LAUGHTER]

PROFESSOR: We'll delete it from our slides, yes. We are powerful, but not that powerful. Other questions, comments, critiques? Yes, in the back.

AUDIENCE: Are each of the exams equally weighted?

PROFESSOR: Are each of the exams equally weighted? Yes, they are. Yes.

AUDIENCE: If we're in 6.874, we have the additional AI problems. Does that mean that we have more questions on the exam, but just as much time to do them?

PROFESSOR: The question was, if you're in 6.874 and you have the additional AI questions on the

exam, does that mean you have more questions, but the same amount of time. And I believe the answer to that is yes.

PROFESSOR: We may revisit that question.

PROFESSOR: We will revisit that question at a later date. But course six students are just so much smarter than everyone else, right? Yes.

AUDIENCE: Can we switch between different versions of the class by the add deadline?

PROFESSOR: I'm sorry, could you say that again?

AUDIENCE: Can we switch between versions of the class by the add deadline?

PROFESSOR: Can you switch between different versions of class by the add/drop deadline? In principle, yes. But if you haven't done the work for that, then you will be in trouble. And there may not be a smooth mechanism for making up missed work that late. Because the add/drop deadline is rather late.

So I'd encourage you, if you're considering doing the more intensive version or not, sign up for the more intensive version. You can always drop back. Once we form groups though, for the projects, then, obviously, there's an aspect of letting down your teammates. So think carefully, now, about which one you want to join. It'll be hard to switch between them. But we'll do whatever the registrar requires us to do in terms of allowing it. Other questions? Yes.

AUDIENCE: If the course sites, can we undergrads access the AI problems just for fun, to look at them?

PROFESSOR: Yes. If the course sites remain separate, will undergrads be able to access the AI problems? The answer is yes. Yes.

AUDIENCE: Should students in 6.874 attend both presentations?

PROFESSOR: Should students in 6.874 attend both presentations? Not necessary. You're welcome to both if you want. But the course six recitation should be sufficient. Other

questions? Yes.

AUDIENCE: So what's covered in the normal recitations?

PROFESSOR: What's covered in the normal recitations? It'll be reinforcing material that's in the lectures. So the goal is not introduce any new material in the course 20, course seven recitations. Just to clarify, not to introduce any new material. Other questions?

OK. Great. And we'll hang around a little bit to answer any remaining questions when they come up.