Once again we would like to thank the reviewers for their feedback. We will use the same color code as in the previous round of revisions. For the record, it seems that reviewer numbers were swapped between the revisions. We have not changed the labels: they are shown here as we received them at the second round of feedback.

The comments of reviewer 1, reviewer 2 and reviewer 3 are in blue.
Our comments are in purple and italics.

Changes to the software or software documentation are in bold italics, with yellow highlight.
Changes in the manuscript are in bold italics.
Actual text changes in the manuscript are in bold green.

Reviewer: 1

Comments to the Author
The authors resolved all the concerns successfully.

We are glad to learn that our response was satisfactory.

Reviewer: 2

Comments to the Author
First, I would like to thank the authors for their follow up on the question of read count simulations to train the SVM and for their decision to implement the idea of user-defined training for the quality control step in future versions of Zerone.

Most of my comments have been answered. However, there are a couple of lingering important comments and two other comments that arose in the authors' rebuttal.

Major Comments:

1- Thank you for adjusting the figures, they are much clearer now. However, Fig2 and Fig3 are still missing X and Y axis labels.

We are not sure what the issue is: on our copies of the figures the axis labels are clearly visible. On the top panel of Figure 3 the labels have been dropped because they are identical to the panel aligned below. Perhaps this is a graphical software issue? ***We have duplicated the axis label in Figure 3****, in case it is what this reviewer meant. In any event, we will make sure that the axes are present and visible on the final print (as judged by the proofs).*

2- In my first review, I asked the authors to compare IDR to the SVM step in Zerone in my major comment 2. The authors state in their rebuttal that they can't see how this can be done. To answer: One can use the IDR diagnostics (explained on the IDR webpage towards the bottom) to decide whether the dataset has good quality or not, which is basically the same as what the SVM in Zerone claims to do. If the authors don't want to use MACS to do this due to time constraints, ENCODE has already processed their data this way (using SPP+IDR) and give a quality indication about each dataset on this link:
http://genome.ucsc.edu/cgi-bin/hgFileUi?db=hg19&g=wgEncodeAwgTfbsUniform

To clarify further, one question can be how often do Zerone's SVM and IDR (dis)agree on the overall quality of a dataset? Knowing this information can be helpful and is very relevant to this manuscript.

This is a very good point. Upon addressing it, we have realized that the SVM presented in the previous version of the manuscript was not well calibrated and it sometimes gave grotesque advice. A little investigation revealed that the chosen features did not port well to new datasets. We have thus picked different features and tested the

stability of the classifier on new data. Among others, we have reduced the number of features to limit overfitting, even though there cannot be any guarantee that performance will be maintained on every dataset. We are particularly indebted to this reviewer, as this critical point would have escaped us without his/her insistence.

The classifier of Zerone has been modified. It now extracts different features from the output.

Paragraph 2.3 and Figure 2 have been changed to match the classifier as implemented in this version of Zerone.

We have performed a comparison of the quality control of MACS+IDR versus the quality control of Zerone. We chose 30 'peaky' profiles generated in H1 ES cell by ENCODE. Experiments were either paired with their replicate, or with another one from this set, generating 60 cases to discretize and classify. ***The detail and the replay are available on the associated Docker image. The paragraph below has been added at the end of section 2.4.***

The quality control of Zerone was compared to a method based on IDR to flag replicates with low consistency or low quality (described in https://sites.google.com/site/anshulkundaje/projects/idr). Briefly, this method uses the script batch-consistency-analysis.r to perform pairwise comparisons between all the replicates to check that the number of reproducible peaks is similar between them. The same comparisons are also performed between random splits of each replicate, and between random splits of a pooled profile. Experiments that do not satisfy minimal criteria are flagged as faulty.

The paragraph below has been added at the end of section 3.1.

Low quality profiles can also be detected using IDR. We ran either Zerone or MACS followed by IDR-based flagging on a test set of 30 chromatin features from H1 ES cells. The agreement between the two methods was 85% (see section 2.4 and associated Docker image). We excluded 5 cases where either MACS or IDR failed to run. Among the 8 cases of disagreement, Zerone accepted 3 and rejected 5, suggesting that it is slightly more stringent than MACS + IDR. It is important to note that the quality control of Zerone is more general than IDR-based flagging since it applies to all types of signal without upper limit on the number of replicates. In contrast, IDR is a tool for pairwise comparisons and it does not apply to profiles with broad domains. Also note that IDR-based flagging in itself can represent a significant computational burden when there are many replicates or many targets sites (section 3.3 shows some examples of the computational cost of IDR).

Please note that the author of the "IDR webpage" (https://sites.google.com/site/anshulkundaje/projects/idr) states that "The analyses in this presentation are NOT supposed to be treated as published results AND should not be used as a formal citation as an argument for one peak caller or the other." We are thus unsure how to properly acknowledge the use of IDR for this purpose, or whether we should take extra precautions before putting up this information in the manuscript. We will conform to the opinion of the editor about this issue.

3- The authors state that they can't run JAMM+IDR because JAMM already descritizes replicates jointly. I don't understand this statement. IDR does not "combine" replicates, it scores peaks on their reproducibility. JAMM combines replicates to find better peak edges and improve peak ranking, and it reports a large list of peaks so that it can be thresholded using IDR. The authors should check the section about IDR in the JAMM manuscript for clarification and also go through the entire IDR pipeline described on the IDR webpage.

Regarding this manuscript, the authors can either:
(1) run JAMM with IDR as described in the JAMM manuscript, since they also run MACS with IDR in the new version of the manuscript. This is the better option.
or (2) at the very least, do as I suggested in my previous review and mention clearly in the manuscript that JAMM's output is expected to be thresholded using IDR. I didn't find any such reference in the new version of the manuscript. For example, JAMM can be indicated as "JAMM(unthresholded)" or so on in the tables and figures.

We did not mean that we cannot run JAMM+IDR (but the comments of this reviewer made us realize that this is how it came across). This point is somewhat of a historical accident that we will briefly explain. When more than two replicates are available and the targets are domain-like, IDR is not applicable so there is no standard way to find the consensus discretized profile. This is one of the reasons we developed Zerone, so we initially planned to include a benchmark in these conditions. The only other software able to do the job was JAMM and it would have to be run without IDR, so it initially seemed fair to us to run JAMM in these conditions. At the first round of revisions, we made a note to edit the text as suggested by this reviewer, but this point slipped away in the middle of other modifications. Our bad: it should not have. Retrospectively, we realize that presenting JAMM this way is not fair. We apologize to the JAMM developers; it was never our intention to damage their reputation, we just stuck to a decision that was made long ago, but that turned out to be no longer relevant.

We chose to do option (1). ***Tables 2, 3 and 4 now contain the performance of JAMM with and without IDR. In Figure 5, the discretization chosen for JAMM is always the one with the best F1 score (without IDR for CTCF and with IDR for Pol2 and H3K4me3).***

Since the computation of the IDR represents a significant amount of time (8-10 days in this case), ***we have updated Figure 7. On the top panel, there are now two bars for JAMM and MACS. One of them indicates the running time with IDR and the other one without.*** *The end of the legend now reads as shown below.*

The bars in dark grey represent the total running time when IDR is also computed (for JAMM and MACS). Note that the logarithmic scale misrepresents the relative fraction of time spent on each task.

Also, **the following sentence has been added to the first paragraph of section 3.3.**

Post-processing the output with IDR did not significantly increase the running time of MACS, but it did so for JAMM, bringing the total running time over a week (Fig. 7, dark grey bars).

Minor comments:

- The authors use GRO-seq now to define positive and negative genes. Thank you for considering this suggestion. However, if I understand correctly, the authors consider any gene with at least one read to be expressed. This is too relaxed. A stricter threshold can be more appropriate.

Thanks for bringing it up, we agree. Reviewer 3 also has some concerns about this part (see our response below). We now use criteria that we think are more reasonable to call a TSS active: they must contain at least 10 GRO-seq reads within a +/- 500 bp window. ***The numbers have been updated accordingly in Tables 3 and 4.*** *The relative performance of Zerone is unchanged, so this did not entail any related change to the text.* ***We have added the following sentence to the legend of Table 3 and Table 4.***

A TSS is called active if a 1 kb window centered around it contains at least 10 GRO-seq reads.

- Regarding my minor comment 1 from the previous review, input is the sonicated crosslinked DNA before subjecting it to IP. Input can more or less correct for such intrinsic biases like structural variations for example. But it won't correct entirely for GC content bias for example which is thought to be a result of PCR amplification. Like the authors themselves state in the rebuttal, my point is: input (or whatever other control) is not expected to be exactly like the IP in terms of its biases. So why use input to fit the shape parameter and the zero inflation parameter?

The answer comes in the authors' rebuttal as well, the sentence starting with "actually, we tried it in an early prototype...". This is the answer to my question and the authors can consider adding this to the manuscript or the supplemental methods, since it can be of general relevance.

Thanks for the clarification, we were not sure which type of answer was expected there. The corresponding sentence in section 2.1 now reads as shown below.

For this reason they are fitted from the negative control profiles before the Baum-Welch cycles (as a side note, fitting them with the Baum-Welch algorithm slows convergence and sometimes gives aberrant solutions).

- In their rebuttal to my minor comment 5, the authors state:
"Provided the experiment is valid, we do not see in which case a discretization capturing less signal (ChIPseq reads) for the same amount of called targets would be better. That said, we would be happy to discuss this issue in the manuscript and bring the necessary corrections if we are given such an example."

I can give two examples: (1) In ChIP-seq a "true" peak is expected to have a bimodal distribution of ± strand peaks. Some peak finders like SISSRS, GPS and Peakzilla exploit this property and preferentially score such locations higher than other locations which do not have this property even if they have more reads. (2) Some peak finders attempt to provide a refined spatial resolution of the data like JAMM, Peakranger and Peakzilla. In such cases, it can happen between two locations "x" and "y". "y" has higher counts than "x" but lies in the trough between two higher peaks while "x" represents the highest point locally. In such cases, "y" will be ignored and "x" will be a peak.

The authors can check the PeakZilla manuscript which can shed light on some of these examples.

Thanks for the detail. We did not think about read shifting before this reviewer brought it up in this context. We now agree that such possibilities exists. Since those cases are somewhat outside the main focus of the manuscript, we prefer not to retell the examples in full detail. We suggest to resolve the issue by adding a cautionary sentence in the discussion. ***We have added the sentence below to the third paragraph from the end of the Discussion****.*

Note, however, that the number of reads per window is not the only criterion for calling targets, so there are cases where a discretization lying far from the front may be preferable to one lying closer to it.

Reviewer: 3

Comments to the Author
Your revisions are quite satisfactory and the inclusion of Docker is to be recommended to many an author.

I have only some minor comments that should be and can easily be improved.

How would you like people should pronounce the name of your software? Give it a footnote on pronunciation.

People may pronounce "Zerone" the way they feel is more comfortable in their language. As a reference, we usually pronounce /zi ˈroʊn/ or / ˈziːron/. We have added a footnote at the first occurrence of the word after the abstract, which reads as shown below.

[1] Pronounced /*zi ˈroʊn*/ or / *ˈziːron*/, *i.e.* as inserting 'ear' in 'zone'.

The sentence:
Zerone detects low quality profiles with 2% of errors.

is imprecise. Do you mean the profiles with 2% errors (whatever they would be) or that the detection levels have 2% errors?

Thanks for pointing out that this sentence is ambiguous. We mean the second. To avoid confusion, we have rephrased this sentence of the abstract as show below.

The result is a classifier reaching 95% accuracy in detecting low quality profiles.

See our response to point 2 of reviewer 1 about the change from 98% to 95%.

The sentence:
There is thus a continuous need to develop and improve

will sound better after this change:
There is thus a need to continuously develop and improve

Thanks. ***We have changed the text as suggested.***

p. 14
parameter theta. What is alpha there? Some uncertainty regarding p-1, or is it Greek ro-1, too.

We have added the following information right after the first formula.

π is the zero-inflation or mixture parameter (the proportion of unmappable windows), α is a shape parameter dictating the distribution of reads and p0 , . . . , pr are probabilities linked by the equality p0 + p1 + . . . + pr = 1 and dictating the average number of reads per window for each replicate.

Provide a reference to forward-backward algorithm.

We have added the reference to the famous tutorial of Rabiner published in 1989.

In the paragraph p17 “Table 2 shows that Zerone…” there is definitely too much of a sales pitch. Zero does not achieve a notable precision. It is third worst! And what about Total?

Sorry for our excess of enthusiasm. ***We have rephrased this paragraph as shown below.***

Table 2 shows that Zerone has the best F1 score (the harmonic mean of precision and recall) and the second highest recall score, close to that of JAMM. On the other hand, JAMM and MACS had much higher precision when used in combination with IDR. JAMM discovers the largest amount of motifs, but it also has the highest number of targets. For this score, Zerone comes in second position, with a much lower amount of targets.

Table 3 gives F1 scores that are rather poor for Active TSS. Any comments on the general nature of the problem?

The issue was the criterion for calling a TSS active (they had to intersect GRO-seq reads). ***We have changed the criteria****, in line with the comment of reviewer 1 (see above). We now observe higher scores for all discretizers because of the increased window size around the TSS.*

And on p. 18, in the paragraph “Table 3 shows that…” Zerone is good, but not outstanding. I suggest that you moderate your language a bit.

We have rephrased this paragraph as shown below.

Table 3 shows that Zerone obtains the highest F1 score. Here the precision of Zerone is almost as high as that of MACS + IDR, with a better recall. Zerone also discovers the highest net number of TSS and active TSS, consistent with its higher position in Figure 5. These results confirm that the performance of Zerone is good on this data set.

It would also be useful to comment on highest Total, highest TSS.

The modification above addresses this point. ***We have also added the following sentence in the second paragraph of section 3.2.3:***

Again, Zerone discovers the highest net number of TSS and active TSS, consistently with its position above the other tools in Figure 5. In this example, Zerone also calls less targets than the other tools.

The active TSS part of the tables seems irrelevant. Any suggestion for the reader why it is there?

As mentioned above, ***the numbers have been updated****. Since their relevance is now clear, we consider that their presence does not need to be justified to the reader.*

p. 19 "The results were similar…" Specify the size of input.

This first sentence of section 3.3 now reads.

We compared the running times of the different programs on discretizing the four data sets used above, containing 7 to 36 million reads (Table 1).

When talking about memory usage, only maxima are really important. Do you ever run into a memory problem?

The numbers in Figure 7 are the maxima. Zerone needs about 500 MB to run the core algorithm (this includes storing the Viterbi path, the emission probabilities, the posterior (smoothing) probabilities and an index of the data to speed up computations). On top of this more or less constant cost, Zerone needs to store the read counts per window. Since this does not increase with the sequencing depth, the memory usage varies very little (neither does the running time). The memory increases as the number of replicates increases and as the window size decreases. A profile with 300 bp windows in the human genome represents 10^7 *integers to store the read counts i.e. 40 MB. Most machines will have enough memory to run Zerone, even if the input has many reads and even if there are many replicates.*

We have added the following two sentences at the end of the first paragraph of section 3.3.

The memory usage of Zerone does not depend on the sequence depth of the experiment, but solely on the number of replicates and the total number of windows in the genome. As an example, for the human genome at 300 bp resolution, the memory footprint of Zerone is expected to be around 500 MB + 40 MB per replicate.
