

Ultrafast and reference-free sequence discovery in single cell data

Corresponding Author: Professor Nikolaus Rajewsky

Any redactions in this file are there to maintain patient confidentiality, the confidentiality of unpublished data, or to remove third-party material.

Parts of this Peer Review File have been redacted as indicated to remove third-party material.

Attachments originally included by the reviewers as part of their assessment can be found at the end of this file.

Version 0:

Reviewer comments:

Referee #1

(Remarks to the Author)

The Malva platform has multiple advantages. It broadens the analytical scope of scRNA-seq data beyond the conventional pre-defined gene-by-cell matrix, allowing flexible queries of expression quantifications or coverage along genomic regions of genes, transcripts, circular RNAs, viral sequences, and sequences with specific mutations or alterations, including those not present in reference genomes. The concept is novel, transforming how researchers explore large-scale sequencing data, and will likely attract strong interest from the research community. However, while Malva represents a useful and innovative computational tool and resource, several aspects currently limit its potential impact—particularly in terms of dataset coverage and accessibility.

* It appears that Malva is intended for commercialization. Tools and resources that are broadly accessible to the research community are more likely to gain wide adoption and achieve high impact.

* Access to many datasets containing raw sequencing data, especially human clinical samples, is often controlled. Consequently, the searchable space available to Malva users is limited. The authors should describe the coverage of major repositories (e.g., Human Cell Atlas, GEO/SRA, ENA) within the Malva index and provide a practical solution for exploring data not yet included, such as users' own datasets. A local or stand-alone version enabling users to expand datasets for customized analysis would be highly valuable.

* Doublets may exist across scRNA-seq datasets. In reference-based pipelines, these can be detected by identifying cells or clusters with signatures of two cell types. However, the current Malva pipeline does not appear to handle doublets explicitly, which may lead to false-positive results.

* In its current form, Malva returns only cells expressing the queried gene or sequence, making downstream analysis challenging. For instance, comparisons of expression abundance across tissues or cell types may be confounded by batch effects, which are not sufficiently addressed. Re-normalization and batch-effect correction require additional information that is missing in the current output—such as the inclusion of cells without expression of the queried gene/sequence, size factors for normalization, and original cell IDs from the source studies. These omissions prevent users from retrieving additional metadata or performing customized downstream analyses.

* Sequencing errors are not explicitly considered in the Malva pipeline. Rigorous evaluation is needed to demonstrate the robustness of Malva to sequencing errors, both in pseudoquantification and in variant detection.

* A particularly promising application of Malva is the direct search for specific sequence alterations in cancer cells. Although the authors present results using a cell line mixture containing a fusion gene (BCAS4–BCAS3), the analysis remains superficial. Demonstrating Malva's ability to detect a broader spectrum of somatic alterations—such as point mutations, small insertions/deletions, fusions, and structural variants—in large cohorts of human tumor samples would substantially increase its scientific value and community interest.

* The main advantage of Malva lies in its sequence-based query capability. However, only a few examples are provided. Systematic evaluation of its performance in applications such as variant detection, isoform usage (especially 3'UTR analysis), and circular RNA detection would strengthen confidence in its capabilities.

* Sequence variants such as small insertions/deletions and gene fusions may not be fully captured through k-mer-based indexing alone. The authors should evaluate these potential limitations or explicitly discuss them as inherent challenges of reference-free analysis in the Discussion section.

Minor points

- * The LLM agent fails to recognize several common biological entities. For example, queries such as "Claudin 18.2," "CLDN18.2," "NM_001002026.3," "CD45RO," "CD45RA," or "transcript CD45RO" return no results.
- * The current web portal does not support visualization of coverage-like profiles along genomic coordinates.
- * Cell type annotation is somewhat confusing. For instance, querying "CD3D" returns top cell types including CD8_T_cell, T_cell, and memory_T_cell, which belong to the same hierarchical structure. This redundancy reduces clarity and limits intuitive comparisons across tissues and cell types.

(Remarks on code availability)

The API code works well and produces results consistent with those from the web portal. A stand-alone version of Malva is currently not available.

Referee #2

(Remarks to the Author)

In this manuscript León-Periñán, Karaikos and Rajewsky introduce *Malva*, an index, tool, and service (a framework) for the reference-free analysis of single-cell sequencing data at the level of "raw" sequence search. The manuscript describes the Malva resource, the indexing strategy, and several different types of analysis that can be carried out using this resource. Specifically, the authors focus on analyses where "standard" reference-based approaches can miss important biological signals, while at the same time demonstrating how reference-free analysis of data can lead to results that are reasonably concordant with standard reference-based analyses.

The work presented is truly impressive in scale. What has been accomplished here is, in my estimation, substantial. I experimented with the command-line tools provided, and after a brief hiccup (see below), it worked quite well. While I have not performed any in-depth analyses with the tool, I did perform some searches with expected biological results, and I obtained query results concordant with my expectations. Overall, I feel that Malva could prove an important resource to researchers, and that it could enable useful complementary analyses beyond what is obtained from standard reference-based approaches.

Some highlights of the approach include:

- Malva Index currently includes approximately 60 million cells from thousands of experiments, and can be accessed online without requiring local storage or reprocessing. Importantly, the index is designed to be incrementally expandable: new single-cell or spatial libraries can be indexed independently and then merged into the global index without re-indexing everything. This design choice is essential, as it enables the resource to grow continuously and keep pace with the rapid accumulation of new datasets each year, while maintaining manageable computational cost.
- The manuscript provides a clear overview of the Malva platform. Supplementary Figure 1 effectively illustrates how raw data are ingested, standardized, and indexed at the cell level (lack of specific technical detail described below notwithstanding), and how queries are subsequently resolved through the public API. In addition, the query summary table is well structured and clearly outlines the range of sequence-based search operations that Malva enables. This presentation is particularly helpful in conveying the functionality and practical utility of the platform.
- The paper demonstrates that Malva enables types of analyses that are not easily supported by existing tools, and the authors provide a comprehensive set of examples to illustrate this point convincingly. First, they evaluate the sensitivity and specificity of Malva Index using known positive and negative controls, and further validate consistency by comparing derived allele frequencies against population-level estimates. Second, they show that Malva enables isoform-level exploration directly from sequence data, including cell-type-specific differential coverage patterns, alternative 3'UTR usage, and the ability to query back-splice junctions to detect circular RNAs.
- The authors introduce a cell-by-sequence representation (achieved via cell-by-bucket) as a complement to the standard cell-by-gene matrix. Using this representation, they perform cell-cell similarity, clustering, and marker discovery, revealing sequence-driven structure that is not captured by gene-level analysis in some settings. While the sequence-based and gene-based embeddings are not always fully concordant, the sequence-based perspective provides useful complementary information, particularly when high-quality reference genomes are lacking.

Though I find the proposed framework compelling, I have several concerns, both major and minor, with the proposed manuscript as written, which I feel should be addressed.

== Major Concerns

1. Having read the manuscript in the presented order, I was surprised to get all the way to the discussion before I read the caveat that "Thus, Malva does not replace state-of-the-art reference-based methods, but extends them with additional layers of information." I deeply agree with this framing of Malva, but this hardly seems how it is presented through most of the manuscript until this point. In the abstract, introduction, and results, the manuscript focuses on how Malva can be used to address both common and atypical tasks in single-cell sequencing analysis. In fact, for much of the manuscript, until I got to

the method section, I was attempting to predict and understand how the analog of gene-level abundances could be obtained before I later learned that these are pre-computed and cached. In fact, given the timings reported in the manuscript and observed in practice, it would seem that while Malva is impressively fast for the main kinds of queries discussed in the manuscript, it is not actually particularly optimized for gene-level "reference-based" analysis. In other words, querying for counts or coverage, or even presence/absence of all reference genes, is likely not faster than state-of-the-art lightweight reference-based counting pipelines. Of course, this is absolutely OK and expected, I just feel strongly that the complementary nature of Malva should be highlighted earlier (and perhaps a bit more frequently) in the manuscript.

2. Availability of core code: The only source code provided to the reviewers, and likewise, the only code that seems to be available, is that for the API and the clients. That is, there is code provided for interacting with the Malva server, but the core code that does indexing, query, aggregation, crawling, classification, etc. seems to be nowhere made available. Is the intent for that code to remain proprietary? If so, that should be made clear and explicit in the manuscript. Even *if* the intent is for that code to remain proprietary, I have serious concerns with no accommodations being made to have that code available to the referees during review. Specifically, the description of the methods presented in the manuscript is at a rather high-level, and insufficient in itself to reproduce the operation of e.g. the Malva Index. A high-level description like this would likely be OK if the code was available for deeper inspection, but if the intent is for the code to remain private, then I feel the computational procedures should be described in more detail. For example, some specific questions that come to mind:

- Are the indexed k-mers filtered in any way (as done in e.g. the SBT, Metagraph, Mantis, work)? If so, what filters are applied? If not, does this mean that the system is also indexing distinct k-mers that result from sequencing error? That suggests that there need be some linear dependence of the index on the total amount of indexed sequence.
- How are the k-mer frequencies encoded in the Malva index? The methods note that the sequence array stores the unique k-mers alphabetically, and that each unique k-mer is associated with entries in a pointer array that points to the composite cell-dataset identifiers where each k-mer appears. All of this sounds like presence/absence encoding. Further, none of the text describing how the sequences are bit-packed mentions encoding of k-mer frequency. How is this handled within Malva, or is frequency not encoded? What if a k-mer appears many times in the same dataset and cell?
- Similar to the above, how are k-mers containing an ambiguous (i.e. 'N') nucleotide handled? Does the indexer skip indexing them? Is the ambiguous nucleotide replaced with a fixed nucleotide or with a pseudo-random nucleotide?

Likewise, important details that may be critical to the practical functioning of the index, such as the manner in which queries are batched, the pattern in which the index is accessed, and the parameters and eviction policy of the LRU cache used, are not described in the manuscript nor capable of being interrogated in the provided code. I have other specific questions about the important details of how the system works, but my larger comment is that either the relevant code should be made available or a much more detailed description should be given (perhaps in the supplement if appropriate).

3. How is cell barcode correction handled in the indexing? The k-mers are indexed by a combination of their dataset ID and their cellular barcode. However, barcodes, like the rest of the sequenced fragments, are subject to both PCR amplification and sequencing errors. This means that the number of observed barcodes in a typical single-cell RNA-seq dataset is very large, often an order of magnitude or more higher than the number of actual live cells that have been assayed. This leads to barcode correction as an important step of pre-processing of these datasets, since appropriate correction of the barcodes can help assign otherwise lost reads to their true cells of origin, and can also eliminate very low-depth barcodes that likely do not correspond to an actual sequenced cell. However, I could not find details in the method section or supplement describing the process by which extracted barcodes are processed or corrected within Malva index. Are barcodes corrected? Are all observed barcodes kept and maintained separately? Are low count barcodes discarded, or is correction to a true barcode within the sample attempted?

4. In the paper, there is only gentle discussion of the fact that the Malva approach is not generally UMI aware. To be clear, the authors never make the positive claim that it is UMI aware. However, the omission of this specific caveat, and the relegation of the effect of this choice on pseudo-quantification to certain qualitative and partially quantitative comparisons, should be addressed. In high-throughput droplet-based scRNA-seq protocols, substantial PCR amplification is performed. Likewise, we know that amplification is not perfectly uniform and that gene and sequence-specific biases result in the preferential amplification of certain sequences / genes. This means that counts derived from non-UMI corrected k-mers may not only differ due to differences between k-mer counting and alignment or pseudo-alignment, but may also differ in sequence and gene-specific ways due to the reference-based pipelines' applications of UMI deduplication compared to Malva's lack thereof. While the authors do propose and implement a saturation-based pseudocount correction factor in their pipelines, and this does generally help to address the most glaring discordance (e.g. in the first-order moments) between the reference-based and Malva-based quantifications, the correction is applied as a global (i.e. cell-wide) correction. This suggests that while, in expectation, counts will be scaled to address the prevalent PCR amplification, the counts of specific genes that may have undergone more or less PCR amplification will not generally be corrected. Specifically, the information that is being omitted here is that, in reference-based pipelines, UMIs are generally deduplicated after alignment, so that reads sharing substantially similar UMIs and originating from the same genomic locus (as evidenced by their alignments) are deduplicated using one of several different algorithms for this purpose. This deduplication is then both cell and locus (i.e. gene)-specific, rather than cell-wide. In fact, the supplementary plots in Supp Fig. 3 b seem to demonstrate that the Malva counts per gene are almost always higher than those of the reference-based pipelines. While the counts do correlate well globally, there is non-trivial density that is substantially off-diagonal, suggesting that some genes are obtaining a much higher count under Malva than the reference-based solution.

On the other hand, devising a proper way, in the reference-free context, to address sequence or target-specific UMI deduplication seems a difficult problem in its own right. One option would be to have the existing identifiers include UMI information in addition to cell barcode and dataset information; yet this is likely to vastly increase the index size and may not

be worth the trouble. The authors very briefly touch upon this current issue in the Limitations section. I think that Malva remains immensely useful, even without a full solution to this problem. As stated in point 1. above, this is much less of a problem when Malva is properly viewed as complementary to, rather than a replacement for, existing reference-based analysis tools. Nonetheless, I feel that the current framing in the manuscript draws insufficient attention to this caveat, and that the paper should clearly state this caveat, warn against deriving specific quantitative results purely from the pseudoquantification procedure, and more thoroughly discuss this limitation in the paper (i.e. explain how gene or sequence-specific differences in PCR amplification are not currently addressed in the method). Additionally, a potentially relevant line of related work that may be worth considering here, but which is not discussed in the current manuscript, is the analysis of binary gene expression matrices in the context of single-cell RNA-seq analysis, as in [1]. That is, for certain analyses, and given the current lack of general methods to properly deduplicate sequence contributions, it may be desirable to simply look at the presence/absence of sequence, rather than to attempt a quantitative assessment of abundance.

5. In the index construction benchmarks, all tools are evaluated under a fixed ~8 GB memory cap, whereas methods such as Fulgor, Themisto, and REINDEER are typically benchmarked with substantially larger memory budgets in their respective publications (MetaGraph's peak memory is less clearly reported). Since indexing performance in de Bruijn graph / colored-dBG systems is known to be sensitive to available memory, this constraint may underrepresent the performance of the comparison tools. The comparison would be strengthened to me by briefly clarifying the reason for choosing the 8 GB limit (e.g., there may be practical deployment constraints or considerations, given that the comparison tools were not originally designed for cell-level indexing), or by additionally showing results under a higher peak memory cap.

In addition, the current performance plots and complexity table are clear and intuitive. To make the comparison even more transparent, it may be helpful to include a small summary table reporting the underlying benchmarking values (e.g., CPU time, peak memory usage) for both indexing and query benchmarks across tools.

6. The claim that cell-by-sequence and cell-by-gene representations yield similar neighborhoods and clustering (Fig. 4c) appears somewhat overstated. In Fig. 4c-left, the sequence-based embeddings generate more clusters, and the reported low NMI values (e.g., ~0.31) indicate that the two representations may be capturing different underlying structures. Without gene location constraints, some of the additional structure in the sequence-based space may be difficult to interpret, and in certain datasets may reflect technical or compositional effects, particularly in low-complexity regions, as the authors noted. Clarifying that the sequence-based embedding provides a complementary rather than directly concordant view would help set expectations and emphasize that careful interpretation is needed.

7. The GitHub link referenced in the manuscript did not appear to be accessible at the time of review, though I presume it contains the same source code for accessing the API programmatically as has been shared with the reviewers via a zip file.

== Minor Concerns

1. The `README.md` seems to be incorrect. It suggests that the client should be registered with the URL `https://malva.mdc-berlin.net`, but this leads the subsequent client calls to fail. In fact, the proper URL seems to be `https://malva.mdc-berlin.de`, just as is used in the web interface.

2. It was somewhat unclear how the W parameter should be chosen for query, as well as how the W parameter interacts with the actual query length during the query procedure. Specifically, the query procedure `_given_ W` is very clear. All k -mers in the window are queried in the index, and the window is considered as present in a cell if $\geq \tau$ fraction of the k -mers are observed. However, how is this applied to sequences where N , the sequence length, is much larger than W ? For example, if I wish to query for a 1000 nucleotide gene, what are the criteria for determining its presence in a cell? Is it sufficient for a single window from the gene to be present? Must a certain threshold of all windows in the gene be present? This was not immediately clear from the description in the manuscript. Additionally, I was somewhat curious about the threshold. For a window to be present, it is said that $\text{score}(W, C)$ must be greater than the threshold, where $\text{score}(W, C) = \frac{|M(W, C)|}{(w-k+1)}$. This score is evaluated over *all* k -mers of the window. However, during indexing, *non-overlapping* k -mers are taken to be indexed. Doesn't this imply that, even if the window is entirely "covered" by indexed k -mers, that only a small fraction of all possible k -mers in that window will actually match to the cell in the index? Is there some other procedure going on here that is overlooked?

3. I appreciate the effort that the authors put into describing the specific space that Malva occupies, and the unique challenges in indexing raw data with the very large number of labels that are necessary for atlas-scale single-cell sequencing indexing. This characteristic sets Malva apart, somewhat, from prior work on large-scale sequence indexing of bulk samples where the label space was, at most, on the order of 100s of thousands to millions. One question I had about the comparison with existing tools is what configuration of Fulgor was tested. Specifically, was it the classic variant, the meta-color variant, the differential variant, or the meta-differential variant? This was clear for the MetaGraph comparison, as supplementary table 1 notes the RowDiff/Multi-BRWT variant of MetaGraph. However, the variant of Fulgor compared against is not mentioned. In general, according to the available literature, the meta-differential variant is the most space-efficient, while being fast to query.

== Bibliography

[1] P. Qiu, "Embracing the dropouts in single-cell RNA-seq analysis," Nature Communications, vol. 11, no. 1, Mar. 2020, doi: 10.1038/s41467-020-14976-9.

(Remarks on code availability)

Please note, as discussed in the main review, the only code that is provided is the documentation for the API and the command line client for interacting with Malva. That code is well-packaged, easy to install, and is commented and generally well-structured. I cannot comment on the critical code that powers Malva (including e.g. the ingestion engine, the indexing code, and the query algorithms), as that code has not been made available.

Referee #3

(Remarks to the Author)

I co-reviewed this manuscript with one of the reviewers who provided the listed reports.

(Remarks on code availability)

The source code made available to reviewers appears to include only the API and client-side components, which are the parts I was able to test. The README.md appears to contain an incorrect server URL. It suggests users to register the client at <https://malva.mdc-berlin.net>, which results in failed requests; the correct endpoint seems to be <https://malva.mdc-berlin.de>, consistent with the web interface.

After updating the URL configuration, I was able to install the local client package and test the three query modes described in the documentation. The gene-name search functions as expected. However, the NL search via the Python API results in a `JSONDecodeError`, and the sequence-based search works only when replacing the example sequence with an alternative one.

The web interface performs as expected.

Referee #4

(Remarks to the Author)

The work presents Malva, a large scale index for querying single cell/spatial datasets. The contribution is positioned alongside existing frameworks like Metagraph or Logan. The paper is well written and even pedagogical, the figures are helpful. The chosen methodology makes, at a global scale, much sense. I'm certain that Malva could become a prime resource for many biologists in the coming years. Key results of the paper include an online resource of about 60 million cells from various experiments.

A major methodological concern is the sampling of non-overlapping k-mers when building the index. While the motivation may be to reduce memory footprint, this approach has critical implications: queries become more fragile, because a sequencing read whose informative k-mers fall between sampled positions may go undetected. Any phase shift, even slight, between input reads and indexed k-mers can cause the system to miss biologically relevant matches. The choice of nonoverlapping k-mers also leads to more convoluted algorithms during the query. Then, why not using methods with guarantees and well established methods like minimizers for sampling?

While Malva adopts an inverted-index-based strategy, this is not a novel design choice, Metagraph also relies on such an architecture. Can the authors describe to which extent single cell datasets are already available in tools like Metagraph and Logan search? I'm also asking this because I'm not too convinced by the fact that methods like Fulgor would not scale to many labels (and I'm not really convinced by the experiment, see later on). I believe it is ok to have your own codebase and solution, but it is more interesting to highlight what's different in content and features (e.g., your query possibilities).

Additionally, the comparison against Metagraph, Fulgor, and others raises fairness questions. If Malva indexes only a sampled subset of k-mers while other tools index the complete set, the comparison is inherently unbalanced. For a fair evaluation, all systems should be tested using identical k-mer inputs or full-coverage indexing. It is unclear whether this was done.

In terms of performances, the manuscript states scalability advantages but does not quantify space usage relative to other tools. It also shows that Malva achieves decent speed although it is not the quickest for queries, but I wondered if the pre computed matrices strategy was involved in the query benchmark or not. Also, did the sequences used for quickness of query response included a mix of positive and negative queries ?

Another important remark is the validation of the AI agent used to translate queries from natural language to a range of possible queries to the Malva API. Usually methods like this are fine-tuned, and their perception of metadata categories or specific vocabulary can still be put to question. I think the paper does not present enough validations on this aspect.

Other remarks:

-“whose complexity is uncoupled” I don't think it is uncoupled, and I don't think your results support this claim

-The manuscript also mentions that highly repetitive k-mers are masked but does not describe: how repetition is defined, which thresholds or statistical tests are used, how the procedure differentiates repeats from highly expressed genes. Without this, reproducibility is impossible.

-The claim of adaptability to modified nucleotide indexing (e.g., m6A) is potentially interesting, yet the manuscript treats this as trivial (“just by changing the alphabet”). In practice, this is unlikely to be straightforward, as modified bases introduce new challenges in encoding, ambiguity handling, and downstream resolution. This point must be expanded technically, otherwise it reads as speculative.

-The mention of Simple Abundance (p.4) is insufficiently explained.

(Remarks on code availability)

Access to source code only via archive rather than GitHub undermines transparency; I expect a justification. Other projects at least as ambitious had a public repository from the start.

Version 1:

Reviewer comments:

Referee #1

(Remarks to the Author)

Although a standalone version of Malva is now available through binaries, restricted access control may still limit the broader impact of this otherwise useful tool. I am not in a suitable position to assess its availability in detail.

Additional suggestions below could further enhance both the tool and the manuscript:

* On the web portal (<https://malva.mdc-berlin.de/>), after submitting a query, the server does not return all cells in the index but only those with counts ≥ 1 . I recommend allowing user-defined parameters (such as `min_counts`) to adjust the returned results. For instance, `min_counts = 0` could return all cells, while `min_counts = 1` (default) would return only those with query hits. Moreover, it seems difficult to map "sample_id" to the original barcode in the returned data — an "original_barcode_id" field appears to be missing. Including normalization factors in the metadata (so that "norm_expr" can be recalculated from "cell_count" and the normalization factors) would also facilitate downstream analyses. For the Coverage Explorer, it would be helpful to add options to select datasets and filter cells according to metadata fields.

* It seems inconsistent that the results concerning somatic point mutation evaluation (Sup. Fig. 11) are presented under the section "Screening for Germline Variation." I suggest reorganizing this part and clarifying that the evaluation pertains specifically to the detection of somatic point mutations.

* Some figures appear to be missing, such as Sup. Fig. 5g and Sup. Fig. 13h.

(Remarks on code availability)

I successfully installed the Python package from the provided wheels using Python 3.11.14 and was also able to run Malva via Apptainer. The example results were reproducible (<https://github.com/malva-bio/malva/tree/main/examples>).

Referee #2

(Remarks to the Author)

= Malva Revision Review

I thank the reviewers for their thorough and thoughtful revisions in response to the comments and suggestions offered by myself and the other reviewers. I find the revised manuscript greatly improved, both in the detail of the exposition (the additional supplementary note describing the indexing and query framework in detail), and in terms of the suite of experiments presented. I also appreciate the revised framing, which I think better places *Malva* in the context of existing work, and highlights its complementary nature with respect to existing reference-based quantification pipelines, while still conveying the broad scope of new types of analyses that it can enable. I still think it unfortunate that *Malva* will not be made available as an open source tool, which may hamper the ability of those, especially in the methods development community, to build upon the work done here. Nonetheless, I appreciate the authors' efforts to somewhat ameliorate these concerns by providing a binary version of their index building and query program, and committing to those components being free for academic use.

Overall, I am largely satisfied with the revision, and commend the authors on the truly impressive scope of their work. I have only a few remaining technical questions / concerns.

== Remaining questions

1. The newly-added exposition on how cell barcodes are handled clarifies a lot that was missing in the previous version of the paper. However, it does raise another question. Since the barcode algorithm looks for exact matches against a user-provided "whitelist", what is the status of technologies where such a whitelist may not be available (e.g. protocols such as variants of split-seq or sci-seq, where there may be no vendor-provided whitelist). In such cases, a common approach is to use a cell calling algorithm; either an initial simple algorithm like the "knee" method, or a more sophisticated approach such as that adopted in 'EmptyDrops' [1], or one of several even more sophisticated methods. Likewise, in addition to not having a vendor-provided whitelist, several of these protocols have rather complicated encodings / embeddings of their technical read information (barcodes & UMIs), requiring more sophisticated processing than extracting fixed length subsequences from known positions (sometimes handled by dedicated pre-processing programs like 'matchbox' [2] 'flexiplex' [3] or 'splitcode' [4]). Does *Malva* support these protocols? Is it anticipated that *Malva* will be expanded to support such protocols? If so, some minor details on how such expanded support is planned would be useful.

2. Likewise, though the 10x chromium platform was initially designed for short-read RNA-seq data, it has become increasingly popular as a platform for long read data, both PacBio and ONT sequencing technologies have been used with

10x kits (and, in fact with other protocols such as split-seq) to produce isoform-level long read single-cell RNA-sequencing data. While the amount of such data is still dwarfed by the amount of short-read single-cell RNA-seq data, it will be a growing data type in coming years. How do you anticipate *Malva* applying to such data? What types of changes would be necessary to make it work (would it suffice to just change the thresholds, window and k-mer sizes, or would more fundamental changes to the search algorithms be required)? Perhaps the answer is simply that the current tool is unsuitable for long-read technology, and such extensions would be out of scope, even in future versions. Regardless, it would be good to mention this in the manuscript as a potential limitation (and/or a future direction).

3. From the software availability perspective, providing only x86-64 executables running on linux (and likely depending on specific versions of libc) is quite constraining. I'd highly recommend the authors to provide a singularity image and / or docker container as well, so that users can run those executables across a broad range of machines, rather than a specific narrow range of linux machines.

Bibliography

[1] A. T. L. Lun, S. Riesenfeld, T. Andrews, T. P. Dao, T. Gomes, and J. C. Marioni, "EmptyDrops: distinguishing cells from empty droplets in droplet-based single-cell RNA sequencing data," *Genome Biology*, vol. 20, no. 1, Mar. 2019, doi: 10.1186/s13059-019-1662-y.

[2] J. Schuster et al., "Powerful read processing with matchbox," Nov. 2025, doi: 10.1101/2025.11.09.685711.

[3] O. Cheng et al., "Flexiplex: a versatile demultiplexer and search tool for omics data," *Bioinformatics*, vol. 40, no. 3, Feb. 2024, doi: 10.1093/bioinformatics/btae102.

[4] D. K. Sullivan and L. Pachter, "Flexible parsing, interpretation, and editing of technical sequences with splitcode," *Bioinformatics*, vol. 40, no. 6, June 2024, doi: 10.1093/bioinformatics/btae331.

(Remarks on code availability)

I have looked at and evaluated the client code (as with the first round of review). I have not reviewed the indexing and query code, as those are provided only as pre-compiled executables (and currently only available for x86-64 linux machines).

Referee #3

(Remarks to the Author)

I co-reviewed this manuscript with one of the reviewers who provided the listed reports.

(Remarks on code availability)

The provided binaries (tested via installation from the wheel file) were successfully installable and functional in my environment. I was able to run both the `malva index` and `malva quant` steps by following the instructions provided, and the core functionality appears to work as described.

However, the hyperlink labeled "Documentation" in the README (<https://github.com/malva-bio/malva>) appears to be invalid. It would be helpful if the authors could verify and update this link.

Referee #4

(Remarks to the Author)

Summary

The authors have addressed several points raised previously and have improved the manuscript in a number of aspects. In particular, they clarified the positioning of *Malva* with respect to existing literature and expanded the manuscript by adding several descriptions that were previously missing, especially regarding the methodological aspects. The authors also added a benchmark following the request regarding the validation of the LLM used to generate the queries, clarified which queries are precomputed and which are computed on demand, and improved the description of the query algorithm used to sample cameras. These changes strengthen the manuscript.

However, several points remain unclear or would benefit from further clarification or improvement.

Major Comments

- about transparency:

The authors should consider making publicly available the queries used to evaluate the LLM's performance, and the prompts employed during the instruction-tuning phase.

Other items are not properly documented, such as the fuzzy dictionary component mentioned in the authors' response is not cited. An other example is Supplementary Figure 1 that refers to the "Malvacrawl" service, but the manuscript does not describe how this service operates. In particular, it remains unclear how the retrieved datasets are processed and how duplicate datasets are detected or avoided.

One limitation of the manuscript is definitely that the code will remain private, although it may be made available upon request for academic researchers. While the authors may have valid reasons for this decision such as costs of maintenance, I must stress that this approach does not fully align with common practices in the bioinformatics community, where open-source code sharing is strongly encouraged to ensure transparency, reproducibility, and reuse.

- comparison with related tool

The response provided regarding comparisons with tools such as Metagraph or Fulgor remains approximate. The authors argue that Reindeer does not directly index all k-mers but rather unitigs (or related structures). Conceptually, however, this does not fundamentally differ from indexing k-mers, since the indexed sequences ultimately represent the same information. Whether the lookup uses minimizer-based strategies or not does not appear to fundamentally change the nature of the indexing approach. As a result, the distinction made in the response remains somewhat unclear and potentially confusing. The authors should clarify more precisely what differentiates their approach from these existing tools.

Table 1 raises concerns. Some of the reported factors are really unclear to me and sound questionable. For example, in Reindeer, the logarithmic factor should not apply to the number of distinct k-mers in each dataset as suggested in the table. Similarly, for Bifrost, it is unclear why a logarithmic factor would appear in the reported complexity. Overall, the assumptions behind these complexity estimates are not sufficiently explained and may be inaccurate. Moreover, the complexity analysis can fall short as much of the speed of these approaches is explained by careful cache-design rather than pure complexities. I would advocate to remove this table.

Minor

- In Supplementary Figure 5, panel needs an x-axis
- type in Supplementary Figure 2, an extra e to a k-mer
- Note: If I understand the hierarchical structure presented in Figure 2 is a B-tree-like data structure.
- Will the authors consider releasing the probes used for the analysis, such as poly(A) tails and EGFR mutations ?

(Remarks on code availability)

The source code is not publicly available. The authors indicate that the code may be made available upon request to academic researchers, but this limited availability prevents an independent evaluation of the software as a community resource, especially for future maintenance.

Version 2:

Reviewer comments:

Referee #1

(Remarks on code availability)

(Remarks to the Author)

Thanks for the open-source code. The proposed idea in the authors' last response letter looks promising, and the redesigned web portal (Malva Platform) has become an impressive tool with enriched functionalities. However, a few technical issues remain: it appears that some claimed functions are not fully implemented, or that certain bugs hinder their proper use.

* In the Malva Platform, for the "Expression" module, the downloaded "Per-cell data" is in fact not per-cell, but aggregated sample-cell type expression data; the normalization factor is also not provided. In addition, there is often a duplicated entry for a given sample and cell type combination: one corresponding to the measured average expression (if the average > 0), and another showing 0 expression ("n_cells" is 0, and the mean or median expression is NA). The documentation should be improved. For example, the meaning of "enrichment" is unclear, and more detailed information about the single-cell results would be helpful for users. For the "Coverage" module, searches are slow when no filter is applied. In addition, the "zoom in" and "zoom out" functions do not work.

* For the malva_client, I installed version 0.3.1 successfully. The search() method works well, but search_cells() does not return data as expected (status = 'no results') when querying the same gene as in search(). This limits its support for downstream analysis.

* For Malva, I installed it successfully from the source code (Python 3.14.0). The open-source release is expected to be well-received by the research community.

One suggestion for rapid improvement would be to encourage extensive use, evaluation, and benchmarking within the research community –not only in terms of user experience and functionality, but also regarding the accuracy of cell type annotation and the quality of the indexed database.

Referee #2

(Remarks to the Author)

The most recent round of revisions have addressed all of my prior concerns, as well as my largest remaining concern regarding the availability of the source code of the tool. I think the authors have made the correct decision on this point, and value their understanding and appreciation of the importance of source code availability as had been raised by several of the referees, including myself. I have no further substantial concerns with the manuscript.

(Remarks on code availability)

I was able to inspect and review the code, though have not had a chance to do so comprehensively.

My first attempt to run the example in the example directory failed because it seems that ``anndata`` was not amongst the listed dependencies during build. Thus, execution failed during the ``1_run_malva.sh`` script. However, once I added ``anndata``, the example ran successfully and completed with the expected results.

Referee #3

(Remarks to the Author)

I co-reviewed this manuscript with one of the reviewers who provided the listed reports.

(Remarks on code availability)

Referee #4

(Remarks to the Author)

The authors have made a clear effort to clarify additional material after my remarks, and made the necessary corrections. Good luck with the final steps before publication!

(Remarks on code availability)

Version 3:

Reviewer comments:

Referee #1

(Remarks to the Author)

The effort to improve the tool is evident. It would be appreciated if the following remaining technical issues could be resolved:

* Add an option to output all cells, not only those positive for the queried genes/sequences, both in the web portal and the API (malva and malva-client). This would allow users to recalculate metrics such as “fraction expressing”, mean normalized expression (including zeros), and z-scores by themselves. Such measurements are useful for downstream analyses, including cross-dataset comparisons. This requirement was proposed in my previous round of comments, but has not yet been implemented.

* For the web portal, the downloaded per-cell expression matrix currently has a limit of 50. I suggest removing this limit.

* For the malva-client, the obtained expression values are always 1. In addition, the API documentation is still insufficient. For example, the meaning of the expression values is unclear. Following the tutorial or quick-start (<https://malva-client.readthedocs.io/en/latest/quickstart.html> or https://github.com/malva-bio/malva_client/tree/main), errors/bugs frequently occur.

Code:

```
result = client.search("SPP1")
cells = client.retrieve_cells(result, sample_ids=[123456])
```

returns the error:

"No cell IDs returned for this search job; No cells returned for this search job".

For the code:

```
value_result = client.search("SPP1", aggregate_expression=False)
```

The returned `value\_result` contains empty data.

* For the coverage explorer, it is still slow. I could not even obtain results for a gene such as PTPRC (11:67435510-67437682).

(Remarks on code availability)

The code works well in most situations, except for those I noted in the comments above

Version 4:

Reviewer comments:

Referee #1

(Remarks to the Author)

All of my concerns have been addressed. I would be delighted to see this tool continue to evolve, benefit the research community, and remain open source.

(Remarks on code availability)

The code works well.

Open Access This Peer Review File is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

In cases where reviewers are anonymous, credit should be given to 'Anonymous Referee' and the source.

The images or other third party material in this Peer Review File are included in the article's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

To view a copy of this license, visit <https://creativecommons.org/licenses/by/4.0/>

Dear Referees,

We thank you for the thoughtful and constructive evaluation of our manuscript. We have now carefully and exhaustively addressed all comments in our point-by-point response below. In several cases we performed extensive additional analyses. We have revised the manuscript (all changes are highlighted) accordingly. The most important points are:

- 1) We clarify the complementary and discovery-tool nature of Malva with respect to existing, reference-dependent tools. Also, we better explain our indexing, query, and barcode-handling strategies. In response to concerns about accessibility and transparency, we now describe repository coverage in detail, provide standalone binaries for local indexing and querying, clarify our plans concerning free access for academic users, and expand the methodological descriptions (including **new Sup. Note 4** and **new Sup. Note 5**) to make the procedures reproducible, even though the main code remains proprietary.
- 2) We address limitations and clarify methodological details. This includes a more explicit treatment of the lack of UMI-aware deduplication and its implications for pseudoquantification and presence/absence analyses, clearer discussion of the limits of probe-based detection (including indels, fusions, and 3' bias, **new Sup. Fig. 14**), new analyses showing high robustness to sequencing errors (**new Sup. Fig. 7**), and clarified benchmarking setup and comparisons to existing indexing frameworks (**new Sup. Table 3**). We have also fixed the reported issues in the web portal and API and enhanced the documentation.
- 3) We have substantially expanded the large-scale validation of Malva's detection capabilities. This includes systematic benchmarking against the polyASite v3.0

database (Moon et al., 2025; NAR) (~1.5M polyA sites across hundreds of Human Cell Atlas samples) (**new Sup. Fig. 10**), validation of somatic mutation detection using the scTML database (Li et al., 2025; NAR) (~100,000 SNVs across 280 tumor samples from 16 cancer types) (**new Sup. Fig. 11**), targeted validation of EGFR mutations in a lung cancer cohort (**new Sup. Fig. 6**), and quantitative assessment of the natural language query interface accuracy across 10,000 synthetic test cases (**new Sup. Fig. 12**).

Referee #1 (Remarks to the Author):

The Malva platform has multiple advantages. It broadens the analytical scope of scRNA-seq data beyond the conventional pre-defined gene-by-cell matrix, allowing flexible queries of expression quantifications or coverage along genomic regions of genes, transcripts, circular RNAs, viral sequences, and sequences with specific mutations or alterations, including those not present in reference genomes. The concept is novel, transforming how researchers explore large-scale sequencing data, and will likely attract strong interest from the research community. However, while Malva represents a useful and innovative computational tool and resource, several aspects currently limit its potential impact—particularly in terms of dataset coverage and accessibility.

* It appears that Malva is intended for commercialization. Tools and resources that are broadly accessible to the research community are more likely to gain wide adoption and achieve high impact.

We appreciate the reviewer for raising this point. While we do intend to commercialize Malva, we are committed to open accessibility for the Academic research community. Specifically, there will be a “free tier” of the platform with reasonable usage limits that can accommodate typical research applications. Paid tiers will be available for high-throughput research or commercial use cases. We now also provide free binaries for academic research purposes (see <https://github.com/malva-bio/malva>, https://github.com/malva-bio/malva_client, and the updated Data and Code availability statement). Referees can find the binaries under <https://drive.google.com/drive/folders/1EpssRmtTPj-o12nm8FsC41Jlks8KucE6?usp=sharing>. Documentation is provided in the GitHub repository and in <http://malva.readthedocs.io>.

* Access to many datasets containing raw sequencing data, especially human clinical samples, is often controlled. Consequently, the searchable space available to Malva users is limited. The authors should describe the coverage of major repositories (e.g., Human Cell Atlas, GEO/SRA, ENA) within the Malva index and provide a practical solution for exploring data not yet included, such as users’ own datasets. A local or stand-alone version enabling users to expand datasets for customized analysis would be highly valuable.

We thank the reviewer for this suggestion. We have now added a description of the current coverage of major repositories (Human Cell Atlas, GEO/SRA, ENA, CNGBdb) to the manuscript.

Regarding user-provided data not yet in the Malva Index, we now provide standalone binaries for indexing and querying, allowing users to build local indices from their own datasets. These can be queried programmatically via the existing API and notebook interfaces, enabling integration of private or controlled-access data without requiring upload to our servers. The binaries are freely available for academic research purposes upon request. We have updated our Data and Code availability statement accordingly.

* Doublets may exist across scRNA-seq datasets. In reference-based pipelines, these can be detected by identifying cells or clusters with signatures of two cell types. However, the current Malva pipeline does not appear to handle doublets explicitly, which may lead to false-positive results.

We thank the reviewer for this point. Malva does not explicitly handle doublets, which we acknowledge in the manuscript as a limitation. As we discuss in more detail below, Malva is designed as a complementary discovery tool to identify cell barcodes containing specific sequences. While doublet detection is desirable, the diversity of available methods and their dataset-specific nature renders this challenging to implement at scale (notably, to the best of our knowledge, even established platforms such as CELLxGENE do not perform doublet filtering). Specifically, we apply filtering during the clustering stage by removing cells with anomalously high total counts or gene counts (above the 99th percentile) and by excluding clusters with mixed marker signatures during cell type assignment. Extending this process is dataset- and analysis-specific and we recommend users to run specialized doublet detection software.

On a more general note, however, explicit doublet handling and other specialized downstream analyses are not part of Malva's main processing pipeline and output. Malva provides access to raw data, enabling users to perform proper quantitative follow-up analysis as needed. Users should therefore not rely on Malva alone for doublet filtering or definitive cell identity assignment, but can use Malva to retrieve barcodes of interest and then apply doublet-aware downstream analyses on the retrieved data.

* In its current form, Malva returns only cells expressing the queried gene or sequence, making downstream analysis challenging. For instance, comparisons of expression abundance across tissues or cell types may be confounded by batch effects, which are not sufficiently addressed. Re-normalization and batch-effect correction require additional information that is missing in the current output—such as the inclusion of cells without expression of the queried gene/sequence, size factors for normalization, and original cell IDs from the source studies. These omissions prevent users from retrieving additional metadata or performing customized downstream analyses.

We thank the reviewer for raising this point. Malva actually returns extensive metadata alongside query results, including study identifiers, sample identifiers, technology, tissue, disease status, and cell type annotations. In particular, through the sample unique identifiers (UUID), users can download the corresponding samples to perform normalization and comparative analyses. This information is available via the web portal and the API, enabling users to perform batch correction, re-normalization, or other customized downstream analyses. Users can also retrieve the original cell identifiers to link back to source studies and obtain additional metadata or raw data as needed.

We have now clarified these important features in the manuscript – Malva provides the retrieval layer and associated metadata, while downstream normalization and batch correction remain the user's responsibility and can be performed using standard tools informed by the metadata Malva returns.

* Sequencing errors are not explicitly considered in the Malva pipeline. Rigorous evaluation is needed to demonstrate the robustness of Malva to sequencing errors, both in pseudoquantification and in variant detection.

We thank the reviewer for this suggestion. Malva's sliding window approach requires only a fraction of k-mers to match (controlled by the τ parameter), providing robustness against sequencing errors. We have now evaluated Malva's robustness by simulating error rates ranging from 0.1% to 10% across our benchmark datasets (spatial low and high resolution data, single-cell, single-nuclei, and the cell line mixture data).

For pseudoquantification, correlations between Malva pseudocounts and reference counts remained consistent until 2% error rate, gradually decreasing with error rates up to 10% (new **Sup. Fig. 7**). For variant detection in the cell line mixture benchmark, sensitivity and specificity were also maintained across error rates (new **Sup. Fig. 5e,f**). This same analysis also underpins our response to Referee 4's concern about non-overlapping k-mers and potential "phase shift" effects (please see below). We note that sensitivity and specificity in this benchmark do not always reach 1.0 even for overlapping k-mer indices. This reflects methodological differences between Malva and Flexiplex (our ground truth): for instance, Flexiplex permits Hamming distance 1 matches, detecting sequences with single-nucleotide mismatches that exact k-mer matching excludes, while Malva's additional detections (not present in Flexiplex output, explained by Malva requiring just a perfect match of 24 bp, while Flexiplex uses probes of ~35 bp) are counted as false positives when using Flexiplex as ground truth.

* A particularly promising application of Malva is the direct search for specific sequence alterations in cancer cells. Although the authors present results using a cell line mixture containing a fusion gene (BCAS4–BCAS3), the analysis remains superficial. Demonstrating Malva's ability to detect a broader spectrum of somatic alterations—such as point mutations, small insertions/deletions, fusions, and structural variants—in large cohorts of human tumor

samples would substantially increase its scientific value and community interest.

This is a great suggestion. We extended our analysis in three directions to demonstrate Malva's ability to detect somatic alterations at scale:

First, we validated probe design for polyadenylation site detection using polyASite v3.0 (Moon et al., 2025; NAR), a curated database that analyzed hundreds of samples from the Human Cell Atlas for polyA site usage. We designed two types of probes for each annotated site: polyA-specific probes (ending with a polyA tract) and control probes (same genomic position without the polyA signal). Querying these against the corresponding samples in Malva Index, we observed consistently higher Spearman correlations for polyA probes compared to control probes across tissues (new **Sup. Fig. 10**), demonstrating that probe design critically determines whether Malva captures polyadenylation site-specific usage versus general regional coverage.

Second, we validated EGFR mutation detection in a lung cancer dataset (He *et al.* 2021; Oncogene) containing five tumor-normal matched samples with known mutation status: four samples with EGFR exon 19 deletions and one with the EGFR L858R point mutation in exon 21. Using STAR alignment as ground truth, we designed Malva probes targeting exons 18-21 and the specific L858R variant. Malva correctly identified mutation-positive cells in the expected samples while maintaining specificity in normal tissue (new **Sup. Fig. 6**).

Third, we performed large-scale validation using the scTML database (Li et al., 2025; NAR), which reanalyzed 280 tumor samples across 16 cancer types using a unified variant calling pipeline based on GATK best practices. We extracted ~100,000 SNVs from exonic and UTR regions and designed 45-bp probes centered on each variant. Across cancer types, Malva achieved median sensitivity of ~0.94 and specificity exceeding 0.99, with performance scaling with the number of mutation-positive cells per sample (new **Sup. Fig. 11**). This sensitivity is comparable to mutation detection benchmarks in bulk RNA-seq: REINDEER/Transipedia reports recall of 0.875–0.945 at high-precision thresholds requiring ≥ 3 matching k-mers (Bessière et al. 2024, Genome Biology, Table 3). The single outlier (PAAD) reflects mouse-derived samples that are, we think, misclassified in the original database (query probes were constructed using the GRCh38 human reference genome from the coordinates provided in scTML; thus, using these to interrogate mouse data leads to low recall).

Additionally, we note that access to large, uniformly processed cancer cohorts with raw sequencing data remains limited since most datasets fall under 'restricted access'; for this reason, the current, public Malva Index aggregates data available across repositories rather than focusing on specific large-scale studies. Nevertheless, as our analyses shows, the aggregated scale of tumor samples in the public Malva Index (spanning multiple cancer types and tissues) provides power for detecting recurrent somatic variants. We now acknowledge in the Discussion that the 3' bias of most

single-cell and single-nuclei data constrains detection of variants outside 3' regions. We anticipate, however, that with the rise of long read technology, Malva will in the future become even more useful for these types of analyses.

* The main advantage of Malva lies in its sequence-based query capability. However, only a few examples are provided. Systematic evaluation of its performance in applications such as variant detection, isoform usage (especially 3'UTR analysis), and circular RNA detection would strengthen confidence in its capabilities.

We thank the reviewer for this suggestion. We now perform systematic evaluation of Malva for variant and isoform detection in **new Sup. Fig. 10** and **new Sup. Fig. 11** (see above). These, together with the previous germline variant, isoform and circRNA analyses, further validate the accuracy of probe-based analysis at scale. We believe that these examples, together with the sequencing error robustness analysis (**new Sup. Fig. 7, see above**), characterize Malva's behavior for probe-based queries.

Additionally, we note that k-mer-based approaches for capturing sequence variation have been extensively benchmarked in prior work. For instance, DE-kupl (Audoux et al., 2017, Genome Biology) demonstrated that k-mer queries can reliably retrieve splice variants, polyadenylation variants, and mutation events from RNA-seq data. Malva follows these principles, thus sequence probes targeting specific junctions or variants can be used to retrieve cells expressing those sequences. Given a well-designed probe, Malva performs exact k-mer matching, with high specificity; sensitivity depends on factors such as probe length, sequence complexity, and the fraction of k-mers required to match (τ parameter); sensitivity decreases when multiple mutations or sequencing errors occur within the probe span.

* Sequence variants such as small insertions/deletions and gene fusions may not be fully captured through k-mer-based indexing alone. The authors should evaluate these potential limitations or explicitly discuss them as inherent challenges of reference-free analysis in the Discussion section.

We agree that indels affect k-mers spanning its position, and fusion detection requires probes designed specifically for the breakpoint junction, which might be non-trivial. We now explicitly address these limitations in the Discussion section. Additionally, we note in this Discussion section that the 3' bias of most (single-cell and spatial) datasets in Malva Index hinders the detection of variants and fusions occurring far upstream from 3' regions of transcripts.

Minor points

* The LLM agent fails to recognize several common biological entities. For example, queries such as “Claudin 18.2,” “CLDN18.2,” “NM_001002026.3,” “CD45RO,” “CD45RA,” or “transcript CD45RO” return no results.

We have now updated the API documentation and added a disclaimer on the web platform to warn about possible limitations of the language model, and upgraded the Language Model system instruction to resolve most common aliases and RefSeq identifiers. More specialized queries, such as CD45RO, require specialized probe design (e.g., targeting the specific exon, or exon-exon junction), and therefore we recommend that it should be manually designed – by default, will be broadly resolved as *PTPRC*.

* The current web portal does not support visualization of coverage-like profiles along genomic coordinates.

Thank you. We have updated the web portal to support visualization of coverage-like profiles along genomic coordinates.

* Cell type annotation is somewhat confusing. For instance, querying “CD3D” returns top cell types including CD8_T_cell, T_cell, and memory_T_cell, which belong to the same hierarchical structure. This redundancy reduces clarity and limits intuitive comparisons across tissues and cell types.

We have updated the cell type annotation output to provide results at different levels of the hierarchy, allowing users to select the granularity appropriate for their analysis and enabling more intuitive comparisons across tissues and cell types.

Referee #1 (Remarks on code availability):

The API code works well and produces results consistent with those from the web portal. A stand-alone version of Malva is currently not available.

A standalone version of Malva is now possible through binaries, and available to researchers within Academia upon request.

Referee #2 (Remarks to the Author):

In this manuscript León-Periñán, Karaïskos and Rajewsky introduce *Malva*, an index, tool, and service (a framework) for the reference-free analysis of single-cell sequencing data at the level of "raw" sequence search. The manuscript describes the Malva resource, the indexing strategy, and several different types of analysis that can be carried out using this resource. Specifically, the authors focus on analyses where "standard" reference-based approaches can miss important biological signals, while at the same time demonstrating how reference-free analysis of data can lead to results that are reasonably concordant with standard reference-based analyses.

The work presented is truly impressive in scale. What has been accomplished here is, in my estimation, substantial. I experimented with the command-line tools provided, and after a brief hiccup (see below), it worked quite well. While I have not performed any in-depth analyses with the tool, I did perform some searches with expected biological results, and I obtained query results concordant with my expectations. Overall, I feel that Malva could prove an important resource to researchers, and that it could enable useful complementary analyses beyond what is obtained from standard reference-based approaches.

Some highlights of the approach include:

- Malva Index currently includes approximately 60 million cells from thousands of experiments, and can be accessed online without requiring local storage or reprocessing. Importantly, the index is designed to be incrementally expandable: new single-cell or spatial libraries can be indexed independently and then merged into the global index without re-indexing everything. This design choice is essential, as it enables the resource to grow continuously and keep pace with the rapid accumulation of new datasets each year, while maintaining manageable computational cost.
- The manuscript provides a clear overview of the Malva platform. Supplementary Figure 1 effectively illustrates how raw data are ingested, standardized, and indexed at the cell level (lack of specific technical detail described below notwithstanding), and how queries are subsequently resolved through the public API. In addition, the query summary table is well structured and clearly outlines the range of sequence-based search operations that Malva enables. This presentation is particularly helpful in conveying the functionality and practical utility of the platform.
- The paper demonstrates that Malva enables types of analyses that are not easily supported by existing tools, and the authors provide a comprehensive set of examples to illustrate this point convincingly. First, they evaluate the sensitivity and specificity of Malva Index using known positive and negative controls, and further validate consistency by comparing derived allele frequencies against population-level estimates. Second, they show that Malva enables isoform-level exploration directly from sequence data, including cell-type-specific differential

coverage patterns, alternative 3'UTR usage, and the ability to query back-splice junctions to detect circular RNAs.

- The authors introduce a cell-by-sequence representation (achieved via cell-by-bucket) as a complement to the standard cell-by-gene matrix. Using this representation, they perform cell-cell similarity, clustering, and marker discovery, revealing sequence-driven structure that is not captured by gene-level analysis in some settings. While the sequence-based and gene-based embeddings are not always fully concordant, the sequence-based perspective provides useful complementary information, particularly when high-quality reference genomes are lacking.

Though I find the proposed framework compelling, I have several concerns, both major and minor, with the proposed manuscript as written, which I feel should be addressed.

Major Concerns

1. Having read the manuscript in the presented order, I was surprised to get all the way to the discussion before I read the caveat that "Thus, Malva does not replace state-of-the-art reference-based methods, but extends them with additional layers of information." I deeply agree with this framing of Malva, but this hardly seems how it is presented through most of the manuscript until this point. In the abstract, introduction, and results, the manuscript focuses on how Malva can be used to address both common and atypical tasks in single-cell sequencing analysis. In fact, for much of the manuscript, until I got to the method section, I was attempting to predict and understand how the analog of gene-level abundances could be obtained before I later learned that these are pre-computed and cached. In fact, given the timings reported in the manuscript and observed in practice, it would seem that while Malva is impressively fast for the main kinds of queries discussed in the manuscript, it is not actually particularly optimized for gene-level "reference-based" analysis. In other words, querying for counts or coverage, or even presence/absence of all reference genes, is likely not faster than state-of-the-art lightweight reference-based counting pipelines. Of course, this is absolutely OK and expected, I just feel strongly that the complementary nature of Malva should be highlighted earlier (and perhaps a bit more frequently) in the manuscript.

We thank the reviewer for this feedback and agree that the complementary nature of Malva should be emphasized earlier and more consistently. We have now revised the manuscript to make this very clear.

In the Introduction, we now state that Malva is designed as a sequence-level discovery layer that complements standard reference-based pipelines. For routine gene-level quantification on individual datasets, established tools like kallisto, STAR, or alevin-fry remain appropriate choices. Malva addresses a different problem: enabling sequence-level queries across tens of millions of cells without requiring users to download and reprocess petabytes of raw data. These analyses cannot be performed

with pre-computed gene count matrices, for instance searching for unannotated isoforms, somatic variants, viral sequences, or arbitrary nucleotide probes across atlas-scale data.

To clarify how Malva operates in practice: upon building Malva Index, we precompute expression matrices for common human and mouse genes (based on, e.g., a specific GENCODE version) and cache these results. When users query a standard gene symbol, these precomputed values are returned directly. On the other hand, arbitrary sequence queries, e.g., a novel splice variant, are computed on demand against the Malva Index. Users can also provide alternative sequences for a given gene (e.g., a candidate splicing isoform not present in a GENCODE release), which are then queried *de novo*. Frequently accessed k-mers may be cached for performance, but non-standard sequences are never precomputed. Similarly, coverage on either arbitrary sequences or reference genomes is not precomputed, but also done *on demand*.

We acknowledge that for the most accurate, sample-specific gene-level quantification, reference-based pipelines remain valuable. Malva's strength lies in enabling on-demand, ultrarapid, sequence-level pseudoquantification across atlas-scale data, which becomes particularly useful in scenarios where references are not readily available, e.g., recent efforts for large-scale biodiversity profiling (e.g., see Seb  -Pedr  s et al., 2025; Nature), as we demonstrate in **Figure 4**.

2. Availability of core code: The only source code provided to the reviewers, and likewise, the only code that seems to be available, is that for the API and the clients. That is, there is code provided for interacting with the Malva server, but the core code that does indexing, query, aggregation, crawling, classification, etc. seems to be nowhere made available. Is the intent for that code to remain proprietary? If so, that should be made clear and explicit in the manuscript. Even *if* the intent is for that code to remain proprietary, I have serious concerns with no accommodations being made to have that code available to the referees during review. Specifically, the description of the methods presented in the manuscript is at a rather high-level, and insufficient in itself to reproduce the operation of e.g. the Malva Index. A high-level description like this would likely be OK if the code was available for deeper inspection, but if the intent is for the code to remain private, then I feel the computational procedures should be described in more detail. For example, some specific questions that come to mind:

- Are the indexed k-mers filtered in any way (as done in e.g. the SBT, Metagraph, Mantis, work)? If so, what filters are applied? If not, does this mean that the system is also indexing distinct k-mers that result from sequencing error? That suggests that there need be some linear dependence of the index on the total amount of indexed sequence.
- How are the k-mer frequencies encoded in the Malva index? The methods note that the sequence array stores the unique k-mers alphabetically, and that each unique k-mer is associated with entries in a pointer array that points to the composite cell-dataset identifiers where each k-mer appears. All of this sounds like presence/absence encoding. Further, none of the text describing how the sequences are bit-packed mentions encoding of k-mer frequency.

How is this handled within Malva, or is frequency not encoded? What if a k-mer appears many times in the same dataset and cell?

- Similar to the above, how are k-mers containing an ambiguous (i.e. `N`) nucleotide handled? Does the indexer skip indexing them? Is the ambiguous nucleotide replaced with a fixed nucleotide or with a pseudo-random nucleotide?

We thank the reviewer for their detailed questions. The core indexing and querying code will remain proprietary; however, we have now prepared new **Sup. Note 4** describing the algorithms in detail enough for full reimplementations of the indexing and querying software by bioinformatics researchers familiar with k-mer methods. Additionally, we now provide binaries for indexing and querying private data free of charge to Academic users upon request. Referees can find the binaries under <https://drive.google.com/drive/folders/1EpssRmtTPj-o12nm8FsC41Jlks8KucE6?usp=sharing>. Documentation is provided in the GitHub repository and in <http://malva.readthedocs.io/>. Moreover, to address the specific questions raised:

(1) During indexing, k-mers containing N nucleotides or consisting of homopolymers are excluded; no other filtering is applied at index time. Frequency-based filtering is deferred to query time, where users specify minimum and maximum cell-count thresholds; this flexibility is important given the diverse analyses Malva supports and the sparsity of the underlying datasets.

(2) We thank the reviewer for pointing this out: indeed, Malva implements presence/absence encoding. Each (k-mer, cell) pair is stored at most once, regardless of read-level frequency. This design reflects single-cell data characteristics where >95% of non-overlapping k-mers appear exactly once per cell due to limited sequencing depth (even when aggregated across hundreds of experiments); we observe that, in practice, the quantitative aspect of *pseudocounts* emerges from the windowed query algorithm rather than stored frequencies.

(3) k-mers containing N are skipped during indexing; no substitution is performed.

We have also included the above information in the new **Sup. Note. 4**.

Likewise, important details that may be critical to the practical functioning of the index, such as the manner in which queries are batched, the pattern in which the index is accessed, and the parameters and eviction policy of the LRU cache used, are not described in the manuscript nor capable of being interrogated in the provided code. I have other specific questions about the important details of how the system works, but my larger comment is that either the relevant code should be made available or a much more detailed description should be given (perhaps in the supplement if appropriate).

Thank you for this comment. As described above, we have now documented these details in **Sup. Note 4**. More specifically, regarding the questions about query batching and caching: k-mers are sorted lexicographically across all sequences in a query and

processed in index order to maximise sequential disk access; this also benefits concurrent queries from multiple users, as the same index instances are accessed in parallel by several workers. Decompressed array blocks are retained in an LRU cache (capacity 2×10^6 blocks). For indices exceeding memory, a hierarchical page structure enables lookup with bounded memory.

3. How is cell barcode correction handled in the indexing? The k-mers are indexed by a combination of their dataset ID and their cellular barcode. However, barcodes, like the rest of the sequenced fragments, are subject to both PCR amplification and sequencing errors. This means that the number of observed barcodes in a typical single-cell RNA-seq dataset is very large, often an order of magnitude or more higher than the number of actual live cells that have been assayed. This leads to barcode correction as an important step of pre-processing of these datasets, since appropriate correction of the barcodes can help assign otherwise lost reads to their true cells of origin, and can also eliminate very low-depth barcodes that likely do not correspond to an actual sequenced cell. However, I could not find details in the method section or supplement describing the process by which extracted barcodes are processed or corrected within Malva index. Are barcodes corrected? Are all observed barcodes kept and maintained separately? Are low count barcodes discarded, or is correction to a true barcode within the sample attempted?

We thank the reviewer for raising this point. Barcodes are matched via exact lookup against technology-specific whitelists (e.g., 10x Chromium v2/v3). During indexing, all barcode-k-mer associations are stored; however, low-abundance barcodes are filtered at the output stage rather than during index construction. Specifically, prior to the sample-specific cell type annotation stage, we apply a UMI-based "knee" method on total counts derived from gene-level analysis to retain only barcodes likely corresponding to true cells, analogous to the filtering performed by tools such as Cell Ranger (see Methods "Barcode preprocessing").

Barcode error correction usually recovers only a fraction of the already retrieved reads. Melsted et al. (2021, Nature Biotechnology) report that Hamming distance 1 correction "rescues, on average, 0.8% of reads" across a benchmark panel of 20 datasets (for similar technologies as covered in Malva Index). For this reason, barcodes with Hamming distance > 0 from the whitelist are discarded by many existing pipelines as well: implementing correction across a wide diversity of barcoding strategies is non-trivial and increases processing time. Since Malva's primary goal is to enable exploratory analysis at large scale rather than to maximize recovery from individual datasets, we reason that ingesting more diverse datasets compensates for per-sample read loss more effectively than implementing complex error correction schemes.

Accordingly, we now acknowledge this limitation more clearly in the Results and Discussion section, and have added a description of barcode handling to the Methods section.

4. In the paper, there is only gentle discussion of the fact that the Malva approach is not generally UMI aware. To be clear, the authors never make the `_positive_` claim that it is UMI aware. However, the omission of this specific caveat, and the relegation of the effect of this choice on pseudo-quantification to certain qualitative and partially quantitative comparisons, should be addressed. In high-throughput droplet-based scRNA-seq protocols, substantial PCR amplification is performed. Likewise, we know that amplification is not perfectly uniform and that gene and sequence-specific biases result in the preferential amplification of certain sequences / genes. This means that counts derived from `_non-UMI corrected_` k-mers may not only differ due to differences between k-mer counting and alignment or pseudo-alignment, but may also differ in sequence and gene-specific ways due to the reference-based pipelines' applications of UMI deduplication compared to Malva's lack thereof. While the authors do propose and implement a saturation-based pseudocount correction factor in their pipelines, and this does generally help to address the most glaring discordance (e.g. in the first-order moments) between the reference-based and Malva-based quantifications, the correction is applied as a global (i.e. cell-wide) correction. This suggests that while, in expectation, counts will be scaled to address the prevalent PCR amplification, the counts of specific genes that may have undergone more or less PCR amplification will not generally be corrected. Specifically, the information that is being omitted here is that, in reference-based pipelines, UMIs are generally deduplicated after alignment, so that reads sharing substantially similar UMIs and originating from the same genomic locus (as evidenced by their alignments) are deduplicated using one of several different algorithms for this purpose. This deduplication is then both cell and locus (i.e. gene)-specific, rather than cell-wide. In fact, the supplementary plots in Supp Fig. 3 b seem to demonstrate that the Malva counts per gene are almost always higher than those of the reference-based pipelines. While the counts do correlate well globally, there is non-trivial density that is substantially off-diagonal, suggesting that some genes are obtaining a much higher count under Malva than the reference-based solution.

On the other hand, devising a proper way, in the reference-free context, to address sequence or target-specific UMI deduplication seems a difficult problem in its own right. One option would be to have the existing identifiers include UMI information in addition to cell barcode and dataset information; yet this is likely to vastly increase the index size and may not be worth the trouble. The authors very briefly touch upon this current issue in the Limitations section. I think that Malva remains immensely useful, even without a full solution to this problem. As stated in point 1. above, this is much less of a problem when Malva is properly viewed as complementary to, rather than a replacement for, existing reference-based analysis tools. Nonetheless, I feel that the current framing in the manuscript draws insufficient attention to this caveat, and that the paper should clearly state this caveat, warn against deriving specific quantitative results purely from the pseudoquantification procedure, and more thoroughly discuss this limitation in the paper (i.e. explain how gene or sequence-specific differences in PCR amplification are not currently addressed in the method). Additionally, a potentially relevant line of related work that may be worth considering here, but which is not discussed in the current manuscript, is the analysis of binary gene expression matrices in the context of single-cell RNA-seq analysis, as in [1]. That is, for certain analyses, and given the current lack of general methods to properly

deduplicate sequence contributions, it may be desirable to simply look at the presence/absence of sequence, rather than to attempt a quantitative assessment of abundance.

Thank you for this comment. We agree that this point deserves more explicit treatment throughout the manuscript. Indeed, Malva is not UMI-aware, and the saturation-based correction we propose addresses global PCR inflation but not gene- or sequence-specific amplification biases. A comprehensive solution to this problem extends, however, beyond the scope of this work.

We have now revised the text to clarify this earlier in the manuscript, not only in the Discussion, but also in the Results section where pseudoquantification is introduced and extensively benchmarked against state-of-the-art gene quantification tools. We emphasize that, as an important outcome from our benchmarking, users can rely on pseudoquantification outputs whenever answering qualitative questions (e.g., cell type composition, pathway activity, detection of marker genes), but should remain cautious upon purely quantitative tasks (e.g., computing exact shift in gene expression of individual genes between different subsets of cells).

We also appreciate the pointer to the work by Qiu (2020) [1] on binary expression matrices. Indeed, for certain analyses, presence/absence of sequence may be more appropriate than quantitative assessment. We offer this as an output mode (in the backend API, `malva.indexes.MalvaIndex.where(..., single_count = True)`), and have added a reference to this work in the Discussion when addressing the scope of pseudoquantification.`

5. In the index construction benchmarks, all tools are evaluated under a fixed ~8 GB memory cap, whereas methods such as Fulgor, Themisto, and REINDEER are typically benchmarked with substantially larger memory budgets in their respective publications (MetaGraph's peak memory is less clearly reported). Since indexing performance in de Bruijn graph / colored-dBG systems is known to be sensitive to available memory, this constraint may underrepresent the performance of the comparison tools. The comparison would be strengthened to me by briefly clarifying the reason for choosing the 8 GB limit (e.g., there may be practical deployment constraints or considerations, given that the comparison tools were not originally designed for cell-level indexing), or by additionally showing results under a higher peak memory cap.

We thank the reviewer for this comment. The performance benchmark was not limited to 8GB of memory, and tools were allowed to use all required resources. The ~8GB figure reported for Malva reflects its actual peak memory consumption, not an imposed cap. We have now clarified this in the main text.

Moreover, we have now added **Sup. Table 3** reporting CPU time, peak memory usage for both indexing and query benchmarks across tools, including the new 100-dataset single-cell benchmark described in our response to Referee 4.

In addition, the current performance plots and complexity table are clear and intuitive. To make the comparison even more transparent, it may be helpful to include a small summary table reporting the underlying benchmarking values (e.g., CPU time, peak memory usage) for both indexing and query benchmarks across tools.

See response to comment above (i.e., **Sup. Table 3**)

6. The claim that cell-by-sequence and cell-by-gene representations yield similar neighborhoods and clustering (Fig. 4c) appears somewhat overstated. In Fig. 4c-left, the sequence-based embeddings generate more clusters, and the reported low NMI values (e.g., ~0.31) indicate that the two representations may be capturing different underlying structures. Without gene location constraints, some of the additional structure in the sequence-based space may be difficult to interpret, and in certain datasets may reflect technical or compositional effects, particularly in low-complexity regions, as the authors noted. Clarifying that the sequence-based embedding provides a complementary rather than directly concordant view would help set expectations and emphasize that careful interpretation is needed.

We agree with the referee that the original phrasing may have overstated concordance. Figure 4c was included precisely to illustrate the range of outcomes, i.e., from high to low agreement, rather than a general claim about similarity.

We have revised the text to emphasize that sequence-based embeddings provide a complementary view that can capture additional structure not present in gene-based representations, and that careful interpretation is warranted, particularly in datasets where low complexity or intronic sequences dominate.

7. The GitHub link referenced in the manuscript did not appear to be accessible at the time of review, though I presume it contains the same source code for accessing the API programmatically as has been shared with the reviewers via a zip file.

Thank you. We have updated the GitHub repository with the relevant documentation, public API code and examples (see <https://github.com/malva-bio/malva> and https://github.com/malva-bio/malva_client).

Minor Concerns

1. The `README.md` seems to be incorrect. It suggests that the client should be registered with the URL `https://malva.mdc-berlin.net`, but this leads the subsequent client calls to fail. In fact, the proper URL seems to be `https://malva.mdc-berlin.de`, just as is used in the web interface.

We thank the reviewer for catching this. The README has been corrected.

2. It was somewhat unclear how the W parameter should be chosen for query, as well as how the W parameter interacts with the actual query length during the query procedure. Specifically,

the query procedure `_given_ W` is very clear. All k-mers in the window are queried in the index, and the window is considered as present in a cell if τ fraction of the k-mers are observed. However, how is this applied to sequences where N , the sequence length, is much larger than W ? For example, if I wish to query for a 1000 nucleotide gene, what are the criteria for determining its presence in a cell? Is it sufficient for a single window from the gene to be present? Must a certain threshold of all windows in the gene be present? This was not immediately clear from the description in the manuscript. Additionally, I was somewhat curious about the threshold. For a window to be present, it is said that $\text{score}(W, C)$ must be greater than the threshold, where $\text{score}(W, C) = (|M(W, C)|) / (w - k + 1)$. This score is evaluated over *all* k-mers of the window. However, during indexing, *non-overlapping* k-mers are taken to be indexed. Doesn't this imply that, even if the window is entirely "covered" by indexed k-mers, that only a small fraction of all possible k-mers in that window will actually match to the cell in the index? Is there some other procedure going on here that is overlooked?

We thank the reviewer for raising this important point. For sequences longer than W , presence in a cell is determined as soon as a single window passes the threshold. That is, if any window along the query has $\text{score}(W, C) > \tau$, the sequence is considered present.

This design reflects the assumption that, given the sparse nature of single-cell data, one passing window typically corresponds to one captured molecule (at least in the limit of $\tau = 1$ and error-free reads).

Regarding the score formula: we acknowledge the confusion and have updated the text to clarify that the denominator refers to non-overlapping k-mers within the window, consistent with our indexing strategy. Thus, for a window of size W with k-mer size k , the number of indexed k-mers per window is LW/kJ (plus one overlapping k-mer at the boundary), not $(W - k + 1)$.

We have added a brief recommendation in the main text on the choice of W and τ , and provide more detailed guidance in **Sup. Table 2**, **Sup. Note. 5** and in the API/web portal documentation.

3. I appreciate the effort that the authors put into describing the specific space that Malva occupies, and the unique challenges in indexing raw data with the very large number of labels that are necessary for atlas-scale single-cell sequencing indexing. This characteristic sets Malva apart, somewhat, from prior work on large-scale sequence indexing of bulk samples where the label space was, at most, on the order of 100s of thousands to millions. One question I had about the comparison with existing tools is what configuration of Fulgor was tested. Specifically, was it the classic variant, the meta-color variant, the differential variant, or the meta-differential variant? This was clear for the MetaGraph comparison, as supplementary table 1 notes the RowDiff/Multi-BRWT variant of MetaGraph. However, the variant of Fulgor compared against is not mentioned. In general, according to the available literature, the meta-differential variant is the most space-efficient, while being fast to query.

We thank the reviewer for this request. For Fulgor, we indeed used the meta-differential variant which, as pointed out, is the most space-efficient while maintaining fast query times. This has now been clarified in the Methods section.

Referee #2 (Remarks on code availability):

Please note, as discussed in the main review, the only code that is provided is the documentation for the API and the command line client for interacting with Malva. That code is well-packaged, easy to install, and is commented and generally well-structured. I cannot comment on the critical code that powers Malva (including e.g. the ingestion engine, the indexing code, and the query algorithms), as that code has not been made available.

Referee #3 (Remarks to the Author):

I co-reviewed this manuscript with one of the reviewers who provided the listed reports.

Referee #3 (Remarks on code availability):

The source code made available to reviewers appears to include only the API and client-side components, which are the parts I was able to test. The README.md appears to contain an incorrect server URL. It suggests users to register the client at <https://malva.mdc-berlin.net>, which results in failed requests; the correct endpoint seems to be <https://malva.mdc-berlin.de>, consistent with the web interface.

We thank the reviewer for the extensive testing and reporting of this error. We fixed this accordingly.

After updating the URL configuration, I was able to install the local client package and test the three query modes described in the documentation. The gene-name search functions as expected. However, the NL search via the Python API results in a `JSONDecodeError`, and the sequence-based search works only when replacing the example sequence with an alternative one.

Thank you very much. This issue has now been fixed.

The web interface performs as expected.

Referee #4 (Remarks to the Author):

The work presents Malva, a large scale index for querying single cell/spatial datasets. The contribution is positioned alongside existing frameworks like Metagraph or Logan. The paper is well written and even pedagogical, the figures are helpful. The chosen methodology makes, at a global scale, much sense. I'm certain that Malva could become a prime resource for many biologists in the coming years. Key results of the paper include an online resource of about 60 million cells from various experiments.

A major methodological concern is the sampling of non-overlapping k-mers when building the index. While the motivation may be to reduce memory footprint, this approach has critical implications: queries become more fragile, because a sequencing read whose informative k-mers fall between sampled positions may go undetected. Any phase shift, even slight, between input reads and indexed k-mers can cause the system to miss biologically relevant matches. The choice of nonoverlapping k-mers also leads to more convoluted algorithms during the query. Then, why not using methods with guarantees and well established methods like minimizers for sampling?

We thank the reviewer for this comment. As noted, we index non-overlapping k-mers for storage efficiency while querying with a sliding window to maintain sensitivity.

The concern about "phase shift" is addressed by our query algorithm: for any query of length $\geq 2k$ (e.g., 48bp for $k=24$), the sliding window is guaranteed to contain at least one indexed k-mer if the target sequence is present. This is a theoretical guarantee equivalent to dense indexing for the query lengths relevant to Malva's use cases (expression quantification, variant detection, isoform analysis). We validated this empirically through the sequencing error robustness analysis, which showed minimal loss of signal even at 10% sequencing error rates (**new Sup. Fig. 7** and **Sup. Fig. 5e,f**). For queries exactly equal to k , sensitivity depends on whether that specific k-mer happens to fall on an indexed position (probability $1/k$ for a random position); however, all analyses we present use probes longer than k , where this limitation does not apply. That is, the default search parameters we showcase in this study (i.e., expression pseudoquantification, variant, and isoform analysis) can be performed with probes of length $> k$.

We also note that other platforms impose similar practical constraints: the public MetaGraph web API requires a minimum query length of 41 bp despite most indices being built with 31-mers. Similarly, the Transipedia/REINDEER framework demonstrates a fundamental sensitivity-specificity trade-off governed by the number of matching k-mers required: with `min_hits=1` (analogous to single k-mer matching), precision drops to 0.20–0.27 for mutation and fusion detection, improving to 0.82–0.98 only when requiring ≥ 3 matching k-mers (Bessi re et al. 2024; Genome Biology, Table 3). This underscores that longer probes with multiple k-mer matches are essential for reliable

detection, a design principle Malva enforces by recommending query lengths $\geq 2k$. Additionally, our sequencing error robustness analysis (**Sup. Fig. 5e-f**, **Sup. Fig. 7**; see also response to Referee 1, page 4) showed minimal loss of signal across error rates up to 10% (including point mutations and indels) for spatial, single-cell, single-nucleus, and cell-line benchmarks.

We also further validated our parameter choices, by performing optimality analysis across k-mer sizes (8–24) and probe lengths (8–44 bp) on the cell-line benchmark (**new Sup. Fig. 14**). F1 scores plateau at >0.90 for k-mer sizes ≥ 18 and probe lengths ≥ 32 bp, confirming that our default parameters (k=24, probe length >24 bp) operate at the regime that optimally balances sensitivity and precision.

Regarding minimizers: they would indeed work, and we do not claim they would have worse accuracy. However, minimizers were designed to guarantee that two sequences sharing a long exact match will share at least one sampled k-mer, a property essential for assembly and overlap detection. Malva's goal is not assembly but rapid detection of known query sequences across a large, continuously growing index. Given our windowed query strategy, minimizers would not improve sensitivity for queries $\geq 2k$ while adding complexity to index construction and merging. Our primary design goal is minimal indexing time and memory usage, not exhaustive k-mer coverage.

Moreover, minimizer detection probability depends on hash rank: k-mers with small hash values are detected frequently, while those with large hash values are detected rarely. This increases variance in sensitivity across arbitrarily short queries. In particular, this becomes problematic precisely for queries of length = k, providing further justification for our requirement that queries be $\geq 2k$. In contrast, non-overlapping sampling exhibits uniform detection probability of $1/k$ for any k-mer, regardless of sequence composition.

There are also practical advantages for our design choice. With deterministic positions, index merging reduces to a simple union operation; minimizer-based indices would require tracking selection context, complicating incremental updates. The indexing algorithm is also simpler: walk through at fixed positions, extract and encode. For minimizers, one needs window tracking, hash comparisons, and selection logic. This is highly valuable to our goal of continuously ingesting new datasets and merging them into a growing Index.

While Malva adopts an inverted-index-based strategy, this is not a novel design choice, Metagraph also relies on such an architecture. Can the authors describe to which extent single cell datasets are already available in tools like Metagraph and Logan search? I'm also asking this because I'm not too convinced by the fact that methods like Fulgor would not scale to many labels (and I'm not really convinced by the experiment, see later on). I believe it is ok to have your own codebase and solution, but it is more interesting to highlight what's different in content and features (e.g., your query possibilities).

We thank the reviewer for this comment. Indeed, Malva's inverted-index-based strategy builds on foundational work by MetaGraph, Fulgor, and others; methods we cite extensively and benchmark against throughout the manuscript (**Sup. Table 1, Sup. Note 1, Sup. Fig. 2**). However, Malva addresses a fundamentally different problem: indexing and querying expression information from large-scale, single-cell data.

The key distinction lies in the expected types of queries, type of outputs, and their resolution. Existing platforms such as Logan and MetaGraph index sequencing data at the sample level: each "label" corresponds to an experiment or biological sample. Logan does indeed contain sequencing reads originating from single-cell runs, but it does not account for single-barcode resolution; all reads from a given run are collapsed into a single identifier. This precludes analyses relating to individual cell types, trajectories, or spatial locations, and makes it impossible to integrate sequence information with cell-level metadata (e.g., cell type annotations, disease status, spatial coordinates).

Regarding scalability: methods like Fulgor require computing unitigs per label, which becomes unfeasible when the number of labels reaches millions (as in single-cell atlases) rather than thousands (as in bulk collections). The bottleneck is not the number of k-mers but the number of unique labels. We re-ran benchmarks with 80 single-cell datasets (~1% of the current Malva Index, ~700,000 cells, ~1.5 TB of raw reads) to illustrate this. While the final index size for Fulgor was just 3-5x larger than Malva's, the more critical limitation is that Fulgor produces an in-memory index, making it impractical for serving many concurrent users at atlas scale. We elaborate on these benchmarks in our response to the comment on "fairness" (see below).

Thus, Malva's core differentiating factor is maintaining cell-level (or spatial-coordinate-level) resolution across tens of millions of labels while preserving query performance. This required: (i) a streaming indexing strategy that processes barcodes without pre-splitting reads into per-cell files (as required by many of the previous methods, and infeasible for spatial data with millions of coordinates), (ii) hierarchical index merging that enables incremental updates as new datasets are released, and (iii) a disk-based query architecture optimized for concurrent access by many users rather than single-user, on-memory batch processing. These design choices, now detailed in new **Sup. Note 4**, enable the biological applications we demonstrate.

We have revised the text to be transparent about these relationships: Malva builds on established inverted-index principles while addressing the specific needs of single-cell data, namely cell-level resolution, scalability to billions of labels, and integration with rich per-cell metadata.

Additionally, the comparison against Metagraph, Fulgor, and others raises fairness questions. If Malva indexes only a sampled subset of k-mers while other tools index the complete set, the

comparison is inherently unbalanced. For a fair evaluation, all systems should be tested using identical k-mer inputs or full-coverage indexing. It is unclear whether this was done. In terms of performances, the manuscript states scalability advantages but does not quantify space usage relative to other tools. It also shows that Malva achieves decent speed although it is not the quickest for queries, but I wondered if the pre computed matrices strategy was involved in the query benchmark or not. Also, did the sequences used for quickness of query response included a mix of positive and negative queries ?

We thank the reviewer for this constructive comment.

Regarding fairness of the comparison: we note that methods like Fulgor and REINDEER do not index all k-mers directly but rather unitigs (from compacted de Bruijn graphs) with minimizer-based lookup; MetaGraph indexes all k-mers but heavily compresses annotations. In that sense, all these methods, including Malva, have full coverage with respect to the information in the input reads, but use different strategies to reduce the lookup table size.

We believe this trade-off is justified for Malva's application scope. As discussed above regarding non-overlapping k-mers, our windowed query strategy recovers full sensitivity for any query of length $\geq 2k$ (e.g., 48bp for $k=24$), which accommodates the vast majority of use cases. The theoretical basis is straightforward: a query window spanning at least two index stride positions is guaranteed to contain at least one indexed k-mer if the target sequence is present. We validated this empirically through the sequencing error robustness analysis (including indels), which showed minimal loss of signal even at 10% error rates. For users requiring exact k-mer queries (length = k), we acknowledge reduced sensitivity proportional to $1/k$; however, such short queries are rare in practice for expression analysis, variant detection, or isoform quantification.

In terms of indexing performance, we illustrate this further in a new benchmark using 100 single-cell datasets from Malva Index (~700,000 cells, ~1.5 TB of raw reads). The number of unitigs indexed by Fulgor was comparable to the number of non-overlapping k-mers indexed by Malva, so indexed content at the lookup-table level was roughly equivalent. These results, including CPU time, peak memory usage, and final index size for Malva and Fulgor on the 100-dataset benchmark, are summarized in the new **Sup. Table 3**.

In terms of query accuracy, we have added results comparing Malva with and without subsampling on the same dataset, demonstrating that our non-overlapping strategy combined with windowed queries achieves equivalent sensitivity to full k-mer coverage at substantially reduced memory cost. Non-overlapping indexing shifts the burden of overlapping k-mer extraction from index construction to query time. This is justified by Malva's design goals: we index sparse, short-read single-cell RNA data and aim to provide a highly available search engine serving many concurrent users, rather than an

index for genomes queried sporadically where memory usage and disk load times are less critical.

Regarding space usage: we have added the sizes of final indices for all compared methods to the supplementary materials. For the 700,000-cell benchmark, Malva's index was >4x smaller than Fulgor's (dBG plus colors combined).

Regarding query benchmarks: the pre-computed gene matrices were not involved; all reported query times reflect on-demand k-mer lookups against Index. We have clarified this in the main text. Query benchmarks included the same both positive and negative queries (50-50%) across all tested methods.

Another important remark is the validation of the AI agent used to translate queries from natural language to a range of possible queries to the Malva API. Usually methods like this are fine-tuned, and their perception of metadata categories or specific vocabulary can still be put to question. I think the paper does not present enough validations on this aspect.

We thank the reviewer for this important comment. Our natural language interface uses an instruction-tuned (not fine-tuned) Llama 3.1 model that receives structured system prompts defining valid query types, gene name formats, and filter specifications. For metadata filtering, we employ a hybrid retrieval approach: categorical variables are matched via fuzzy dictionary lookup combined with vector similarity search against an embedding database built from all unique values in our metadata store, a pattern that's becoming standard in modern information retrieval systems.

We agree that systematic validation of the natural language query translator is essential, and we have now added a quantitative assessment of our backend model. We generated 10,000 synthetic test cases using a local instance of gpt-oss:120b, distributed across eight query categories that reflect real usage patterns: simple gene queries, gene queries with explicit filters, cell type marker requests, sequence searches, database identifier lookups, pathway queries, non-biological queries that should be rejected, and "over-inference traps" designed to test whether the model incorrectly adds filters based on biological knowledge rather than explicit user intent.

We evaluated performance using four metrics: Query Type Accuracy (correct classification of query intent), Gene List F1 (precision/recall on gene extraction), Filter Precision (avoiding over-inference of constraints), and Filter Recall (capturing explicitly stated filters). Running the evaluation on Llama 3.1 8B (our current backend model) yielded correct routing of queries to specific processors in >95% of cases, and extracted entities with ~90% precision (new **Sup. Fig. 12**).

Other remarks:

-“whose complexity is uncoupled” I don’t think it is uncoupled, and I don’t think your results support this claim

We agree with the reviewer that our original phrasing was imprecise. We have rewritten this to avoid the term "uncoupled" and instead describe the specific characteristics of Malva's design.

-The manuscript also mentions that highly repetitive k-mers are masked but does not describe: how repetition is defined, which thresholds or statistical tests are used, how the procedure differentiates repeats from highly expressed genes. Without this, reproducibility is impossible.

We thank the reviewer for pointing this out. We mask low-complexity k-mers using dustmasker with default parameters, not highly repetitive k-mers based on abundance. We have now corrected this in the manuscript.

-The claim of adaptability to modified nucleotide indexing (e.g., m6A) is potentially interesting, yet the manuscript treats this as trivial (“just by changing the alphabet”). In practice, this is unlikely to be straightforward, as modified bases introduce new challenges in encoding, ambiguity handling, and downstream resolution. This point must be expanded technically, otherwise it reads as speculative.

We agree that our original phrasing was overly simplistic. We have revised the text to frame this as a future direction rather than a straightforward extension.

-The mention of Simple Abundance (p.4) is insufficiently explained.

We have rephrased this to clarify that we meant the choice between base-level coverage profiles or a single abundance value per sequence. The revised text now reads "base-level coverage or aggregated counts" to avoid ambiguity.

Referee #4 (Remarks on code availability):

Access to source code only via archive rather than GitHub undermines transparency; I expect a justification. Other projects at least as ambitious had a public repository from the start.

Thank you for this comment. We have enabled the public GitHub repositories containing the API code (<https://github.com/malva-bio/malva> and https://github.com/malva-bio/malva_client). Referees can find the binaries under <https://drive.google.com/drive/folders/1EpssRmtTPj-o12nm8FsC41Jlks8KucE6?usp=sharing>. Documentation is provided in the GitHub repository and in <http://malva.readthedocs.io/>.

References

1. Qiu, “Embracing the dropouts in single-cell RNA-seq analysis,” *Nature Communications*, vol. 11, no. 1, Mar. 2020, doi: 10.1038/s41467-020-14976-9.
2. Audoux et al., “DE-kupl: exhaustive capture of biological variation in RNA-seq data through k-mer decomposition,” *Genome Biology*, vol. 18, no 243, 2017, doi: 10.1186/s13059-017-1372-2.
3. He et al., “Single-cell RNA sequencing reveals heterogeneous tumor and immune cell populations in early-stage lung adenocarcinomas harboring EGFR mutations,” *Oncogene*, 40(2):355-368, Jan. 2021, doi: 10.1038/s41388-020-01528-0
4. Moon et al., “PolyASite v3.0: a multi-species atlas of polyadenylation sites inferred from single-cell RNA-sequencing data,” *Nucleic Acids Research*, Volume 53, Issue D1, 6 January 2025, Pages D197–D204, doi: 10.1093/nar/gkae1043
5. Li et al., “scTML: a pan-cancer single-cell landscape of multiple mutation types,” *Nucleic Acids Research*, Volume 53, Issue D1, 6 January 2025, Pages D1547–D1556, doi: 10.1093/nar/gkae898
6. Bessière et al., “Transipedia.org: k-mer-based exploration of large RNA sequencing datasets and application to cancer data,” *Genome Biology*, Volume 25, article number 266, (2024), doi: 10.1186/s13059-024-03413-5
7. Sebé-Pedrós et al., “The Biodiversity Cell Atlas: mapping the tree of life at cellular resolution,” *Nature*, Volume 645, 2025, Pages 877–885, doi: 10.1038/s41586-025-09312-4

Referee #1 (Remarks to the Author):

Although a standalone version of Malva is now available through binaries, restricted access control may still limit the broader impact of this otherwise useful tool. I am not in a suitable position to assess its availability in detail.

We thank R1 for this comment. As described in the general response above, we have decided to release the full source code under an Academic-use license. In addition, any academic user with an ORCID account will get access to the platform. We hope this fully addresses the concern about broader impact.

Additional suggestions below could further enhance both the tool and the manuscript:

* On the web portal (<https://malva.mdc-berlin.de/>), after submitting a query, the server does not return all cells in the index but only those with counts ≥ 1 . I recommend allowing user-defined parameters (such as `min_counts`) to adjust the returned results. For instance, `min_counts = 0` could return all cells, while `min_counts = 1` (default) would return only those with query hits. Moreover, it seems difficult to map “sample_id” to the original barcode in the returned data — an “original_barcode_id” field appears to be missing. Including normalization factors in the metadata (so that “norm_expr” can be recalculated from “cell_count” and the normalization factors) would also facilitate downstream analyses. For the Coverage Explorer, it would be helpful to add options to select datasets and filter cells according to metadata fields.

We thank R1 for these detailed and helpful suggestions. We have now implemented all of them. Specifically:

1. Results are stored in sparse format by design, so only cells with count > 0 are returned by default; alternatively, we have now added an API endpoint that returns all cells in the Index, yielding behavior equivalent to `min_counts = 0`.
2. We have added a new API endpoint to provide the full mapping between integer sample IDs and original cell barcodes.
3. Similarly, normalization factors are now included in all downloadable result objects and accessible via the Python API.
4. We have re-implemented the user interface of the web application for increased clarity and additional filtering capabilities. As well, the Coverage Explorer now supports the same metadata-based filtering capabilities as the rest of the frontend.

* It seems inconsistent that the results concerning somatic point mutation evaluation (Sup. Fig. 11) are presented under the section “Screening for Germline Variation.” I suggest reorganizing this part and clarifying that the evaluation pertains specifically to the detection of somatic point mutations.

Thanks for pointing this out. We have now moved these results to the section “Benchmarking Malva’s indexing and query performance”.

* Some figures appear to be missing, such as Sup. Fig. 5g and Sup. Fig. 13h.

We thank R1 for catching this. **Sup. Fig. 14** was wrongly referenced as **Sup. Fig. 5g** and **Sup. Fig. 13h** refers to **Sup. Fig. 13f**. We have now fixed these accordingly.

Remarks on code availability

I successfully installed the Python package from the provided wheels using Python 3.11.14 and was also able to run Malva via APTainer. The example results were reproducible (<https://github.com/malva-bio/malva/tree/main/examples>).

Referee #2 (Remarks to the Author):

I thank the reviewers for their thorough and thoughtful revisions in response to the comments and suggestions offered by myself and the other reviewers. I find the revised manuscript greatly improved, both in the detail of the exposition (the additional supplementary note describing the indexing and query framework in detail), and in terms of the suite of experiments presented. I also appreciate the revised framing, which I think better places *Malva* in the context of existing work, and highlights its complementary nature with respect to existing reference-based quantification pipelines, while still conveying the broad scope of new types of analyses that it can enable. I still think it unfortunate that *Malva* will not be made available as an open source tool, which may hamper the ability of those, especially in the methods development community, to build upon the work done here. Nonetheless, I appreciate the authors' efforts to somewhat ameliorate these concerns by providing a binary version of their index building and query program, and committing to those components being free for academic use.

Overall, I am largely satisfied with the revision, and commend the authors on the truly impressive scope of their work. I have only a few remaining technical questions / concerns.

We thank R2 for their comments about the revised manuscript, and for expressing their concerns about code availability. As described above, we now provide full source code in our repository and as a supplementary file, so the non-profit Academic community can now freely browse and use the code. In addition, any academic user with an ORCID account will get access to the platform.

Remaining questions

1. The newly-added exposition on how cell barcodes are handled clarifies a lot that was missing in the previous version of the paper. However, it does raise another question. Since the barcode algorithm looks for exact matches against a user-provided "whitelist", what is the status of technologies where such a whitelist may not be available (e.g. protocols such as variants of split-seq or sci-seq, where there may be no vendor-provided whitelist). In such cases, a common approach is to use a cell calling algorithm; either an initial simple algorithm like the "knee" method, or a more sophisticated approach such as that adopted in `EmptyDrops` [1], or one of several even more sophisticated methods. Likewise, in addition to not having a vendor-provided whitelist, several of these protocols have rather complicated encodings / embeddings of their technical read information (barcodes & UMIs), requiring more sophisticated processing than extracting fixed length subsequences from known positions (sometimes handled by dedicated pre-processing programs like `matchbox` [2] `flexiplex` [3] or `splitcode` [4]). Does *Malva* support these protocols? Is it anticipated that *Malva* will be expanded to support such protocols? If so, some minor details on how such expanded support is planned would be useful.

We thank R2 for raising this point, which we have now clarified in the Discussion and Methods. Malva's indexing step indeed requires a barcode whitelist as input, that is, a predefined list of valid cell barcodes against which R1 reads are matched exactly (e.g. 10x Chromium), or where barcodes are not used (e.g. Smart-seq2). We initially focused on ingesting these kind of data as it constitutes >80% of all publicly available single-cell runs. For technologies that do not provide a vendor-supplied whitelist, such as split-seq or sci-seq, the whitelist must be derived as a preprocessing step before indexing with Malva. This is consistent with how other pipelines (e.g., STARsolo, spacemake) would handle such protocols, and as the reviewer points out, tools like matchbox, flexiplex, or splitcode can be used for this purpose. We have now clarified this in the manuscript (Discussion of limitations, and Methods). In the future, we will update the `malvacrawl` service to integrate such a step prior to indexing, and support automated ingestion from these diverse technologies.

2. Likewise, though the 10x chromium platform was initially designed for short-read RNA-seq data, it has become increasingly popular as a platform for long read data, both PacBio and ONT sequencing technologies have been used with 10x kits (and, in fact with other protocols such as split-seq) to produce isoform-level long read single-cell RNA-sequencing data. While the amount of such data is still dwarfed by the amount of short-read single-cell RNA-seq data, it will be a growing data type in coming years. How do you anticipate *Malva* applying to such data? What types of changes would be necessary to make it work (would it suffice to just change the thresholds, window and k-mer sizes, or would more fundamental changes to the search algorithms be required)? Perhaps the answer is simply that the current tool is unsuitable for long-read technology, and such extensions would be out of scope, even in future versions. Regardless, it would be good to mention this in the manuscript as a potential limitation (and/or a future direction).

We thank R2 for raising this point, which we have now added to the manuscript. Malva has no restriction on read length during indexing; the index construction treats reads as sequences of k-mers regardless of length. The primary difference for long-read protocols arises from the higher base-calling error rate of older ONT/PacBio chemistries/basecalling. This, on the one hand, reduces the fraction of reads yielding exact barcode matches against the whitelist. We plan to address this with a barcode correction preprocessing step, as outlined for the comment above. On the other hand, during query time, the windowed voting parameters (in particular, `sliding_size`, which can be set to approximately the read length, and `min_pct`) could be adjusted to account for elevated error rates, similar to the parameter choices already described for short-read data and supported by our simulated sequencing error analysis. We now address these points in the Discussion and more extensively in **Sup. Note 5**.

3. From the software availability perspective, providing only x86-64 executables running on linux (and likely depending on specific versions of libc) is quite constraining. I'd highly recommend the authors to provide a singularity image and / or docker container as well, so that users can run those executables across a broad range of machines, rather than a specific narrow range of linux machines.

We thank R2 for this practical suggestion. Since we now provide the full source code with compilation instructions, users can build Malva on their platform of choice. However, maximum performance (as reported in the manuscript) can only be ensured in Linux-based systems, as there are a number of platform- and kernel-specific optimizations during indexing and query time that we have not tested in other OSs. We provide an Apptainer image with the same configuration as our production-level service.

Bibliography

- [1] A. T. L. Lun, S. Riesenfeld, T. Andrews, T. P. Dao, T. Gomes, and J. C. Marioni, “EmptyDrops: distinguishing cells from empty droplets in droplet-based single-cell RNA sequencing data,” *Genome Biology*, vol. 20, no. 1, Mar. 2019, doi: 10.1186/s13059-019-1662-y.
- [2] J. Schuster et al., “Powerful read processing with matchbox,” Nov. 2025, doi: 10.1101/2025.11.09.685711.
- [3] O. Cheng et al., “Flexiplex: a versatile demultiplexer and search tool for omics data,” *Bioinformatics*, vol. 40, no. 3, Feb. 2024, doi: 10.1093/bioinformatics/btae102.
- [4] D. K. Sullivan and L. Pachter, “Flexible parsing, interpretation, and editing of technical sequences with splitcode,” *Bioinformatics*, vol. 40, no. 6, June 2024, doi: 10.1093/bioinformatics/btae331.

Remarks on code availability

I have looked at and evaluated the client code (as with the first round of review). I have not reviewed the indexing and query code, as those are provided only as pre-compiled executables (and currently only available for x86-64 linux machines).

Referee #3 (Remarks to the Author):

I co-reviewed this manuscript with one of the reviewers who provided the listed reports.

Referee #3 (Remarks on code availability):

The provided binaries (tested via installation from the wheel file) were successfully installable and functional in my environment. I was able to run both the ``malva index`` and ``malva quant`` steps by following the instructions provided, and the core functionality appears to work as described.

However, the hyperlink labeled “Documentation” in the README (<https://github.com/malva-bio/malva>) appears to be invalid. It would be helpful if the authors could verify and update this link.

Thank you for pointing this out. The broken link has been corrected in the repository.

Referee #4 (Remarks to the Author):

The authors have addressed several points raised previously and have improved the manuscript in a number of aspects. In particular, they clarified the positioning of *Malva* with respect to existing literature and expanded the manuscript by adding several descriptions that were previously missing, especially regarding the methodological aspects. The authors also added a benchmark following the request regarding the validation of the LLM used to generate the queries, clarified which queries are precomputed and which are computed on demand, and improved the description of the query algorithm used to sample cameras. These changes strengthen the manuscript.

However, several points remain unclear or would benefit from further clarification or improvement.

Major Concerns

- about transparency:

The authors should consider making publicly available the queries used to evaluate the LLM's performance, and the prompts employed during the instruction-tuning phase.

Other items are not properly documented, such as the fuzzy dictionary component mentioned in the authors' response is not cited. An other example is Supplementary Figure 1 that refers to the "Malvacrawl" service, but the manuscript does not describe how this service operates. In particular, it remains unclear how the retrieved datasets are processed and how duplicate datasets are detected or avoided.

We thank R4 for these detailed transparency comments. We have addressed all of them. The LLM evaluation queries and instruction-tuning prompts are now publicly available (see Data Availability statement). These are available at <https://zenodo.org/records/19455194>, and with the referee-link https://zenodo.org/records/19455194?token=eyJhbGciOiJIUzUxMiJ9.eyJpZCI6ImU4YzViZTBmLTM4NGEtNGRjYy04Yzg5LTZjYzhmYWU2NmE5ZCIsImRhZGEiOnt9LCJyYW5kb20iOiI3Mzk3ZjJmYmEwYjg0YjllMDlkNGViNjY0OTlwZTU0NSJ9.X5i24WB7sVubDQfZ1yCeghlw9dECBMqwxBe-CLKWwpbWsHBAqwS31gBlcPR6fnnrT_beMOM_16dFF7p_D9FkhQ. We have now expanded the Methods section with a full description of the 'malvacrawl' service, how we handle data discovery and download, metadata parsing and harmonization, and deduplication.

One limitation of the manuscript is definitely that the code will remain private, although it may be made available upon request for academic researchers. While the authors may have valid reasons for this decision such as costs of maintenance, I must stress that this approach does not fully align with common practices in the bioinformatics community, where open-source code sharing is strongly encouraged to ensure transparency, reproducibility, and reuse.

As described in the general response, we have decided to release the full source code under an Academic-use license. In addition, any academic user with an ORCID account will get access to the platform.

- comparison with related tool

The response provided regarding comparisons with tools such as Metagraph or Fulgor remains approximate. The authors argue that Reindeer does not directly index all k-mers but rather unitigs (or related structures). Conceptually, however, this does not fundamentally differ from indexing k-mers, since the indexed sequences ultimately represent the same information. Whether the lookup uses minimizer-based strategies or not does not appear to fundamentally change the nature of the indexing approach. As a result, the distinction made in the response remains somewhat unclear and potentially confusing. The authors should clarify more precisely what differentiates their approach from these existing tools.

We thank this reviewer for raising this point. We agree with the reviewer that all tools in this family ultimately index similar information (which k-mers appear in which labels), and that distinctions based on whether the unit of indexing is a k-mer, unitig, or monotig are not fundamental. We have revised the Introduction and Results accordingly.

We clarify that existing tools (MetaGraph, REINDEER, Fulgor, Themisto, SBT, among others) were designed for bulk collections of hundreds to hundreds of thousands of samples, where they excel. However, at the single-cell scale (reaching tens of millions of labels/cells), three properties become limiting. First, construction typically requires a compacted de Bruijn graph from the union of all input data, a global operation that must be partially repeated when new data are added. Second, annotation storage may grow with total number of labels: MetaGraph uses, during construction, an k-mers-by-labels matrix compressed via RowDiff/Multi-BRWT (e.g., whose construction pipeline requires special handling beyond $\sim 10^6$ columns, as noted in the MetaGraph documentation); REINDEER returns a count vector proportional to the total number of labels per query regardless of sparsity; and Fulgor/Themisto use per-unitig inverted lists whose aggregate size and in-memory footprint grow with total number of labels, requiring very large amounts of RAM at the scale that Malva operates (in our 100-sample benchmark alone, Fulgor's in-memory index was already 326.1 GB vs 0.3 GB for Malva; **Sup. Table 3**). Third, approximate methods (SBT, BIGSI, COBS) maintain one probabilistic structure per label, which has not been achieved for tens of millions of labels (to the best of our knowledge).

Malva addresses these challenges through three design choices now described in the manuscript (Methods, **Sup. Note 4**). Non-overlapping k-mer sampling eliminates de Bruijn graph construction while preserving query sensitivity (**Sup. Note 5**). Per-k-mer sparse inverted lists store only the expressing cell identifiers, and the on-disk prefix-partitioned layout supports constant-memory queries (**Sup. Note 4**). Independent per-sample indexing with incremental merging enables continuous corpus growth without rebuilding. We also note that algorithmic complexity alone does not determine real-world performance; for instance, Malva combines its on-disk layout with kernel-level

`io_uring` for efficient disk access, which is critical for achieving low query latency at scale. For this reason, and in agreement with the reviewer, we have removed Supplementary Table 1 and instead rely on empirical benchmarks.

In summary, the differences operate at two levels. During construction, Malva bypasses de Bruijn graph assembly and instead simply stores sorted pairs of (k-mer, cell-ID) that can be efficiently merged across samples via streaming union; the non-overlapping sampling strategy reduces the number of pairs by a factor of $\sim k$ relative to tools that index all overlapping k-mers, directly translating into faster construction and smaller intermediate storage. During storage and query, Malva writes a simple posting list per k-mer containing only the cell identifiers that express it. In single-cell data, expression is extremely sparse: a typical k-mer appears in a few hundred cells out of tens of millions, and the posting lists are correspondingly short. This sparsity leaves little room for the more elaborate compression schemes used by bulk-oriented tools (color class deduplication, unitig monochromaticity, RowDiff topology sparsification) to provide meaningful additional compression; the data is already sparse and the lists are already short. Our empirical results confirm this: when compared on identical single-cell data (~ 1.5 TB), Fulgor did not yield a substantially smaller index than Malva, while requiring longer construction times and a larger memory footprint (**Sup. Table 3**). This validates that, in the single-cell regime, the simpler approach is not only faster to build and query but also comparably compact.

Table 1 raises concerns. Some of the reported factors are really unclear to me and sound questionable. For example, in Reindeer, the logarithmic factor should not apply to the number of distinct k-mers in each dataset as suggested in the table. Similarly, for Bifrost, it is unclear why a logarithmic factor would appear in the reported complexity. Overall, the assumptions behind these complexity estimates are not sufficiently explained and may be inaccurate. Moreover, the complexity analysis can fall short as much of the speed of these approaches is explained by careful cache-design rather than pure complexities. I would advocate to remove this table.

We thank R4 for pointing out the potential pitfalls of estimating the computational complexity of the different algorithms. As noted above, we have removed **Supplementary Table 1** and instead focus on the existing empirical comparisons across different tasks and scales; we updated **Supplementary Note 1** accordingly.

Minor Concerns

- In Supplementary Figure 5, panel needs an x-axis
- type in Supplementary Figure 2, an extra e to a k-mer
- Note: If I understandThe hierarchical structure presented in Figure 2 is a B-tree-like data structure.
- Will the authors consider releasing the probes used for the analysis, such as poly(A) tails and EGFR mutations ?

We thank R4 for these careful observations. We have now corrected the Sup. Figs accordingly.

The supplementary note figure has now been reorganized for clarity, i.e., we indicate the hierarchical index structure (and other structures) are a conceptual generalization, and we specify the efficient implementation strategies that are used in Malva; thus, the figure has been replaced with a more clear explanation in written form.

Probes for polyA tail analysis are now publicly available (see Data Availability statement), and the *EGFR* mutation probes have been included in the methods section of the manuscript.

The source code is not publicly available. The authors indicate that the code may be made available upon request to academic researchers, but this limited availability prevents an independent evaluation of the software as a community resource, especially for future maintenance.

Referee #1 (Remarks to the Author):

Thanks for the open-source code. The proposed idea in the authors' last response letter looks promising, and the redesigned web portal (Malva Platform) has become an impressive tool with enriched functionalities. However, a few technical issues remain: it appears that some claimed functions are not fully implemented, or that certain bugs hinder their proper use.

We thank this reviewer for their very constructive comments and their extensive testing of our platform. We have looked very carefully into the technical points raised below and we are happy to report we resolved these in the current version of the platform and in malva_client 0.3.2 (see below for a point-by-point explanation).

* In the Malva Platform, for the "Expression" module, the downloaded "Per-cell data" is in fact not per-cell, but aggregated sample-cell type expression data; the normalization factor is also not provided. In addition, there is often a duplicated entry for a given sample and cell type

combination: one corresponding to the measured average expression (if the average > 0), and another showing 0 expression ("n_cells" is 0, and the mean or median expression is NA). The documentation should be improved. For example, the meaning of "enrichment" is unclear, and more detailed information about the single-cell results would be helpful for users. For the "Coverage" module, searches are slow when no filter is applied. In addition, the "zoom in" and "zoom out" functions do not work.

* For the malva_client, I installed version 0.3.1 successfully. The search() method works well, but search_cells() does not return data as expected (status = 'no results') when querying the same gene as in search(). This limits its support for downstream analysis.

* For Malva, I installed it successfully from the source code (Python 3.14.0). The open-source release is expected to be well-received by the research community.

Thank you very much for the extensive stress-testing of our platform, and the very detailed bug report. Specifically, here are the changes/fixes we implemented:

- The download functionality in the Expression module has been reorganised into three clearly labelled options (summary data, metadata, and full single-cell expression matrices), with improved file structure, clearer file names, and an internal MANIFEST.txt better explaining the layout; the single-cell download now includes full per-cell expression matrices with all metadata and normalisation factors.
- Documentation has been improved throughout, including a new UI element (with a "?" icon) clarifying the meaning of the different fields and results.
- A bug affecting the zoom in and zoom out functions in the Coverage Explorer (in some browsers, e.g., Google Chrome in Windows) has been fixed.
- We note that the Coverage Explorer involves data transfers larger than for the Expression Explorer, which explains the slower response times R1 observed; we are actively working on further performance improvements.
- Finally, in malva_client version 0.3.2 we have introduced a new retrieve_cells function that returns the full single-cell expression matrix, replacing the now retired search_cells.

One suggestion for rapid improvement would be to encourage extensive use, evaluation, and benchmarking within the research community –not only in terms of user experience and functionality, but also regarding the accuracy of cell type annotation and the quality of the indexed database.

We very much welcome this suggestion to encourage community use and benchmarking, and will promote this through the existing feedback form in the Malva Explorer, as well as new, interactive communication channels via Gitter Chat and our GitHub repositories.

Referee #2 (Remarks to the Author):

The most recent round of revisions have addressed all of my prior concerns, as well as my largest remaining concern regarding the availability of the source code of the tool. I think the authors have made the correct decision on this point, and value their understanding and appreciation of the importance of source code availability as had been raised by several of the referees, including myself. I have no further substantial concerns with the manuscript.

We thank this reviewer for their many comments, very thoughtful suggestions and kind communication throughout the peer-review process.

Referee #2 (Remarks on code availability):

I was able to inspect and review the code, though have not had a chance to do so comprehensively.

My first attempt to run the example in the example directory failed because it seems that ``anndata`` was not amongst the listed dependencies during build. Thus, execution failed during the ``1_run_malva.sh`` script. However, once I added ``anndata``, the example ran successfully and completed with the expected results.

Thank you very much for raising this point. We have now added the previously missing dependency on `anndata`.

Referee #3 (Remarks to the Author):

I co-reviewed this manuscript with one of the reviewers who provided the listed reports.

We thank this reviewer for their very extensive testing of our platform throughout the peer-review process.

Referee #4 (Remarks to the Author):

The authors have made a clear effort to clarify additional material after my remarks, and made the necessary corrections. Good luck with the final steps before publication!

We thank this reviewer for their comprehensive and useful feedback and scientific discussion that encouraged us to greatly improve the quality of our manuscript and of Malva as a whole.

Referee #1 (Remarks to the Author):

The effort to improve the tool is evident. It would be appreciated if the following remaining technical issues could be resolved:

* Add an option to output all cells, not only those positive for the queried genes/sequences, both in the web portal and the API (malva and malva-client). This would allow users to recalculate metrics such as “fraction expressing”, mean normalized expression (including zeros), and z-scores by themselves. Such measurements are useful for downstream analyses, including cross-dataset comparisons. This requirement was proposed in my previous round of comments, but has not yet been implemented.

First of all, we thank the referee for this careful and extensive testing of our platform. Regarding this suggestion, we addressed this point in our point-by-point response #2 to referee #1:

“[...] we have now added an API endpoint that returns all cells in the Index, yielding behavior equivalent to `min_counts = 0`.”

We preferred to retain the design choice not to return all cells with the results, as user misconfiguration might lead to excessive load on our servers when performing large batch searches.

Rather, we had (response #2) implemented a function that we have however now better documented, see

`client.get_cells_by_metadata(include_all_database_cells=True, include_cell_metadata=True)` in `malva_client`, which allows to download all cells across the Index, with their sample ID of origin, cell ID, cell type labels and normalisation factors. These can be merged with the non-zero single-cell matrices.

We have now also added a "Download all cells" button in the web UI (limited to 50 samples, see explanation below).

* For the web portal, the downloaded per-cell expression matrix currently has a limit of 50. I suggest removing this limit.

We thank the reviewer for this suggestion. We benchmarked that removing the limit of samples per download negatively impacts the performance of the web platform when many users are running analyses in parallel. Therefore, we decided to keep this limit for now (potentially increasing it as experience and server capacity evolves, as an engineering exercise). In the meantime, we encourage users who require large analysis to use the Python API, which has no limits. This follows standard practice of other tools, e.g., BLAST, MetaGraph...

* For the `malva-client`, the obtained expression values are always 1. In addition, the API documentation is still insufficient. For example, the meaning of the expression values is unclear. Following the tutorial or quick-start (<https://malva-client.readthedocs.io/en/latest/quickstart.html> or https://github.com/malva-bio/malva_client/tree/main), errors/bugs frequently occur.

Code:

```
result = client.search("SPP1")
cells = client.retrieve_cells(result, sample_ids=[123456])
```

returns the error:

"No cell IDs returned for this search job; No cells returned for this search job".

For the code:

```
value_result = client.search("SPP1", aggregate_expression=False)
```

The returned ``value_result`` contains empty data.

Thank you for the careful stress-testing of our documentation. These `sample_ids` (e.g., 123456) were meant as placeholders, and we have now fixed it accordingly. We also note that `aggregate_expression` is deprecated in favor of `client.retrieve_cells`.

* For the coverage explorer, it is still slow. I could not even obtain results for a gene such as PTPRC (11:67435510-67437682).

Thank you for the extensive testing of *Coverage Explorer*. We benchmarked via API, and response times for the chr11:67435510-67437682 locus were around 30 seconds. Rendering on the web browser additionally takes 1-10 seconds, depending on the browser and machine; we are actively working on improving the UI/UX of this component. Please note that coverage matrices are much larger (cells-by-nucleotide instead of cells-by-gene) and therefore take longer to generate and display (compared to Expression Explorer).

Moreover, occasional, slower performance can also be due to temporary throttling of an user based on usage. That is, if an account/token generates too many requests (>30 per minute), these are throttled down to avoid overload. This is common in most web-based bioinformatics tools. We now relaxed the throttling factor from 0.1 to 0.5, to avoid excessive slowdown that might lead to the observed timeouts.

We anticipate gathering more experience in setting these parameters as more researchers use our platform, and we apologize for the inconvenience this mechanism may have caused.