PROFESSOR: So good afternoon, once again. And welcome back to Computational Systems Biology, lecture number seven. And today we're going to put to good use two things that we have learned. We've learned how to align high throughput reads to genomes. And we've learned how to take a collection of high throughput reads and assemble a genome.

And today we're going to delve into the mysteries of transcriptional regulation. And so what I'd like to discuss with you today is a very important set of techniques that allows us to elucidate exactly how genes are regulated. It's probably the most important technique for allowing us to get at that question. And I'm sure many of you are familiar with the idea of transcriptional regulators, which are proteins that bind in a sequence specific way to the genome and act as molecular switches.

And we'll return to some aspects of these when we talk about proteomics later in the term. But suffice to say, these proteins have domains that interact in a sequence specific way with the DNA bases in one of the grooves of DNA. They also typically contain a domain which is an activation domain or repression domain that interacts with other proteins that can cause the genome to fold up. And it can also help recruit the RNA polymerase holoenzyme to actually turn on a gene transcription.

So here we have a figure of a collection of Pit1 molecules interacting with the genome. And of course, there are many flavors of genomic regulators. It's estimated that humans have about 2,000 different proteins that act as these molecular switches, some as activators, some as repressors. And we're going to be talking about, fundamentally, today the idea of how these molecules interact with the genome and control gene expression through the analysis of where they actually interact with specific genome loci.

So if we were to draw a picture of how we understand gene regulation in cartoon form, if we have a gene here with the transcription start site and we can imagine RNA molecules being produced off of this genomic template, we know that there are non-coding regions of a gene that permit for the binding of these regulators. And my definition of a gene is all of the DNA that's required to make a specific transcriptive protein. So that includes not only the coding parts of the gene but the non-coding, regulatory parts as well.

So we can imagine out here that there are a collection of regulators, perhaps just one, that bind to a sequence that in turn activate this gene, producing the RNA transcript, which then in turn this turned into another protein. This protein may undergo some sort of post-translational modification by signaling pathway or other mechanism that activates it. So regulators need to be activated before they could bind and some do not. And this activated regular combine to get another gene and cause another RNA to be expressed.

And it may be, in the second context, that we need two different proteins to bind to activate the gene. And so during the course of the term, we're going to be talking about many aspects of these regulatory networks, including things like what the regulatory code of a genome is, that is where these binding sites are and how they're occupied. And we'll return to that later on in today's lecture.

We'll talk about the dynamics of binding of proteins, including how concentration [? they ?] dependent are. And we'll talk about combinatorial control, whether or not, for example, both of these have to be present or just one of them needs to be present for a gene to be transcribed, and how you can use these regulatory sequences to implement very complex computational functions.

But suffice to say, the most important thing that we have to identify is the programming that underlies the genome, which includes these regulatory sequences and exactly how they're occupied by regulatory proteins. Ideally what we would like to be able to do is to do a single experiment that elucidated all of the regulatory sites in the genome and which ones were occupied by which proteins.

But presently, that's technically not possible.

So we'll begin today with a technique that allows us to consider one protein at a time and identify where it occupies the genome. Now there are other kinds of proteins that we can identify in terms of where they are associated with the genome that are not transcriptional regulators, per se. For example, we all know that chromatin is organized on spools called nucleosomes. These nucleosomes are composed of eight different histones that have tails on them, and there can be covalent marks put on these tails. We'll return to this later on when we talk about epigenetics.

But I did want to mention today that it's possible to identify where there are histones with specific barks that are present in the genome and what genome sequences they are associated with. So we can look at sequence specific regulators. We could look at a more general epigenetic marks, all using the same technology.

And this slide simply recapitulates what we just talked about over here on the left-hand side of the board. But we want to know basically what and where, in terms of these genomic regulators, what they are and where they are in the genome. And today we're going to assume a fairly simple model, which is that regulars that are proximal to a gene, most probably regulated. Although we know in practice, actually, it appears that roughly one third of the regulators that are proximal to a gene actually skip it and regulate a gene further down the genome.

It's not really understood very well how the genome folds in three space to allow these transit regulatory interactions to occur. But I'll just point out to you that the simplistic model that proximal binding regulates proximal genes doesn't always hold, especially when we get into mammals and other higher organisms.

And as I mentioned, another aspect of this is that certain proteins may need to be modified to become active, and thus there are signaling pathways. You can imagine signaling pathways responding to environmental stimuli outside of cells. These signaling pathways can interact with transcriptional activators and modify what targets they seek in the genome.

So these sorts of regulatory networks will be talked about specifically in a separate lecture later in the term, but they're extraordinarily important. And a foundational aspect of them is putting together the wiring diagram. And the wiring diagram has to do with where the regulators occupy the genome and what genes those regulators regulate.

And in order to do that, we're going to utilize a technique today called ChIP-seq, which stands for chromatin immunoprecipitation followed by sequencing. And we can now reliably identify where regulars bind to the genome within roughly 10 base pairs. So the spatial resolution has gotten exceptionally good with high throughput sequencing, as we'll see, which really is a fantastic era to be in now, because this really wasn't possible 5 or even 10 years ago. And so we now have the tools to be able to take apart the regulatory occupancy of the genome and discern exactly where these proteins are binding.

The way that this is done, as I'll describe the protocol to you, in general, and then I'm going to pause for a second and see if anybody has any questions about the specifics. But the essential idea is that you have a collection of cells. Typically you need a lot of cells. We're talking 10 million cells.

So for certain kinds of marks, you can get down below a million or even to 100,000 cells. But to get robust signals, you need a lot of cells at present. And all these cells obviously have chromosomes inside of them with proteins that are occupying them.

And the essential idea is that you take a flash photography picture of the cell while it's alive. You add a cross linking agent, and that cross links proteins creates bonds between the proteins and the genome, the DNA, where those proteins are sitting. And so you then isolate the chromatin material, and you wind up with pieces of DNA with proteins occupying them, all the proteins. So not just some of the proteins, but all the proteins are non-selectively cross-linked to the genome.

You then can take this extract and fragment it. Typically you fragment it by using sonication, which is mechanical energy, which causes the DNA to break at random locations. There are more modern techniques that we'll touch on at the end of today's lecture where you could enzymatically digest these fragments right down to where the protein is.

But suffice to say, you get small fragments, which you then can immunopurify with an antibody to a protein of interest. So one condition of using this technology is either A, you have a good antibody to a protein that you care about as regulatory, or B, you have tagged this protein such that it has a flag tag, myc tag, or some other epitope tag on it which allows you to use an antibody or other purification methodology for that specific tag. So either you have a good antibody or you have a tag on the protein.

One problem with tags on proteins is that they can render the proteins nonfunctional. If they're nonfunctional, then, of course, they're not going to bind where they should bind. And one has to be careful about this, because if you introduce a tag and you have a couple good copies of the protein that are untagged and one copy that is tagged, it's hard to tell whether or not the tagged version is actually doing what you think it does, and one has to be careful.

But suffice to say, assuming that you have some way of immunopurifying this protein, you can then use the antibodies to simply purify those fragments that have the protein of interest. After you've purified the protein of interest, you can reverse the cross linking, have a collection of fragments which you then sequence using a high throughput sequencing instrument.

Now recall that, in the usually applied protocol, the fragmentation is a random. So you're going to be sequencing both ends of these molecules. For each one, you probably only sequence one end. You're going to sequence an end of these molecules which gives you a sequence tag that is near where the event occurred, but not exactly at it.

And we're going to take those tags, and if we have our genome-- here represented as this short, horizontal chalk line-- we'll take our reads, and we will align them to the genome, and try and discern from those aligned reads where the original proteins were binding. Now our job is to do the best possible alignment or discovery of where the proteins are binding, given this evidence. So we have a collection of evidence exhibited by the read sequences.

The other thing that we will do is we will take the original population of molecules, and we will sequence them as well, sometimes called the whole cell extract sequence, as a control. And we'll see why we need this control a little bit later on. But this is going to be a purified so-called IP for immunoprecipitate fraction, which we'll sequence. And this will be the whole cell extract, which should not be enriched for any particular protein.

Now before I go on, I'd be happy to entertain any questions about the details of this protocol, because it's really important that you feel comfortable with it before we talk about the computational analysis of the output. So if anybody has any questions, now would be a great time to ask. Yes.

AUDIENCE: I have more of a scientific question. This assumes that we know a transcriptioned factor. Are there are ways, methods to figure out transcription factors so that you can design antibodies to bind to it?

PROFESSOR: So the question is, this assumes that we know the regulators that we're interested in ahead of time. And is there a de novo way of discovering heretofore unknown regulators that are binding to the genome? The answer to that question is sometimes, as is usually the case.

Later in the term, we'll talk about other methodologies for looking at the regulatory occupancy of the genome that don't depend upon immunopurification, in which case we'll get an understanding of what's going on with the genome at the level of knowing what particular sequences are occupied without knowing what's there. From the sequence, sometimes we can infer the family of the protein that is sitting there.

But in general, the holy grail of this, which has not really fully materialized, would be as follows, which is instead of purifying with an antibody and then sequencing, why not purify with a nucleic acid sequence and then do mass spec, to actually take the proteins off of the DNA, run them through mass spec, and figure out what's there.

And we and others have attempted this. And at times, you get good results. But mass spec is improving greatly, but it's till a fraught process with noise.

And there's a paper just published in *Nature Methods* late last year on something called the CRAPome. Have you heard of this paper before? It is all the junk you get when you run mass spec experiments.

And so when you run a mass spec experiment, you can just take all the stuff in the CRAPome out of it, and it actually helps you quite a bit. That gives you an idea what the state of the art of mass spec is. It's a little bit noisy.

But I think your question is great. I think we need to get there. We need to get to the place where we can take portions of the genome and run them through mass spec and figure out what is populating it de novo without having to know ahead of time. Any other questions? OK. Great.

So the figure on the slide up there also describes the ChIP-seq protocol. And I'll also say that some people believe this would never work. They actually thought that when you did the purification, you would just get so much background sequence that when you map it to the genome, you could never discern any signal whatsoever. And so there are lively debates about this until somebody actually made it work, and then the argument was over because it made everything else completely and totally obsolete. So it wasn't good to be on the wrong side of that argument, I'll tell you that. I wasn't, but all right. I was on the right side.

But suffice to say, here's a close-up picture of Mr. Protein-- Ms. Protein-- and what happens when there is breakage around that site, followed by sequencing. And as you can see, the little black lines connecting between the protein and the DNA is supposed to indicate contacts sites. And you can see the little yellow arrows are supposed to indicate breakage sites of the DNA that are being caused by, in this case, mechanical breakage through sonication.

And you get reads from both strands of the DNA. Remember that a sequencing instrument always sequences from five prime to three prime. So you're going to get the reads on the red strand and on the blue strand, shown here. And when we do the mapping, we know which strand they're mapped on. And the profile is shown in the lower plot, showing the density of map reads versus distance from where we believe the protein is sitting.

And the tag density refers to tags or sequence tags or reads that are aligned using the methodology we discussed two lectures ago to the genome we assembled last lecture. So the characteristic shape shown in this picture is something that is not the same for all proteins. It is something that varies from protein to protein. And thus, one of the things that we'll want to do during our discovery of where these proteins are binding is always learn the shape of the read distribution that will come out of a particular binding event.

So just to show you some actual data, so you'll get a feel for what we're talking about, this is actual data from the Oct4 protein, which is an embryonic regulator, pluripotency factor, binding to the mouse genome around the SOCS2 gene. And you can see the two distinct peaks on the upper track, both the plus strand reads and the minus strand reads, shown in blue and red respectively. Each one of the black and white bars at the top-- probably can't be read from the back of the room-- but each one of those is 1,000 bases, to give you some idea about sort of the scale of the genome that we're looking at here.

You can see the SOCS2 gene below. The exons are the solid bars. And then you see the whole cell extract channel, which we talked about earlier. And the whole cell extract channel is simply giving us a background set of reads that are nonspecific.

And so you might have a set of, say, 10 or 20 million reads, something like that, for when these experiments that you map to the genome and get a picture that looks like this. So now our job is to take the read sets that we see here, genome wide, and figure out every place that the Oct4 protein is binding to the genome.

Now there are several ways that we could approach this question. One way to approach the question would be to simply say, where are the peaks? And so you hear this kind of exploration often described as peak finding. Where can you find the peaks? And where is the middle of the peak?

Now the problem with this approach is that it works just fine when a peak represents a single binding event. So imagine that these two fingers here are binding events, and they're fairly far apart of the genome. Now as they come closer and closer and closer together, what will happen is that, instead of having two peaks, we're going to wind up having one broad peak.

And thus, there's a lot of biology present in this kind of binding of the same protein proximal to itself. So we need to be able to take these sorts of events that occur underneath a single enrichment, or single peak, into two separate binding events. And this is shown in the next couple of slides, right where we look at what we would expect from a single event, in terms of a read enrichment profile once it's aligned to the genome.

And we think about a possibility that there are two events, here shown in indiscernible gray and blue and red. And we note that each one of these will have its own specific profile. And then you can consider them to be added together to get the peak that we observe.

Now one of the reasons this additive property works is that, remember, we're working with a large population of cells, and regulators don't always occupy a site. And thus, what we're looking at in terms of the reads are the sum of all of the evidence from all of the cells. And so even though the proteins are close to one another, we often can find an additive effect between that proximal binding.

So how can we handle this? Well, what we're going to do is we're going to do two key algorithmic things. We're going to model the spatial distribution of the reads that come out of a specific event, as I suggested earlier. And we're going to keep that model up to date.

That is, we can learn that model by first running our method using a common distribution of reads, identify a bunch of events that we think are bindings of a single protein, take those events and use them to build a better model of what the redistribution looks like, and then run the algorithm again. And with the better distribution, we can do a much better job at resolving events out of multiple events out of single peaks.

And the next thing we're going to do is we're going to model the genome at a single base pair level. So we're going to consider every single base as being the center point of a protein binding event, and using that model, try and sort through how we could have observed the reads that we are presented with. And first, the spatial distribution that we build is going to look something like this.

And we consider a 400 base pair window, and we learn the distribution of reads, and we build an actual empirical distribution. We don't fit the distribution to it, but rather we can keep an exact distribution or histogram of what we observe, averaged over many, many events. So when we're fitting things, we have the best possible estimate. Yes.

AUDIENCE: I was just trying to remember to origin of the [INAUDIBLE] clearly. So within the protocol, are you sequencing with the proteins bound to these fragments?

PROFESSOR: No. See, you can't do that. You have to reverse the cross-linking from the DNA. And then there's a step here which we omitted for simplicity, which is, we amplified the DNA.

AUDIENCE: So I was just wondering, if there's no protein, why doesn't the polymerase just read through the whole thing from one side? Why is there a peak? There seems to be a loss of signal right where the protein is bound.

PROFESSOR: Well, that depends upon how long the reads are and how hard you fragment. And in fact, that can occur. But typically, we're using fairly short reads, like 35 base pair reads. And we're fragmenting the DNA to be, say, perhaps 200 to 300 base pairs along.

So we're reading the first 35 base pairs of the DNA fragment, but we're not really all the way through. We could read all the way through if we wanted, but there really wouldn't be a point to that. The thing that we're observing here is where the five prime end of the readers, where the leftmost edge of the read is. So even though it might be reading all the way through, we're just seeing the left edge of it. Does that answer your question?

AUDIENCE: Yeah.

PROFESSOR: OK, great. Any other questions? Yes, at the back.

AUDIENCE: So to clarify, this distribution that's being shown up here on both the positive and negative strand--

PROFESSOR: Yes.

AUDIENCE: This is the position of where the reads started, not the count of the number of times that particular base was shown in the sequencing result. Is that correct?

PROFESSOR: Let me repeat the question. What we're observing in the distribution is where the read starts and not the number of times that base shows up in the sequencing data. It is the number of reads whose five prime position start at that base. OK So each read only get one count. Does that help? OK.

The other thing that we're going to do, for simplicity, is we're going to assume that the plus and the minus strand distributions are symmetric. So we only learn one distribution, and then we flip it to do the minus strand. And that's shown here, where we can articulate this as this empirical distribution, where the probability or read given a base position is described in terms of the distance between where we are considering the binding of it may have occurred and where the read is.

So it's important for us to look at this in some detail so you're comfortable with it. Here's our genome again. And let's assume that we have a binding event at base m along the genome. And we have a read here, r sub n, at some position. The probability that this read was caused by this binding event can be described as probably a read n given the fact that we're considering an event at location m.

Now of course, it could be that there are other possible locations that have caused this read. And let us suppose that we model all of those positions along the genome as a vector pi. And each element of pi describes the probability or the strength of a binding event having occurred in a particular location. So we can now describe the probability of a read sub n given pi is equal to the summation where i equals 1 to big M, assuming that there is 1 to M bases in this genome, of a p rn given m pi m, like this.

So we are mixing together here all of the positions along the genome to try and explain this read. So the probably of the read, given this vector pi, which describes all the possible events that could have created this read, is this formulation, which considers the probability of each position times the probability that an event occurred at that position subject to the constraint that all of a pi i's sum to 1. So we're just assigning probability mass along the genome from whence all of the reads originally came.

So this is considering a single read, r sub n, and where that might have originated from. Yes.

AUDIENCE: Does the constraint basically state that only one event occurred in this fragment point?

PROFESSOR: The question is, does this constraint imply that only one event occurred? No. The constraint is implying that we're going to only have one unit of probability mass to assign along the genome which will generate all of the reads. And thus, this vector describes the contribution of each base to the reads that we observe.

So let us say simplistically that it might be that there are only two events in the genome, and we had a perfect solution. Two points in the genome, like m1 and m2, would have 0.5 as their values for pi. And all the other values in the genome would be 0. We're going to try to make this as sparse as possible, as many zeroes as possible.

So only at the places in the genome where protein is actually binding will pi i be nonzero. Does that make sense? These are great questions. Yes. And if people could say their names first, that would be great. Yes.

AUDIENCE: Sara. Just to clarify, the pi vector is completely empirical?

PROFESSOR: This distribution is completely empirical. Yes, that's right, Sara, completely empirical. I'll also say, just so you know, that there are many ways of doing this kind of discovery, as you might imagine. The way we're going to describe today was a way that was selected as part of the ENCODE 3 pipeline for the government's ENCODE project.

And so what I'm going to talk about today is the methodology that's being used for the next set of data for ENCODE 3 followed by IDR analysis, which is also part of ENCODE 3. So what you're hearing about today is a pipeline that's being used that will be published next year as part of the ENCODE project.

These papers, this method's been published. But the analysis of all the Encyclopedia of DNA Elements-- which is what Encode stands for-- the third phase of that is utilizing this. OK, any other questions? Yes.

AUDIENCE: Does the shape of this binding event tell you anything about the topology of the actual protein?

PROFESSOR: It does, actually. And we'll return to that. But the shape of this binding can tell you something about the class of protein, which is still an area of active research. But also note that that is a little bit confounded by the fact that when you have homotypic binding, which means you have these closely spaced binding events, you get these broader peaks. And so there's a lot of research into what the shapes on the genome mean and what biological function of mechanism they might imply. Yes.

AUDIENCE: Can you explain pi one more time?

PROFESSOR: Yeah, explain pi one more time, sure. So pi is describing where there are binding events along the genome. So for example, if we just had two binding events, m1 and m2, then pi of m1 would be equal to 0.5, and pi of m2 would be equal to 0.5, and all the other values of pi would be 0. So we're just describing with pi where the events are occurring. That gives us the location of the events. OK?

And you'll see in a moment why we articulated it that way. But that's what pi is doing. Does that answer your question? OK. Any other questions? Yes.

AUDIENCE: In cases when you have two peaks really close together, to a point that there's some sort of [INAUDIBLE] between, how do you constrain your pi?

PROFESSOR: How do you constrain the pi when you have closely spaced peaks?

AUDIENCE: Yeah.

PROFESSOR: Well, you don't constrain it. You actually want--

AUDIENCE: [INAUDIBLE]

PROFESSOR: Well, I'm going to show you some actual examples of this algorithm running. So I'm going to give you an animation of the algorithm running, so you can actually watch what it does. And you'll see, first it's going to be something that isn't too pleasant, and then we'll fix it.

But the key thing here is sparsity. What we want to do is we want to enforce pi being as sparse as possible to explain the data. One of the common problems in any approach to machine learning is that, with a suitably complex model, you can model anything, but it's not necessarily interpretable.

So what we want to do here is to make pi as simple as possible to be able to explain the data that we see. However, if a single event cannot explain and observe redistribution at a particular point in the genome, we'll need to bring another event in.

All right, so that is how to think about a single read. And now we can just say the probability of our entire read set given pi is quite simple. It's simply the product over all the reads. Sorry. Like so.

So this is the probability of the entire read set. So we had this previously, which is the probability of a single read. And we take the product of the probability for each individual read to get the likelihood of all the reads.

So now all we need to do to solve this problem is this. We need to say that pi is equal to the arg max pi of P R pi, which gives us the maximum likelihood estimate for pi. Now it's easy to write that down, just find the setting for pi that maximizes the likelihood of the observed reads, and proof, you're done, because now you've come up with a pi that describes where the binding events are along the genome at single base pair solution, assuming that pi is modeling every single base. Does everybody see that? Any questions about that? Yes.

AUDIENCE: [INAUDIBLE]

PROFESSOR: n is the number of reads. m is a number of bases in the genome. n is the number of reads.

AUDIENCE: The length of pi, right? Is that equal to the number of binding events? Or is it equal to just the number of [INAUDIBLE]?

PROFESSOR: The length of pi?

AUDIENCE: Yeah.

PROFESSOR: It's the number of bases in the genome, and hopefully the number of bindings is much, much, much smaller than that. Typical number of binding events is 5 to 30,000 across a genome that has 3 billion bases, something like that. So it's much, much smaller. OK?

So this is what we would like to solve. And another way to look at this model, which it may be somewhat more confusing, is as follows, which is that we have these events spread along the genome. And we have the reads shown on the lower line. And the events are generating the reads.

So this is called a generative model, because the derivation of the likelihood directly follows from, if we knew where the events were, we could exactly come up with the best solution for the assignment of pi, and thus for the likelihood. So this is what we have on the board over here. But we can solve this directly. Yes. Question in the back.

AUDIENCE: Hi. My name is Eric. I have a little question. What is the definition of GPS here?

PROFESSOR: Oh, sorry. Yeah, GPS is the name of this algorithm. I told it to an editor and they hated it. But at any rate, it's called the genome positioning system.

[LAUGHTER]

Yeah, so you don't like it either, right, Eric? Oh, well. But yes, it locates proteins on the genome, right? Yeah. Good question.

So this is going to be, I think, our first introduction to the EM algorithm as a way of solving complex problems like this. And here is the insight we're going to use to solve this problem, which is that imagine I do the function g. So g is going to be this function that tells us what reads came from which events and where those events are in the genome.

So if you knew g exactly, this would be a really trivial problem to solve. It would tell you, for every single read, which particular binding event caused it. And if we knew that, it would be great, because then we could say something like this.

We could say, well, we knew g. Then the number of reeds created by a binding event at location m would simply be-- it's not a very good summation sign here-- we would sum over all the reads and count up the number of them where read n was caused by an event at location m. And we just summed up this number. We'd have the number of reads caused by an event at location m.

Is everybody cool with that? No. Do you see the definition of there of g on the overhead projector? So every time a read n comes from event m, that function is going to be equal to 1. So we just count the number of times it's 1. For a given location m, we find out how many reads came from that event-- could be 5, could be 10, could be 100.

We don't know exactly how many were generated by that particular event, but there's going to be some number. And we have 10 million reads, and we have 100,000 events, we're going to get about 100 reads per event, something like that. You can give that kind of order of magnitude.

Everybody OK? Any questions about the details of this? Because the next step is going to be causing leaps of faith, so I want to make sure that everybody's on firm ground before we take the leap together. Yes.

AUDIENCE: So you are in forced sparsity yet?

PROFESSOR: I'm not in forced sparsity yet. You like sparsity. You're going to keep me to that, right? So later on, you'll hold me to it, all right? That's your job. What's your name?

AUDIENCE: [INAUDIBLE]

PROFESSOR: OK. That's your job, all right? Mr. Sparsity. I like that. He's been sparse. I like it.

So if this is the number of reads assigned to a particular location, then we know that pi sub m will simply be n sub m over the summation of all of the reads assigned to all of the events. And some of these are going to be zero. A lot of them will be zero.

So here, I don't know this assignment of reads to events. It's latent. It's something I have invented which I do not know. But if I did know it, I'd be able to compute pi directly, like so, because I'd be able to figure out for each location on the genome how many reads were there. And that's how much responsibility that location had for generating all of the reads. Is everybody with me so far? Any questions at all? OK.

So remember, this is latent. We don't know what it is, but if we did know, we'd be in great shape. So our job now is to estimate g. And if we could estimate g, we can estimate pi. And once we get a better estimate for pi, I'd like to suggest to you we can get a better estimate for g. And we can go between these two steps.

And so that's what the expectation, maximization algorithm does in the lower part here, which is that the left part is estimating gamma, which is our estimate for g, which is looking at the number of reads that we think are softly assigned to a location m over the total number of softly assigned reads from all locations. And that gives the fraction of a read at n assigned to event m.

So we are computing an estimate of g. But to compute this estimate, we have to have pi. It's telling us where we believe that these reads are coming from. And once we have gamma, which is an estimate, we can compute pi, just in the same way we computed it here, if we knew g.

So we compute the expectation of this latent function, which we don't know, and then we maximize pi to maximize our likelihood. And you can derive this directly by taking the log of that likelihood probability up there and maximizing it. But suffice to say, you wind up with this formulation in the EM framework.

So I can go between these two different steps-- the E step, the M step, the E step, the M step. And each time as I go through it, I get a better and better approximation for pi, until I ultimately get it within some preset level of convergence. And then I'm finished. I actually can report my pi.

But before I leave the EM algorithm, and this formulation of it, are there any questions at all about what's going on here? Yes, question in the back.

AUDIENCE: So the solution to the EM algorithm doesn't always [INAUDIBLE]?

PROFESSOR: It gives you a solution. And it is the optimum solution given the starting conditions that we give it.

AUDIENCE: But it depends on the starting conditions, right?

PROFESSOR: It does depend on the starting conditions. Any other questions? Yes, in the back.

AUDIENCE: Is this dependent upon you having discrete binding events? So if you have a protein that has more diffuse localization across the genome, for example, RNA Pol II, would this model break down?

PROFESSOR: So the question is, does this depend upon you having discrete binding events? What biological assumptions are we making in this model about whether or not a protein is actually fixed and welded to the genome in a particular location or whether or not it's drifting around? And this might be of particular interest, for example, if we're looking at histones, which are known to slide up and down the genome, and we're immunoprecipitating them. What will this give us?

We are making the assumption that we're dealing with proteins that have punctate binding properties, that actually are point binding proteins. And thus what we're going to get out of this, as you'll see, is going to be a single location. There are other methodologies, which I won't describe today, for essentially deconvolving the motion of a protein on the genome and coming up with its middle or mean position while allowing it to move. But today's algorithm is designed for point binding proteins. Good question, OK?

All right. So just to be clear, what we're going to do is we're going to initialize everything, such that we'll begin by initializing pi to be 1 over the size of the genome. And at the very end of this, the strength of a binding event will be simply the number of reads assigned to that event by summing up our estimate of g.

And the nice thing about this is that because the number of reads assigned to an event, in some sense, is scaled relative to the total number of reads in the experiment, we can algorithmically take our genome and chop it up into independent pieces and process them all in parallel, which is what this methodology does. All right.

So let's have a look at what happens when we run this algorithm. So here we are. This is synthetic data. I have events planted at 500 and 550 base pairs. The x-axis is between 0 and 1,400 base pairs. And the y-axis is pi from 0 to 1.

So here we go. It's running. And this is one base pair resolution. And it is still running-- and stop. Well, we get sort of a fuzzy mess there, don't we? And the reason we're getting a fuzzy mess there is that we have got a lot of reads we've created, and there are a lot of different genome positions that are claiming some responsibility for this, which my friend, Mr. Sparsity, doesn't like, right? We need to clean this up somehow.

And let's see what happens when we actually look at actual data. So here is that original data I showed you for the two Oct4 binding events next to the SOCS2 gene. And I'll run the algorithm on these data. And you can see it working away here.

And once again, it's giving us a spread of events, and you can see it even beginning to fill in along where there's noise in the genome and stopped here. And it's assigning the probability mass of pi all over the place. It is not sparse at all.

So does anybody have any suggestions for how to fix this? We've worked all might. We've got our algorithm running. We thought it was going to be totally great. When we run it, it gives us this unfortunate, smeary kind of result. What could we do? Any ideas at all? Yes.

AUDIENCE: Add a prior where most of the reads are.

PROFESSOR: Add a prior. You saw the word no prior, a little tip off. Yeah, absolutely, good point. Add a prior.

So what we're going to do is we're going to try and add a prior that's going to prejudice pi to be 0 to create sparsity. So we like pi being 0. So we'll add what's called a negative Dirichlet prior, which looks like this, which is the probability of pi there is proportional to 1 over pi to the m raised to the alpha power. And as pi gets smaller, that gets much, much bigger.

And thus, if we add this prior, which happens to be a very convenient prior to use, we can force pi to be 0 in many cases. And when we take that prior into account and we compute the maximum a posteriori values for pi, the update rules are very similar, except that what's interesting is that the rule on the left for computing our estimate of g is identical. But on the right, what we do is we take the number of reads that we observe at a particular location and we subtract alpha from it, which was that exponent in our prior.

And what that simply means is that if you don't have alpha reads, you're history. You're going to get eliminated. So this is going to do something called component elimination. If you don't have enough strength, you're going to get zapped to zero.

Now you don't want to do this too aggressively early on. You want to let things sort of percolate for a while. So what this algorithm does is it doesn't eliminate all of the components that don't have alpha reads at the outset. It lets it run for a little while. But it provides you with a way of ensuring that components get eliminated and set to zero when they actually don't have enough support in the read set.

So once again, all we're going to do is we're going to add this prior on pi, which we will wind up multiplying times that top equation. We'll get the joint probability of r and pi in this case, and then we're going to maximize it using this EM adaptation. And when we do so, let me show you what happens to our first example that we had.

OK, that's much cleaner, right? We actually only have two events popping out of this at the right locations. All right. So that's looking good. And the probability is summed to 1, which we like. And then we'll run this on the Oct4 data. And you can see now, instead of getting the mushing around those locations, what's going to happen is that most of the probability mass is going to be absorbed into just a couple components around those binding events.

Now another way to test this is to ask whether or not, if we looked at what we believe are closely spaced homotypic events, in this case, for CTCF, with that prior, we still can recover those two events being next to each other in the genome. And so if we run this, you can see it running along here.

Each one of these iterations, by the way, is an EM step. And there you go. And you can see that even from the sparse data you see above-- that's the actual data being used, all those little bars up there, read counts for the five prime ends of the reads-- we can recover the position of those binding events. Question, yes.

AUDIENCE: [INAUDIBLE]

PROFESSOR: It depends upon the number of reads, but it's somewhere around three or four. I mean, if you look at the GPS paper, which I posted, it tells you how alpha is set. Yes.

AUDIENCE: Do you use wholesale extract data when you are trying to compute?

PROFESSOR: Question, do we use wholesale extract data? I told you all how important it was, and I haven't mentioned it again. We're about to get to that, because that's very important. I'm glad you asked that question. Yes.

AUDIENCE: It looks like, along the genome coordinates, each event is only one base pare, but binding is usually more than one base pair. So is that the center of the binding?

PROFESSOR: Yes. Well, it's the center of the read distribution function. Whether or not it's the center of the binding of the protein, I really can't tell. OK, great.

And I'll just point out that a power of this method is its ability to take apart piles of reads like this and to these so-called homotypic binding events, where you can see where the motif is. And you can see this method will take apart that pile of reads into two independent events, whereas other popular methods, of which there are many for analyzing these type of ChIP-seq data, don't take apart the event into multiple events, because most of them make the assumption that one pile of reads equals one binding event.

Now back to our question about wholesale extract. What happens if you get to a part of the genome that looks like this? We're going to go back to our Oct4 data. And we have our Oct4 IP track, which is above, and then our wholesale extract tract is down there on the bottom.

What would you say? Would you say those are really Oct4 binding events? Or do you think something else is going on there? If so, what might it be? Any ideas about what's going on here? Yes.

AUDIENCE: Regions with a lot of repeats in the genome?

PROFESSOR: Regions with repeats in the genome? Yes. In fact, you can see repeats annotated in green right there. And as you recall from last time when we talked about assembly, it's very difficult to actually get the number of repeat elements correct. And a consequence of that is if you underestimate the number of repeat elements in the genome and you collapse them, when you do sequencing of the genome and you remap it, you're going to get regions of the genome were a lot of reads pile up, because they're actually coming from many different places in the genome, but they only have one place to align.

And these towers of reads, as they're called, can create all sorts of artifacts. So in order to judge the significant of events, what we'll do is a two-step process. First, we will run our discovery pipeline and discover where all the events are in the IP channel without regard to the wholesale extract channel, in this particular case. And then we will filter the events and say which of the events are real in view of the data from the wholesale extract channel.

And so the way to represent this is what is the likelihood we would have seen those IP reads at random given the wholesale extract channel? So how can we formulate this problem? Well imagine that we take all of the reads that we see in both channels and we add them together. So we get a total number of reads at a particular location in the genome.

If, in fact, they were going to occur at random, we'd be doing a coin flip. Some reads will go on the IP channel. Some reads will go on the wholesale extract channel. So now we can ask what's the probability we observed greater than a certain number of reads on the IP channel at chance, assuming a coin flipping model where we've added the reads together.

Another way to view that is what's the chance that we have observed less than a certain number of reads in the wholesale extract channel assuming a coin flipping model? And that will give us a p-value under the null hypothesis that, in fact, there is no binding event and simply what's going on is that we have reads being flipped between the two channels out of a pool of the total number of reads.

And if we take that approach, we can formulate it as the binomial in the following way. We can compute a p-value for a given binding event using the data from both the IP channel and from the control channel. And the way that we do this is that we look at the number of reads assigned to an event, because we already know that number. We've computed that number.

And we take a similar window around the control channel. We count the number of reads in it and ask whether or not the reads in the IP channel is significantly larger enough than the reads in the control channel to reject the null hypothesis that, in fact, they occurred at random.

So this is the way that we compute the p-values for events. So once we've computed the p-values for events, we still have our multiple hypothesis correction problem, which is that if we have 100,000 events, we need to set our p-value's cut off and appropriate value. And I think that you may have heard about this before in recitation. But the way that this is done in this system is to use a Benjamini-Hochberg correction.

And the essential idea is that we take all of the p-values and we organize them from smallest to largest. And we're going to take a set of the top events, say 1 through 4, and call them significant. And we call them significant under the constraint that p-value sub i is less than or equal to i over n, where n is the total number of events times alpha, which is our desired false discovery rate.

So this allows us in a principled way to select the events that we believe are significant up to a false discovery rate, which might be, for example, 0.05 for something that's typically used for a false discovery rate. So we've talked about how to discover events, how to judge their significance individually, how to take a ranked list of events from a given experiment and determine which ones are significant up to a desired false discovery rate.

The false discovery means that, let's say, that we have 1,000 events that come out. If this false discovery rate was 0.1, we would expect 100 of them to be false positives. So it's the fraction of positives that we detect that we think are going to be false. And we can set that to be whatever we want.

Now let's talk about the analysis of these sort of data. There's a question up here. Yes.

AUDIENCE: So this randomness, does that correct to the fact that in your previous slide the coin isn't necessarily fair?

PROFESSOR: Oh, the null hypothesis assumed that the coin was fair, that reads could either occur. That in this case, for example, that the reads were either occurring in the control channel or in the IP channel with equal likelihood, assuming because they were coming from the same process, which was not related to the IP itself. They're coming from some underlying noise process.

AUDIENCE: So typically you'd have some fixed number of reads, and so in the IP channel, the bulk of your reads was in your peaks. Then you would have fewer in your--

PROFESSOR: So the question is, how do you normalize for the number of reads and how they're being spent? Because in the IP channel, you're spending the reeds on IP enriched events. In the wholesale extract channel, you're not spending the reads on those, so you have more reads to go around for bad things, so to speak.

So what this algorithm does-- which I'm glad you brought it up-- is it takes regions of the genome that are event free, and it matches them up against one another and fits a model against them to be able to scale the control reads so that the number of reads in the IP and extract channel are the same in the regions that are free of events. And so it matches things, so it is 0.5. OK?

AUDIENCE: Even if wasn't 0.5, wouldn't this rank list correct with that?

PROFESSOR: No, it would not. Let us suppose that we made a horrible mistake and that we let these events through because our p was wrong. They would have a very large number of reads, and they would be very significant. As a consequence, they would float up to the top of this list as having the lowest p-values, or the most significant, and they would pop through this. So we would report them as binding events, which is not desirable.

These are all great questions. Any other questions at all? So we've talked about this.

The next question you might have would be, you've done two replicates of the same chip. You always want to do two of every experiment, if not three or four. And I know it's expensive, but you don't know where you are if you only do one of something. Trust me.

So assuming you've done two experiments that were identical ChIP experiments, and you would like to know whether or not the results between those two experiments are concordant. How might you go about determining this? Does anybody have any ideas?

Let me suggest something to you. Imagine that we create two lists-- one for experiment x and one for experiment y. And we're going to put in this list, in rank order, the strongest events and where they occur in the genome. And let us suppose that we are able to match events across experiments when they're in the same location.

So what we'll do is we'll say Event one occurs here. Event 13 occurs here and here. Event 9 occurs here and here. Event 10 is matched. Event 11 occurs here. Event 57 occurs here.

And these are ranked in order of significance. They're not completely concordant, and they're ranked according to the significance in this particular experiment and this particular experiment. And we would like to know whether or not we think these two experiments are good replicates of one another.

Now the nice thing about converting this into a rank test is that we are no longer sensitive to the number of reads or any specific numeric values represented by each one of these events. We're simply ranking them in terms of their importance and asking whether or not they appear to be quite similar.

So one way to do this is simply to take a rank correlation, which is a correlation between the ranks of identical events in the two lists. And so, for example, imagine in the lower right-hand corner of this slide shows you an example of X and Y values, although there's not really a linear relationship between them, and the Pearson correlation coefficient is 0.88.

If we look at the right correlation, which is defined by the equation on the left, which is really simply the correlation between their rank values, the Spearman correlation for rank is 1. They're perfectly matched. So if you want to consider whether or not two experiments are very similar, I'd suggest you consider rank based tests. But we can do even more than this.

Imagine that we would like to know at what point along this list things are no longer concordant, that we would like to consider all the events that are consistent among these two experiments. And we assume that there is a portion of the experiment results that are concordant and part that is discordant. And we want to learn where the boundary is.

So what we'll do is this. We will look at the number of events that are concordant up to a particular point. Let's see here, make sure we get the parametrization right. So if we have lists that are n long, psi of n of t is the number of events that are paired in the top n times t events.

So t is simply the fraction. If is 0.25, that would mean that in the top 25% of the events, the number of events that are paired. Actually, it's the fraction of events. Sorry.

So assuming that we had perfect replicates, what we would see would be that psi of n of t would look like this. Yes?

AUDIENCE: What is t?

PROFESSOR: t is the fraction of the events that we're considering. So this t might be equal to 0.25. If this was 10, then this would be-- oh, sorry. I think this t equal 0.5. If n was equal to 10, this would be the first five events. So t is the fraction. n is the total number of events.

So if this is t, which is the fraction which goes from 0 to 1, and this is psi of n of t, this would be a perfectly matched set of replicates. A 0.5, 0.5 of the events are matched, which is perfect. Can't do any better than that.

So this is simply telling us, as we take larger and larger fractions of our event, what fraction of them are matched up to that point? Question.

AUDIENCE: So this assumes that the rank order is important for the correlation, right? The higher rank they are, we expect them to be better correlated. For example, if in the middle they were the best correlated, but not the top or bottom in terms of the ranking, then you might have a weird looking--

PROFESSOR: Yeah. The question is, doesn't this assume that we care most about the events up here, as opposed to the events, for example, in the middle. And this analysis is done to consider how many of the top events we take that we think are consistent across replicates.

So it starts at the top as a consequence of that assumption. This is the definition of psi n, which is the fraction of the top events that are paired in the top events. It's roughly linear from the point where events are no longer reproducible. And psi prime of n is the derivative of that function.

And graphically, if we look at these, we can see that there's perfect correspondence on the left point up to some point. And then the hypothesis of this irreproducible discovery rate is at that point things stop being in correspondence because you've hit the noise part of the event population. and you get to junk. And as soon as you get to junk, things fall apart.

And what this methodology does is it attempts to fit the distribution to both the correspondence and the non-correspondence parts of the population, and you get to set a parameter, which is the probability that you're willing to accept something that's part of the junk population of the non-correspondence population.

And the way this is used, for example, in the ENCODE project is that all ChIP-seq experiments are done in parallel. Event discovery is done independently on them. All of the events from each independent experiment are ranked from 1 to n. And then IDR is done on the results to figure out which of the events are consistent across replicates. And obviously, if you've had a very bad replicate, you get a very small number of events, if any at all. But if you have good replicates, then you get a large number of events.

A secondary question is, imagine you had processing algorithm A and processing algorithm B. It might be that processing algorithm A typically gave you more reproducible events than processing algorithm B, that whenever you ran A, you could get much further down this curve before things fell apart. Does that mean that A is necessarily better, or not? What do you think? Would you always pick algorithm A if that was the case? Any insight on this? Yes?

AUDIENCE: No, probably, because there's experimental considerations to it.

PROFESSOR: Yeah. I would go with [INAUDIBLE] experimental considerations. But the reason you might not pick it is it might be that if algorithm A always gave you the same wrong answers in the same order, it would score perfectly on this, right? It doesn't mean that it's right. It just means it's consistent.

So for example, it could be calling all the towers as events, and it could always give you, for every experiment, the same results. In fact, one interesting thing to do is to run this kind of analysis against your experiment, and an outlier shouldn't match your experiment. It should fail. It's always good to run negative controls when you're doing data analysis.

So this IDR analysis simply allows you to pick alpha, which is the probability of the rate of repairs for the irreproducible part of the mixture that you're willing to accept. And the way this is used is, as I said, is that here you see different ways of analyzing ChIP-seq data. And the little vertical tick mark indicates how many peaks they get before they can't go any further because they hit the IDR bound alpha. And some methodologies can produce far more events consistently than other methods can.

So that's just something for you to keep in mind. And I will not have time to present all the rest of this. But I did want to point out one thing to you, which is that the methodology we talked about today is able to resolve where things are in the genome at exceptional spatial resolution, within 20 base pairs genome wide. And it's always run genome wide at single base pair solution.

And it can do things like compute for pairs of transcription factors that have been profiled using ChIP-seq, the spacing between them for spacings that are significant. And so you can see that we're beginning to understand the regulatory grammar of the genome in terms of the way factors organize together and they interact to implement the combinatorial switches that' we've talked about.

That's it for today. You guys have been totally great. We'll see you on Tuesday for RNA-seq. And have a great weekend until then.