2 - Noise correction and genotype labeling

Code ▾

This is the second part of the tutorial for processing GoT-ChA genotyping libraries. The Gotcha R package required for this tutorial can be downloaded here.

Use of GotchaLabeling function

This is the second part of the pipeline, which starts from the wild type and mutant read counts as obtained in the *"1- How to run Gotcha in slurm clusters with parallel computing"* tutorial shown here.

First, we load the Gotcha R package and other packages required for this tutorial:

Hide

```
library(Gotcha)
```

```
Checking if r-reticulate-gotcha virtual environment is available...
Virutal environment r-reticulate-gotcha is available | use_virtualenv(r-reticulate-gotcha)
```

To perform the noise correction and genotype assignment, we run the *GotchaLabeling* function. This can be run on the full list of detected barcodes (including both empty droplets and true cells). This takes longer, but can help identify the cluster of non-genotyped cells when performing the cluster assignment. Here, we run the function only on barcodes that were assigned as true cells based on the scATAC-seq data. The data used as input is generated by the *BatchMutationCalling* function followed by the *MergeGotchaOuts* function, as shown in the "Running Gotcha with parallel computing in slurm" tutorial.

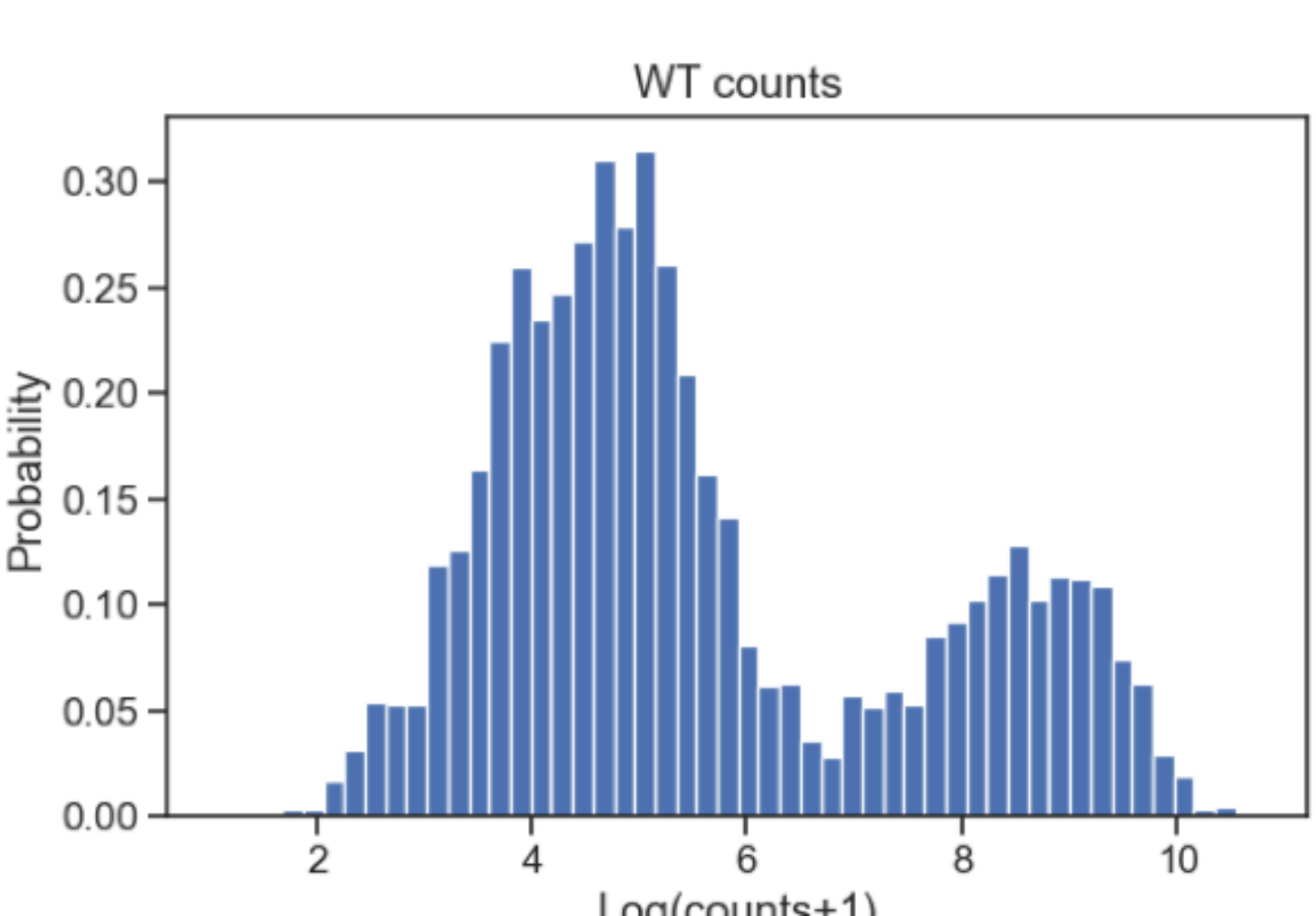
The metadata used as input should contain the following columns: Sample, GeneName_WTcount, and GeneName_MUTcount as shown below:

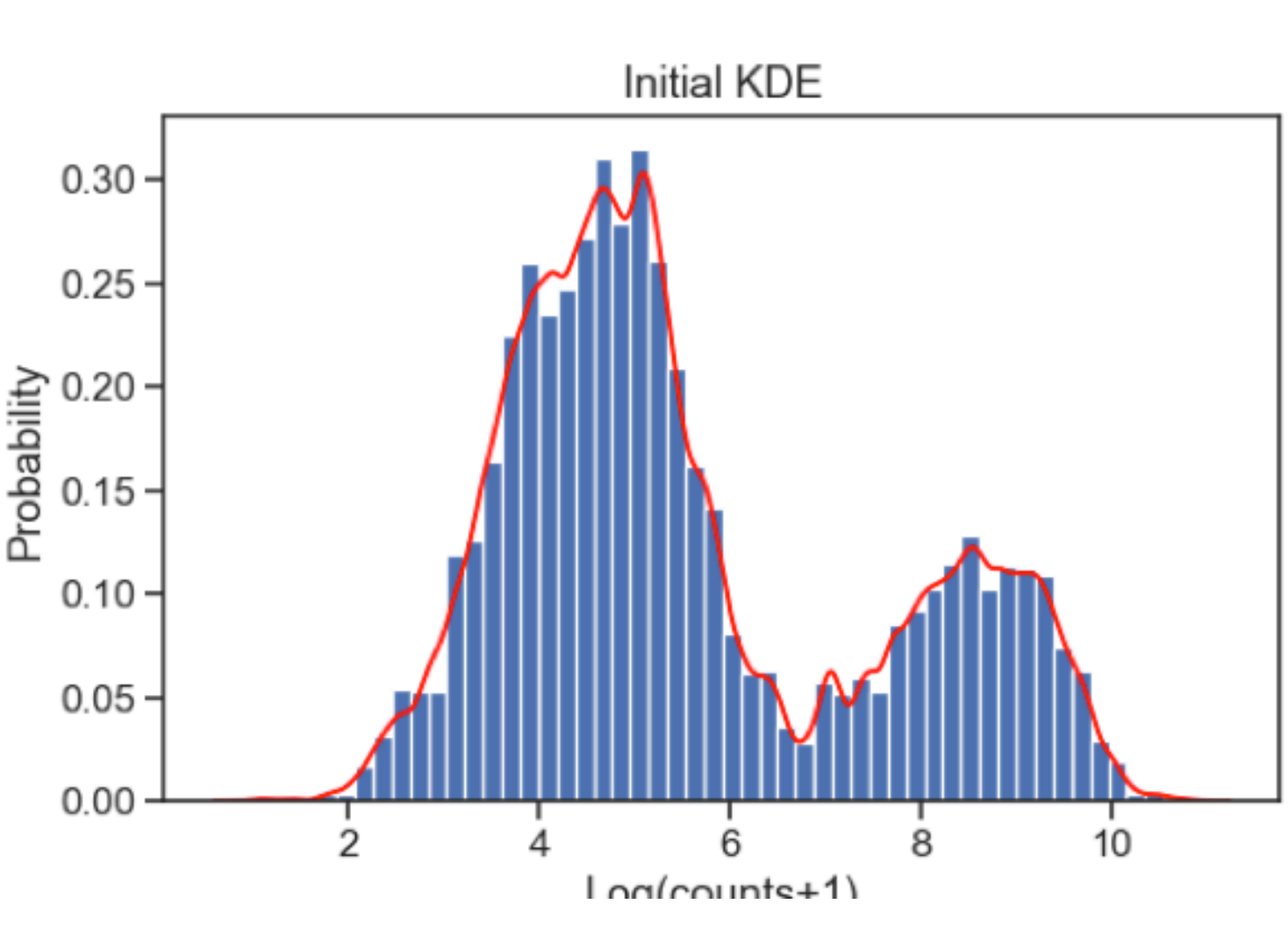
X	Sample	TP53_WTcount	TP53_MUTcount
RM30#AACATCGAGGCCTCGT-1	RM30	2383	136
RM30#CCCACATAGGCGATTG-1	RM30	112	2920
RM30#CAGCTAATCTATCCTA-1	RM30	311	771
RM30#GCACGGTGTCGAGGAT-1	RM30	548	2582
RM30#CCAATGACAGGCAGAT-1	RM30	214	17655
RM30#TTGTTGTCAGGTAACG-1	RM30	251	493

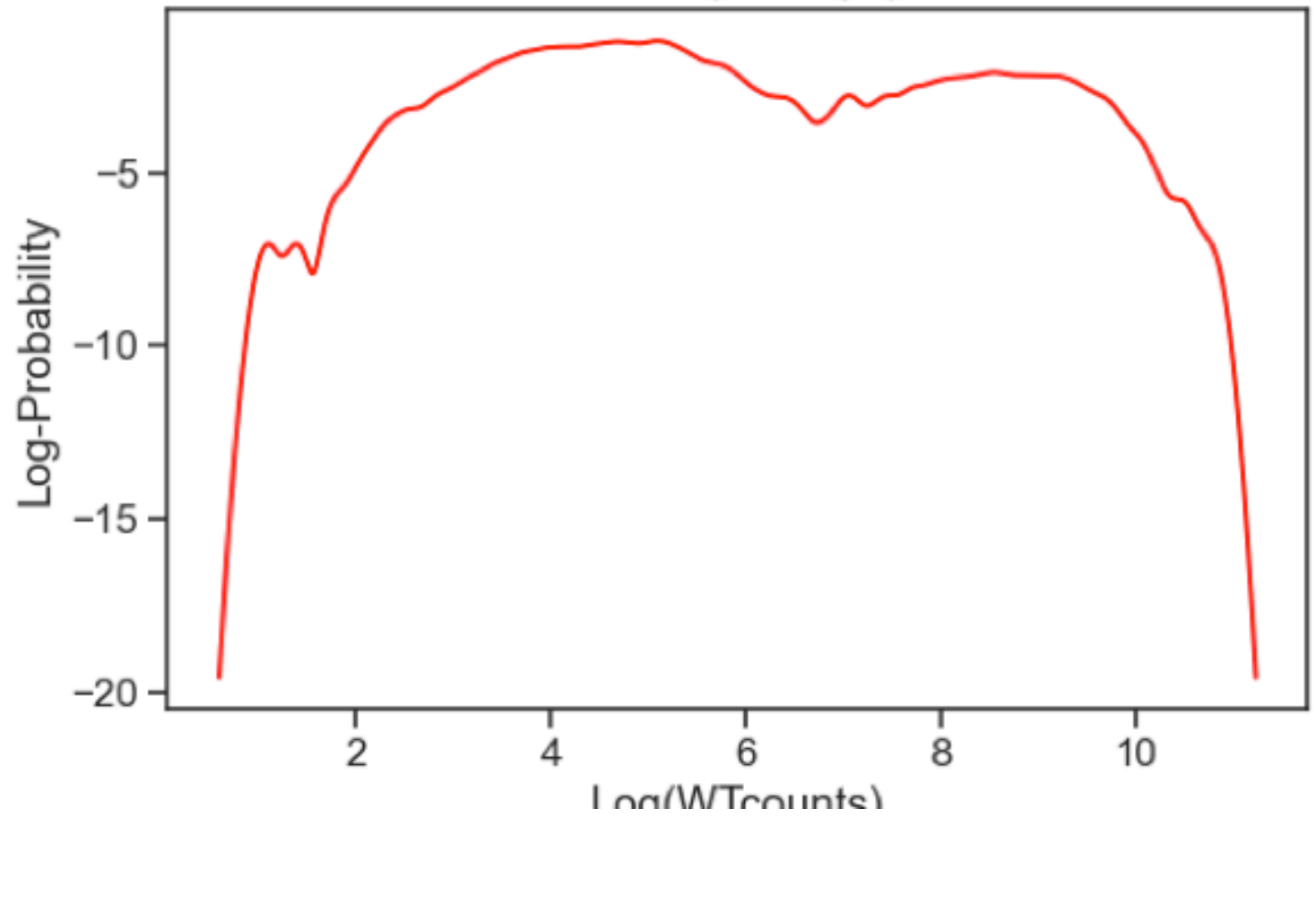
To run the *GotchaLabeling* function:

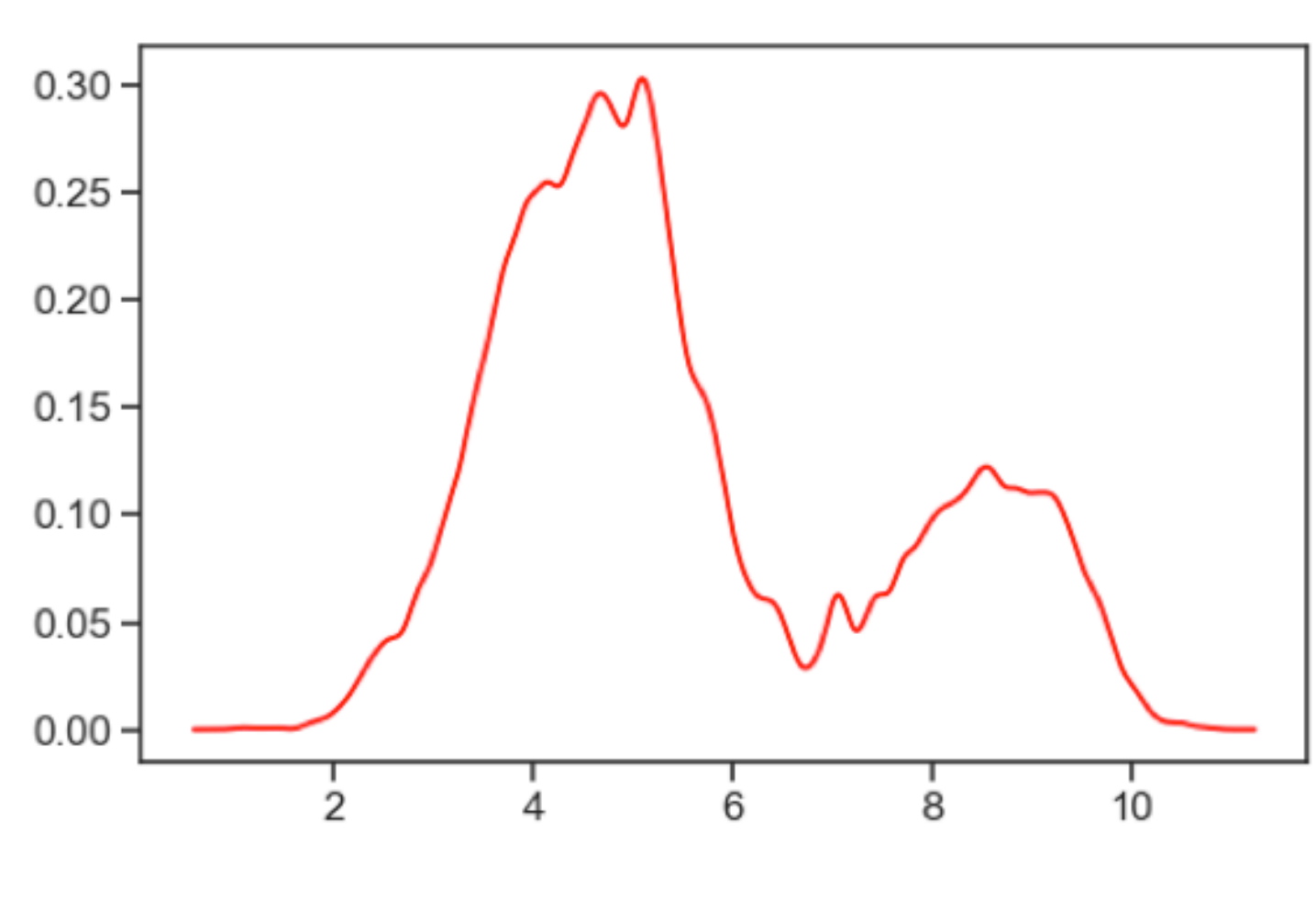
Hide

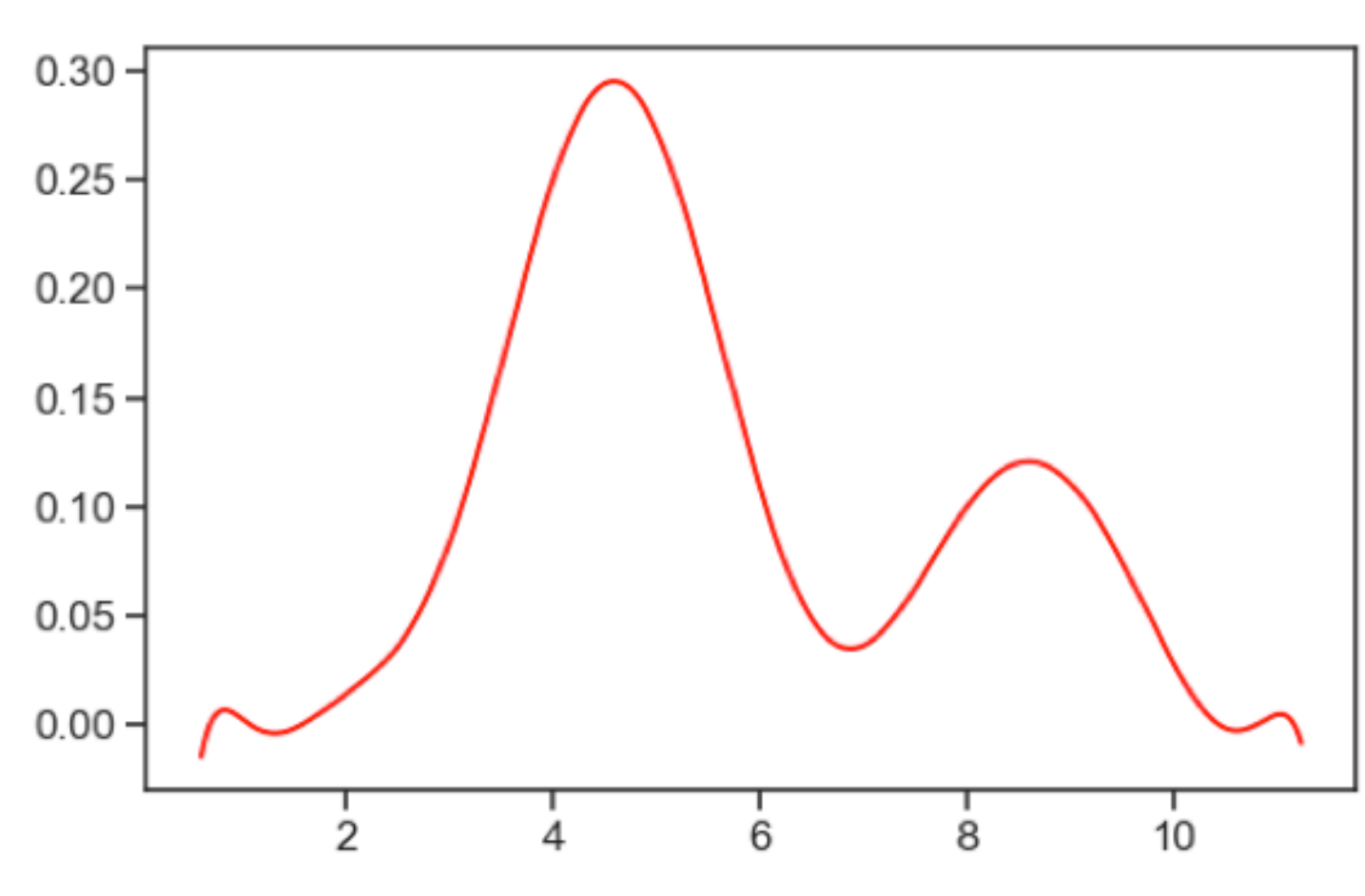
```
genotypes <- GotchaLabeling(path = "/Users/francoizzo/Documents/GoTChA_2022/Tutorial/",
                            infile = "metadata_cells.csv",
                            gene_id = "TP53",
                            sample_id = "RM30")
```

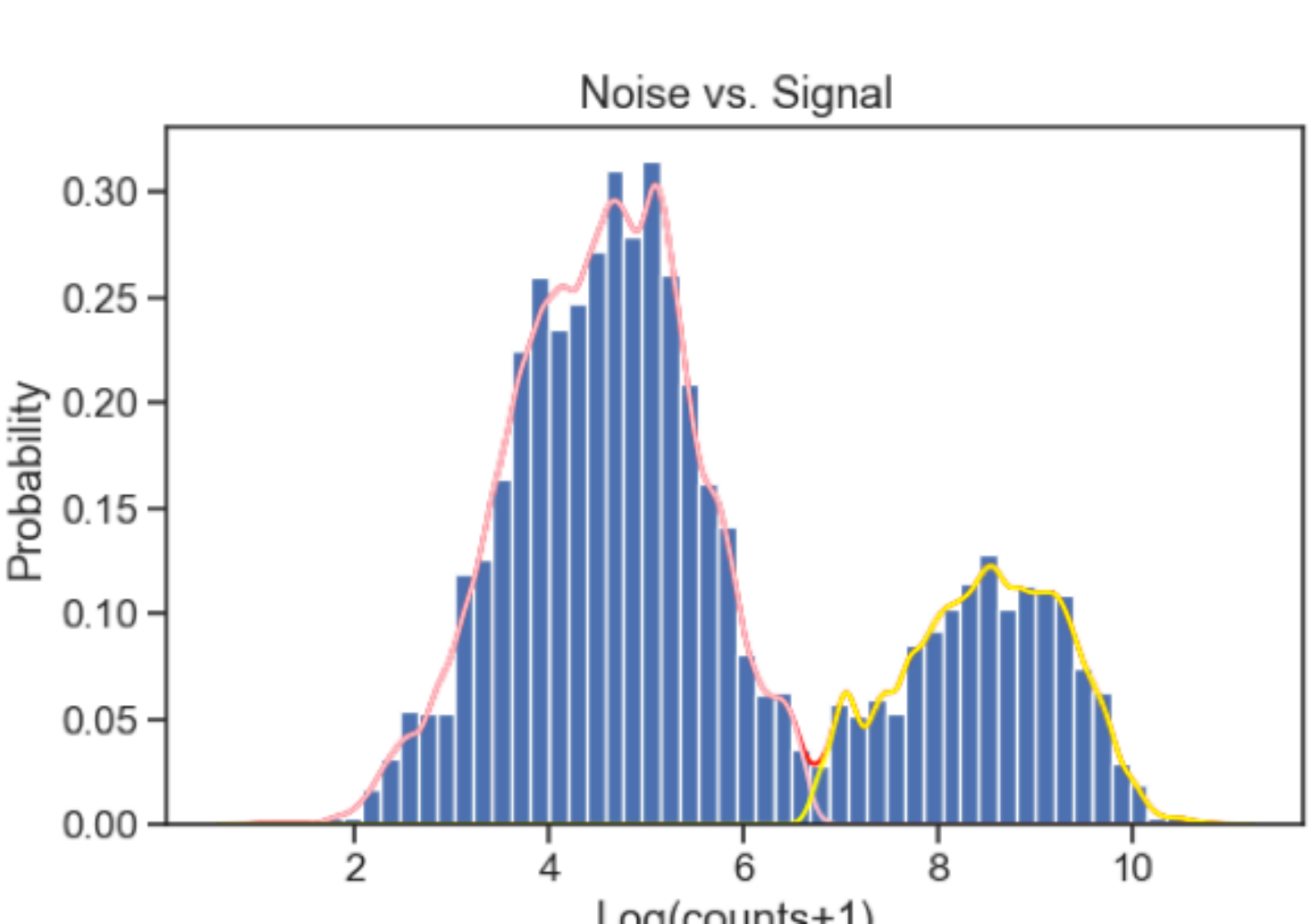


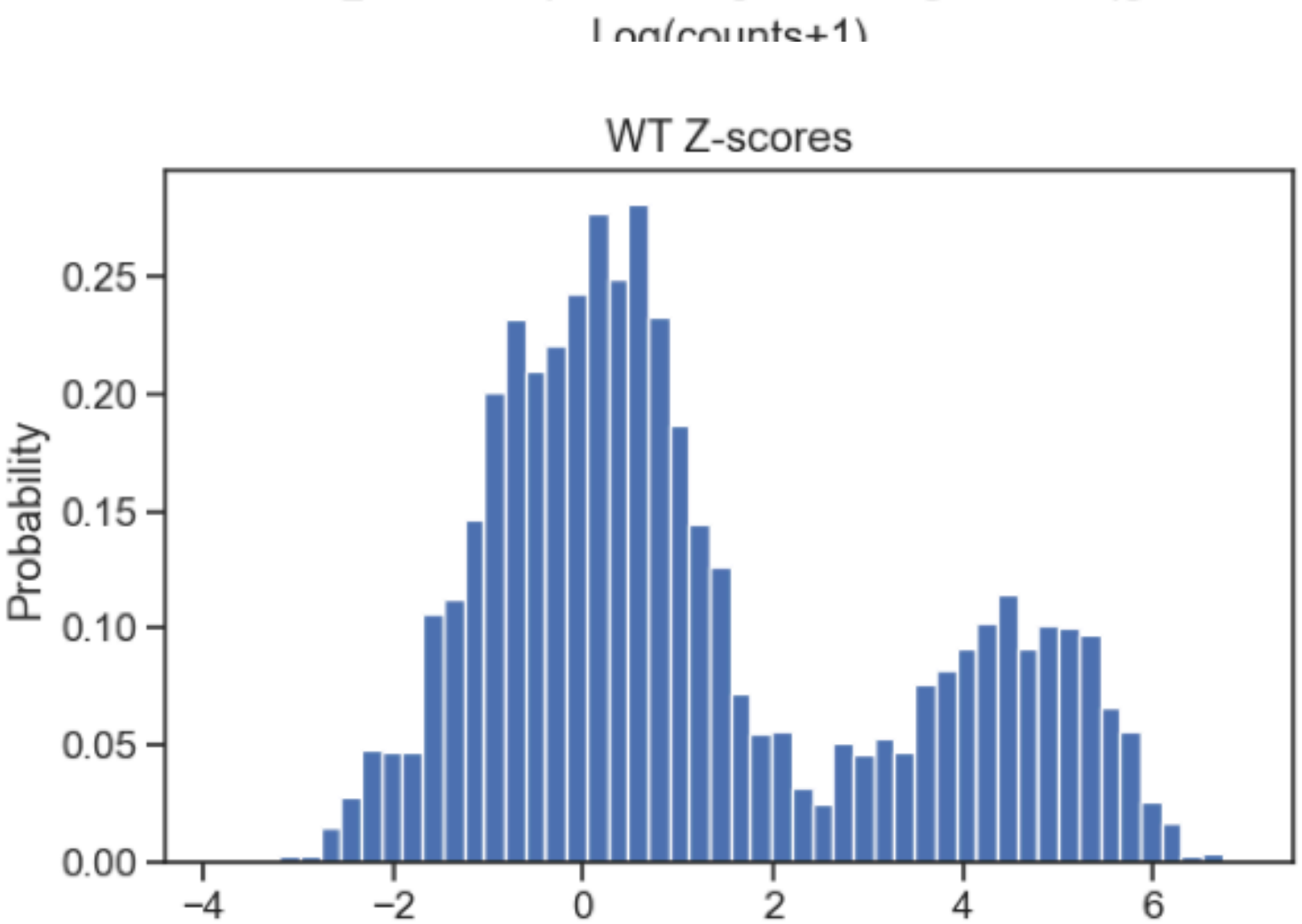


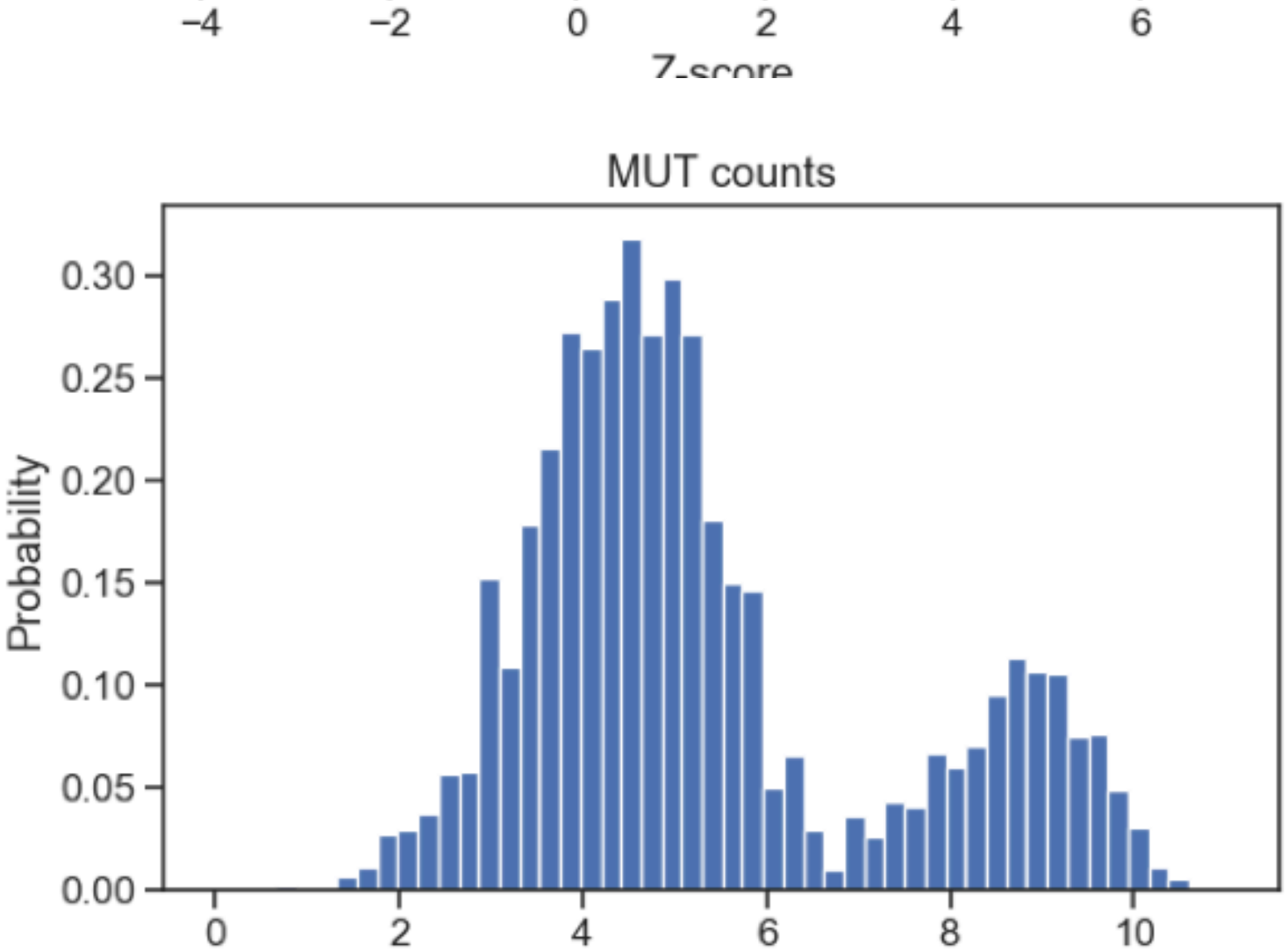


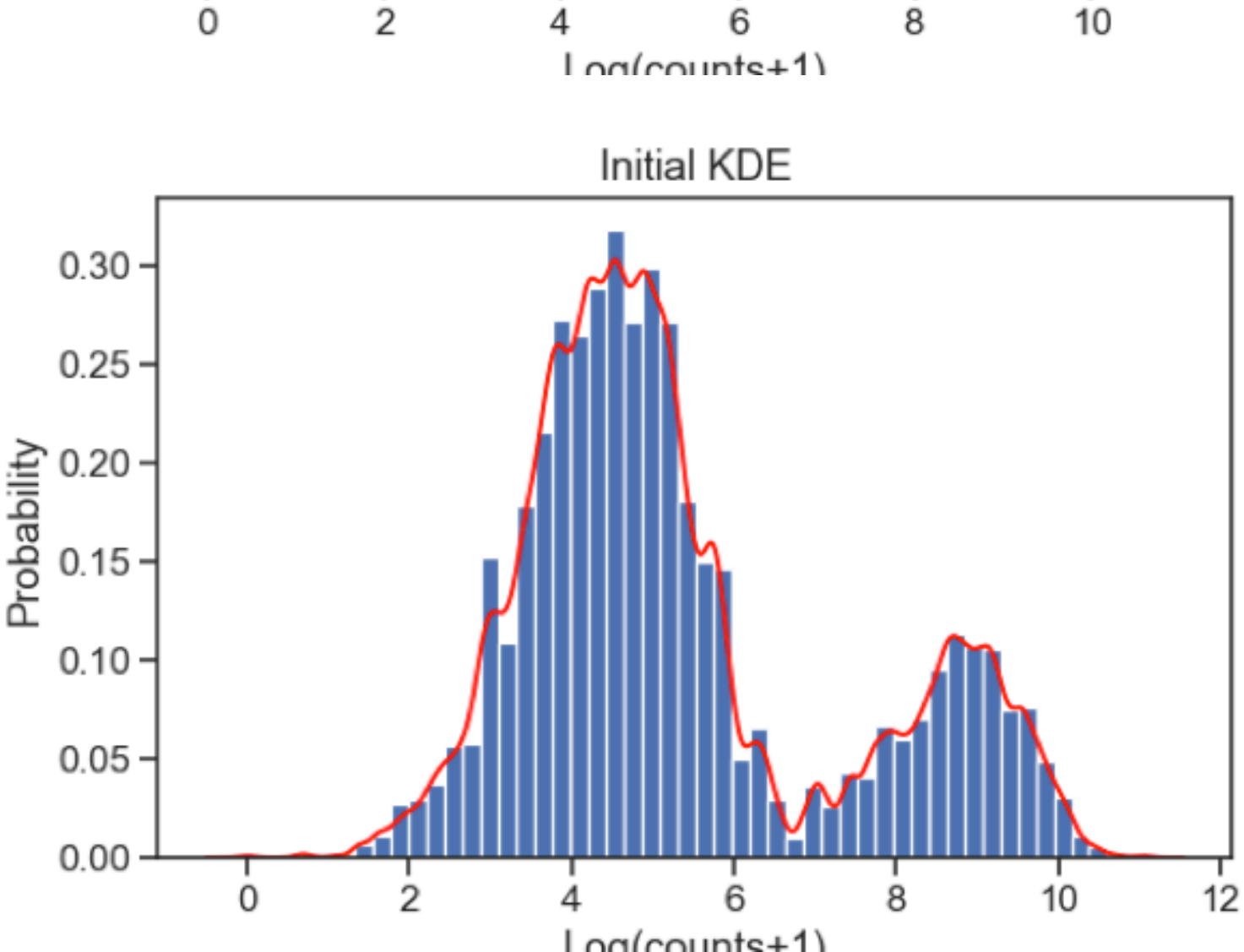


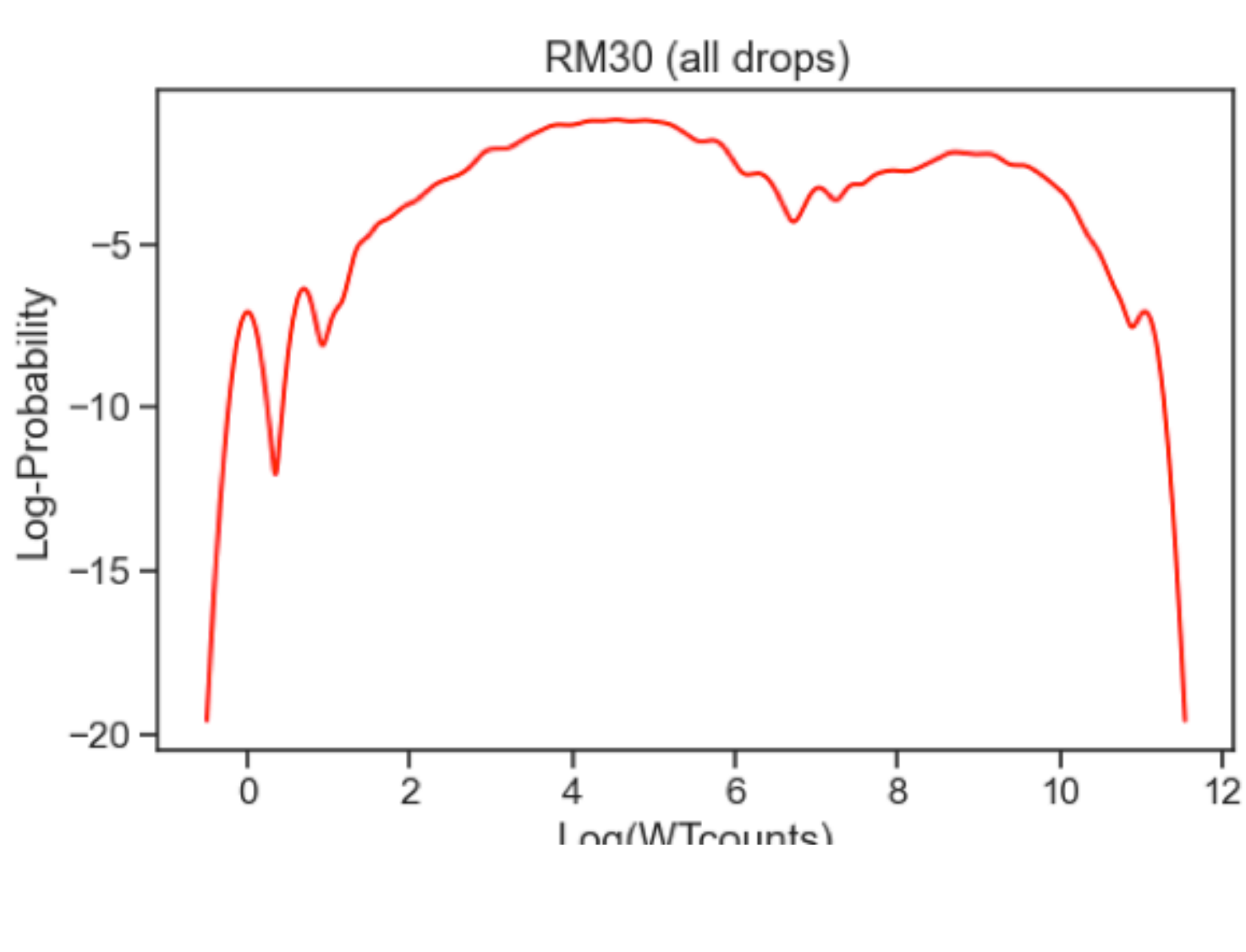


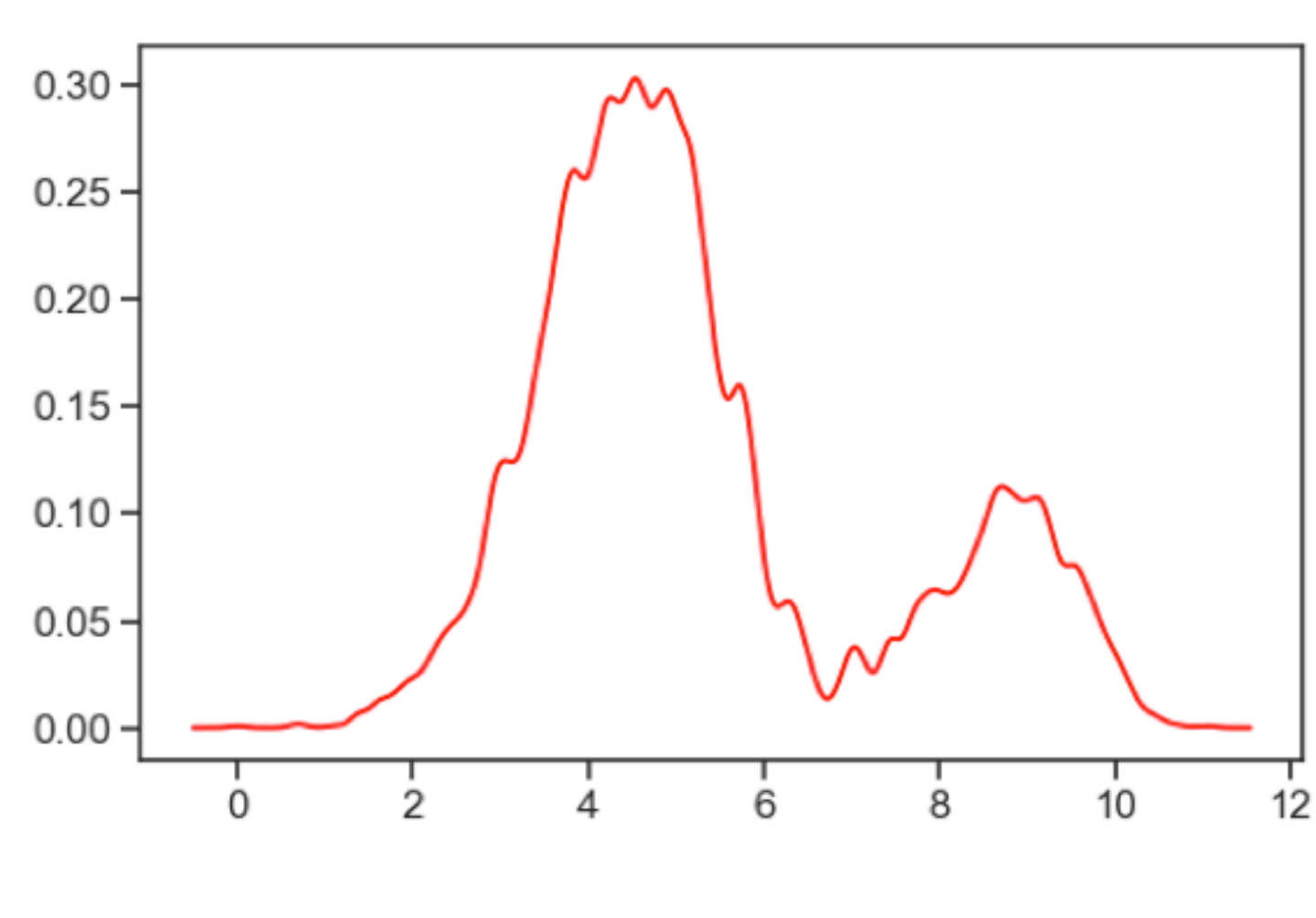


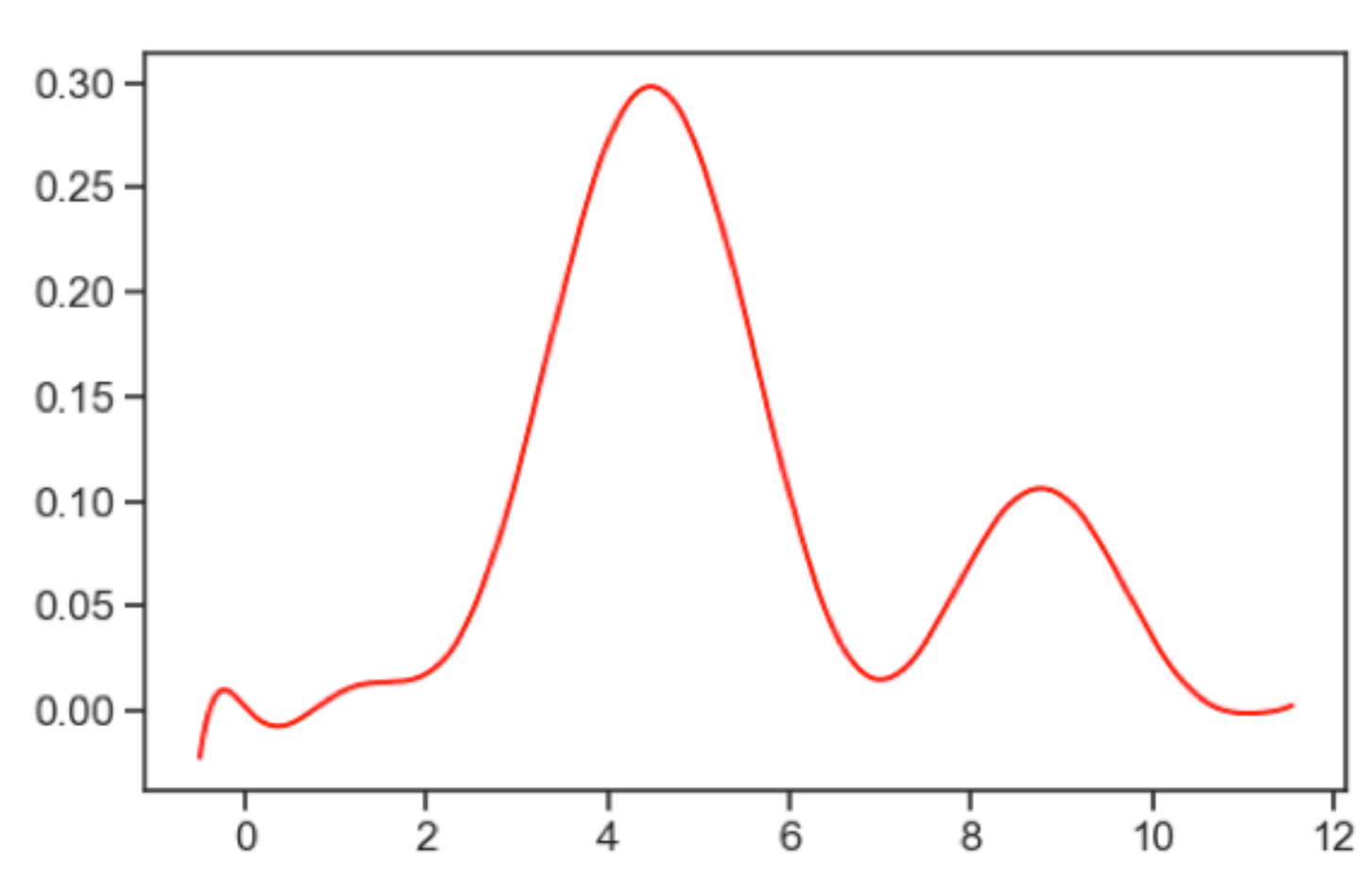


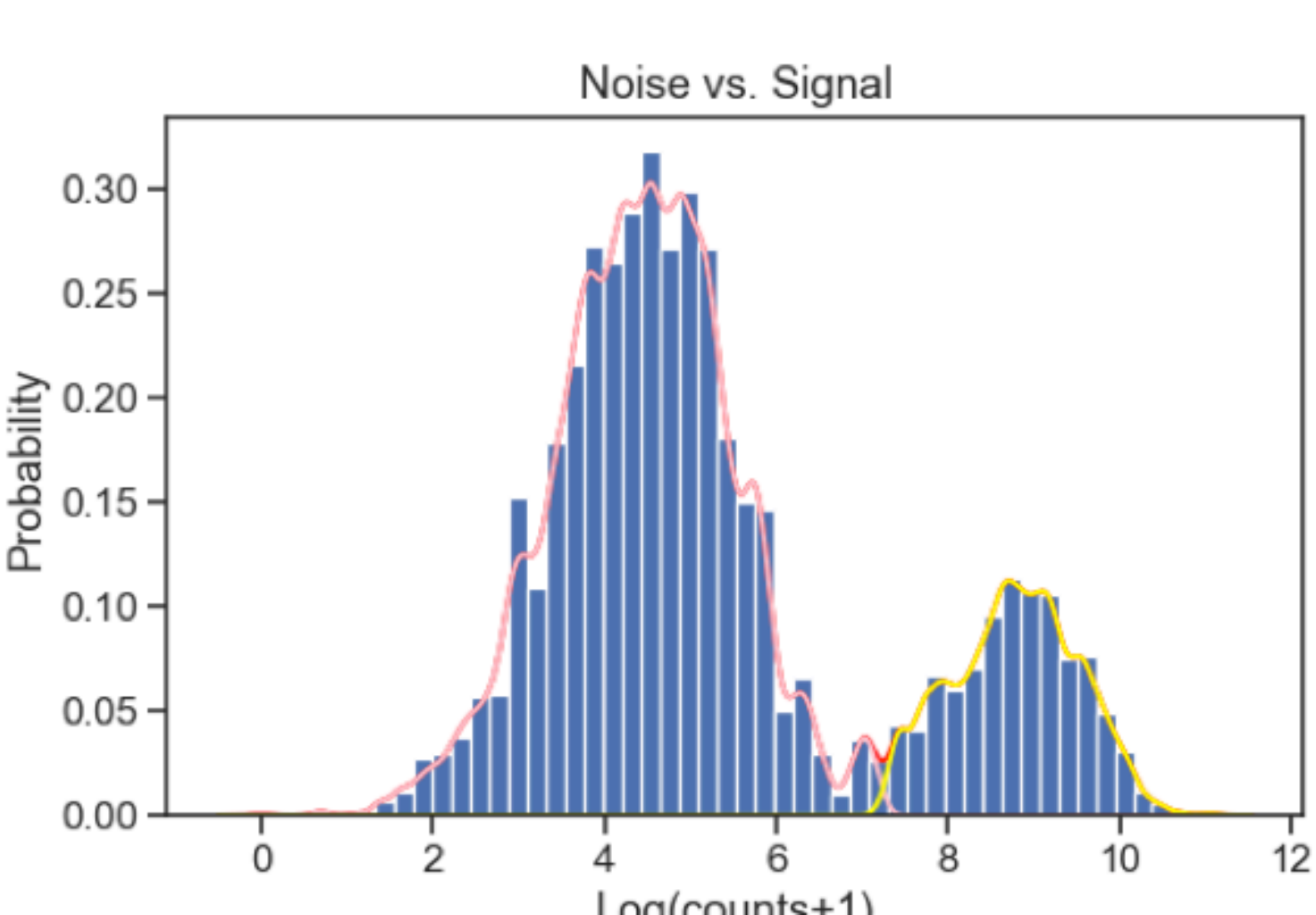


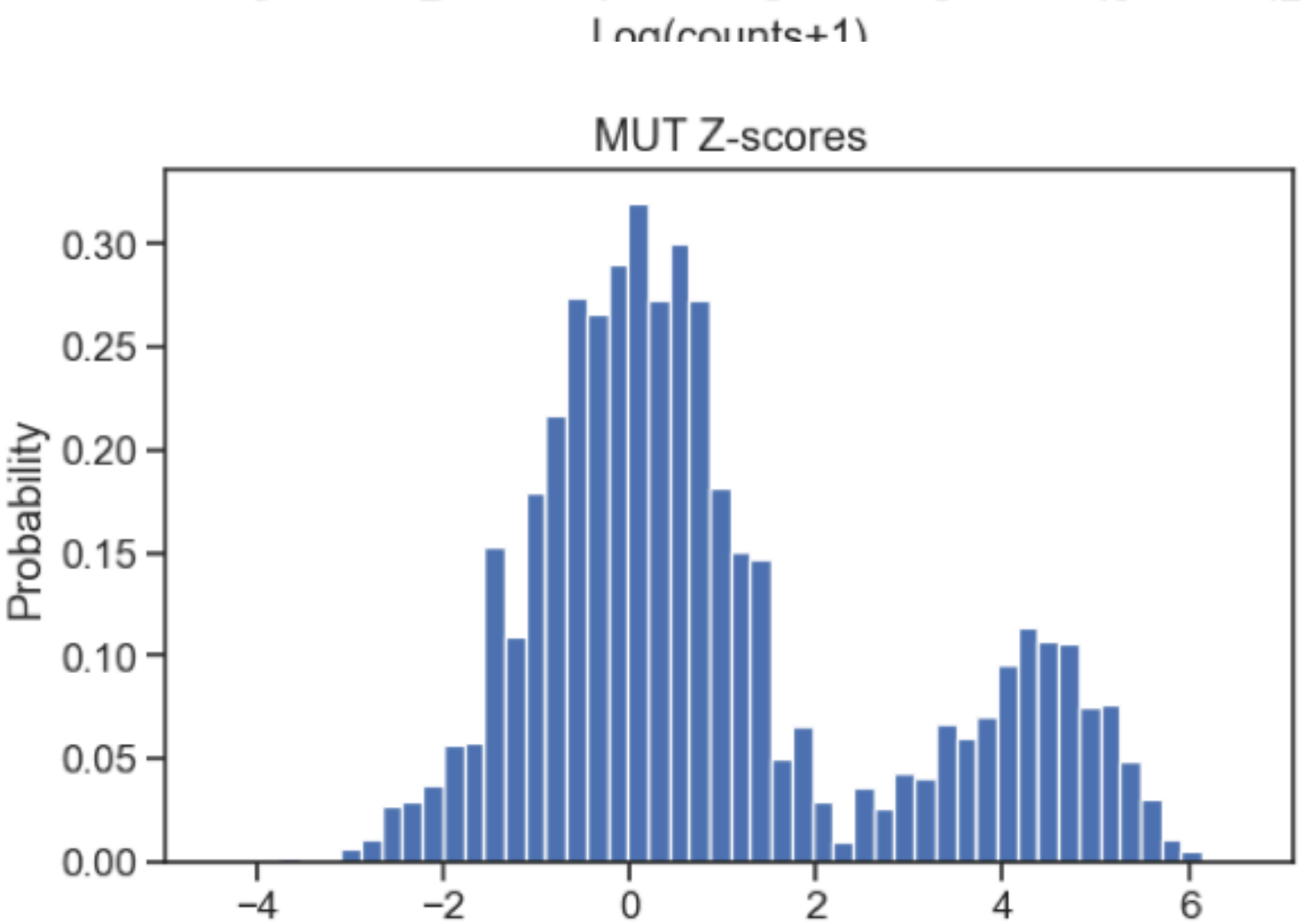


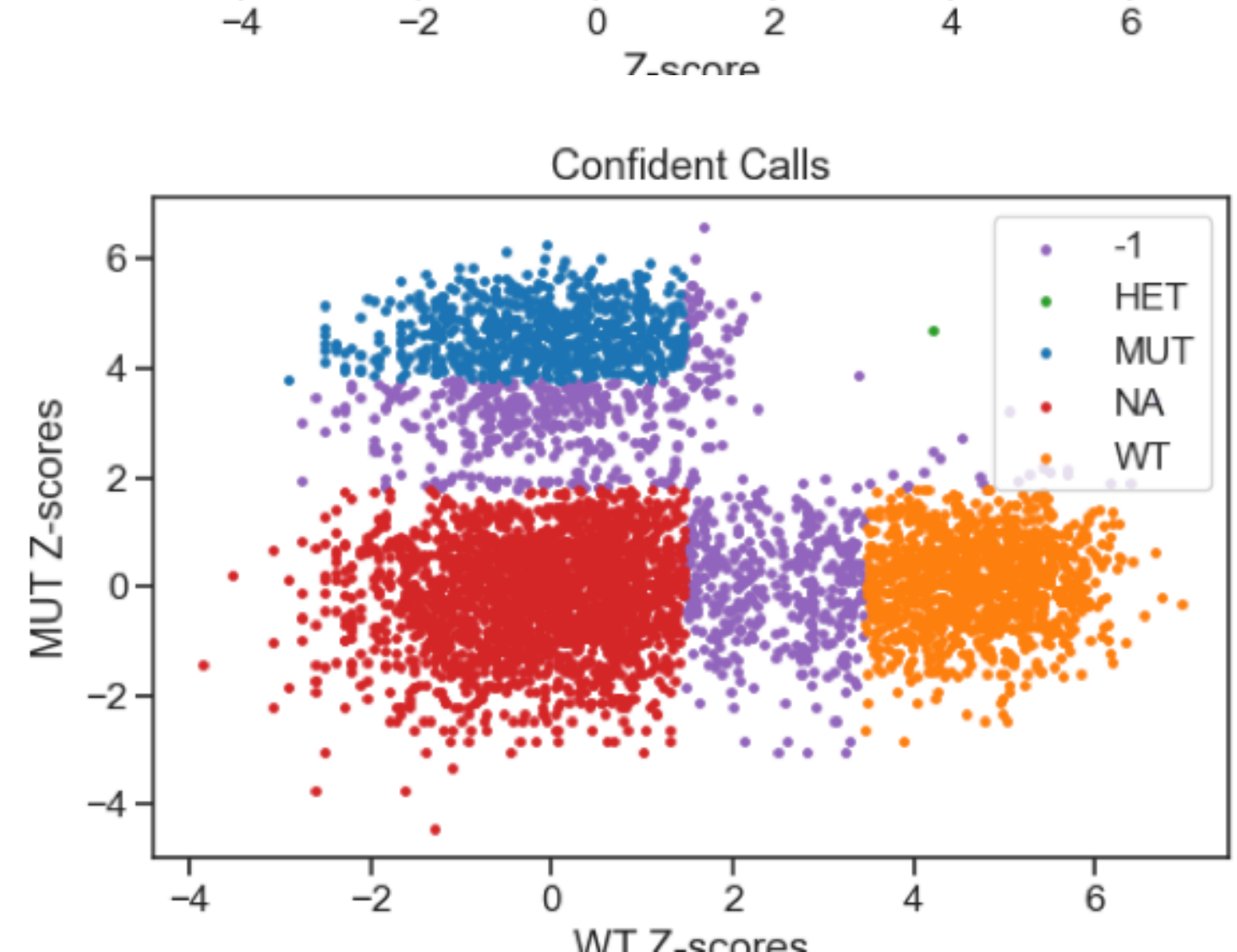


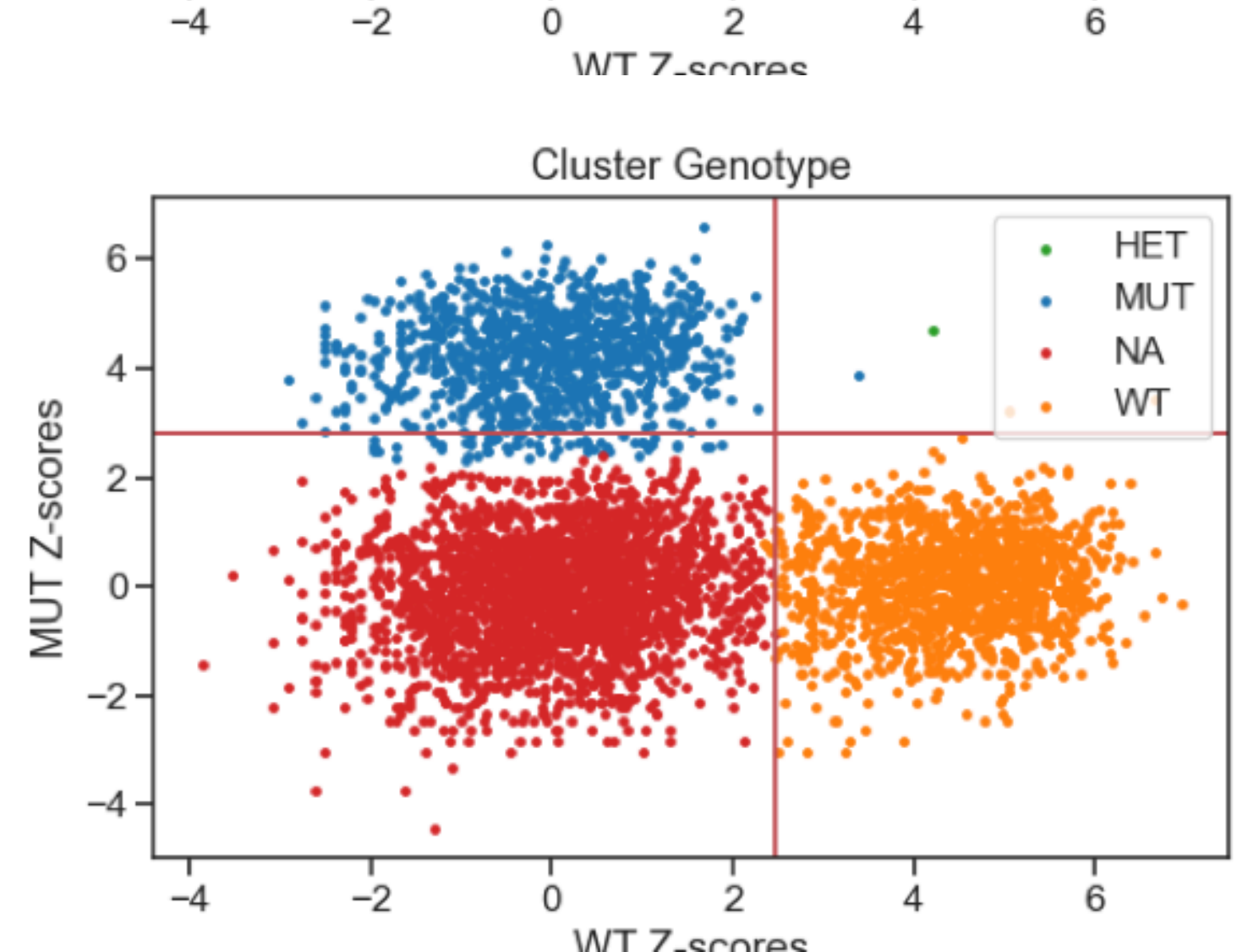




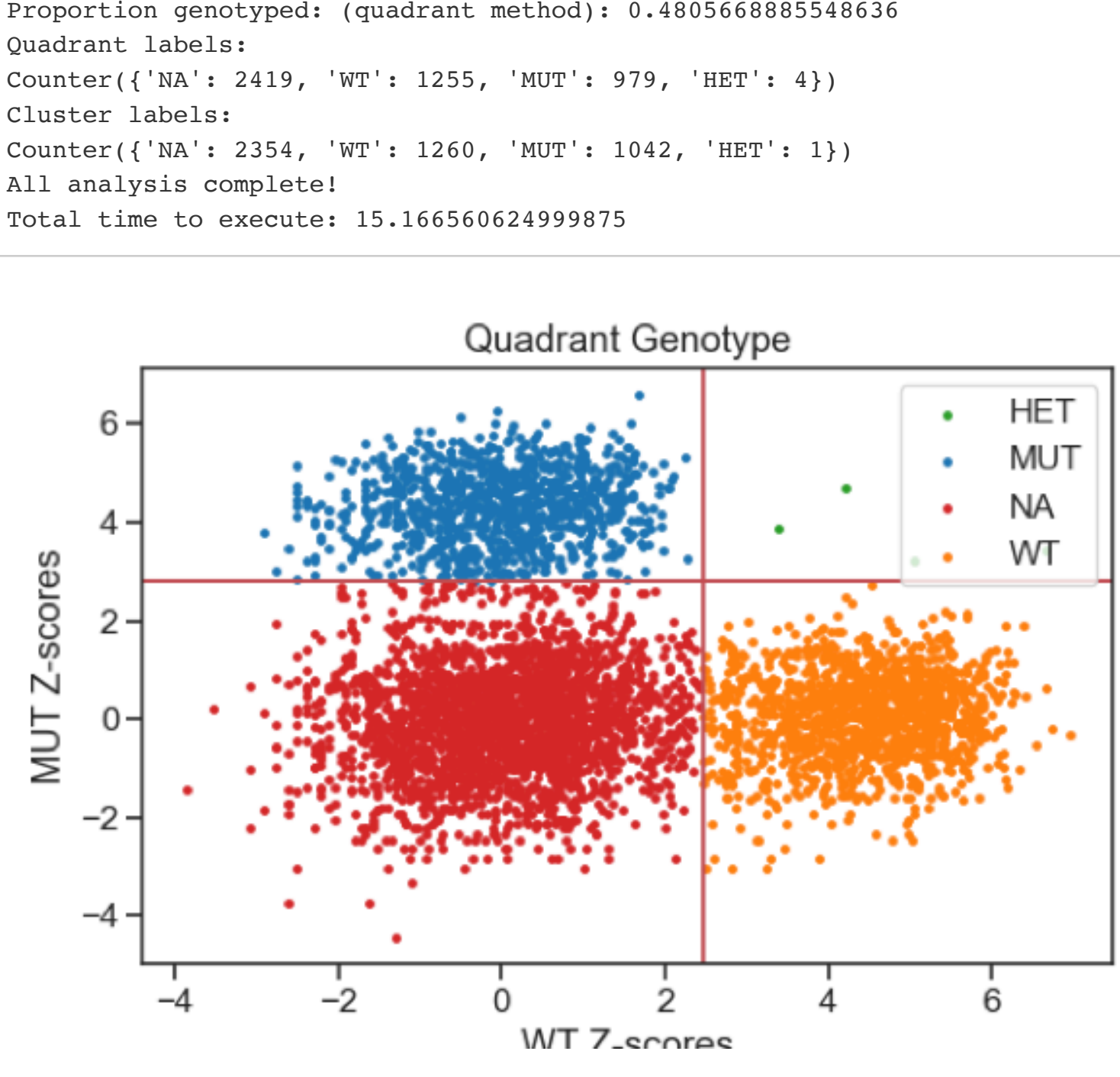








```
Reading in file.
Number of cells: 4657
Noise correcting WT read counts.
5.3210505715729095
6.875730081223762
Optimization terminated successfully.
         Current function value: 0.028554
         Iterations: 14
         Function evaluations: 29
6.734723897917415
Noise mean, noise var:
4.536318113057001 0.8870008623680512
2.462512606706242
Noise correcting MUT read counts.
6.02166087787291
7.002238955397624
Optimization terminated successfully.
         Current function value: 0.025923
         Iterations: 13
         Function evaluations: 26
7.236701131357777
Noise mean, noise var:
4.43795266627485 1.0981572650587275
2.787402763976981
Performing quadrant genotyping.
Computing KNN-based clusters.
[1255, 979, 2419, 4]
Nearest neighbors: 18
Remove uncertain labels and re-label with self-training.
Label propagation completed in 4.77876854200008 seconds.
Proportion genotyped: (cluster method): 0.4945243719132489
Proportion genotyped: (quadrant method): 0.4805668885548636
Quadrant labels:
Counter({'NA': 2419, 'WT': 1255, 'MUT': 979, 'HET': 4})
Cluster labels:
Counter({'NA': 2354, 'WT': 1260, 'MUT': 1042, 'HET': 1})
All analysis complete!
Total time to execute: 15.166560624999875
```



The output of the function is a data frame containing the assigned genotypes based on either the quadrant genotype (quadrant_class) or the predicted genotype (genotype_pred):

Hide

```
head(genotypes)
```

	WTcount	MUTcount	transf_WT	transf_MUT	quadrant_class	genotype_pred
RM30#AACATCGAGGCCTCGT-1	2383	136	3.62948	0.48007	WT	WT
RM30#CCCACATAGGCGATTG-1	112	2920	0.21402	3.52737	MUT	MUT
RM30#CAGCTAATCTATCCTA-1	311	771	1.35165	2.20207	NA	NA
RM30#GCACGGTGTCGAGGAT-1	548	2582	1.98463	3.40490	MUT	MUT
RM30#CCAATGACAGGCAGAT-1	214	17655	0.93455	5.31923	MUT	MUT
RM30#TTGTTGTCAGGTAACG-1	251	493	1.11242	1.75743	NA	NA

This data frame can be added to the metadata of a Signac or ArchR object to perform downstream analysis. The "genotype_pred" column contains the assigned genotype label after noise correction and labeling.