So before joining UCLA in 2013, Production work. On the script is include differentiation factors, asymmetric (indistinct) replication, p-value-free false discovery (indistinct), and a high dimensional variable selection. And on the bio (indistinct) application side, her research include all single cell (indistinct) genomics and (indistinct). Research published. She’s an MIT Technology Review certified in 2020, and she has received from Harvard. So it’s my pleasure to talk about some of our recent work related to how statistic rigor is important in genomics. So I want to say that when I was a student,
especially I think most of our audience here are students,
I want to give you this motivation.
When I was a student back in 2007,
that was when I just started my PhD
and I was interested in bioinformatics.
I had a lot of questions about bioinformatics methods
after I took statistics classes.
I think some of the questions I listed here
include are P values valid?
Because P values are so widely used
in genomics bioinformatics.
And also, we have a lot
of bio bioinformatics methods developed for data analysis.
And I wonder why don’t we use classical statistical methods
in textbooks?
And the third thing is,
when we use statistical test to understand the question,
to answer some pivot question,
what is the proper null hypothesis?
So you will see those questions in the topics
I will talk about next.
So this talk will focus on the multiple testing problem.
See, multiple testing, what it means
is that we have multiple hypothesis tests,
and the criteria we use in this problem are P values,
which we have one P value per test.
So we know that the requirement for a valid P value
is that P values should follow the uniform distribution
between zero one under the null hypothesis.
Or we may relax this to be super uniform.

Just for your information, super uniform means that the P values have higher density toward one and lower density towards zero.

So that’s still okay for type one error control, even though you may have a larger than expected type two error.

So given the many, many P values, we need one criterion to set a cutoff on the P values. And the most commonly used criterion for multiple testing correction is called a false discovery rate, short as FPR.

So the definition here is the expectation of this ratio, and this ratio is the number of false discoveries over the number of discoveries.

So this notation means the maximum between the number of discoveries and one.

In other words, we don’t allow the denominator to be zero, if we don’t make any discovery.

So this is to avoid the dividing zero issue.

And this ratio has a name called false discovery proportion.

In other words, we can have this proportion for one particular data set.

However, as you know, we don’t observe this ratio because we don’t know which discoveries are false.

So therefore, this ratio is only a hypothetical concept, but not really computable.

And here, the expectation
0:04:31.92 –> 0:04:35.1 is taken over all possible data set
0:04:35.1 –> 0:04:38.13 from the same distribution as our data set.
0:04:38.13 –> 0:04:40.26 So this is the frequentist concept
0:04:40.26 –> 0:04:43.98 because we have imaginary potential data sets.
0:04:43.98 –> 0:04:46.95 So therefore, the phenomena paper
0:04:46.95 –> 0:04:49.83 by Benjamini and Hochburg gave us a way
0:04:49.83 –> 0:04:53.28 to control this expectation called FDR
0:04:53.28 –> 0:04:56.73 under a claimed level, say, 5%,
0:04:56.73 –> 0:05:00.6 even though we couldn’t realize this ratio itself.
0:05:00.6 –> 0:05:02.4 But we could control its expectation.
0:05:02.4 –> 0:05:04.83 So that’s the magic of statistics.
0:05:04.83 –> 0:05:07.02 So Benjamini Hochburg algorithm allows us
0:05:07.02 –> 0:05:11.19 to set a cutoff on the P values to control the FDR.
0:05:11.19 –> 0:05:14.79 But I want to emphasize that the FDS’s only controlled
0:05:14.79 –> 0:05:17.28 when P values satisfy this assumption,
0:05:17.28 –> 0:05:19.32 otherwise, it may not be.
0:05:19.32 –> 0:05:24.32 So I want to say three common causes of ill-posed P
values,
0:05:24.36 –> 0:05:27.48 which make P values don’t satisfy this assumption
0:05:27.48 –> 0:05:30.217 in genomics, and I’ll go through them one by one.
0:05:31.11 –> 0:05:34.17 The first issue is what I call the formulation
0:05:34.17 –> 0:05:37.4 of a two sample test problem as a one sample test.
0:05:37.4 –> 0:05:39.06 What does this mean?
0:05:39.06 –> 0:05:42.09 So I will use the common genomic analysis
0:05:42.09 –> 0:05:44.67 of ChIP-seq data as an example.
0:05:44.67 –> 0:05:45.99 So in ChIP-seq data,
0:05:45.99 –> 0:05:50.16 we want to measure where a protein binds in the
genome.
0:05:50.16 –> 0:05:53.52 So you can consider the X axis as the genome
0:05:53.52 –> 0:05:56.79 and the Y axis as the protein binding intensity
0:05:56.79 –> 0:05:58.68 measured by ChIP-seq.
So here, we have experimental sample, the condition of our interest, say, a certain cell line. And the background sample is what we know that's no protein, so there should be no protein binding. But we still want to measure the noise from the experiment. So we need this contrast. And here, we want to say that the region in the red box, this interval, we want to call it as a peak, if we see the intensity in the experimental sample is much larger than the intensity in the background sample. So we do the comparison and we want to cut this at a peak. That's the purpose of this analysis. And I wanna say that, in the field, because ChIP-seq has become popular since 2008, Macs and Homer are probably the two most popular software for cutting peaks. Even though they have very complex procedures for processing the sequencing data that in a statistical part to call a region as a peak or not, I can say, their formulation is as follows. Given a region, we count its number of ChIP-seq reads in the background sample and in the experimental sample. So let's just summarize this intensity as a count, a count here, a count here, and both are now negative. So I call the background count as big X,
experimental count as big Y.

And in our data, we have the observations, right?

We refer to them as small x, small y.

Then, the P value in both software

is essentially this probability, the probability

that big Y is greater or equal than the observed small y,

where the big Y follows upon some distribution

with mean parameter as the small x.

Now, when I look at this formula back in 2008,

the Macs paper, I wonder whether this is correct.

And I don’t think so.

Because the reason, if you look at it,

is what is the null hypothesis?

The null hypothesis is essentially, okay,

let’s assume the experimental count

is our test statistic, okay?

We assume it follows a Poisson distribution

with mean lambda.

And here, the null hypothesis is lambda is equal to small x.

Alternative is lambda greater than small x.

So what’s the problem with here?

Essentially, we are using small x as a fixed parameter

instead of a random observation.

So in other words,

the randomness in the background count is ignored.

We only consider experimental count as the random variable.

So in other words, where use the two sample testing problem

to a one sample testing problem
because we only consider the randomness

But this is not something our textbook teaches us.

The reason is because if we consider background

as one condition, experimental has another condition,

under each condition, our sample size is only one.

So therefore, the T test will not apply

because a central limit here clearly doesn’t apply.

So how do we calculate P value, any ideas?

I think one possibility that we could still assume

Poisson distribution for both background X

and experimental Y.

You have two Poisson, under the independence,

we can probably derive the distribution

for Y minus X, right, and what’s the null distribution.

That’s the only way.

But, if you think about it, how can we verify

whether the Poisson distribution is reasonable?

You only have one observation from it.

The distribution could be anything, right?

So assuming a parametric distribution seems quite,

I will say, aggressive.

So I think P value calculation is challenging here.

And also, I even wonder, in this case,

for this one versus one comparison,

should we use a P value?

Or is this really a testing problem that’s feasible?

So I would say, over the years, I gradually realized

that here we looked at many, many regions,

not just one region.

So the goal or the criterion that’s ultimately used
0:10:08.16 → 0:10:09.24 is actually FDR.
0:10:09.24 → 0:10:12.18 And in this process,
0:10:12.18 → 0:10:15.96 P values are just intermediate for FDR control,
0:10:15.96 → 0:10:18.18 instead of our final target.
0:10:18.18 → 0:10:21.09 So do we have to stick with P values?
0:10:21.09 → 0:10:25.11 This motivated me to write this paper with my students
0:10:25.11 → 0:10:30.11 to propose a way to achieve p-value-free FDR control
0:10:30.18 → 0:10:34.23 by leveraging the theory in Barber and Candès paper,
0:10:34.23 → 0:10:35.61 their knockoff paper,
0:10:35.61 → 0:10:38.58 so we could actually doing FDR control
0:10:38.58 → 0:10:41.19 in this example without using P value.
0:10:41.19 → 0:10:43.17 So I will talk about this later in my talk,
0:10:43.17 → 0:10:46.83 but this is one motivation for the Clipper paper.
0:10:46.83 → 0:10:49.95 The second issue with P values is that we observe,
0:10:49.95 → 0:10:51.68 sometimes, P values are not valid
0:10:51.68 → 0:10:56.68 because the parametric model used may not fit the data well.
0:10:57 → 0:11:00.6 So this is an example for this commonly used
0:11:00.6 → 0:11:04.62 differential expression analysis on RNA sequencing data.
0:11:04.62 → 0:11:06.6 So for this task,
0:11:06.6 → 0:11:09.75 the two popular softwares are DESeq2 and edgeR.
0:11:09.75 → 0:11:12.42 So the data usually looks like this.
0:11:12.42 → 0:11:15.39 So we want to compare two conditions
0:11:15.39 → 0:11:18.78 and seeing which genes are differentially expressed.
0:11:18.78 → 0:11:21.63 So condition one, we have three samples,
0:11:21.63 → 0:11:23.4 which we cause to replicate,
0:11:23.4 → 0:11:25.41 condition two, three replicates.
0:11:25.41 → 0:11:29.07 So every row is one replicate,
0:11:29.07 → 0:11:31.17 while every column is one gene.
0:11:31.17 → 0:11:33.9 So to call a gene as differentially expressed,
0:11:33.9 → 0:11:36 we need to compare its three values
from condition one, two, three values from condition two.

So clearly, we can see the left one may be a D gene, the right one may not be a D gene, right?

That’s our intuition.

And we want to make this more formal by doing a statistical test.

But in both edgeR and DESeq2, you can see that to compensate the small sample size, like three versus three,

they assume a gene follows a negative binomial distribution under each condition.

So essentially, these three values are assumed to follow one negative binomial distribution.

These three values follow another negative binomial distribution.

And the null hypothesis is the two negative binomial distributions have the same mean, that’s the problem.

Okay, so we actually discovered an issue with popular methods from this data set.

And thanks to my collaborator Dr. Wei Li who is a computation biologist at UC Irvine.

So actually, from this patient data, we have a much larger sample size, 51 patients before the treatment of some immunotherapy medicine,

58 patients on treatment.

So we want to compare the RNA sequencing data of these two groups of patients.

So essentially, when we apply DESeq2 or edgeR to this data,
0:13:00.84 –> 0:13:05.253 the red dots indicate the number of D genes identified.
0:13:06.3 –> 0:13:11.3 To verify whether we can still identify D genes
0:13:11.64 –> 0:13:12.84 from permuted data,
0:13:12.84 –> 0:13:15.15 because the reason is that we want to see
0:13:15.15 –> 0:13:18.78 whether the permuted data is actually really,
0:13:18.78 –> 0:13:20.22 because we know the permuted data
0:13:20.22 –> 0:13:21.84 shouldn’t give us any signals.
0:13:21.84 –> 0:13:23.82 If we just disrupt the two groups,
0:13:23.82 –> 0:13:25.83 we shouldn’t expect any D genes.
0:13:25.83 –> 0:13:29.07 But surprisingly, we found that each method
0:13:29.07 –> 0:13:33.54 can identify sometimes even more D genes from per-
0:13:33.54 –> 0:13:37.23 muted data.
0:13:37.23 –> 0:13:39.43 So the bar and the error bars show the distribution
0:13:39.43 –> 0:13:40.35 of D genes identified from permuted data.
0:13:40.35 –> 0:13:43.71 So this is something quite unexpected.
0:13:43.71 –> 0:13:46.53 And to look into the reason, our first thought
0:13:46.53 –> 0:13:49.92 is to check the negative binomial assumption.
0:13:49.92 –> 0:13:51.78 Because now, under each group,
0:13:51.78 –> 0:13:54.9 we have 51 and 58 sample sizes,
0:13:54.9 –> 0:13:58.68 so we could check the distribution, and here’s what we get.
0:13:58.68 –> 0:14:02.753 You see that for the genes that are frequently identified
0:14:02.753 –> 0:14:06.99 from permuted data, if we run the goodness-of-fit test,
0:14:06.99 –> 0:14:09.87 we check the negative binomial distribution,
0:14:09.87 –> 0:14:12.36 these genes have very small P values,
0:14:12.36 –> 0:14:15.09 indicating that this fit is not good.
0:14:15.09 –> 0:14:16.35 Well, if you look at the genes
0:14:16.35 –> 0:14:19.95 that are rarely identified from permuted data,
0:14:19.95 –> 0:14:22.86 the P values are bigger and the goodness-of-fit is better.
0:14:22.86 –> 0:14:25.2 So we do see this relationship
0:14:25.2 –> 0:14:28.74 between the goodness-of-fit of negative binomial
and the frequency that a gene is identified from permuted data. So negative binomial model seems to not fit well on this patient data. Because here, the 51 patients shouldn’t be regarded as replicates, they’re not experimental replicates, they are individuals. So therefore, the theory for deriving negative binomials usually assume as a Gamma-Poisson Mixture model, Gamma-Poisson Hierarchical model. That one may no longer hold, and that’s why we think the parametric model is not applicable to this patient data. So what’s the consequence, right? So we want to convince the scientist what’s the consequence of doing this analysis in this problematic way. We show that if we just use the D genes found by DESeq2 and edgeR, which are the genes corresponding to the red dot, around the so called gene oncology analysis, that is to check which functional terms are enriched in those two gene sets, we can see many functional terms are related to immune functions. Which would suggest that if we trust these two methods’ results, we may conclude that, yes, between the two groups of patients, there are differences in immune responses, right? That seems to confirm our scientific hypothesis. However, now, we see many of these genes
were also identified from permuted data, then, that will make the results dubious. So what we tried is that, even the sample size is so large, we tried the classical Wilcoxon rank sign test, which everybody learned, right? So non parametric two sample test that doesn’t assume a parametric distribution. And here, it’s self consistent, it doesn’t identify D genes from real data, but also, it doesn’t identify D genes from permuted data. So there’s no contradiction here. And this result motivated me to ask this question, which I had years ago, should we always use popular bioinformatics tools? Like, check the citation of these two methods, super highly cited. Should I reuse popular method or should we consider general statistical methods, like Wilcoxon. So our recommendation is sample size matters, right? We may have different methods suitable for different sample sizes, and essentially, why statistics has so many methods, paramedic, non parametric, is because we have different scenarios in our data. That’s the first thing we should realize. It’s not like one method can do all the things. And the second thing is sanity check. We should always consider doing some sanity check
to make sure we trust the results
instead of just take the results for granted.
So these things were summarized in our paper
published earlier this year.
And since its publication,
we have received a lot of discussions on Twitter,
if you are interested.
But anyway, so it means that many people are interested
especially many people, users believe
that popular bioinformatics tools are the state-of-the-art,
right, the way, standard methods.
But if you are bio statisticians, you may not like this.
Because we want to develop new methods.
Otherwise, what’s our job, right?
So in this case, we need to really find the loopholes,
the limitations, or the gap between current approach
and the data scenarios,
and try convinces people that, yes,
we do need careful thoughts when we choose method.
It’s not always one method.
And a related question is,
in Wilcoxon, definitely doesn’t have a strong assumption,
and (indistinct) have a reasonable power
when the sample size is large.
But what if sample sizes are small, right?
So when it’s small, we know,
non parametric tests like Wilcoxon doesn’t have power.
So in this case, we actually proposed Clipper again,
so it can work as a downstream correction tool
0:18:34.05 –> 0:18:36.3 for DESeq2 and edgeR.

0:18:36.3 –> 0:18:38.7 Because they are supposed to be quite powerful,

0:18:38.7 –> 0:18:41.01 even though they find probably too many.

0:18:41.01 –> 0:18:44.19 So hopefully, we could use that to borrow their power,

0:18:44.19 –> 0:18:47.31 but help them improve the FDR control.

0:18:47.31 –> 0:18:50.31 So I’ll show the results later in my talk.

0:18:50.31 –> 0:18:51.63 That’s the second cause.

0:18:51.63 –> 0:18:53.76 And the third cause for ill-posed P values

0:18:53.76 –> 0:18:55.95 is a little more complicated.

0:18:55.95 –> 0:18:59.67 And this is the issue commonly observed in single cell data,

0:18:59.67 –> 0:19:01.08 single cell RNA-seq data.

0:19:01.08 –> 0:19:02.91 So I will use this analysis

0:19:02.91 –> 0:19:07.91 called pseudotime differentially expressed genes as example.

0:19:08.19 –> 0:19:09.858 What is a pseudotime?

0:19:09.858 –> 0:19:13.11 Pseudotime means it’s not real time, it’s pseudo, right?

0:19:13.11 –> 0:19:15.72 So it’s something we inferred

0:19:15.72 –> 0:19:17.67 from single cell RNA-seq data,

0:19:17.67 –> 0:19:20.43 so those cells are measured all at once.

0:19:20.43 –> 0:19:25.43 But we want to infer some time trajectory from the cells.

0:19:25.92 –> 0:19:28.83 So I’ll just use the screenshot from Slingshot,

0:19:28.83 –> 0:19:33.83 which is a method for inferring pseudotime for explanation.

0:19:34.05 –> 0:19:39.05 So here, this is a two-dimensional PCA plot of cells,

0:19:39.18 –> 0:19:41.28 and the cells are pre-clustered,

0:19:41.28 –> 0:19:44.25 so each color represents one cluster.

0:19:44.25 –> 0:19:47.1 So the Slingshot algorithm does the following,

0:19:47.1 –> 0:19:50.61 first, it takes the cluster means’ centers,

0:19:50.61 –> 0:19:52.77 and connect them using the algorithm
called minimum spanning tree.

So if you’re not familiar with that, it has an equivalence with hierarchical clustering actually.

So with the minimum spanning tree, you get this tree, and then, they smooth out the tree using principle curves.

So we have two curves, and then for every cell, we find the closest curve and project the cell to the curve.

So therefore, in each curve, the projections are called pseudotime values.

And usually, it’s normalized between zero and one, so we need to find the root and call it zero, the other end is called one.

The whole process is called pseudotime inference.

In other words, after it, we will give every cell a pseudotime value in each trajectory.

Okay, so one thing I want to emphasize is that in this pseudotime inference, we used gene expression values already.

So it’s not like we observe pseudotime as external variable, but it’s from the same data.

So I want to show what we could do after the pseudotime.

So a typical analysis is to identify which genes are differentially expressed along the pseudotime.

Like the left one, we see, it has this upward trajectory, so we may call it differentially expressed.
And here, we want to say the pseudotime represent some cell immune response, and this is an immuno-related gene, so we expect to see the upward trajectory. For the right gene, we expect to see something constant, so we don’t want to come right (indistinct) a D gene, that’s the intuition. And I want to say that we must realize, pseudotime values are random simply because the cells is a random sample, right? We need to consider randomness, and we want to show this to people by doing subsampling. So you can see that sampling variation would get into pseudotime values. Here, every row is a cell. If I randomly subsample, say, 80% of cells from the left cells and redo the pseudotime trajectory inference, we can see that for the cells in the subsamples that include it, its values will vary to some degree. So it’s not a constant. Okay, so realizing this, we should consider the randomness of pseudotime from the data. However, existing methods all treat pseudotime as an observed covariate. So our goal here is to fix this, and we proposed this method called PseudotimeDE, which actually does the inference, which infers whether one gene is differentially expressed along pseudotime,
0:22:36.45 –> 0:22:40.62 So what we did exactly is that, here,
0:22:40.62 –> 0:22:43.83 to see whether a gene changes with pseudotime,
0:22:43.83 –> 0:22:45.48 what’s the intuition?
0:22:45.48 –> 0:22:48.27 We should do regression, right, do a regression analysis
0:22:48.27 –> 0:22:52.53 by treating a gene’s expression value as Y,
0:22:52.53 –> 0:22:54.66 pseudotime as X, and regular regression.
0:22:54.66 –> 0:22:57.54 Yeah, this is exactly what existing methods did.
0:22:57.54 –> 0:22:59.43 And to make sure the regression
0:22:59.43 –> 0:23:01.74 is not restricted to be linear,
0:23:01.74 –> 0:23:04.92 and also account for that the gene expression values
0:23:04.92 –> 0:23:06.57 are non negative counts.
0:23:06.57 –> 0:23:11.57 So actually, we choose the generalized additive model,
0:23:11.61 –> 0:23:14.1 which is also used in an existing method,
0:23:14.1 –> 0:23:15.75 which I will show very soon.
0:23:15.75 –> 0:23:20.4 So this is a very flexible and interpretable model.
0:23:20.4 –> 0:23:24.06 So generalized means Y can be non Gaussian
0:23:24.06 –> 0:23:25.23 and the other distribution,
0:23:25.23 –> 0:23:27.51 just like generalized linear model.
0:23:27.51 –> 0:23:28.74 But additive means
0:23:28.74 –> 0:23:32.34 that we make the linear model more general,
0:23:32.34 –> 0:23:36.553 so every feature can be non linearly transformed,
0:23:37.86 –> 0:23:41.55 but the features after transformations are still added.
0:23:41.55 –> 0:23:44.34 So that’s additive, short as GAM.
0:23:44.34 –> 0:23:47.31 So essentially, once we have a set of cells,
0:23:47.31 –> 0:23:49.35 we first infer the pseudotime,
0:23:49.35 –> 0:23:52.17 so we order the cells along the pseudotime,
0:23:52.17 –> 0:23:53.43 and for gene J,
0:23:53.43 –> 0:23:56.55 we check how the gene changes with pseudotime,
0:23:56.55 –> 0:23:59.76 so we run the generalized additive model
0:23:59.76 –> 0:24:01.5 to obtain a test statistic.
Please know that generalized additive model has its theory, so we could use the theory to calculate to use the null distribution and calculate P value. And that was done in an existing method. We want say that this may be problematic because this whole null distribution considers pseudotime to be fixed. So to address this, we need to consider pseudotime inference as part of our test statistic calculation. So to do this, we use the top part. We actually do subsampling of the cells. The reason we didn’t do bootstrap is simply because we want the method to be flexible for pseudotime inference method. Like I show here, there are Slingshot, Monocle3, and a few others. We want it to be flexible, and some methods don’t allow cells to be repetitive, so bootstrap doesn’t apply here. And we use subsampling with percentage pretty high, like 80%, 90%, and we did a robustness analysis. And then, on each subsample, we do pseudotime inference. With this, how do we get a null distribution of the test statistic? What we did is to permute the cells, so any relationship between the gene J and the pseudotime is disrupted. So this can be considered from the null,
and then, we did the same GAM model, and then, we calculate the values of the test statistic on these permuted subsamples, that gave us a null distribution. So together, we can get a P value, this is what we did. And we can show that this approach indeed can control the P values, make the P values uniformly distributed on the null, while the existing method that uses GAM, but only the theoretical distribution called tradeSeq, they have some distortion for P values. And then, you may wonder, what’s the consequence? We can show that, oh, and I should say, Monocle3 uses generalized linear model and not uncertainty. So you can see that even though it’s not as bad as tradeSeq, still, some distortion. So we wanna show that by calibrating the P value using our way we can actually discover more functional terms in our differentially expressed genes. It means that we can find some new biological functions that were missed by this new method. Which shows that FDR control not just help with FDR control of P value calibration, not just help with FDR control, but may also boost some power. So I just quickly talk about this PseudotimeDE, but I want to say that its computational time is the biggest limitation.
Because here, our P value calculation requires many rounds of subsampling, pseudotime inference, and permutation.

So let’s say we want the P value with resolution 0.001, we need at least 1000 rounds of such things, right? That will take time.

So the natural question is can we reduce the number of rounds, right, and still achieve FDR control? That becomes similar to my first goal. Can we get rid of the higher resolution P values, control the FDR, and then, we will use Clipper again.

Clipper is used throughout all the motivations, that’s why we proposed it, and I’ll talk about it in the next minute.

And the second question we didn’t address is that what if the cells don’t follow a trajectory at all? So clearly in our null hypothesis, we are assuming there is a trajectory, it’s just that gene J doesn’t change with the trajectory. But what if the trajectory doesn’t exist?

So this whole idea of this trajectory pseudotime inference doesn’t make sense, right?

We need to consider that. But I don’t think we have a good way to do it, unless we can change the cells to have a null where the cells don’t follow a trajectory.

So this motivated us to generate cells that don’t follow a trajectory, and we used a simulator.
So which it will be the last part I will talk about today. Okay, PseudotimeDE is one such a problem where pseudotime is inferred from the same data. Another common problem is to do clustering on single cells to identify cell clusters, and between cell clusters, we identify differentially expressed genes. We call this problem ClusterDE. But this is also using the data twice, right? So people have called this term double dipping, meaning that the same data used for twice. To tackle this problem, we need to consider the uncertainty in cell clustering, and there are three existing papers that try to address this problem. They either need to assume a distribution, like genes follow Gaussian distribution in every cluster or every gene follows a Poisson distribution here and they need to do count splitting. So I won’t talk into the couple of details here, but I just want to say that the count splitting approach in my opinion tackles a different problem. It is conditional on the observed data matrix, rather than considered to be random. But I will not talk about the detail here. So motivated by the challenge in this problem, and we want to propose something not distribution-specific. We want to use our simulator to generate the null data.
and then use Clipper to achieve the FDR control.
So we want to do this non parametrically.
So I think the idea was motivated
by two phenomenal statistical papers.
One is the gap statistic paper,
which was proposed to find the number of clusters
in the clustering problem.
And if you read a paper, I think the smart idea there
is they try to generate data points without clusters
as the negative control.
Then, you can control your number of clusters
with some statistic,
versus what if there’s no clusters, right,
and do the comparison and find the gap.
That’s the gap statistic.
And knockoffs gave the theoretical foundation
for FDR control without using high resolution P values.
Okay, so the halftime summary
is that I talked about three common causes
of ill-posed P values.
Hopefully, I have convinced you
that we need something to avoid this problem.
So I talked about Clipper,
the p-value-free FDR control for genomic feature screening.
And as I said, it was motivated and enabled
by the FDR control procedure from this paper.
But the difference here is that we focus
on marginal screening of interesting features.
So in other words, we look at one feature at a time.
In my previous examples, a feature could be a region or a gene. So in the original knockoff paper, their goal is to generate knockoff data just like fake data for multiple features jointly. And that’s the very challenging part. But in our case, we don’t need that because we are looking at one feature at a time, so it’s not a multi-varied problem, but it’s a marginal screening problem. So our goal is to get rid of high resolution P values. So the advantage of this is we don’t need parametric distribution assumptions, or we don’t need large sample sizes to enable non parametric tests, these are not needed. We just need to summarize every feature into a contrast score, and then, set a cutoff on the contrast scores. So what do I mean by contrast score? So every feature, say, I have total d features, they have C contrast scores shown as C1 to Cd, so I’m calling the histogram of the distribution of contrast scores. So if the theoretical assumption is satisfied, then the features that are null features should follow a symmetrical distribution around the zero, okay? And for the features that are interesting should be large and positive on the right tail.
So the theory of the FDR control just says, we can find the contrast score cutoff as \( t \), such that this ratio is controlled under \( q \). We ought to find the minimum \( t \) for this. What this means is can you can consider this ratio as a rough estimator of FDR. So the denominator is just the left tail, the red part plus one, sorry, the numerator is the right tail plus one, the denominator is the, sorry, the left tail is, sorry, the numerator is the left tail plus one, the denominator is the right tail with maximum with one. So in other words, still trying to avoid dividing zero. And the idea is that we want to find a threshold \( t \), so that the right tail will be called discoveries and the left tail represent false discoveries. That’s the intuition. Because we know, if the feature’s null, then it will be randomly positive or negative. And the sign is independent of the absolute value. So that just replaces the uniform distribution requirement for \( P \) values, we change that to symmetry. And another thing is that the feature, if it’s large positive, we want to discover it, right? So this will be the discovery set and this represents the negative, false discovery set. So that’s the idea intuition behind this approach. But the theory to really prove it, we need to use Martingale in probability to prove it.
And some of the technique was used for the Benjamini Hochburg procedure still based on Martingale.

So anyway, this allows us to really control the FDR just using contrast scores.

And another thing I found as appealing is that if you visually inspect the contract scores, you can see whether the assumption seems to be reasonable because you expect to see something symmetrical plus a heavy right tail.

Okay, so we are currently writing to make this more formal, so we could actually check whether the assumption is reasonably holding.

So with this approach, we can make a lot of the comparison analysis easier because the key is to find a reasonable contrast score that satisfies this assumption.

And I can say that there may be multiple contrast scores that satisfy, not just the unique one.

Then the difference is power, right? So we may have a better power if you have a heavier right tail.

Okay, so for a ChIP-seq peak calling analysis, we can say that the contrast score will be comparing the target data from experimental condition to the null data, which is the background condition.

They serve a natural pair of contrast, and we could apply any pipeline to each data,
the same pipeline and then do the contrast, right?
You can imagine, if there’s no peak,
then these two values will be,
which one is bigger is equally likely.
And for the RNA-seq analysis,
here, I showed we could use permuted data as the null data.
actual data as a target data.
So if we run some test on actual data
to get a test statistic,
we use the same test on permuted data
to get a test statistic, and they serve as a contrast.
And finally, for the PseudotimeDE and ClusterDE,
the single cell problem,
actual data will give us some comparison,
either PseudotimeDE
or the between ClusterDE test statistic.
And if we have some similar data
that represents the null,
like null trajectory, null cluster,
we could run the same pipeline and then do the contrast.
So you see, this actually free us
from saying we need to derive P values
and we need to know the distribution
by either theory or by numerical simulation, right?
These are all relieved
because we just need to do a contrast.
And the power is gained from the many, many tests,
we look at them together.
So that’s why
this idea (background noise drowns out speaker).
Okay, so as I said, we tried to implement Clipper as a way to improve FDR control, and we did achieve this for the popular software Macs and Homer, with inflation. So you see that they did have inflated FDR, so the Y axis is the actual FDR, with our Clipper as an add-on to be used downstream of what they output and do the contrast, we can largely reduce the FDR to the target and still maintain quite good power. So that’s the usage of Clipper as an add-on. And for the single cell part, I didn’t finish about the null data generation. How do we do it? Our simulator was proposed partly for this reason, but it has more uses. So I just want to say that it’s called scDesign3 because we have scDesign and scDesign2 as two previous work. Now, focus on scDesign2 because it is the direct predecessor of scDesign3. So what scDesign2 two does is it tries to fit a multi-gene probabilistic model for each cell type, and then, every gene assumes to follow a parametric distribution within the cell type.
And the major contribution is that we capture gene-gene correlations using Gaussian copula. That will make the data more realistic. Here is the comparison. This is the real data used for fitting the model. This is the lab (indistinct) test data used for validation, and this is the synthetic cells using copula. If we remove the copula, the cells will look like this. So not realistic at all. And our data is more realistic than other simulators that did not explicitly capture gene-gene correlation. Although, they have some implicit mechanism, but the model is different. Okay, so we realize that scDesign2 is doing a good job for displaying cell types, but it cannot generate data like this from a continuous trajectory. What we could do is to force the cells to be divided and then use scDesign2. But then, you can see the cells are kind of in clusters, right, not in real data. But with our generalization to the version three, we now can generate cells from a continuous trajectory. And I can quickly say that we basically generalize this, this count distribution per cell type to a generalized additive model, which I already said. So we could make it more flexible in general, and scDesign2 becomes a special case of scDesign3. And one more thing we could do is we actually use the technique vine copula,
so we could get the likelihood of how the model fits to the real data, so we can get the likelihood of the model, which can also give us more information. So besides the single cell trajectory data, we can also use the idea to generate spatial data. So here, the modification is that for every gene we assume a Gaussian process in the 2D space, so it can have a smooth function for (indistinct) expression (indistinct).

And also, my other student help with making the simulator to generate reads, sequencing reads, not just counts. So we can go from counts to reads, and this will give us more functionality to benchmark some low level tools. So in short, the scDesign3 simulator has two functionalities. One is to do, of course, simulation. We can generate single cell data from cell types, discrete, continuous trajectories, or even in the spatial domain.

We could generate feature modalities we call multi-omics, including RNA-seq, ATAC-seq, which is a technology for open chromatin measurement, CITE-seq, which includes both protein and RNA, and also DNA methylation. These are the examples we tried, but we could do even more. We could allow it to generate data with experimental designs.
including sample covariate, conditions, or even batches. So these can make us generate cases for more types of benchmarking. And for interpreting real data, scDesign3 can give us model parameters, so we can know whether a gene has different means in two cell types, whether a gene has a certain change on a pseudotime, or a gene has a certain change in two dimensional space. And also, as I said, we can output a likelihood that can give us a way to calculate the basic information criterion BIC, so we could evaluate whether some pseudotime describes data well, whether the algorithm for pseudotime inference does a good job, or whether the clusters explain data well. So these are the things we could do. And finally, to generate the null data for the Clipper (indistinct), we can alter the model parameters. Like this is what we fit from real data, we could change the model parameters to make the gene no longer differentially expressed, have the same mean in two subtypes. Or, after we fit a real data with two cell types or two clusters, we could change the cluster parameter to make sure the cells come from one cluster instead of two clusters. So these are the things we could do with the model.
And so this is how our paper, but more details are in our paper, which has been posted, if you are interested. And I want to just quickly show how the ClusterDE analysis could be done. This is the real data with two clusters. I want to say that this is the case where permutation wouldn’t work. If you just permute the cluster labels, the cells will look like the same cells, right? They’re still two clusters. But if you use our simulator, we could generate cells from one cluster that reflects the complete null, there’s no cluster. And the use of this can be shown in this example. There’s only one cluster, but if we use clustering algorithms, like these two choices, Seurat is a popular pipeline, Kmeans is the standard classical algorithm, using either to force the cells into two clusters, we are using gene expression data. So no wonder that if you look at a gene’s expression between the two clusters, you may call it DE, but that’s not interesting, since there’s no clusters. So if we use scDesign3 to generate null data, in this case, null data should be very similar to real data. It still has only one cluster. Then, if we run Seurat or Kmeans, similarly, on null data,
we would divide the cell in a similar way, and then, if you do a contrast of the two sets of results, you should see no big difference. That’s the idea for controlling FDR.

So indeed, in that example, if we’re just naively wrong, the Seurat pipeline clustering followed by some tests like t, Wilcoxon, bimodal, yeah, you will see FDR is one. The reason is you keep finding D genes, even though there’s no cluster. But using our approach, we could control the FDR reasonably well. So that’s the predominant results for this purpose for this task, so that summarizes my talk today.

And finally, I just want to make a few notes to give some messages. I talk about multiple testing, but in many scientific problems, I think the key is whether it should be formulated as a multiple testing problem. So actually, to address this question, I wrote a prospective article with my collaborator Xin Tong at USC. We try to clarify statistical hypothesis testing from machine learning binary classification. They seem similar because both would give you a binary decision, right? But I can say that testing is an inference problem, classification is a prediction problem. So if you really think about it, their fundamental concepts are different.
0:44:27 –> 0:44:30.9 So that’s why we wrote this to really talk with biologists, computational people who use this simultaneously. If you’re interested, you can check it out. And finally, I wanna say that, so if it’s a multiple testing problem, I talked about three common causes of ill-posed P values, and I propose a solution, Clipper, for simplifying this problem by just using contrast scores, and then, set a cutoff. And the simulator, which we hope to be useful for the single cell and spatial omics field because this field is so popular, we have more than 1000 methods already. So benchmarking seems to be something quite necessary. Because if there’s no benchmarking, maybe new methods wouldn’t have much of a chance. Because people may still use the older method that are better cited. Okay, so these are the papers related to my talk today. And so, finally, I want to say that, so if you’re interested, you want to check them out, and let me know if you have any questions. So finally, I’ll just say this, this is something quite interesting. It’s another paper we just recently wrote, and I can say, you should be online in genome biology very soon. So we actually did this benchmark.
0:45:43.11 –> 0:45:47.22 for the so called QTL analysis in genetics, right?
0:45:47.22 –> 0:45:49.77 Quantitative Trait Locus mapping.
0:45:49.77 –> 0:45:51.33 So in this analysis,
0:45:51.33 –> 0:45:55.32 a common procedure is to infer hidden variables
0:45:55.32 –> 0:45:57.36 from the data, like genes expression matrix,
0:45:57.36 –> 0:46:00.06 want to do hidden variable improvements.
0:46:00.06 –> 0:46:03.39 Besides the most part, (indistinct) has the classical
PCA,
0:46:03.39 –> 0:46:06.69 several methods propose specific (indistinct).
0:46:06.69 –> 0:46:09.99 And my student Heather, actually gave her the full
credit,
0:46:09.99 –> 0:46:12.93 she was so careful and she really wanted to understand
0:46:12.93 –> 0:46:14.4 the method before using it,
0:46:14.4 –> 0:46:16.56 then that lead to this project.
0:46:16.56 –> 0:46:19.35 She wants to see, huh, do I really see advantages
0:46:19.35 –> 0:46:22.29 of this new method even compared to PCA?
0:46:22.29 –> 0:46:23.88 But that’s what she found, right?
0:46:23.88 –> 0:46:26.37 PCA still seems to be the most stable,
0:46:26.37 –> 0:46:29.61 robust, and also faster algorithm,
0:46:29.61 –> 0:46:32.4 but this is one of the reviewer’s comments
0:46:32.4 –> 0:46:34.05 I wanna share with you.
0:46:34.05 –> 0:46:36.52 These results may come as a surprise to some,
0:46:36.52 –> 0:46:39.45 given the nearly un-contestable status
0:46:39.45 –> 0:46:42.06 that method A has achieved within the community.
0:46:42.06 –> 0:46:43.8 But sadly, they reflect the fact
0:46:43.8 –> 0:46:46.74 that computational biology methods can rise to fame
0:46:46.74 –> 0:46:50.28 almost by accident rather than by sound statistic arg-
ments.
0:46:50.28 –> 0:46:51.57 So if you’re interest,
0:46:51.57 –> 0:46:53.91 you can check out this paper, it’s on bio archive.
0:46:53.91 –> 0:46:56.58 But anyway, I think it says how important it is
0:46:56.58 –> 0:46:59.88 for statisticians to convey our message, right?
Why do we need statistical rigor, why does it matter?
So for our students, if you want to know more about GAM and copulas, there are two books I want to recommend. They’re very good introductory textbooks, so you can know the (indistinct).
Finally, I want to thank my collaborator at UC Irvine, my students for all their tremendous work I talk about today and also the funding agencies for giving us the support. So thank you very much.
A question?
Yes. So I was really curious about the analysis of like the large patient sample. I know that there has in fact been extensive discussion on it. Yeah, yeah. Which is interesting, to say the least, how it’s gone down. But I was kinda curious, the way that it was presented here made me think about like, apologies, if this is like a path that’s already been tread, so, yeah, the bar graph. Yeah. Yeah, so it sort of, it makes me wonder about the application of the term false discovery in different contexts. And taking patients, you can imagine, there can be like unintended structure within those populations.
And by (interference drowns out speaker) chance, if there is 30,000 potential transcripts that you’re looking at, there might actually be, between individuals who are not isogenic, truly differentially expressed genes between even permuted groups. And so I’m wondering if there’s a useful distinction between a false discovery and a true, but uninteresting discovery. I think it depends on how you define truth. I think that’s the key. What is the definition of D genes? To be exact, the definition of D genes in DESeq2, edgeR, and that of Wilcoxon is different. Because in Wilcoxon, the D gene is defined, okay, if a gene, it has two distributions, one under each condition, and if I randomly take one observation from each distribution from each condition, is the chance that one is bigger than the other equal to 0.5? That’s the Wilcoxon question. While DESeq2 and edgeR, their D gene definition is the negative binomial means are different. But clearly, you can see, it only depends on that negative binomial is a reasonable distribution, that’s the key. So that’s why in theory, if negative binomial is no longer valid or reasonable, then why should we define a D gene
Based on negative binomial mean indifference? I think that’s kind of my answer to your question. But the tricky thing about statistical inference compared to supervised learning is that we don’t observe the truth, that’s always the case. So we’re making a guess. Frequentist people have one way to guess, Poisson people have another way of guess. And so one issue I’ve seen in the Twitter discussion is that several people try to, maybe not intentionally, confuse frequentist concept with Poisson concept, but they’re not really comparable, right? You cannot talk about them in the same ground. That’s a problem, and here, our criterion, false discovery rate is a frequentist criterion, it relies on P values, right? So therefore, you cannot use Poisson arguments to argue against such frequentist way. Because you are doing frequentist, right? But whether frequentist makes sense or not, that’s a different topic. Hopefully, that answers your question. Yes. Hello,

thank you much for your talk, and I think that is very interesting. However, I have a question on slide 26 actually. It’s about what you said that, maybe 26.
Okay, yeah. Yeah, you said that like it is a multi-gene probabilistic model for cell type. However, I’m a little bit confused about how you define the cell type. But basically, from my own understandings, that after you get, for example, the single cell rise data, you will use the route to get the cluster. Yeah. And you will annotate this cluster based on the- Knowledge, yeah. Gene. And then, if this model based on your annotation of, okay. Essentially, yeah, we need cell cluster to be pre-defined. So if it’s not reasonable, then, yes, it will affect the results for sure. Because the key is that you need to make sure it is reasonable to assume a gene follows one of the four distribution within a cluster, right? So that’s why there are methods that try to refine clustering by checking the negative binomial distribution. So there are several research on that, and they’re trying to refine that. But basically, we are sitting on those methods to do the simulation, that’s what we do.
But again, so that’s why this is the problem with scDesign2, but scDesign3 sort of tries to address this problem by providing the BIC. So if the input clusters are bad, then you can see that in the BIC. Because the likelihood will not be there, yeah. A similar question.

I have another question is that basically I assumed (indistinct) about the experiments have duplicates, however, in some situations, maybe we do not have the replication. But in this situation, how could we control the FDR, if we do not have replicates, then we cannot get the P value. That’s exactly the point of this talk. The only part that has replicates is the RNA-seq part. The second part, that’s the only part we have replicates. In the first part, when we do the ChIP-seq, it’s just one replicate per condition, right? That’s why I said P value calculation would be helpful. Right, so the reason we could control the FDR without using P values is just because we have many, many tests. So that’s why we’re doing this large scale testing. I think the idea, if you check it out, Bran Efron has talked about it extensively in his book, it’s called, so his idea of Empirical Bayes is very similar to this. We try to borrow information across tests.
0:54:10.71 –> 0:54:12.505 to set a threshold.
0:54:12.505 –> 0:54:14.758 Yeah, hopefully that answers your question.
0:54:14.758 –> 0:54:15.591 Yeah?
0:54:15.591 –> 0:54:19.758 (interference drowns out speaker)
0:54:20.821 –> 0:54:22.657 Yeah, sounds good, thank you.
0:54:22.657 –> 0:54:25.649 (interference drowns out speaker)
0:54:25.649 –> 0:54:26.482 Thank you.