WEBVTT

1 00:00:03.580 --> 00:00:06.090 - Right, so I think while we're waiting,

2 00:00:06.090 --> 00:00:09.473 I'll just give a very brief introduction about Jingshu.

3 00:00:10.520 --> 00:00:14.200 Jingshu is an Assistant Professor from the Stats Department

4 00:00:14.200 --> 00:00:16.850 at University of Chicago.

5 00:00:16.850 --> 00:00:21.650 And today she's going to present some very exciting work

6 00:00:21.650 --> 00:00:25.233 on trajectory inference for the single cell data.

7 00:00:26.700 --> 00:00:28.893 I'm very excited to hear about her work.

8 00:00:30.990 --> 00:00:31.823 I think...

9 00:00:33.500 --> 00:00:35.230 Lets wait for two minutes

10 00:00:35.230 --> 00:00:39.720 and then we'll start with Jingshu's work.

11 00:00:39.720 --> 00:00:43.747 So if you have any other related questions about Jingshu

12 00:00:44.590 --> 00:00:48.320 before the talk start, you're free to ask as well.

13 00:00:48.320 --> 00:00:50.730 (chuckles)

14 00:00:50.730 --> 00:00:53.050 - Hi Lynn. Can you make me a co-host.

15 00:00:53.050 --> 00:00:56.423 - Oh right. Thanks for reminding me.

16 00:00:58.260 --> 00:00:59.093 Let me see.

17 00:01:13.858 --> 00:01:16.191 So one more minute to start.

18 00:01:41.502 --> 00:01:43.150 - So you can see my screen, right?

19 00:01:43.150 --> 00:01:46.593 - Yes. I can see your screen. Looks good to me.

20 00:01:47.680 --> 00:01:50.333 Maybe I'll hand it over to you now Jingshu I think,

21 00:01:52.227 --> 00:01:54.193 if (indistinct) late I think,

22 00:01:55.810 --> 00:01:58.160 they can ask questions if you miss any details.

23 00:01:59.340 --> 00:02:01.380 - Yes. Okay.

24 00:02:01.380 --> 00:02:03.900 So thanks everyone for coming

25 00:02:03.900 --> 00:02:07.523 and settling for the introduction invitation.

26 00:02:08.410 --> 00:02:12.690 Today I will talk about the Single-Cell RNA Sequencing Data

27 00:02:12.690 --> 00:02:15.610 and how we can learn the cell dynamics

28 00:02:15.610 --> 00:02:18.433 from the single-cell RNA sequencing.

29 00:02:23.110 --> 00:02:26.240 So the single-cell RNA sequencing is a relatively,

30 00:02:26.240 --> 00:02:29.840 is a newly development, newly-developed

31 00:02:29.840 --> 00:02:32.800 but also relatively mature technology

32 00:02:32.800 --> 00:02:37.800 for measuring the RNA expression levels in the cells.

33 00:02:37.890 --> 00:02:42.890 And the traditional microarrays or bulk RNA sequencing

34 00:02:43.300 --> 00:02:45.250 matches the gene expressions

35 00:02:45.250 --> 00:02:49.150 as the average across all cells in a tissue.

36 00:02:49.150 --> 00:02:53.560 However, a cell is made of many cells and, Oh, sorry.

37 00:02:53.560 --> 00:02:55.680 A tissue is made of many cells

38 00:02:55.680 --> 00:02:58.940 and the cell population is typically not homogenous

39 00:02:59.830 --> 00:03:02.050 and the cells can have different functions

40 00:03:02.050 --> 00:03:03.810 and different cell types.

41 00:03:03.810 --> 00:03:08.010 So in contrast, in single-cell RNA sequencing,

42 00:03:08.010 --> 00:03:11.570 we have measured the transcriptional profile

43 00:03:11.570 --> 00:03:13.960 in each individual cell.

44 00:03:13.960 --> 00:03:16.130 So we can expand this vector

45 00:03:16.130 --> 00:03:19.410 of gene expressions for a tissue to a matrix

46 00:03:19.410 --> 00:03:23.490 of the gene inspections in the cells.

47 00:03:23.490 --> 00:03:26.310 And each entry is a mattered RNA count

48 00:03:27.520 --> 00:03:31.060 for a particular gene or a particular cell.

49 00:03:31.060 --> 00:03:31.893 So that's...

50 00:03:31.893 --> 00:03:34.360 So the benefit is that we have no,

51 00:03:34.360 --> 00:03:38.550 we have a more detailed understanding of what is going on

52 00:03:39.610 --> 00:03:41.030 in the tissue.

53 00:03:41.030 --> 00:03:44.160 So the benefit of single-cell RNA sequencing,

54 00:03:44.160 --> 00:03:47.820 is that it can give you a relatively unbiased

55 00:03:47.820 --> 00:03:51.300 and complete picture of the cell population.

56 00:03:51.300 --> 00:03:54.290 And this is particularly useful

57 00:03:54.290 --> 00:03:57.290 when the cell population is complicated.

58 00:03:57.290 --> 00:04:02.203 For example when the cells are experiencing dynamic changes.

59 00:04:03.560 --> 00:04:06.390 And as an application of the method

60 00:04:06.390 --> 00:04:11.390 that I will introduce today in this lecture, in this talk,

61 00:04:11.460 --> 00:04:16.367 I will focus on the study of the mouse neocortex.

62 00:04:17.840 --> 00:04:22.840 This is a cartoon showing the migration and generation

63 00:04:22.930 --> 00:04:27.590 of the projection neurons in the mouse neocortex.

64 00:04:27.590 --> 00:04:32.590 Yeah, you guys see that this is quite a complicated process

65 00:04:32.857 --> 00:04:36.540 and there are still a lot of things that are unknown

66 00:04:36.540 --> 00:04:40.390 about the neuronal diversity and the mechanism

67 00:04:40.390 --> 00:04:43.143 of how the projection neurons are generated.

68 00:04:44.540 --> 00:04:45.373 And the goal,

69 00:04:45.373 --> 00:04:48.770 is that we want to use the single-cell RNA sequencing

70 00:04:48.770 --> 00:04:52.480 so that we can have a more complete understanding

71 00:04:52.480 --> 00:04:56.503 of this, the neuronal diversity and the neuron development.

72 00:04:57.388 --> 00:05:02.388 So you can see that here in this cartoon, this shapes,

73 00:05:03.360 --> 00:05:05.970 there are different shapes and colors,

74 00:05:05.970 --> 00:05:10.970 to represent different cell types in the neocortex,

75 00:05:11.810 --> 00:05:16.680 as the cells are experiencing the continuous dynamic changes

76 00:05:16.680 --> 00:05:18.760 actually in the real cell population,

77 00:05:18.760 --> 00:05:20.833 it is much complicated than that.

78 00:05:22.030 --> 00:05:24.530 There is not clear boundaries

79 00:05:24.530 --> 00:05:27.870 between different cell types and there may be...

80 00:05:27.870 --> 00:05:31.657 There even, it's not a clear definition of cell type.

81 00:05:32.640 --> 00:05:34.890 So, what we hope,

82 00:05:34.890 --> 00:05:38.000 is that we want to use single-cell RNA sequencing

83 00:05:38.000 --> 00:05:43.000 to first recover the trajectory of the dynamic changes

84 00:05:43.260 --> 00:05:45.150 or the developmental process

85 00:05:45.150 --> 00:05:46.803 that the cells are experiencing.

86 00:05:48.400 --> 00:05:52.340 So specifically we focus on two datasets.

87 00:05:52.340 --> 00:05:56.040 One data set, we name it as data set A.

88 00:05:56.040 --> 00:05:57.590 So this is a data set

89 00:05:57.590 --> 00:06:01.330 that is recently collected by my collaborator.

90 00:06:01.330 --> 00:06:03.960 And so we have samples...

91 00:06:05.480 --> 00:06:07.180 The cells from the mouse neocortex

92 00:06:08.110 --> 00:06:10.990 at six different embryonic days.

93 00:06:10.990 --> 00:06:13.180 And before our data,

94 00:06:13.180 --> 00:06:17.410 there is another dataset we call it, we name it data set B.

95 00:06:17.410 --> 00:06:21.790 And this dataset is a smaller dataset than ours

96 00:06:21.790 --> 00:06:22.623 but they are...

97 00:06:22.623 --> 00:06:26.880 They have also sequenced a very similar brain region

98 00:06:27.820 --> 00:06:31.820 of the mouses and they have a sequence of cells

99 00:06:31.820 --> 00:06:34.420 from four different embryonic days.

100 00:06:34.420 --> 00:06:37.320 So you can see that our,

101 00:06:37.320 --> 00:06:40.830 most of the days that are sequenced in our dataset

102 00:06:40.830 --> 00:06:44.380 and with the other dataset B, do not overlap.

103 00:06:44.380 --> 00:06:48.500 And so it would be beneficial if we can have with...

104 00:06:48.500 --> 00:06:50.835 If we can combine the two there datasets

105 00:06:50.835 --> 00:06:55.835 and so that we can make use of the cells from both studies.

106 00:06:56.200 --> 00:06:58.880 For instance, for our dataset,

107 00:06:58.880 --> 00:07:03.590 we don't have these cells from the day 11,

108 00:07:03.590 --> 00:07:06.240 which is quite important day.

109 00:07:06.240 --> 00:07:10.520 For example here, day 11 are the day that,

110 00:07:10.520 --> 00:07:15.397 there are projection neurons that are, beginning time,

111 00:07:16.860 --> 00:07:19.560 well, there are projection neurons that are generated.

112 00:07:21.190 --> 00:07:25.750 And so this E11 cells are sequenced from the other dataset.

113 00:07:25.750 --> 00:07:27.690 So it would be beneficial

114 00:07:27.690 --> 00:07:31.760 if we can perform a choice analysis of the two datasets

115 00:07:31.760 --> 00:07:35.170 and learn a shared developmental trajectory

116 00:07:35.170 --> 00:07:36.400 as these two datasets,

117 00:07:36.400 --> 00:07:39.913 are actually sequencing the same mouse brain region.

118 00:07:42.410 --> 00:07:47.350 So as you may have imagined, if we don't do anything,

119 00:07:47.350 --> 00:07:51.200 if we just concatenate the cells from two datasets

120 00:07:51.200 --> 00:07:53.830 and treat them as datasets from the same lab,

121 00:07:53.830 --> 00:07:56.700 then these two datasets actually will not,

122 00:07:56.700 --> 00:07:58.910 the cells will not merge

123 00:07:58.910 --> 00:08:03.370 because of the bash effects between the two datasets.

124 00:08:03.370 --> 00:08:05.110 Because these are from two labs

125 00:08:05.110 --> 00:08:07.810 and they have different sequencing machines

126 00:08:07.810 --> 00:08:09.510 so the cells become different,

127 00:08:09.510 --> 00:08:12.393 though they are coming from the same brain region.

128 00:08:13.990 --> 00:08:16.700 And this is a figure called the UMAP

129 00:08:16.700 --> 00:08:19.540 which is a two-dimensional projection

130 00:08:19.540 --> 00:08:20.720 of the high dimensional,

131 00:08:20.720 --> 00:08:24.080 observed single-cell RNA sequencing data

132 00:08:24.080 --> 00:08:26.550 so that we can have a visualization

133 00:08:26.550 --> 00:08:28.372 of the cell population.

134 00:08:28.372 --> 00:08:29.320 (clears throat)

135 00:08:29.320 --> 00:08:33.240 And using our marker which is called vitae

136 00:08:33.240 --> 00:08:35.140 that I will introduce later

137 00:08:35.140 --> 00:08:40.140 or we can merge the cells from two different sources.

138 00:08:41.430 --> 00:08:43.570 And as I will show later,

139 00:08:43.570 --> 00:08:48.570 we can also keep the uniqueness, the unique characteristics

140 00:08:48.810 --> 00:08:51.220 that only exist in one of the datasets.

141 00:08:51.220 --> 00:08:54.624 So we can keep the biological meaningful differences

142 00:08:54.624 --> 00:08:57.261 between the two datasets.

143 00:08:57.261 --> 00:08:58.094 And our method is actually not just,

144 00:09:00.960 --> 00:09:03.240 data integration approach.

145 00:09:03.240 --> 00:09:06.653 So what we can do, is that we can also simultaneously,

146 00:09:07.650 --> 00:09:11.620 learn a shared trajectory structure

147 00:09:11.620 --> 00:09:15.690 and we can at the same time do the disintegration

148 00:09:15.690 --> 00:09:18.950 or more generally correct for confounding effects

149 00:09:18.950 --> 00:09:23.823 such as the data source and other various like cell cycles.

150 00:09:25.290 --> 00:09:26.910 And in the...

151 00:09:26.910 --> 00:09:27.950 In this figure,

152 00:09:27.950 --> 00:09:32.490 the arrows show the direction of the developmental process

153 00:09:33.750 --> 00:09:38.290 and the line width represents the score for an edge.

154 00:09:38.290 --> 00:09:42.053 So it shows how confident we are in,

155 00:09:43.190 --> 00:09:45.460 in like whether there's a transition

156 00:09:45.460 --> 00:09:50.460 between the two states that the line connects.

157 00:09:54.900 --> 00:09:59.030 So our method actually belongs to a larger group

158 00:09:59.030 --> 00:10:03.180 of computational tools for single-cell RNA sequencing

159 00:10:03.180 --> 00:10:05.460 which is called the trajectory inference.

160 00:10:05.460 --> 00:10:07.822 So here we call it...

161 00:10:07.822 --> 00:10:09.020 So it is called trajectory inference

162 00:10:09.020 --> 00:10:12.610 that is different from statistical inference.

163 00:10:12.610 --> 00:10:15.460 So it's a computational tool

164 00:10:15.460 --> 00:10:20.250 so that we can understand in the, our cell lineage

165 00:10:20.250 --> 00:10:24.640 and the cell fate decisions in biological process,

166 00:10:24.640 --> 00:10:26.800 such as cell differentiation

167 00:10:26.800 --> 00:10:30.293 as what we have already seen in the mouse neocortex,

168 00:10:31.256 --> 00:10:33.190 and some other biological process,

169 00:10:33.190 --> 00:10:35.810 such as immune response, cancer expansion

170 00:10:35.810 --> 00:10:39.293 and many more are using single-cell RNA sequencing data.

171 00:10:41.260 --> 00:10:44.340 In general, the trajectory inference approaches,

172 00:10:44.340 --> 00:10:48.410 they will infer or they start with a,

173 00:10:49.820 --> 00:10:53.060 a type of the underlying trajectory structure

174 00:10:53.060 --> 00:10:54.790 and other methods,

175 00:10:54.790 --> 00:10:59.580 they will assume a specific type of the trajectory structure

176 00:10:59.580 --> 00:11:02.570 for the underlying developmental process,

177 00:11:02.570 --> 00:11:06.460 such as a linear structure, a linear topology

178 00:11:06.460 --> 00:11:11.460 or a bifurcating, a bifurcation or tree-like trajectory.

179 00:11:12.570 --> 00:11:14.320 And as the cell populations

180 00:11:14.320 --> 00:11:15.740 that we are trying to understand,

181 00:11:15.740 --> 00:11:18.160 become more and more complicated,

182 00:11:18.160 --> 00:11:21.210 recent methods also try to infer,

183 00:11:21.210 --> 00:11:23.060 the type of the trajectory structure

184 00:11:23.060 --> 00:11:25.603 from the observed single-cell RNA sequencing data.

185 00:11:27.130 --> 00:11:30.480 And whilst we have learned the trajectory structure,

186 00:11:30.480 --> 00:11:33.032 then this trajectory inference approaches,

187 00:11:33.032 --> 00:11:35.907 will computationally project

188 00:11:35.907 --> 00:11:39.190 and order the cells along the trajectory.

189 00:11:39.190 --> 00:11:43.273 And the right order of the cells along the trajectory,

190 00:11:43.273 --> 00:11:46.060 are called the pseudotime of the cells.

191 00:11:46.060 --> 00:11:47.590 So the trajectory inference,

192 00:11:47.590 --> 00:11:50.633 is also called the pseudotime analysis.

193 00:11:53.420 --> 00:11:54.870 And since...

194 00:11:54.870 --> 00:11:59.557 So the first trajectory inference method is proposed in 2014

195 00:12:00.790 --> 00:12:04.850 and since then it has become a very popular tool

196 00:12:04.850 --> 00:12:08.820 that are used in analyzing single-cell RNA sequencing data.

197 00:12:08.820 --> 00:12:13.820 And in this study, it calculates, it summarizes the number

198 00:12:14.620 --> 00:12:16.390 of single-cell RNA sequences studies

199 00:12:16.390 --> 00:12:18.650 that are published per month.

200 00:12:18.650 --> 00:12:21.110 And you can see that in recent years,

201 00:12:21.110 --> 00:12:22.660 more than half of the published

202 00:12:23.680 --> 00:12:25.660 single-cell RNA sequencing studies

203 00:12:25.660 --> 00:12:28.954 will have some investments of the pseudotime

204 00:12:28.954 --> 00:12:31.490 and trajectories in the cell population

205 00:12:31.490 --> 00:12:33.583 that they are investigating.

206 00:12:35.380 --> 00:12:39.050 And there has also been a lot of methods

207 00:12:39.050 --> 00:12:40.910 for trajectory inference.

208 00:12:40.910 --> 00:12:45.140 And in this,

209 00:12:45.140 --> 00:12:48.080 there is a comprehensive benchmarking paper,

210 00:12:48.080 --> 00:12:49.950 recently in "Nature Biotech"

211 00:12:49.950 --> 00:12:54.580 and it has summarized 70 different trajectory methods.

212 00:12:54.580 --> 00:12:56.810 And in your paper they have compared

213 00:12:56.810 --> 00:12:59.683 about 45 different trajectory inference methods

214 00:12:59.683 --> 00:13:00.943 from different aspects.

215 00:13:03.270 --> 00:13:04.140 So you may wonder,

216 00:13:04.140 --> 00:13:07.070 since there are so many trajectory inference methods

217 00:13:07.070 --> 00:13:10.900 that are already there, why do we still want to develop,

218 00:13:10.900 --> 00:13:12.803 a new trajectory inference method?

219 00:13:14.440 --> 00:13:17.150 So the first point is that,

220 00:13:17.150 --> 00:13:19.400 although we have 70 different methods,

221 00:13:19.400 --> 00:13:21.610 many trajectory inference methods,

222 00:13:21.610 --> 00:13:24.320 they are assuming a specific type

223 00:13:24.320 --> 00:13:25.930 of the trajectory structure.

224 00:13:25.930 --> 00:13:30.870 So many methods only work for a sound developmental process.

225 00:13:30.870 --> 00:13:33.880 If you consider the methods that can work for...

226 00:13:33.880 --> 00:13:35.240 They have the flexibility

227 00:13:35.240 --> 00:13:38.920 if you work for a variety of the trajectory structures

228 00:13:38.920 --> 00:13:43.140 then we don't have that many methods that are available.

229 00:13:43.140 --> 00:13:48.140 And another concern that I have is that most methods,

230 00:13:48.380 --> 00:13:50.660 these trajectory inference methods,

231 00:13:50.660 --> 00:13:54.790 do not have explicit statistical models.

232 00:13:54.790 --> 00:13:56.130 So what I mean is that,

233 00:13:56.130 --> 00:13:58.540 though people are kind of clear

234 00:13:58.540 --> 00:14:01.390 about what's the biological signal

235 00:14:01.390 --> 00:14:05.063 that we want to find in the trajectory inference,

236 00:14:05.950 --> 00:14:10.950 it is actually, many methods are actually pretty vague about

237 00:14:11.080 --> 00:14:15.670 from the aspect of like for the single-cell data matrix,

238 00:14:15.670 --> 00:14:19.250 what can be the definition of the trajectory

239 00:14:19.250 --> 00:14:21.070 that they want to infer.

240 00:14:21.070 --> 00:14:24.540 So, and how that they are generating,

241 00:14:24.540 --> 00:14:28.300 and how the data it can be modeled and generated

242 00:14:28.300 --> 00:14:30.040 with the trajectory structure.

243 00:14:30.040 --> 00:14:33.780 So as the statistician, I think it would be beneficial,

244 00:14:33.780 --> 00:14:37.920 if we have a model-based trajectory inference approach,

245 00:14:37.920 --> 00:14:40.830 so that we can better understand the profit,

246 00:14:40.830 --> 00:14:43.560 how good our estimations are

247 00:14:43.560 --> 00:14:45.800 and have some certain qualification

248 00:14:45.800 --> 00:14:49.753 of the trajectories or slow times that we infer.

249 00:14:53.940 --> 00:14:56.240 And the third point is that

250 00:14:56.240 --> 00:14:59.110 as you have shown at the beginning,

251 00:14:59.110 --> 00:15:01.190 there is also a growing need,

252 00:15:01.190 --> 00:15:04.530 to efficiently align trajectories

253 00:15:04.530 --> 00:15:06.550 or do a joint analysis

254 00:15:06.550 --> 00:15:10.360 from multiple single-cell RNA sequencing datasets.

255 00:15:10.360 --> 00:15:14.140 As the, as the studies...

256 00:15:14.140 --> 00:15:17.950 As the single-cell RNA sequencing datasets are expanding,

257 00:15:17.950 --> 00:15:21.203 there has already been a lot of studies for datasets,

258 00:15:21.203 --> 00:15:25.140 they are for the same tissue or for the same cell type.

259 00:15:25.140 --> 00:15:26.443 And it will be...

260 00:15:26.443 --> 00:15:27.276 (clears throat)

261 00:15:27.276 --> 00:15:31.110 And we can learn a better picture of, on this,

262 00:15:31.110 --> 00:15:35.470 the biological process in the tissue or for the cell time,

263 00:15:35.470 --> 00:15:39.090 if we can use all available datasets.

264 00:15:39.090 --> 00:15:41.100 And so there's a strong need,

265 00:15:41.100 --> 00:15:46.050 an increasing need to do this joint trajectory analysis

266 00:15:46.050 --> 00:15:47.173 for multiple datasets.

267 00:15:50.280 --> 00:15:52.390 So because of these reasons,

268 00:15:52.390 --> 00:15:57.390 we develop a new statistical framework and a new method,

269 00:15:57.460 --> 00:15:59.100 and we call it VITAE,

270 00:15:59.100 --> 00:16:01.920 which is short for variational inference

271 00:16:01.920 --> 00:16:04.610 for trajectory by autoencoders.

272 00:16:04.610 --> 00:16:07.923 And it is a model-based trajectory inference approach.

273 00:16:11.310 --> 00:16:14.870 So our model starts with a definition

274 00:16:14.870 --> 00:16:17.960 of the trajectory backbone.

275 00:16:17.960 --> 00:16:22.030 So we use a graph to define the trajectory backbone.

276 00:16:22.030 --> 00:16:25.140 So we start with a complete graph G,

277 00:16:25.140 --> 00:16:30.140 well the vertices are the distinct cell states and cell type

278 00:16:31.020 --> 00:16:34.460 and an edge denotes a possible transition

279 00:16:34.460 --> 00:16:37.673 between two cell states and or cell types.

280 00:16:38.540 --> 00:16:43.490 And then we can define a cell position on the graph

281 00:16:43.490 --> 00:16:46.710 which is a vector, which is a landscape vector.

282 00:16:46.710 --> 00:16:47.543 And it's...

283 00:16:47.543 --> 00:16:50.707 A K is the number of vertices on the graph,

284 00:16:50.707 --> 00:16:51.850 in the graph.

285 00:16:51.850 --> 00:16:54.773 So if a cell is exactly,

286 00:16:54.773 --> 00:16:57.660 belongs to one cell state or cell type,

287 00:16:57.660 --> 00:17:01.330 then it is on cell vertex.

288 00:17:01.330 --> 00:17:05.410 And if the cell is experiencing a transition

289 00:17:05.410 --> 00:17:08.150 between two cell states or cell types,

290 00:17:08.150 --> 00:17:12.973 then we denote it as on the edge between two vertices.

291 00:17:15.560 --> 00:17:18.273 And then we can define the trajectory backbone

292 00:17:18.273 --> 00:17:20.410 as a subgraph of G.

293 00:17:20.410 --> 00:17:23.280 So we only include an edge or vertex

294 00:17:23.280 --> 00:17:27.390 if we really observe cells that are on the edge.

295 00:17:27.390 --> 00:17:30.170 So though there are many possible transitions

296 00:17:30.170 --> 00:17:33.120 between the cell types or cell states,

297 00:17:33.120 --> 00:17:37.600 We believe, we include a transition only

298 00:17:37.600 --> 00:17:39.210 when we do observe cells

299 00:17:39.210 --> 00:17:42.330 that are experiencing such transitions.

300 00:17:42.330 --> 00:17:47.330 So though there can be many edges in our complete graph,

301 00:17:47.600 --> 00:17:51.540 on the sub graph, it's a sparse success of the other edges

302 00:17:51.540 --> 00:17:54.367 that are possible on the graph.

303 00:17:56.560 --> 00:18:00.720 And a benefit of the above definition,

304 00:18:00.720 --> 00:18:04.920 is that we can allow any types of the trajectory structure.

305 00:18:04.920 --> 00:18:07.440 So it can be either a linear structure,

306 00:18:07.440 --> 00:18:12.370 a bifurcate chain or a tree-like structure or a cycle,

307 00:18:12.370 --> 00:18:16.330 it completely depends on how the data shows.

308 00:18:16.330 --> 00:18:17.840 And so we allow,

309 00:18:17.840 --> 00:18:22.840 we want the data to automatically determine the other,

310 00:18:23.710 --> 00:18:26.310 the trajectory structure or topology

311 00:18:26.310 --> 00:18:30.773 of the underlying dynamic process.

312 00:18:33.370 --> 00:18:38.340 And we can also define the pseudotime for each cell.

313 00:18:38.340 --> 00:18:42.600 I have not written down the exact definition here

314 00:18:42.600 --> 00:18:46.150 but the idea is that we first need a root vertex.

315 00:18:46.150 --> 00:18:51.150 So a root vertex is the start of this dynamic process.

316 00:18:51.420 --> 00:18:54.270 And it can be given by the user,

317 00:18:54.270 --> 00:18:56.630 depending on looking at the marker genes

318 00:18:56.630 --> 00:18:59.760 or other side biological information.

319 00:18:59.760 --> 00:19:01.500 And later we will also...

320 00:19:01.500 --> 00:19:03.760 I will also show you that for some datasets,

321 00:19:03.760 --> 00:19:07.500 we can automatically determine the root vertex.

322 00:19:07.500 --> 00:19:09.190 And with a given root vertex,

323 00:19:09.190 --> 00:19:13.240 then the graph becomes a directed graph.

324 00:19:13.240 --> 00:19:16.920 And we had defined the pseudotime of the cell

325 00:19:16.920 --> 00:19:21.630 as the shortest path from the root to a specific cell

326 00:19:21.630 --> 00:19:26.233 along the trajectory, along the trajectory backbone.

327 00:19:28.470 --> 00:19:33.180 So this graph defines the trajectory structure.

328 00:19:33.180 --> 00:19:34.490 And the next step,

329 00:19:34.490 --> 00:19:38.170 is that we want to link the trajectory structure

330 00:19:38.170 --> 00:19:41.100 with the data generation model.

331 00:19:41.100 --> 00:19:46.100 So the single-cell RNA sequencing data matrix,

332 00:19:46.220 --> 00:19:49.530 is typically a high dimensional matrix

333 00:19:49.530 --> 00:19:51.318 because for each cell,

334 00:19:51.318 --> 00:19:54.461 we typically observe tens of thousands of genes

335 00:19:54.461 --> 00:19:59.410 and there are also complicated dependency relationships

336 00:19:59.410 --> 00:20:01.020 among the genes.

337 00:20:01.020 --> 00:20:02.133 And what we assume,

338 00:20:02.133 --> 00:20:05.940 is that we assume that these dependencies across genes,

339 00:20:05.940 --> 00:20:10.270 can be explained by a latent variables, Z(i)

340 00:20:10.270 --> 00:20:13.110 in a low dimensional space.

341 00:20:13.110 --> 00:20:17.530 And we assume that these latent variables,

342 00:20:17.530 --> 00:20:20.420 are following our normal distributions

343 00:20:20.420 --> 00:20:25.210 and they also have the graph structure.

344 00:20:25.210 --> 00:20:29.264 So U here, are the positions of the vertices on the graph

345 00:20:29.264 --> 00:20:31.970 in this low dimensional space,

346 00:20:31.970 --> 00:20:34.180 and the meaning of Z(i),

347 00:20:34.180 --> 00:20:37.680 is a linear combination of these vertices,

348 00:20:37.680 --> 00:20:40.850 depending on, of the positions of the vertices,

349 00:20:40.850 --> 00:20:45.580 depending on the cell's graphic position on the graph.

350 00:20:45.580 --> 00:20:46.413 And,

351 00:20:48.500 --> 00:20:51.580 what I want to emphasize here is one point,

352 00:20:51.580 --> 00:20:55.222 is that we assume a non-linear marking

353 00:20:55.222 --> 00:20:59.328 from the latent space to the high dimensional observed data,

354 00:20:59.328 --> 00:21:03.100 because we think that though in the low dimensional space,

355 00:21:03.100 --> 00:21:06.836 we can represent the trajectory as these linear lines,

356 00:21:06.836 --> 00:21:10.352 it is very likely a manifold on the observed data.

357 00:21:10.352 --> 00:21:12.101 So this non-linear mapping,

358 00:21:12.101 --> 00:21:15.671 can map this linear lines to hertz

359 00:21:15.671 --> 00:21:18.177 in the high dimensional space.

360 00:21:18.177 --> 00:21:20.204 And now to consider,

361 00:21:20.204 --> 00:21:23.487 to account for the confounding covariates,

362 00:21:23.487 --> 00:21:26.661 such as the data source or cell cycle,

363 00:21:26.661 --> 00:21:30.360 we also allow this non-linear mapping,

364 00:21:30.360 --> 00:21:33.060 to depend on this covariates.

365 00:21:33.060 --> 00:21:34.870 And here we are...

366 00:21:34.870 --> 00:21:36.570 Because the observed data count,

367 00:21:36.570 --> 00:21:39.820 we assume it follows an active binomial distribution,

368 00:21:39.820 --> 00:21:41.270 and L(i) are...

369 00:21:41.270 --> 00:21:43.760 Oh, sorry. L(i) here should be known library size.

370 00:21:43.760 --> 00:21:46.312 Sorry for the typo. It should be known library sizes.

371 00:21:46.312 --> 00:21:50.890 And CRJ, and the CRG, the dispersion parameter switch gene,

372 00:21:50.890 --> 00:21:52.587 are unknown parameters.

373 00:21:52.587 --> 00:21:55.113 And so in this, in the current model,

374 00:21:55.113 --> 00:21:56.837 the unknown parameters we have,

375 00:21:56.837 --> 00:22:00.314 are these cell, the vertex positions,

376 00:22:00.314 --> 00:22:03.001 U, the cell positions on the graph,

377 00:22:03.001 --> 00:22:05.740 the W(i) the non-linear mapping

378 00:22:05.740 --> 00:22:07.970 and this unknown dispersion parameters.

379 00:22:07.970 --> 00:22:09.623 So we have a lot of parameters.

380 00:22:11.010 --> 00:22:14.870 So to further simplify our estimation,

381 00:22:14.870 --> 00:22:18.000 we assume that there are a mixture prior

382 00:22:18.000 --> 00:22:19.840 on the cell positions.

383 00:22:19.840 --> 00:22:23.230 So it's a very tactical idea.

384 00:22:23.230 --> 00:22:27.273 So we assume that first the cell,

385 00:22:27.273 --> 00:22:30.275 there are some latent variables, Ci for each cell.

386 00:22:30.275 --> 00:22:35.275 And so Ci determines which edge or vertex a cell chooses.

387 00:22:35.410 --> 00:22:39.410 So the cell has some probability to choose a specific edge

388 00:22:39.410 --> 00:22:42.170 or a specific vertex,

389 00:22:42.170 --> 00:22:45.520 and if it chooses an edge,

390 00:22:45.520 --> 00:22:48.903 then, it then eventually choose the location of,

391 00:22:50.677 --> 00:22:53.317 the relative location on the edge..

392 00:22:55.030 --> 00:22:58.883 So that's what becomes a mixture prior on this diagram two.

393 00:22:59.840 --> 00:23:00.673 And,

394 00:23:02.375 --> 00:23:06.250 for these non-linear mapping functions,

395 00:23:06.250 --> 00:23:10.060 we're including non-linear mappings.

396 00:23:10.060 --> 00:23:14.533 We model these F(G(i)) functions by a neural network.

397 00:23:16.500 --> 00:23:18.240 So, Oh, do you?

398 00:23:18.240 --> 00:23:22.480 So our parameters now, our known parameters now are,

399 00:23:22.480 --> 00:23:24.287 this U the positions of the vertices

400 00:23:24.287 --> 00:23:26.550 on the low-dimensional space,

401 00:23:26.550 --> 00:23:31.550 this PI, the probability of each vertex and edge,

402 00:23:34.040 --> 00:23:36.727 and the non-linear mappings,

403 00:23:36.727 --> 00:23:39.190 which are the waste in the neural network

404 00:23:39.190 --> 00:23:41.810 and this, this dispersion parameters.

405 00:23:41.810 --> 00:23:42.820 And we...

406 00:23:42.820 --> 00:23:47.270 I space these parameters by combining our mixture model

407 00:23:47.270 --> 00:23:48.733 with a variational autoencoder.

408 00:23:51.280 --> 00:23:52.920 So the variational autoencoder.

409 00:23:52.920 --> 00:23:57.590 So the autoencoder has been a very popular model

410 00:23:57.590 --> 00:23:59.190 for in deep learning.

411 00:23:59.190 --> 00:24:02.230 So what it can do is, is that it can,

412 00:24:02.230 --> 00:24:06.520 can have some non-linear mapping

413 00:24:06.520 --> 00:24:11.120 of the observed data to a low-dimensional space

414 00:24:11.120 --> 00:24:14.200 and we want the low-dimensional space to best,

415 00:24:14.200 --> 00:24:18.060 recover our observed time rational data.

416 00:24:18.060 --> 00:24:21.010 And here, we also have such a task.

417 00:24:21.010 --> 00:24:22.800 We have the low-dimensional space

418 00:24:22.800 --> 00:24:27.023 and we want to best to recover our observed data.

419 00:24:27.023 --> 00:24:30.710 And what's different is that we also have a prior

420 00:24:30.710 --> 00:24:33.770 on the latent space, because we have the prior,

421 00:24:33.770 --> 00:24:38.300 so we use the variational autoencoder model

422 00:24:38.300 --> 00:24:39.440 in deep learning.

423 00:24:39.440 --> 00:24:42.060 So the classical variational autoencoder

424 00:24:43.420 --> 00:24:45.960 in deep learning, we'll assume that the latent space,

425 00:24:45.960 --> 00:24:49.650 has the standard normal distribution as the prior.

426 00:24:49.650 --> 00:24:53.370 And here we just modify it so that we have the...

427 00:24:53.370 --> 00:24:56.260 So that the latent space have the mixture prior

428 00:24:56.260 --> 00:25:01.260 that are assumed in our previous mixture models.

429 00:25:01.520 --> 00:25:03.840 And we use the same approach

430 00:25:03.840 --> 00:25:07.760 as the variational autoencoder, the variational path

431 00:25:07.760 --> 00:25:10.250 which is, though the,

432 00:25:10.250 --> 00:25:13.150 to approximate the posteriors of the latent space.

433 00:25:13.150 --> 00:25:15.881 So though, because of the complicated priors

434 00:25:15.881 --> 00:25:19.290 and non-linear mappings,

435 00:25:19.290 --> 00:25:22.870 this prior, the posterior of the latent space conditional

436 00:25:22.870 --> 00:25:26.590 on the observed data and the confounding covariates,

437 00:25:26.590 --> 00:25:28.110 it can be complicated,

438 00:25:28.110 --> 00:25:32.000 we approximate it by our normal distributions

439 00:25:32.000 --> 00:25:36.220 and the mean and the variances of the normal distributions

440 00:25:36.220 --> 00:25:39.650 which are functions of the observed data Y(i)

441 00:25:39.650 --> 00:25:41.130 and the covariance Xi,

442 00:25:41.130 --> 00:25:43.720 are also non-linear functions

443 00:25:43.720 --> 00:25:46.080 and we model them by the neural network

444 00:25:46.080 --> 00:25:47.320 and that's the encoder.

445 00:25:47.320 --> 00:25:52.320 So the decoder is the nominal mapping function F(G(i)),

446 00:25:52.770 --> 00:25:56.080 mapping the latent space to the observed data.

447 00:25:56.080 --> 00:25:58.240 And encoder are neural networks

448 00:25:58.240 --> 00:26:03.040 that are approximating the posteriors of the latent space.

449 00:26:03.040 --> 00:26:05.983 - Hi, Jingshu, can I ask a very quick question?

450 00:26:06.860 --> 00:26:08.170 If I understand correctly,

451 00:26:08.170 --> 00:26:11.470 up to now you have not used the time information,

452 00:26:11.470 --> 00:26:12.810 is this true?

453 00:26:12.810 --> 00:26:15.600 Or you have considered to include the time information

454 00:26:15.600 --> 00:26:17.390 in the covariate X?

455 00:26:17.390 --> 00:26:18.583 - Oh, oh, yes.

456 00:26:18.583 --> 00:26:23.583 So we will not use time information

457 00:26:24.120 --> 00:26:26.390 in our trajectory inference.

458 00:26:26.390 --> 00:26:29.840 If I have time, I may have a last slide which is a,

459 00:26:30.810 --> 00:26:34.200 which is a review of the literature into data inference.

460 00:26:34.200 --> 00:26:36.163 So into data inference,

461 00:26:37.060 --> 00:26:41.030 the most commonly used and best performing methods,

462 00:26:41.030 --> 00:26:44.160 they will tend to not use the time information

463 00:26:44.160 --> 00:26:45.270 because there is...

464 00:26:45.270 --> 00:26:49.320 Though the time information is typically correlated

465 00:26:49.320 --> 00:26:54.320 with developmental like, timing of the cells,

466 00:26:54.380 --> 00:26:57.070 but because at each collectional time,

467 00:26:57.070 --> 00:27:00.440 is a mixture of cells at different environmental time.

468 00:27:00.440 --> 00:27:03.040 So it's a big, complicated relation

469 00:27:03.040 --> 00:27:05.230 and some methods use that information

470 00:27:05.230 --> 00:27:08.830 but many methods do not use that.

471 00:27:08.830 --> 00:27:13.327 And so our approach is the methods that do not use the,

472 00:27:14.270 --> 00:27:16.470 the collection time information.

473 00:27:16.470 --> 00:27:21.470 And, we use it only when we decide

474 00:27:21.970 --> 00:27:24.420 which vertex is the root.

475 00:27:24.420 --> 00:27:27.100 And I will show that later.

476 00:27:27.100 --> 00:27:27.933 - Thanks.

477 00:27:30.210 --> 00:27:31.980 - So our...

478 00:27:33.130 --> 00:27:37.470 So the last function is, is composed of three parts.

479 00:27:37.470 --> 00:27:41.610 The first part is this likelihood based reconstruction loss.

480 00:27:41.610 --> 00:27:46.610 So this evaluates how goods are our latent spaces to,

481 00:27:47.890 --> 00:27:52.273 to reconstruct the high-dimensional observed data.

482 00:27:54.460 --> 00:27:57.190 And the second part is the KL divergence

483 00:27:57.190 --> 00:28:02.190 between the posterior distribution and the prior.

484 00:28:02.300 --> 00:28:06.070 And you can think of it as a regularization term.

485 00:28:06.070 --> 00:28:07.400 And so to regularize

486 00:28:07.400 --> 00:28:11.980 if the posterior is very far away from the prior,

487 00:28:11.980 --> 00:28:16.980 also when for variational autoencoders so beta equal to one,

488 00:28:17.060 --> 00:28:20.828 it can be also viewed as a lower bound

489 00:28:20.828 --> 00:28:22.170 of the marginalized data.

490 00:28:23.140 --> 00:28:27.040 So here we make the,

491 00:28:27.040 --> 00:28:29.740 we add the training parameter beta,

492 00:28:29.740 --> 00:28:33.920 and in practice we said, beta are larger than one,

493 00:28:33.920 --> 00:28:38.900 so that we can encourage the posterior, the regularization,

494 00:28:38.900 --> 00:28:41.147 so that the posteriors of the I will,

495 00:28:41.147 --> 00:28:43.700 are more likely to tend to align

496 00:28:43.700 --> 00:28:45.310 along the edges and vertices.

497 00:28:45.310 --> 00:28:48.470 And that's the idea that has been used in deep learning

498 00:28:48.470 --> 00:28:49.720 which is called the beta.

499 00:28:52.670 --> 00:28:54.650 And the third term,

500 00:28:54.650 --> 00:28:58.460 it's a term for adjusting for the covariance.

501 00:28:58.460 --> 00:29:00.167 So the covariance...

502 00:29:01.340 --> 00:29:03.417 So this covariance * covers.

503 00:29:03.417 --> 00:29:07.610 So we want our latent space C to be kind of,

504 00:29:07.610 --> 00:29:10.410 be correlated with the covariance,

505 00:29:10.410 --> 00:29:11.243 While...

506 00:29:11.243 --> 00:29:14.150 So we, certain...

507 00:29:14.150 --> 00:29:18.160 So we want to maximize the reconstruction of the data

508 00:29:18.160 --> 00:29:20.420 by only by the covariates.

509 00:29:20.420 --> 00:29:23.870 And setting the tuning parameter alpha larger than zero,

510 00:29:23.870 --> 00:29:27.500 we can help decorrelate Z(i) from Xi.

511 00:29:31.010 --> 00:29:34.730 So another art in our training is that,

512 00:29:34.730 --> 00:29:37.710 we need a good internalization of the graph.

513 00:29:37.710 --> 00:29:40.390 So specifically we need to determine,

514 00:29:40.390 --> 00:29:42.360 how many vertices there are,

515 00:29:42.360 --> 00:29:44.750 and also the positions of the vertices

516 00:29:44.750 --> 00:29:47.190 in the low-dimensional space.

517 00:29:47.190 --> 00:29:48.780 That's not an easy job.

518 00:29:48.780 --> 00:29:51.750 And if we just randomly,

519 00:29:51.750 --> 00:29:55.610 because our final graph depend on,

520 00:29:55.610 --> 00:29:59.503 the total number of vertices that we set at the beginning.

521 00:30:00.420 --> 00:30:04.610 So how we pretrain the model to return it's initial value?

522 00:30:04.610 --> 00:30:08.400 To get the initial values of the unknown parameters,

523 00:30:08.400 --> 00:30:11.860 is that we first trained with beta equal to zero,

524 00:30:11.860 --> 00:30:13.860 so that we don't make use of any,

525 00:30:13.860 --> 00:30:16.580 these prior distributions of the I.

526 00:30:16.580 --> 00:30:17.540 So it's only...

527 00:30:17.540 --> 00:30:20.460 We're only looking at the reconstruction loss

528 00:30:20.460 --> 00:30:22.120 from the likelihood of the data.

529 00:30:22.120 --> 00:30:27.120 So it's like the normal, the classical autoencoder.

530 00:30:27.190 --> 00:30:30.697 And from that we can get some initial estimate of Z(i),

531 00:30:32.740 --> 00:30:34.390 The latent space variables.

532 00:30:34.390 --> 00:30:38.430 And then we perform clustering on Z(i),

533 00:30:38.430 --> 00:30:40.540 and we let the clustering algorithm,

534 00:30:40.540 --> 00:30:43.350 to automatically determine the number of clusters

535 00:30:43.350 --> 00:30:46.187 and we use that as the number of vertices.

536 00:30:46.187 --> 00:30:48.650 And we also use the cluster centers

537 00:30:48.650 --> 00:30:51.633 as the initialization, as the initial values of U.

538 00:30:53.150 --> 00:30:54.840 So that's the main part,

539 00:30:54.840 --> 00:30:57.110 the key ideas in our pre-training start,

540 00:30:57.110 --> 00:31:01.070 so that we can have a good initial addition to,

541 00:31:01.070 --> 00:31:02.570 for the start of the training.

542 00:31:04.600 --> 00:31:07.590 And another trick that we have taken,

543 00:31:07.590 --> 00:31:10.156 is that in practice, sorry,

544 00:31:10.156 --> 00:31:15.156 the best performing existing trajectory inference methods.

545 00:31:15.600 --> 00:31:17.280 They will attempt...

546 00:31:17.280 --> 00:31:19.620 So they are typically very fast.

547 00:31:19.620 --> 00:31:23.880 And in order to have comparable computational costs

548 00:31:23.880 --> 00:31:25.010 of these methods,

549 00:31:25.010 --> 00:31:28.420 we also have accelerated version of our algorithm

550 00:31:28.420 --> 00:31:32.453 which is a simply to reduce the input,

551 00:31:33.900 --> 00:31:35.930 the dimension of the input space,

552 00:31:35.930 --> 00:31:38.870 so we can replace Y(i),

553 00:31:38.870 --> 00:31:43.660 the high-dimensional vector of the gene expressions

554 00:31:43.660 --> 00:31:46.220 with it's principal components.

555 00:31:46.220 --> 00:31:49.740 Now, principal component, principal scores,

556 00:31:49.740 --> 00:31:52.520 which is a low-dimensional vector L.

557 00:31:52.520 --> 00:31:57.147 We, by default we will take the first 64 dimensions

558 00:31:59.136 --> 00:32:00.670 for the principal scores.

559 00:32:00.670 --> 00:32:04.660 And so we replace the elected binomial distribution

560 00:32:04.660 --> 00:32:08.590 by a normal gaussian distribution assumption

561 00:32:08.590 --> 00:32:09.890 of these principal scores.

562 00:32:11.303 --> 00:32:14.760 And as you will see later in our,

563 00:32:14.760 --> 00:32:19.360 in our evaluations with real and synthetic data,

564 00:32:19.360 --> 00:32:23.600 we see that we actually have comparable performance

565 00:32:23.600 --> 00:32:25.900 with our previous likelihood,

566 00:32:25.900 --> 00:32:30.440 with our standard likelihood based methods

567 00:32:30.440 --> 00:32:32.333 for this accelerated version.

568 00:32:34.850 --> 00:32:37.810 So after the final step is that

569 00:32:37.810 --> 00:32:40.380 after the training the autoencoder,

570 00:32:40.380 --> 00:32:43.510 we have approximated distributions,

571 00:32:43.510 --> 00:32:46.900 posterior distributions of the latent space

572 00:32:46.900 --> 00:32:50.790 and also the cell positions that

573 00:32:51.872 --> 00:32:53.243 and which vertex,

574 00:32:54.220 --> 00:32:58.027 the vertex or the posterior distribution of Ci,

575 00:32:58.027 --> 00:33:03.027 well Ci* is which vertex or edge the cell is from.

576 00:33:04.290 --> 00:33:07.200 And we need to use those information,

577 00:33:07.200 --> 00:33:10.430 to determine the trajectory backbone

578 00:33:10.430 --> 00:33:13.120 and to project each cell

579 00:33:13.120 --> 00:33:15.723 on our inferred trajectory backbone.

580 00:33:16.960 --> 00:33:20.933 So how we do that is, first we calculate an edge score.

581 00:33:22.430 --> 00:33:25.800 So this edge score is...

582 00:33:27.070 --> 00:33:29.840 So we have different scores for an edge,

583 00:33:29.840 --> 00:33:31.780 and that is determined

584 00:33:31.780 --> 00:33:34.750 on looking at the posteriors of cells.

585 00:33:34.750 --> 00:33:38.470 How many cells from the posterior distribution?

586 00:33:38.470 --> 00:33:42.970 How many cells choose to lie on that specific edge?

587 00:33:42.970 --> 00:33:44.600 If there are a lot of cells then that means

588 00:33:44.600 --> 00:33:47.450 that it's very likely that edge exist.

589 00:33:47.450 --> 00:33:49.110 If there are very few cells

590 00:33:49.110 --> 00:33:51.610 then very likely that edge should not be,

591 00:33:51.610 --> 00:33:56.610 the edge that is included in the trajectory backbone.

592 00:33:57.920 --> 00:34:01.323 And the denominator is that we want to,

593 00:34:02.980 --> 00:34:05.580 give a relatively high fair score

594 00:34:05.580 --> 00:34:10.493 for the edges of that connecting to small clusters,

595 00:34:11.370 --> 00:34:13.035 to small cell types,

596 00:34:13.035 --> 00:34:17.210 such as we want to also capture the transition

597 00:34:17.210 --> 00:34:20.460 between two rare cell types.

598 00:34:20.460 --> 00:34:23.233 So that's why we have this regularization,

599 00:34:24.930 --> 00:34:28.690 waiting by with, in the denominator.

600 00:34:28.690 --> 00:34:31.970 And we include an edge in the trajectory backbone

601 00:34:31.970 --> 00:34:35.337 if it's edge score is larger than some*.

602 00:34:37.690 --> 00:34:41.983 And when we have an inferred trajectory backbone,

603 00:34:43.420 --> 00:34:45.080 then the next step is,

604 00:34:45.080 --> 00:34:49.890 we want to project the cells on the inferred trajectory,

605 00:34:49.890 --> 00:34:54.320 and we do it by looking at the...

606 00:34:54.320 --> 00:34:57.920 Based on the posterior, approximated posterior distributions

607 00:34:57.920 --> 00:35:00.890 of the cell positions that *.

608 00:35:00.890 --> 00:35:05.770 We want to find the closest point on the inferred trajectory

609 00:35:08.020 --> 00:35:12.140 for this cell based on the posterior distributions.

610 00:35:12.140 --> 00:35:13.413 And the distance,

611 00:35:14.270 --> 00:35:18.000 this expectation we can also use as a evaluation

612 00:35:18.000 --> 00:35:20.800 of the, some uncertainty quantification

613 00:35:20.800 --> 00:35:22.260 of this projection

614 00:35:23.860 --> 00:35:25.333 or the cell positions.

615 00:35:27.310 --> 00:35:28.770 And the third thing we need to,

616 00:35:28.770 --> 00:35:32.880 because our final results is some kind of directed graph.

617 00:35:32.880 --> 00:35:36.220 So we need to determine the root vertex.

618 00:35:36.220 --> 00:35:41.220 So the root vertex can be either given by the user, or if

619 00:35:42.550 --> 00:35:43.870 as I feel I asked,

620 00:35:43.870 --> 00:35:48.400 for some datasets like the data at the beginning of my talk,

621 00:35:48.400 --> 00:35:51.861 the cells are collected in the time series,

622 00:35:51.861 --> 00:35:53.870 and we can make use of that time series,

623 00:35:53.870 --> 00:35:56.993 to determine the root vertex.

624 00:35:58.129 --> 00:36:00.690 The rough idea is that for each vertex,

625 00:36:00.690 --> 00:36:03.150 we can calculate a fraction time score,

626 00:36:03.150 --> 00:36:05.530 which is an average of the cells

627 00:36:05.530 --> 00:36:06.890 that belong to the vertex

628 00:36:06.890 --> 00:36:11.390 or projected on the edge that connects to the vertex,

629 00:36:11.390 --> 00:36:14.720 depending on the distance from the cell to the vertex.

630 00:36:14.720 --> 00:36:19.720 And so we can have some vertex collection time,

631 00:36:19.840 --> 00:36:21.277 collection time score for each vertex.

632 00:36:21.277 --> 00:36:25.351 And we choose the root vertex

633 00:36:25.351 --> 00:36:30.351 as the vertex that has the smallest collection time score.

634 00:36:31.730 --> 00:36:35.300 And with the roots and with our inferred trajectory,

635 00:36:35.300 --> 00:36:37.872 it's straightforward to calculate the pseudo-times

636 00:36:37.872 --> 00:36:38.705 for each score.

637 00:36:39.610 --> 00:36:42.400 So that's the whole process

638 00:36:42.400 --> 00:36:46.343 of our model-based methods for trajectory inference.

639 00:36:47.530 --> 00:36:51.570 And now let's look at some benchmarking results.

640 00:36:51.570 --> 00:36:54.200 So first we...

641 00:36:54.200 --> 00:36:57.980 So our benchmarking includes both some real datasets

642 00:36:57.980 --> 00:36:59.900 and some synthetic datasets.

643 00:36:59.900 --> 00:37:02.040 And we follow the...

644 00:37:02.040 --> 00:37:05.360 Most of the benchmarking follows the same framework

645 00:37:05.360 --> 00:37:08.920 as this well known benchmarking paper

646 00:37:08.920 --> 00:37:11.920 in the "Nature of Biotech" in 2019.

647 00:37:11.920 --> 00:37:13.940 And our datasets are selected

648 00:37:13.940 --> 00:37:17.010 as a subset of the datasets that they have tried

649 00:37:17.895 --> 00:37:19.490 and our criteria is that these datasets,

650 00:37:19.490 --> 00:37:24.490 maybe have enough number of cells not too few cells.

651 00:37:24.960 --> 00:37:29.540 And we wanted to cover different types of topologies.

652 00:37:32.980 --> 00:37:37.120 And this is the benchmarking results.

653 00:37:37.120 --> 00:37:38.660 So the columns, sorry.

654 00:37:38.660 --> 00:37:41.700 So the rows are the different datasets

655 00:37:42.544 --> 00:37:43.644 that I have mentioned.

656 00:37:45.280 --> 00:37:49.687 And we compare five different methods.

657 00:37:49.687 --> 00:37:51.830 So we have, would come first compare two versions

658 00:37:51.830 --> 00:37:53.400 of our approach.

659 00:37:53.400 --> 00:37:58.400 Vitae one as the original elected binomial likelihood base.

660 00:38:00.020 --> 00:38:02.920 Vitae and accelerated version,

661 00:38:02.920 --> 00:38:06.920 replacing the gene expression vectors

662 00:38:06.920 --> 00:38:08.483 by principal scores.

663 00:38:10.960 --> 00:38:13.490 Then we compare it with three different,

664 00:38:13.490 --> 00:38:16.333 state of the arch trajectory inference methods.

665 00:38:18.680 --> 00:38:20.350 The monocle PAGA.

666 00:38:20.350 --> 00:38:24.230 So the monocle series is from the lab that,

667 00:38:24.230 --> 00:38:27.200 who developed the first trajectory inference methods,

668 00:38:27.200 --> 00:38:29.670 and there, and then they further,

669 00:38:29.670 --> 00:38:32.000 the first take monocle for one

670 00:38:32.000 --> 00:38:33.450 and now they have monocle three.

671 00:38:33.450 --> 00:38:36.450 So the monocle series are always commonly used

672 00:38:36.450 --> 00:38:40.260 in these single-cell RNA sequencing papers.

673 00:38:40.260 --> 00:38:43.990 And two, I expect the performing,

674 00:38:43.990 --> 00:38:47.700 trajectory inference methods in the benchmarking paper,

675 00:38:47.700 --> 00:38:49.663 the PAGA and Slingshot.

676 00:38:51.380 --> 00:38:54.590 And all these methods,

677 00:38:54.590 --> 00:38:59.380 do not use this time information explicitly.

678 00:38:59.380 --> 00:39:02.543 So it's a fair comparison between these methods.

679 00:39:03.685 --> 00:39:04.518 And so the...

680 00:39:06.680 --> 00:39:11.120 And for all the methods they are given to by those.

681 00:39:11.120 --> 00:39:13.130 We give them the two number

682 00:39:13.130 --> 00:39:17.443 of clusters or the vertices to start from,

683 00:39:18.940 --> 00:39:20.653 and the two root vertex.

684 00:39:22.160 --> 00:39:25.020 And we, and each column is,

685 00:39:25.020 --> 00:39:29.550 we compare it's measurement, it's metric

686 00:39:30.480 --> 00:39:35.480 for the evaluation of the performance of each method.

687 00:39:35.860 --> 00:39:37.440 So the first two columns,

688 00:39:37.440 --> 00:39:42.340 are the matrix for recovery of the trajectory topology

689 00:39:42.340 --> 00:39:44.493 or the trajectory structure.

690 00:39:45.620 --> 00:39:48.810 And next two columns are the evaluation

691 00:39:48.810 --> 00:39:52.770 of the cell position, estimation accuracy.

692 00:39:52.770 --> 00:39:54.690 And the last metric is

693 00:39:54.690 --> 00:39:58.030 for evaluating the pseudotime accuracy.

694 00:39:58.030 --> 00:40:01.410 And a larger score means a better performance,

695 00:40:01.410 --> 00:40:06.010 and a lower score means like, a worse performance.

696 00:40:06.010 --> 00:40:10.600 So you can see that, our approach first is,

697 00:40:10.600 --> 00:40:13.480 our approach has much better performance

698 00:40:13.480 --> 00:40:16.853 in recovery of the trajectory topology.

699 00:40:17.980 --> 00:40:21.807 We also have some benefits of the cell position estimates,

700 00:40:22.670 --> 00:40:24.180 and because of both,

701 00:40:24.180 --> 00:40:28.550 we have a better performance in the pseudotime accuracy.

702 00:40:28.550 --> 00:40:33.220 And the other thing you can see is that our,

703 00:40:33.220 --> 00:40:36.410 our accelerated version have comparable,

704 00:40:36.410 --> 00:40:39.740 slightly worse but comparable performance,

705 00:40:39.740 --> 00:40:44.740 compared to the, our likelihood based vitae.

706 00:40:45.950 --> 00:40:48.373 And though it has a much quicker,

707 00:40:48.373 --> 00:40:50.973 much less computational cost.

708 00:40:53.810 --> 00:40:55.193 So finally,

709 00:40:56.090 --> 00:41:01.003 let's come back to the case study on mouse neocortex.

710 00:41:01.930 --> 00:41:04.273 So this is the,

711 00:41:05.420 --> 00:41:09.879 the visualization of merging the raw data.

712 00:41:09.879 --> 00:41:14.060 And this is the performance of our methods.

713 00:41:14.060 --> 00:41:19.060 And for comparison, we compare is, another very popular use,

714 00:41:20.597 --> 00:41:23.580 data integration method called Seurat.

715 00:41:23.580 --> 00:41:28.550 So Seurat is the software, the most often used software,

716 00:41:28.550 --> 00:41:30.530 for single-cell RNA sequencing analysis.

717 00:41:30.530 --> 00:41:32.220 Their lab have different,

718 00:41:32.220 --> 00:41:34.950 have developed a series of computational tools

719 00:41:34.950 --> 00:41:37.630 for analyZ(i)ng the single-cell RNA sequencing data.

720 00:41:37.630 --> 00:41:40.340 And this is from their integration methods.

721 00:41:40.340 --> 00:41:43.640 So you can see that both methods can,

722 00:41:43.640 --> 00:41:46.533 is able to integrate the both two datasets,

723 00:41:49.550 --> 00:41:51.690 but for some details, I think,

724 00:41:51.690 --> 00:41:54.580 because we are assuming this trajectory structure,

725 00:41:54.580 --> 00:41:56.830 we have a slightly better performance.

726 00:41:56.830 --> 00:42:00.553 For example, this group of cells are the layer one neurons,

727 00:42:01.476 --> 00:42:05.727 where the group of here, are here in Seurat.

728 00:42:06.690 --> 00:42:09.280 So you can see that because they come from,

729 00:42:09.280 --> 00:42:12.170 because the outer layer parts and the layer parts,

730 00:42:12.170 --> 00:42:16.000 come from two datasets.

731 00:42:16.000 --> 00:42:20.270 Because as I mentioned earlier in dataset B,

732 00:42:20.270 --> 00:42:24.280 they have collected cells from, at day 11.

733 00:42:24.280 --> 00:42:26.706 So this are, we can take a look

734 00:42:26.706 --> 00:42:30.133 of the collection days of each cell.

735 00:42:31.514 --> 00:42:33.605 So you can see that these cells, they are,

736 00:42:33.605 --> 00:42:35.455 the layer one parts come from day 11,

737 00:42:36.320 --> 00:42:38.850 And the rest parts is a mixture

738 00:42:38.850 --> 00:42:40.493 of cells from both two datasets.

739 00:42:41.921 --> 00:42:43.310 And by the way they all belong to the layer one.

740 00:42:43.310 --> 00:42:45.210 So we know that they belong to layer one

741 00:42:45.210 --> 00:42:48.210 by looking at the marker genes expression

742 00:42:48.210 --> 00:42:49.920 which I did not show here.

743 00:42:49.920 --> 00:42:54.920 So it's because we encourage the cells to align together

744 00:42:55.010 --> 00:43:00.010 if they are along the address, if they are similar cells.

745 00:43:02.610 --> 00:43:04.603 And you can see here, that's,

746 00:43:06.494 --> 00:43:10.480 so the two datasets, they have this interpolation

747 00:43:10.480 --> 00:43:14.270 of the pseudo, of the collection time.

748 00:43:14.270 --> 00:43:16.050 And you can see for example,

749 00:43:16.050 --> 00:43:18.320 for these projected cells,

750 00:43:18.320 --> 00:43:21.503 we can see this continuous positions,

751 00:43:23.730 --> 00:43:26.930 like alignments of the cells of different days

752 00:43:26.930 --> 00:43:31.930 from so the most the dark is the cells from day ten.

753 00:43:33.656 --> 00:43:36.248 And the red ones are the cells from day 18

754 00:43:36.248 --> 00:43:39.790 and even days are, are from dataset A,

755 00:43:39.790 --> 00:43:41.660 and odd days are from dataset B.

756 00:43:41.660 --> 00:43:43.150 So you can see that though they're coming

757 00:43:43.150 --> 00:43:46.310 from two different sources,

758 00:43:46.310 --> 00:43:50.733 we can, we are able to align them in the right order.

759 00:43:54.240 --> 00:43:55.073 And,

760 00:43:56.170 --> 00:43:58.270 and as another comparison.

761 00:43:58.270 --> 00:44:03.160 So we compare our estimation of shared trajectory,

762 00:44:03.160 --> 00:44:05.550 with another partisan approach

763 00:44:05.550 --> 00:44:10.550 which is we're first to do data integration with Seurat

764 00:44:11.500 --> 00:44:13.950 and then we can use Slingshots,

765 00:44:13.950 --> 00:44:18.147 to perform trajectory inference on the integrated data.

766 00:44:19.375 --> 00:44:20.630 And you can see that this,

767 00:44:20.630 --> 00:44:23.830 we have a much cleaner trajectory structure.

768 00:44:23.830 --> 00:44:28.830 And we also have a comparable computational cost.

769 00:44:29.090 --> 00:44:30.720 So Seurat and Slingshots,

770 00:44:30.720 --> 00:44:33.540 they cannot be, they do not need regularization.

771 00:44:33.540 --> 00:44:38.180 And with one CPU, they, it takes about 12 minutes.

772 00:44:38.180 --> 00:44:42.470 And for our accelerated VITAE,

773 00:44:42.470 --> 00:44:44.813 generating this figure, we have,

774 00:44:44.813 --> 00:44:46.750 we can, we take about three minutes

775 00:44:46.750 --> 00:44:51.750 on one GPU port at about 10 minutes on eight CPU cores

776 00:44:51.859 --> 00:44:54.770 which is, the eight CPU cores are like currently,

777 00:44:54.770 --> 00:44:56.550 like most of our laptops,

778 00:44:56.550 --> 00:45:00.280 but we'll have such computational resources.

779 00:45:00.280 --> 00:45:03.940 So we have comparable computation cost

780 00:45:03.940 --> 00:45:06.373 with this state of our methods.

781 00:45:07.270 --> 00:45:10.660 And in addition, because we are...

782 00:45:10.660 --> 00:45:14.330 Based on this approximated posterior distributions

783 00:45:14.330 --> 00:45:18.761 we also have some kind of uncertainty quantifications

784 00:45:18.761 --> 00:45:20.020 on the cell positions.

785 00:45:20.020 --> 00:45:23.143 For example, here, it shall say some parts of the cells,

786 00:45:24.300 --> 00:45:26.120 these cell positions

787 00:45:26.120 --> 00:45:29.573 along the trajectory are not very reliable.

788 00:45:31.660 --> 00:45:36.660 And that will help us to evaluate our, the estimate,

789 00:45:36.960 --> 00:45:40.290 how we think our estimate in pseudotime.

790 00:45:41.710 --> 00:45:43.610 And finally,

791 00:45:43.610 --> 00:45:47.820 this is showing some gene expression change

792 00:45:48.744 --> 00:45:50.030 along the pseudotime order.

793 00:45:50.030 --> 00:45:54.550 And, and we look at some top markers

794 00:45:54.550 --> 00:45:57.030 that are changing along the pseudotime order

795 00:45:57.030 --> 00:46:02.030 for some trajectories in the whole trajectory structure.

796 00:46:03.110 --> 00:46:05.250 And you can see,

797 00:46:05.250 --> 00:46:10.250 here we separately fish the curve for two datasets,

798 00:46:10.550 --> 00:46:13.260 but you can see that they overlap

799 00:46:13.260 --> 00:46:15.360 with each other quite well.

800 00:46:15.360 --> 00:46:18.220 And so that's also an evidence showing

801 00:46:18.220 --> 00:46:20.563 that we can have a good,

802 00:46:23.032 --> 00:46:26.140 a good performance in aligning the two datasets.

803 00:46:28.160 --> 00:46:30.753 So the take home message is,

804 00:46:30.753 --> 00:46:35.753 first we perform this model-based trajectory inference,

805 00:46:36.170 --> 00:46:38.250 to understand cell dynamics.

806 00:46:38.250 --> 00:46:41.200 And our, the second is our methods.

807 00:46:41.200 --> 00:46:43.460 So our method is a model-based approach.

808 00:46:43.460 --> 00:46:48.460 We can combine the mixture prior model, Oh, sorry.

809 00:46:49.990 --> 00:46:53.556 We can combine the collected mixture structure

810 00:46:53.556 --> 00:46:57.600 for defining the trajectory structure

811 00:46:57.600 --> 00:47:00.410 with the variational autoencoders

812 00:47:00.410 --> 00:47:03.333 so that we can efficiently,

813 00:47:04.710 --> 00:47:07.940 efficiently solve the mixture model

814 00:47:07.940 --> 00:47:11.910 and have enough flexibility to fit the data well.

815 00:47:11.910 --> 00:47:14.060 And so our,

816 00:47:14.060 --> 00:47:16.970 trajectory inference approach features,

817 00:47:16.970 --> 00:47:19.220 the analysis of integrating

818 00:47:19.220 --> 00:47:21.900 multiple single-cell RNA sequencing datasets.

819 00:47:21.900 --> 00:47:25.590 And if you are anxious to know more details,

820 00:47:25.590 --> 00:47:29.310 we have our paper, a manuscript,

821 00:47:29.310 --> 00:47:32.740 a manuscript already available on bio archives

822 00:47:32.740 --> 00:47:37.740 and we also have our package codes on VITAE.

823 00:47:38.050 --> 00:47:39.640 And that's all. Thank you.

824 00:47:39.640 --> 00:47:44.401 And if you have any questions, I'm happy to answer them.

825 00:47:44.401 --> 00:47:47.353 - Thanks Jingshu for this excellent, excellent talk.

826 00:47:49.090 --> 00:47:50.340 I wonder whether the audience,

827 00:47:50.340 --> 00:47:52.240 have any questions for Jingshu.

828 00:47:58.112 --> 00:48:01.550 Okay. So Jingshu I have some maybe minor questions.

829 00:48:01.550 --> 00:48:03.650 I recall that in the model,

830 00:48:03.650 --> 00:48:07.410 you have this actual term encouraging X,

831 00:48:07.410 --> 00:48:11.680 the data explained by X the covariates,

832 00:48:11.680 --> 00:48:13.228 the confounding covariates,

833 00:48:13.228 --> 00:48:18.228 to be orthogonal to the leading factor, right?

834 00:48:18.620 --> 00:48:21.330 And then there is a penalty term alpha.

835 00:48:21.330 --> 00:48:22.693 So I wonder,

836 00:48:24.980 --> 00:48:28.280 what's the motivation for you to including this term

837 00:48:28.280 --> 00:48:33.009 instead of first removing the effect from the i directly.

838 00:48:33.009 --> 00:48:36.473 And in practice, how should we have set alpha?

839 00:48:38.940 --> 00:48:42.770 - So, so, so, so the thing is a bit tricky here

840 00:48:44.170 --> 00:48:45.194 from a statistical point of view.

841 00:48:45.194 --> 00:48:50.194 So we want to remove these confounding effects, right?

842 00:48:50.960 --> 00:48:53.320 But the other fact is that,

843 00:48:53.320 --> 00:48:55.930 these confounding effects, the X,

844 00:48:55.930 --> 00:48:58.850 are not exactly orthogonal with Z,

845 00:48:58.850 --> 00:49:03.140 because for instance for the two datasets that I have,

846 00:49:03.140 --> 00:49:06.240 we cannot say that the signal is completely orthogonal

847 00:49:06.240 --> 00:49:08.630 to each dataset they have come from

848 00:49:08.630 --> 00:49:11.989 because there are two biological differences

849 00:49:11.989 --> 00:49:12.950 between the two datasets.

850 00:49:12.950 --> 00:49:14.293 So, so here,

851 00:49:16.877 --> 00:49:18.900 so here, the problem is not completely identifiable

852 00:49:18.900 --> 00:49:21.870 but people do it in practice a lot.

853 00:49:21.870 --> 00:49:26.870 So we want to kind of decorrelate Z and X to some extent

854 00:49:29.250 --> 00:49:31.840 so that we can remove,

855 00:49:31.840 --> 00:49:36.370 remove the batch effects that we do not want

856 00:49:36.370 --> 00:49:38.690 but keep the two biological differences.

857 00:49:38.690 --> 00:49:41.410 So I think the underlying assumption

858 00:49:41.410 --> 00:49:43.590 the big assumption is that we are assuming

859 00:49:43.590 --> 00:49:48.328 that the two biological differences are large enough

860 00:49:48.328 --> 00:49:50.002 so that compared to the...

861 00:49:50.002 --> 00:49:52.918 So we remove the smaller differences, the batch effects

862 00:49:52.918 --> 00:49:56.010 but we can still pick the two biological differences,

863 00:49:56.010 --> 00:49:56.980 to some extent.

864 00:49:56.980 --> 00:50:01.030 So there's no guarantee that it will work,

865 00:50:01.030 --> 00:50:05.808 but in practice, it work on a lot of datasets.

866 00:50:05.808 --> 00:50:06.641 I think that's...

867 00:50:08.485 --> 00:50:11.110 So we will inherit this idea from this paper

868 00:50:11.110 --> 00:50:16.110 by Nancy Huang and her students.

869 00:50:16.300 --> 00:50:19.410 So I think in removing the batch effects.

870 00:50:19.410 --> 00:50:21.290 So I think the idea is that,

871 00:50:21.290 --> 00:50:25.027 we hope that it can work for a lot of datasets.

872 00:50:25.027 --> 00:50:28.340 And the reason that we want to have this penalty,

873 00:50:28.340 --> 00:50:31.580 is that if we don't add any penalty,

874 00:50:31.580 --> 00:50:36.210 then, because this autoencoder is trained by this,

875 00:50:36.210 --> 00:50:38.730 by this stochastic gradient descent.

876 00:50:38.730 --> 00:50:42.863 So sometime it may not find the optimal global solution.

877 00:50:43.860 --> 00:50:48.860 So if we don't encourage it, the X and Z to be decorrelated,

878 00:50:48.880 --> 00:50:50.600 it sometimes may not be able,

879 00:50:50.600 --> 00:50:53.770 it may give you a solution that is not,

880 00:50:53.770 --> 00:50:57.610 that's the Z still are highly correlated with X

881 00:50:57.610 --> 00:50:59.320 and the batch effects are still there.

882 00:50:59.320 --> 00:51:04.297 So then this alpha, I think in practice we set it to be 0.0,

883 00:51:05.850 --> 00:51:10.850 so it's a very small penalty so that we can put some,

884 00:51:11.180 --> 00:51:15.910 some kind of penalty to regularize that.

885 00:51:15.910 --> 00:51:18.104 - Small amount, I guess you mentioned.

886 00:51:18.104 --> 00:51:19.313 - Yes, yes.

887 00:51:20.670 --> 00:51:22.007 And it may be that does not work,

888 00:51:22.007 --> 00:51:25.930 and then in practice you can choose another alpha and try it

889 00:51:25.930 --> 00:51:29.787 and see if it gives you the best results that you want.

890 00:51:29.787 --> 00:51:31.930 - Right. So, so I want...

891 00:51:31.930 --> 00:51:33.590 I guess, right.

892 00:51:33.590 --> 00:51:35.770 So I guess my question is like,

893 00:51:35.770 --> 00:51:40.250 since it's not entirely a supervised problem,

894 00:51:40.250 --> 00:51:41.650 like how, right.

895 00:51:41.650 --> 00:51:45.170 So I'm not sure how to check,

896 00:51:45.170 --> 00:51:48.250 what is a good alpha in the sense,

897 00:51:48.250 --> 00:51:50.622 but if you tell me like a small alpha,

898 00:51:50.622 --> 00:51:51.455 well you're going to be fine

899 00:51:51.455 --> 00:51:53.663 then I just take it to be small alpha.

900 00:51:53.663 --> 00:51:56.550 - Yeah. I think the way you check it is that,

901 00:51:56.550 --> 00:52:00.333 for example the way we check it here is that,

902 00:52:01.820 --> 00:52:04.340 so sometimes we have some referenced cell types,

903 00:52:04.340 --> 00:52:08.600 so that, you know roughly what you are doing.

904 00:52:08.600 --> 00:52:10.020 So here, for example,

905 00:52:10.020 --> 00:52:12.513 here this reference cell types, these are,

906 00:52:13.737 --> 00:52:17.930 these are not used in the modeling approach,

907 00:52:17.930 --> 00:52:20.900 so these are for evaluation the performance.

908 00:52:20.900 --> 00:52:23.300 And for some datasets, you can't,

909 00:52:23.300 --> 00:52:25.660 we can mark our genes and do some class points,

910 00:52:25.660 --> 00:52:27.803 so, you know, roughly like which though,

911 00:52:29.136 --> 00:52:32.070 (indistinct)

912 00:52:32.070 --> 00:52:33.870 and for two datasets, you can see,

913 00:52:33.870 --> 00:52:37.350 like whether you can correctly merge the cell types

914 00:52:37.350 --> 00:52:39.780 that are shared among the datasets

915 00:52:40.847 --> 00:52:41.680 but keep the cell types

916 00:52:41.680 --> 00:52:44.533 that are unique to different cells.

917 00:52:45.585 --> 00:52:48.840 Then for our trajectory inference is slightly complicated

918 00:52:48.840 --> 00:52:51.890 because these cell types are not well separated.

919 00:52:51.890 --> 00:52:55.150 So another way that we can check our performance,

920 00:52:55.150 --> 00:52:58.717 is that you can see here that we can correctly,

921 00:52:59.900 --> 00:53:02.560 for these projected cells we can correctly like,

922 00:53:02.560 --> 00:53:06.460 order the wrong days in the right order.

923 00:53:06.460 --> 00:53:07.660 So that we know that we keep

924 00:53:07.660 --> 00:53:11.273 some biological meaningful signals there.

925 00:53:12.893 --> 00:53:16.033 I think there still can be some bias. Yeah.

926 00:53:17.670 --> 00:53:19.023 - Okay, great. Thanks.

927 00:53:20.870 --> 00:53:25.230 So, thanks Jingshu again for this excellent talk.

928 00:53:25.230 --> 00:53:28.153 And if you have any question you want to ask Jingshu

929 00:53:28.153 --> 00:53:31.260 that you cannot think about for now,

930 00:53:31.260 --> 00:53:33.280 you can always email her offline

931 00:53:33.280 --> 00:53:35.520 and if you want to use her software,

932 00:53:35.520 --> 00:53:38.530 I think she'll be more than happy to answer your question.

933 00:53:38.530 --> 00:53:39.363 - Yes. Yes.

934 00:53:40.582 --> 00:53:41.960 (chuckles)

935 00:53:41.960 --> 00:53:43.693 - So I guess that's all for today.

936 00:53:45.640 --> 00:53:47.000 Thank you everyone joining.

937 00:53:47.000 --> 00:53:49.260 Thank you Jingshu for being here,

938 00:53:49.260 --> 00:53:51.513 and have a nice remaining day.