### Referee 1:

# Referee general comments:

*In this Application note, the authors described a program, MOAT, usable to detect regions of a genome with significantly more mutations than expected. The null expectation is based on random permutation of the mutations within a local region. To offer significant speed, the authors provide a GPU implementation of their program.*

***Author’s Response:***

We appreciate the comments of the reviewers.

### -- Conceptual questions/suggestions --

# Referee conceptual comment 1:

*With the best of my effort to understand, the problem that MOAT is aiming to solve is unclear. Due to this very broad definition of the problem, the solution chosen by the authors is difficult to understand and support. If the problem is to detect local density of variants, why algorithms typically used in peak-finding are not appropriate (knowing that they are very much quicker than permutations)?*

***Author’s Response:***

We thank the reviewers for their suggestions. Due to the page limits, a detailed discussion of the problem setting is a little bit tough. However, we have updated our text (with accompanying supplemental figures) to for a tighter problem focus, and a discussion on the suitability of peak calling algorithms as follows:

“This nonparametric scheme randomly permutes the variants (or target regions) on a relatively large scale where the BMR is assumed to be constant to provide robust burden analysis in ***cancer driver detection***.”

“A common analysis strategy in cancer driver detection is to look for genomic elements with a high variant accumulation across many patients.”

“On the surface, an analysis of whole genome variant density would appear to be a facile adaptation of standard peak calling algorithms that are more commonly used for finding regions of high read density. However, due to the aforementioned BMR heterogeneity, a mere scan for variant aggregations would not be a sufficiently specific evaluation of significantly elevated mutation burdens. This heterogeneity takes many forms, as indicated in Supp Fig XX, resulting in patient-specific rates, region-specific rates, and local nucleotide context-specific rates. Hence, a proper mutation burden evaluation must accommodate a comparison of observed variant density to the expected density under these context-specific background rates, which is not possible with standard peak calling algorithms.”



/var/folders/5n/hj490cl15lx_3ytsnsb56p6h0000gr/T/com.apple.Preview/com.apple.Preview.PasteboardItems/MOAT_08.02_supp_figure_pack (dragged).pdf

# Referee conceptual comment 2:

*Some comparisons with existing tools and a relevant example showing the unique approach of the authors is needed.*

***Author’s Response:***

Here we applied our method on a set of XXX Breast cancer patients published by XXX. Since there is only a few methods focusing on the noncoding regions of the genome, we compared MOAT with some of the newest methods, such as OncodriveFML\cite{} and LARVA\cite{}. OncodriveFML compares the functional impact score of the real variants against the randomized variants at 50kb resolution to get empirical P values. On the other sides, LARVA adopts a parametric scheme by assuming the number of mutations within bins follows a beta binomial distribution.

First we selected the TSS sites of all coding transcripts (100bp upstream of transcription start site), and merged the overlapping ones using bedtools\cite{}. OncodriveFML reports no mutationally burdened regions after multiple test correction. LARVA only reports AGAP5 as significant. However, our method reports 12 TSS sites as significant, with 10 out of the 12 reported genes have been documented as cancer associated. However, currently there is golden standard for real somatic burden analysis. We believe that the high percentage of cancer-associated genes sololy discovered by MOAT implies its better performance on this dataset.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Gene Name | Documented role with cancer | Pubmed ID |
| 1 | SLC3A1 | Cysteine transporter SLC3A1 promotes breast cancer tumorigenesis | 28382174 |
| 2 | ADRA2B | reduce cancer cell proliferation, invasion, and migration | 25026350 |
| 3 | SIL1 | subtype-specific proteins in breast cancer | 23386393 |
| 4 | TCF24 | NA | NA |
| 5 | AGAP5 | significant mutation hotspots in cancer | 25261935 |
| 6 | TMPRSS13 | Type II transmembrane serine proteases in cancer and viral infections | 19581128 |
| 7 | ERO1L | Overexpression of ERO1L is Associated with Poor Prognosis of Gastric Cancer | 26987398 |
| 8 | MYLK3 | Methylation of MYLK3 gene promoter region: a biomarker to stratify surgical care in ovarian cancer | 28350786 |
| 9 | SERPINF2 | potential biomarkers for advanced breast cancer patients | 26673961 |
| 10 | ETV2 | expression profiles to predict distant metastasis of lymph-node-negative primary breast cancer | 15721472 |
| 11 | SIPA1L3 | Highly expressed in preinvasive breast lesions such as high-grade ductal carcinoma in situ, and could be involved in signaling pathways that lead to cell proliferation | 15318932 |
| 12 | SBK2 | NA | NA |

### -- Technical questions/suggestions --

# Referee technical comment 1:

*The authors mentioned they can preserve the trinucleotide context when choosing the new variant location. To my understanding the statistics is based on counts of variant over a region: how the precise location within a trinucleotide can be taken into account by only restricting the permutation location?*

***Author’s Response:***

We thank the reviewer for pointing out this presentation issue. In the updated version, we provide a new supplemental schematic to demonstrate the trinucleotide context preservation in the variant permutation procedure, as follows. Specifically, we first index the whole genome and know what is the local tri-nucleotide for each location. When we are permuting the variants, we only move the variant around positions in the designated bin with exactly the same local tri-nucleotide. Hence, the tri-nucleotide is preserved to make a fair comparison.

/var/folders/5n/hj490cl15lx_3ytsnsb56p6h0000gr/T/com.apple.Preview/com.apple.Preview.PasteboardItems/MOAT_08.02_supp_figure_pack (dragged) 1.pdf

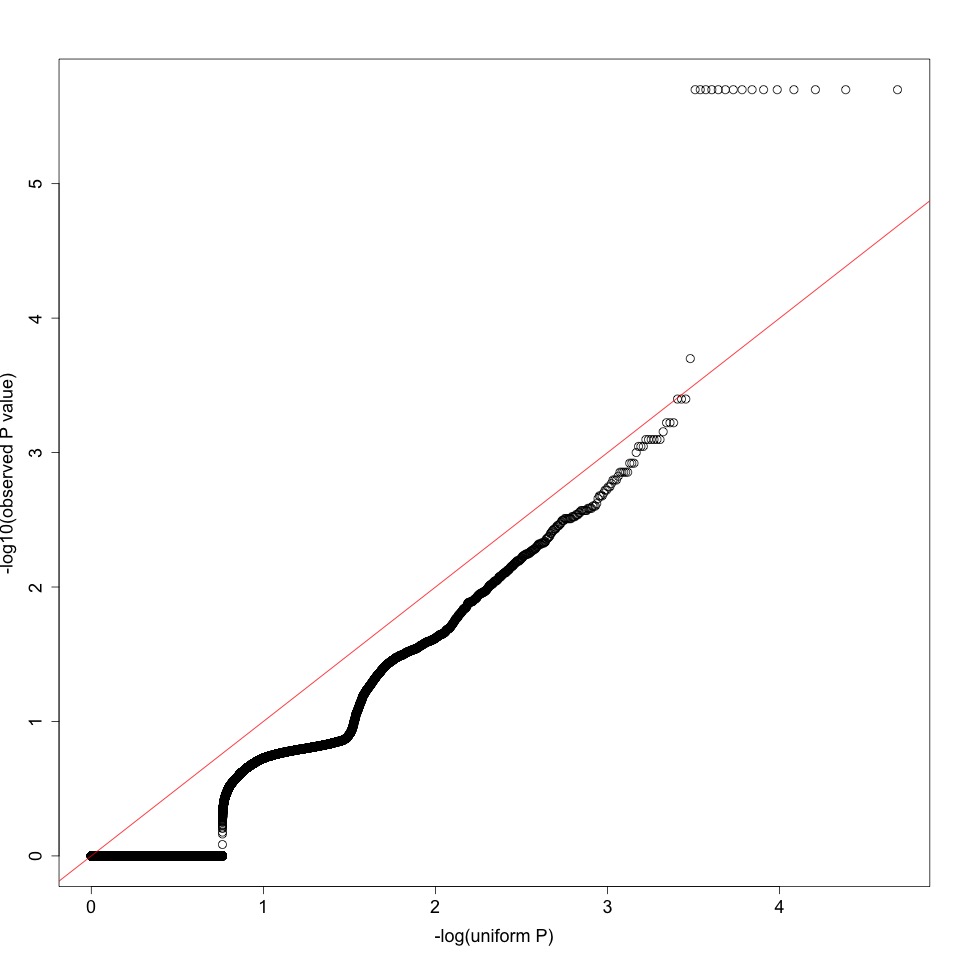
# Referee technical comment 2:

*For MOAT-a, the authors try to demonstrate the accuracy of the tool by stating that it can detect mutation burdens on known cancer-associated TSS. But no complete evaluation is reported. What is the specificity? What are the scores/ranks of these sites compared to all sites?*

***Author’s Response:***

We thank the reviewer to point out this issue. Due to the page limit in the main manuscript, it is very tough in the first version to incorporate a detailed discussion and full figures. Hence we provided a supplementary info and also put this on our website at <https://sites.gersteinlab.org/moat/>.

Similar to the comparison of methods, here we applied our method on a set of XXX Breast cancer patients published by XXX. First we selected the TSS sites of all coding transcripts (100bp upstream of transcription start site), and merged the overlapping ones using bedtools\cite{}. We provide the following Q-Q plot of MOAT-a’s raw p-values. Note that the majority of P values strictly follows the diagonal line with theoretical uniform distribution, with a few outliers as true signals. Hence there is no significant P value inflation. Again, since till now there is no golden standard in the genome-wide analysis, we did a literature search to show that the majority of associated genes of TSS sites solely discovered by MOAT are supported by previous work, justifying MOAT’s power to detect sensible genes.



### Referee 2:

# Referee general comments:

*The authors developed a tool MOAT to perform mutation burden analysis with great speed. Generally, this tool is promising and useful. The paper is well organized and prepared.*

***Author’s Response:***

We thank the reviewer for these comments.

### Referee 3:

# Referee general comments:

*MOAT is a computational system for identifying significant higher-than-expected mutation burdens in genomic regions with an empirical, nonparametric method, which is highly useful for whole genome/exome studies. The expected mutation count is derived by simulating the expected distribution of background mutations. To produce this expected distribution, MOAT offers two types of permutation algorithm: one permutes the locations of annotations (MOAT-a), and one permutes the locations of variants (MOAT-v). MOAT utilizes GPU/OpenMPI parallelization to address running time issues in the scale of whole genome-wide analysis. The manuscript is well-written and concise.*

***Author’s Response:***

We thank the reviewer for these comments.

# Referee comment 1:

*MOAT relies on the assumption that the BMR (background mutation rate) remains approximately the same within a local context. Both MOAT-v and MOAT-a require the user to define the “local context” ([dmax], [dmin] in MOAT-a; [width] in MOAT-v). It may be very difficult for some users to set these parameters, can there be recommended values as default? How do these key parameters influence MOAT’s performance?*

***Author’s Response:***

We have added discussions to the text concerning the best practices for choosing runtime parameters for each of MOAT-a, MOAT-v, and MOAT-sim, as follows:

MOAT-a:

“The boundaries of the intervals for choosing permuted annotations—d\_min and d\_max—are adjustable to allow users to scale the surrounding genome context with respect to the size of the original annotation. The permutation intervals ideally will provide enough range to enable non-overlapping sampling. For example, in our analysis of transcription start site (TSS) mutation burdens, where TSSes are roughly 100 bp in length, we used a d\_min of 2kb and a d\_max of 10kb. As a rule of thumb, the choice of d\_min must be large enough to avoid the possibility that a potential mutation burden signal may "bleed" into the permutation intervals, while the choice of d\_max must be small enough that the BMR covariates are approximately constant within the permutation intervals.”

MOAT-v:

“The ability to adjust the width of the whole genome bins in MOAT-v enables users to select a width that represents regions in which the BMR covariates are expected to be approximately constant. Hence, the permutations created by MOAT-v will honor the regional mutation density expected due to these covariates. Our analysis of a few of the most significant covariate, DNA replication timing, indicate that it really changes at a relatively larger scale, for example 100kb to half a million bp.”

MOAT-sim:

“As with MOAT-v, bin width should be chosen based on the resolution at which the input covariates are approximately constant (50-100kb by default).”

We also made this into the default setting of our software and clearly list it into the online documentation.

# Referee comment 2:

*The MOATsim algorithm uses k-means clustering, how was the value k decided? Also, this program is called Simulated Somatic Variant Datasets, does this tool only works with somatic variants?*

***Author’s Response:***

We have added text to the manuscript explaining the choice of *k*, along with a supplemental figure:

“The whole genome bins are clustered using k-means clustering, which uses the distances between the bins' covariate signal profiles to group the bins into a predefined number of clusters. We selected k based on a within cluster sum of squares analysis (Fig XX)—the ratio of the within cluster sum of squares to the between cluster sum of squares did not change appreciably after k=30, and was subject to stochastic variation at higher values.”

/var/folders/5n/hj490cl15lx_3ytsnsb56p6h0000gr/T/com.apple.Preview/com.apple.Preview.PasteboardItems/MOAT_08.02_supp_figure_pack (dragged) 2.pdf

With regard to germline analysis, MOAT's various programs are optimized for use with somatic variants. Adapting this framework for use with germline variant data is not straightforward. Germline variant distributions are influenced by linkage disequilibrium, and MOAT would require substantial additional development and optimization to accommodate this background mutation model. We plan to investigate such developments in future research.

# Referee comment 3:

*In Table 1, why the GPU is slower (0.86x) in ~14,000 MOAT-a? Is there a conceptual threshold number over which GPU will work better than CPU?*

***Author’s Response:***

At small input sizes, the overhead of setting up the computation on the GPU outweighs the speedup of the actual GPU computation. This is a common observation of many algorithms: given a naïve algorithm A and an optimized algorithm B, B will be faster than A in most cases, but there is a certain “crossing point” where very small inputs below a certain size will run faster in A than in B.

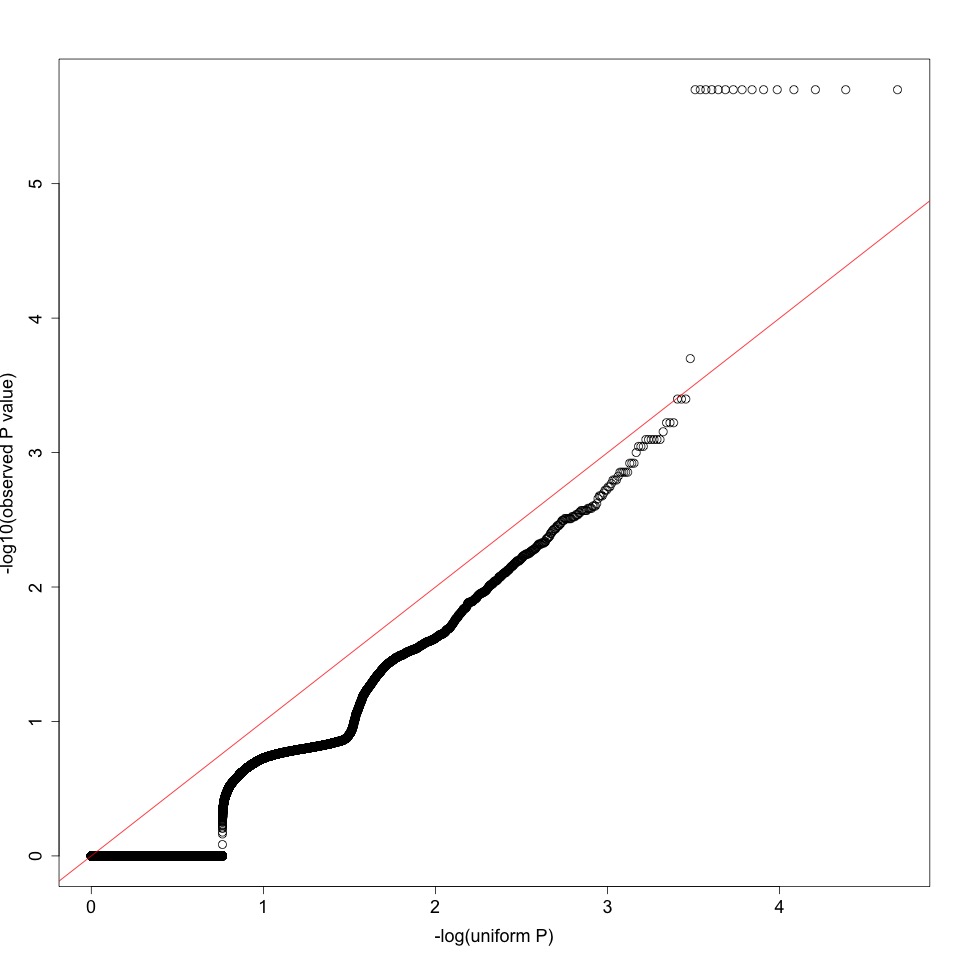
# Referee comment 4:

*In the cancer variant case study, the authors mentioned that several known cancer-associated elements were found to have significant mutation burdens, which is great. Can the authors provide a QQ plot of this study to show the power and there is no bias (in/deflation)?*

***Author’s Response:***

We thank the reviewer to point out this issue. Due to the page limit in the main manuscript, it is very tough in the first version to incorporate a detailed discussion and full figures. Hence we provided a supplementary info and also put this on our website at <https://sites.gersteinlab.org/moat/>.

Similar to the comparison of methods, here we applied our method on a set of XXX Breast cancer patients published by XXX. First we selected the TSS sites of all coding transcripts (100bp upstream of transcription start site), and merged the overlapping ones using bedtools\cite{}. We provide the following Q-Q plot of MOAT-a’s raw p-values.



# Referee comment 5:

*Users may want some sample runs to get started. Sample input files for a complete run and recommended parameter values would be helpful. For example, can the authors provide a pipeline with all parameters they have used in their cancer variant case study? Also, can the parameter [blacklist file] set to be optional or disabled if there is no region list we want to remove? It would be even better if the authors could provide such a file that contains regions of “poor mappability, such as centromeres and telomeres, among others”.*

***Author’s Response:***

We thank the reviewers for this great suggestion. We have made changes accordingly on our website at <https://sites.gersteinlab.org/moat/>. Specifically,

1. We provided a test run with simulated dataset and command lines.
2. For the blacklist regions, there are quite a lot of repeats and “N” in the genome. The quality of variant calling are usually not quite high and the existence of “N” would cause problem when we are trying to fix the local tri-nucleotide effect. Hence we feel it might be better to be precautious on such regions. However, we do add an output to tell the users which regions we actually excluded in our analysis.
3. We also put the black list region online for users to download directly.

# Referee comment 6:

*Minor typo in the README.txt file: “We provide precomputed Funseq scores on the MOAT website at: []”, the URL is missing*

***Author’s Response:***

We apologize for this oversight. The correct URL has been added to the README.