Published in final edited form as:
Nat Methods. 2021 March 01; 18(3): 262–271. doi:10.1038/s41592-021-01076-9.

Fast searches of large collections of single cell data using scfind

Jimmy Tsz Hang Lee[#1], **Nikolaos Patikas**[#1,2], **Vladimir Yu Kiselev**[1], **Martin Hemberg**[1,*]

[1]Wellcome Sanger Institute, Wellcome Genome Campus, Hinxton CB10 1SA, UK

[2]UK Dementia Research Institute, Department of Clinical Neurosciences, University of Cambridge, Cambridge, CB2 0AH, UK

[#] These authors contributed equally to this work.

Abstract

Single cell technologies have made it possible to profile millions of cells, but for these resources to be useful they must be easy to query and access. To facilitate interactive and intuitive access to single cell data we have developed scfind, a single cell analysis tool that facilitates fast search of biologically or clinically relevant marker genes in cell atlases. Using transcriptome data from six mouse cell atlases we show how scfind can be used to evaluate marker genes, to perform *in silico* gating, and to identify both cell-type specific and housekeeping genes. Moreover, we have developed a subquery optimization routine to ensure that long and complex queries return meaningful results. To make scfind more user friendly, we use indices of PubMed abstracts and techniques from natural language processing to allow for arbitrary queries. Finally, we show how scfind can be used for multi-omics analyses by combining single-cell ATAC-seq data with transcriptome data.

Introduction

Single cell technologies have made it possible to profile large numbers of cells [1–6], and there are currently several ongoing efforts to build comprehensive atlases of humans [7] and other organisms [1,2,6,8]. One ambition of these projects is to create references that can be used as a foundation for both basic research and clinical applications [7]. To achieve this goal and to ensure that cell atlases can fulfill their potential, it is critical that they are easy to access for many users, ranging from computational biologists interested in large scale analyses to wet lab biologists and clinicians who are interested in a specific gene and regulatory pathways from a ChIP-seq experiment, or disease associated with genetic variants.

[*]Corresponding author: mh26@sanger.ac.uk.

Author contributions
MH conceived of the project and supervised the research. JL, NP, VK and MH contributed to the code, JL, NP and MH analyzed the data, JL and MH wrote the manuscript with input from NP and VK.

Conflicts of interest
There are no conflicts of interest.

Many of the challenges in working with single cell data stem from its large size. Typically, a single cell RNA-seq (scRNA-seq) dataset is represented as a gene-by-cell expression matrix with ~20,000 genes and may include millions of cells. For epigenetic data, e.g. scATAC-seq [5], the situation is often worse as the matrix can have many more rows, each representing a peak. Even though the matrix is sparse, working with such a dataset places high demands on computer hardware. Many operations are not only time consuming, but require advanced bioinformatics skills. To allow for both interactive and high-throughput queries of a large cell atlas, novel algorithms and data structures are required.

To ensure that large single cell datasets can be accessed by a wide range of users, the underlying software must (i) allow for complex queries, (ii) take full advantage of the single-cell resolution of the data, (iii) support different types of assays besides RNA-seq, (iv) be fast and easy to use, (v) be possible to run interactively through a web browser to access large repositories of data, (vi) be possible to run locally by a user interested in analyzing their own data (vii) be easy to interface with other resources such as the genome wide association study (GWAS) catalog [9], the gene ontology (GO) knowledgebase [10], the disease ontology Medical Subject Headings (MeSH) [11], and collections of clinically important genetic variants [12,13]. However, none of the computational tools available today for collections of scRNA-seq datasets, e.g. Panglao [14], the UCSC Cell Browser [15], and the EBI Single Cell Expression Atlas [16], provide the required functionality and versatility.

Here, we present scfind, an R package that leverages natural language processing techniques to make single cell data accessible to a wide range of users by enabling sophisticated queries for large datasets through an interface which is both very fast and familiar to users from different background. The central operation carried out by scfind is to identify the set of cells that express a set of genes or peaks (i.e. the query) specified by the user. By themselves, such searches can be very powerful as they can identify the cell type that is most enriched for cells corresponding to the query within hundredths of a second. Note that the searches carried out by scfind differ from existing methods such as CellAtlasSearch [17] and CellFishing [18]. These methods take as input the expression levels of a cell and determine the most similar cell or cell-type in a reference dataset, and consequently they cannot be directly compared to scfind. Furthermore, we demonstrate how the fast searches open up new possibilities for global analyses of cell atlases, e.g. for identifying housekeeping genes and tissue-specific genes. By taking advantage of information from PubMed abstracts and natural language processing techniques, scfind allows a user to input various type of query and automatically translates it into a gene list which is used for the search. Finally, we show how scfind can be applied to both scRNA-seq and scATAC-seq atlases together to identify putative cell type specific enhancers.

Results

Efficient compression allows for fast queries with single cell resolution

The input for building an scfind index is one or more non-negative matrices where rows represent genes, peaks, or another feature which are used as query terms and each column represents a cell. There are no assumptions regarding normalization, but it is beneficial if the matrix is sparse. To build a compact index to store the matrix, scfind uses the row names as keys, and compresses each row in two steps. The first step, which is lossless, stores the indices of the cells with non-zero values as binary strings using Elias-Fano coding [19]. In the second, lossy step, the approximate values are stored with a user-defined precision (Figure 1a, Supplementary Figure 1). The combination of lossless and lossy strategies means that the most important information, whether or not a key is present in a cell, is perfectly retained, whereas, only an approximation of the value is stored to save space. Since the number of bits used per entry in the second step is decided by the user, it is possible to tune the trade-off between compression and information loss. If the cells have been assigned labels, e.g. using unsupervised clustering or projection methods, these are stored and used to group the cells returned from a query. Importantly, the index is modular, making it possible to add new datasets without having to reprocess the ones that have already been indexed.

We applied scfind to five large mouse scRNA-seq datasets: the Mouse Organogenesis Cell Atlas (MOCA) [6], the Mouse Cell Atlas (MCA) [2], two collections, FACS and 10X, from the Tabula Muris (TM) [20] which are sampled from the whole body and the Zeisel [4] dataset (BCA), which contains cells from the brain. We also indexed the sciATAC-seq mouse atlas [5] using the genomic coordinates of the 167,013 unique peaks as keys instead of gene names. When searching, the user does not need to specify the exact coordinates of the peaks; scfind will automatically find all peaks inside an interval to use as a query. Compared to the full expression matrix, scfind achieves compression ratios of 10-300, and is more compact than other file formats (Figure 1b, Supplementary Figure 2). Both the relative compression and the absolute size of the index depends on the sparsity of the dataset, with the highest compression for MOCA where >95% of the expression matrix consists of zeros (Figure 1c). Since an index for 100,000 cells takes up no more than ~150 MB of disk space, scfind makes it possible to analyze very large datasets on a standard laptop. Since the four largest scRNA-seq datasets contain only ~10^3 reads/cell, the range of expression values is limited, allowing them to be well approximated with only two bits/cell for the lossy compression (Figure 1d). The compression scheme used by scfind is not just memory efficient, but also fast; it typically takes less than a minute for each tissue to create an index (Figure S3).

To identify the cells that match a query, scfind decompresses the strings associated with each key to retrieve the cells with non-zero expression. If cell labels have been provided, scfind will automatically group the cells and a hypergeometric test is used to determine if the number of cells found in each cell type is larger than expected by chance. Search times are well below one second, even for the MOCA dataset with ~2 million cells, and they scale linearly both with the number of genes in the query and the number of cells in the reference (Figure 1e,f).

As an example, we consider the genes *Il2ra*, *Ptprc*, *Il7r* and *Ctla4*, which correspond to T cell surface markers commonly used in FACS experiments. For the TM FACS dataset the results show that T cells in the bone marrow are the most highly enriched group, and further inspection reveals that 24 of the 38 cells that express all four genes are T cells from different tissues, suggesting that this is the most relevant hit (Figure 2a).

The default is to require that cells contain all keys in a query, but the user can specify for each key if it is to be used with OR or NOT logic. The logical operators make it possible to form complex queries and one application is to carry out *in silico* gating of cells, i.e. subsetting cell states of one cell type by their gene expression status by taking advantage of the fact that scfind returns individual cells matching a query. Based on the expression of *Il2ra*, *Ptprc*, *Il7r* and *Ctla4*, we defined five non-overlapping sub-clusters as suggested by Golubovskaya and Wu [21], and they were detected in the thymus for both TM datasets. (Supplementary Figure 4).

Fast and flexible identification of marker genes

The ability to identify all cells expressing a gene in hundredths of a second makes it possible to exhaustively scan the entire transcriptome. That is, scfind can be used for optimization problems by evaluating all genes based on how well they perform for a specific task. One such application is to search for marker genes. To evaluate gene g_i as a marker for cell type c_j, we defined four quantities: true positives are cells that belong to c_j and express g_i, false positives are cells from other cell types expressing g_i, false negatives are cells from c_j that do not express g_i, and true negatives are cells from other cell types that do not express g_i. From these quantities precision, recall and F1 score can be computed to rank the genes. One can also combine multiple marker genes through AND or OR logic. Using AND to improve precision will typically result in reduced recall (and vice versa for OR), so combining markers will alter the trade-off between precision and recall, but it rarely improves the F1 score.

To illustrate the marker genes search, we have carried out a large scale comparison of cell type markers identified by Seurat and the manually curated CellMarker database [22] for the TM FACS dataset. We have also carried out the comparison of cell type markers from the database for the other four datasets to demonstrate that the results of scfind is not biased toward the genes used in cell type clustering (Supplementary Figure 5-7). Interestingly, these marker genes have moderate recall values but low precision values (Figure 2b, c, Supplementary Figure 5-7). By contrast, scfind can identify marker genes that result in either high precision, recall, or F1 score (Figure 2b, c, Supplementary Table 1, 2). For each cell type ranking all genes took only an average of 0.28 s for TM FACS, 0.18 s for TM 10X, and 0.74 s for MCA (Supplementary Figure 8), suggesting that an entire atlas can be ranked in minutes. To compare how well marker genes perform across mouse cell types, we identified the best combinations of up to five marker genes in all three adult whole body atlases. When we modified the search to only compare against cell types from the same tissue, the scores increased (Figure 2d, Supplementary Table 1, 2).

The set of marker genes is defined as the smallest number of genes that allows a cell type to be reliably identified. Instead, one can use the largest possible set of genes that are expressed above a threshold, and we define the maximal set of marker genes as those genes with recall>0.9. Since recall is defined as TP/(TP + FN) the maximal marker set is independent of the other cell types in the atlas, unlike a definition based on precision or F1. The cardinality of the maximal marker gene set gives an indication of heterogeneity as a homogeneous cluster will have a larger number of genes. For example, the mean number of maximal markers for the TM FACS cell types is 238, but T cells, which we know have different subsets (Supplementary Figure 4) have only on average, 114 maximal markers (Supplementary Table 4).

We consider the five cardiomyocyte markers that we identified previously. Comparison with a list of 1,296 surface markers [23] reveals that none of them can be found on the surface, but scfind's similarity search function returns five other markers for which antibodies are available (Supplementary Figure 5). Additionally, scfind features lists of reported transcription factors [24] and chemokines [25], allowing for searches using only those genes.

Identification of housekeeping genes and cell type specific genes

In addition to identifying sets of genes that are specific to a cell type, we can also evaluate how many different cell types express a specific gene. We define a gene as present if it is found in at least $max(10, 0.1N_i)$ cells, where N_i is the total number of cells of type i. Consistent with previous studies based on bulk RNA-seq [26] we find a bimodal pattern such that many genes are either widely expressed or found in only a few cell types (Figure 2e). The number of cell types where a gene is expressed is strongly correlated between the different atlases (Spearman's rho = .63 on average), suggesting that the findings are robust despite the differences in sequencing depth and cell type annotations.

Of particular interest are those genes expressed across most or all cell types, a.k.a. housekeeping genes. A widely used definition of housekeeping genes was obtained from several bulk RNA-seq experiments, and it constitutes 3,505 genes [27]. We compared the 3,144 genes from this list which were present in the MCA and the two TM datasets with the same number of widely expressed genes from each of the three atlases. This revealed a high degree of similarity, as 912 genes were found on all four lists (Figure 2f, Supplementary Table 3). An additional 740 genes were found in all three atlases but not in the literature curated list, suggesting that there are a substantial number of ubiquitously expressed genes that are not traditionally considered as housekeeping genes. Moreover, 1,345 of the genes from the literature were not identified as widely expressed in any of the three atlases, implying that the results can differ substantially depending on whether bulk or single cell data is used. Since the ranking of the genes takes only ~1 second/cell type (Supplementary Figure 9), it will be straightforward to continuously refine lists of housekeeping genes as more data becomes available. We also found 4,966 genes present in only one or two cell types in all three atlases (Supplementary Table 3). Closer inspection of the cell types containing the largest number of specific genes shows that they are mainly found in hepatocytes, brain and testis (Supplementary Table 5).

Frequent pattern growth algorithm ensures that long queries return meaningful results

One challenge in using queries involving even a moderate number (>5) of terms for sparsely sequenced datasets is that it is very likely that an empty set of cells will be returned. To ensure that meaningful results are returned without requiring the user to manually modify the query through trial and error, scfind features a query optimization routine. This procedure identifies subsets of the original query terms, hereafter referred to as subqueries, that are guaranteed to return non-empty sets of cells. Since the number of possible subqueries may be very large, evaluating all possible combinations is intractable. Instead, scfind uses a strategy inspired by the frequent pattern (FP) growth algorithm [28] to identify subqueries that return many cells. Compared to the brute-force algorithm, FP-growth can speed up the procedure for approximately 2 orders of magnitude when the original query contains 20 genes (Supplementary Figure 10). To help the user decide which subquery to use, they are ranked by a score inspired by the term frequency-inverse document frequency metric [29]. To illustrate the subquery optimization we consider eight genes which can distinguish cardiac muscle cells in the heart [30]. A search of the TM FACS dataset returns no cells, but the subquery optimization ranks the query involving *Actc1*, *Myh6* and *Nppa* as the best one and heart cardiac muscle cells is the only significantly enriched cell type (Figure 3a).

To carry out a high throughput involving more realistic and biologically meaningful positive control queries, we considered the GO database [10] where each term is associated with a manually curated list of genes. Although some terms are unlikely to be cell type specific (e.g. “DNA binding”), for some terms it is reasonable to expect that they should be enriched for one or more cell types. For example, after using the subquery optimization we find that the best matches for “lung saccule development” (GO:0060430) has the highest enrichment in several different cell types found in the lung for the MCA, MOCA, TM 10X and TM FACS datasets. We queried all terms containing between 5 and 25 genes against the three mouse atlases, and report the p-values of all cell types for the best subquery (Figure 3b, Supplementary Figure 11, 12). Biclustering of the resulting matrix reveals that the rows, corresponding to the cell types, are not grouped by tissue. Instead, functionally similar cell types, e.g. immune cells, are grouped together, regardless of their tissue of origin. By contrast, the columns which represent the GO terms, form a hierarchy which corresponds well to the GO annotation evidenced by the fact that the parent categories are not mixed. This result is reassuring since the GO terms are hierarchical and can be represented as a tree structure. Another important observation is that the subquery optimization takes <10 seconds even for queries involving up to 25 genes (Figure 3c). Taken together, we have demonstrated that subquery optimization can yield biologically meaningful results in a timely manner.

Queries involving biomedical keywords

One of the central purposes of a cell atlas is to facilitate the identification of cells or cell types associated with a specific query. We have demonstrated that scfind provides this functionality given a query formed by a list of genes. Although such queries are convenient for researchers with expertise in genetics and genomics, it restricts the number of people that can use the cell atlas. To allow for a much wider range of queries, scfind leverages the knowledge that has been accumulated over decades in the form of abstracts from the biomedical research literature stored in NIH’s PubMed database [31]. The PubTator [32] resource has parsed all PubMed abstracts and extracted gene names, genetic variants, disease names and chemicals for each PubMed ID. We assume that if a keyword, i.e. variant, disease or chemical, is mentioned in the same abstract as a gene, then the gene is relevant for queries involving that keyword. Based on this logic we have created dictionaries where each keyword is associated with a list of three or more genes sorted by how often they co-occur. In total, the dictionaries map >300,000 keywords to gene lists (Table 1).

The variant dictionary makes it possible to combine text mining and expression analysis to identify the cell type where a variant is most likely to have an impact [33,34]. Results for five variants queried against the TM FACS data (Supplementary Table 6) are well supported by literature. Many diseases are referred to by many different names, a redundancy which can be resolved by mapping each of the 171,162 disease names to one of the 6,639 categories defined by Medical Subject Headings (MeSH) and Online Mendelian Inheritance in Man (OMIM). With the same approach, we provide five example queries to illustrate this feature (Supplementary Table 7). For example, a search for “Legionella infection” returns cell types from the lung as the top hits. Considering the etiology of Legionnaire’s disease, a bacterial infection causing pneumonia, this is clearly a relevant result. Combining MeSH IDs and Chemical Entities of Biological Interest [35], scfind allows searches for drugs, hormones, toxins, etc and the results for five searches are reported here (Supplementary Table 8). To allow for word and phrase searches, other than disease names and chemicals, that are common in the biomedical literature, scfind includes a dictionary, based on PubMed Phrases [36]. (Supplementary Table 9).

PubTator and PubMed phrases expand the number of possible queries, but there are still substantial limitations as the user is required to enter the precise keywords found in the dictionaries. To allow for greater flexibility, scfind employs a strategy from natural language processing, word2vec [37], to map queries that are not found in the dictionary to the existing keywords (Figure 4a). The idea behind word2vec is to embed words in a vector space which makes it possible to match words and phrases based on similarity of their associated vectors. The neural network carrying out the embedding needs to be trained on a large corpus of text, and scfind uses one based on PubMed abstracts [38].

Addition of the basic gene queries to the logic operators, the subquery optimization, the dictionaries based on PubMed abstracts, and the word2vec functionality, results in a versatile search engine that can identify cell types corresponding to complex queries at speeds that allow for an interactive workflow. When given a complex query, scfind first identifies gene names and keywords that exist in one of the dictionaries. Any remaining terms are then mapped using word2vec to the nearest keyword, and the gene names obtained are then used as a query (Figure 4a).

The dictionaries mapping keywords to gene names can be inverted to provide a set of keywords associated with each gene. The inverted dictionaries are useful for interpreting a list of genes, and an intuitive way to visualize the results is through a word cloud where the size of each word is related to how frequently it is associated with the gene set. One application is to make sure that the gene list obtained following subquery optimization still represents the user’s original intent. As an example, we consider a set of markers discriminating between atrial and ventricular heart tissue [30]. Subquery optimization for the TM FACS datasets suggests *Hyal2* and *Myl2* as the best query, and it shows cardiomyocytes and heart epithelial cells as the most enriched cell types. In the resulting word cloud (Figure 4b), we note the prominence of the term “ventricular” which suggests that the sub query still reflects one of the key properties of the original gene list. Moreover, the word cloud contains several variants, e.g. rs104894368 and rs104894369 that have been associated with hypertrophic cardiomyopathy [39,40], thus providing novel suggestions and interpretations to the user.

Application to single cell ATAC-seq data identifies cell type specific enhancers

Super enhancers, sometimes referred to as stretch enhancers [41,42], are defined as regions >3 kb that harbor multiple enhancer sites. Super enhancers are important for gene regulation and although they have been extensively studied, previous studies have largely been based on bulk data. We hypothesized that scATAC-seq data could make it possible to identify what cell type a super enhancer is associated with. From dbSuper [43] we obtained 4,328 super enhancer loci from eight tissues, and for each locus we identified the most significantly enriched cell types by searching for overlapping peaks using an OR query, thereby assigning 2,726 loci to a unique cell type (Figure 5a, Supplementary Table 10). This includes several known loci, e.g. *Hnf4a*, *Nr5a2*, *Ppara* and *Rxra* in hepatocytes [44] and *Nr4a1* in monocytes [45], that were correctly identified.

One of the biggest challenges in studying gene regulation in metazoans is to identify the targets of distal enhancers. As enhancer targets cannot be predicted from sequence alone, information both about chromatin and gene expression is required [46]. Since proximity can be a useful guide, we hypothesize that open chromatin peaks that are only found in one cell type are likely to target nearby genes that are specifically expressed in the same cell type and searched the TM FACS atlas for genes that are highly expressed in only one cell type for each tissue. For each of those genes, we searched the sciATAC-seq atlas for peaks that were unique to the same cell type and within 100 kb of one the start sites (Figure 5b). This procedure identified 15,583 open chromatin-gene pairs from seven tissues (Supplementary Table 11). We found support in the literature for several of the identified candidates (Figure 5b) [43]. As a validation, we searched for enriched transcription factor binding motifs in the putative enhancers for three cell types where more than 500 loci were identified. (Figure 5c).

Scfind can also be applied to multi-omics datasets. We created an index for 5,081 cells from mouse neonatal cerebral cortex where both open chromatin and transcriptome were profiled using SNARE-seq [47]. To demonstrate the advantage of joint searches with scRNA-seq and scATAC-seq, we compared to using only one modality, and we found that utilizing both modalities improves the precision (Supplementary Figure 16). We modified the marker gene search method to carry out an AND query for each gene along with each of the peaks found within 1 Mb of its TSS. This allowed us to identify putative enhancer-gene pairs that are specific to each cell type. As an example, we identified 983 putative genes-enhancer pairs for Cajal-Retzius neurons and 974 pairs for endothelial cells. Our analysis identifies several loci that have been shown to be relevant [48]. Our search for enriched motifs at the cell type specific open chromatin sites returned several hits (Figure 5d, Supplementary Figure 17).

Discussion

As the size of single-cell datasets grows exponentially, there is an urgent need for computational tools to allow for fast and efficient analysis. We have presented scfind, a method for searching large collections of single cell data to identify sets of cells that correspond to diverse criteria. Although the resource requirements for scfind still scale exponentially, the rate is substantially lower than for alternative approaches making it feasible to work with datasets containing millions of cells using standard hardware. Scfind is able to carry out complex queries in less than a second, suggesting that scfind can support real-time exploration of cell atlases. If scfind is going to be used in a manner analogous to how Internet search engines are used, it will require that datasets are indexed and made available through a website such as Panglao [14], the UCSC Cell Browser [15], or the EBI Single Cell Expression Atlas [16], where new datasets are regularly incorporated. However, our R package can be run locally using either pre-indexed datasets or an index constructed by the user as detailed at https://github.com/hemberg-lab/scfind.

Our subquery optimization routine combined with the PubMed based dictionaries makes it possible to resolve arbitrary queries, ensuring that scfind can be used by people without expertise in genetics and genomics. The searches are difficult to evaluate quantitatively, but our tests suggest that they frequently provide reasonable results. Nevertheless, there are several ways by which results could be improved. The way in which the information is extracted is not very sophisticated, and associations are based solely on co-occurrence without considering the context. For example, an article may incorrectly put forward evidence that a specific gene is *not* important for a disease. Although we cannot rule out that some searches will be confounded, we conjecture that the impact of this may be limited since positive result bias is well documented in biomedical science [49]. More advanced methods for processing natural language will make it possible to further exploit this knowledge when interpreting and analyzing high-throughput datasets. Another shortcoming is that most of the testing has been carried out using mouse datasets, even though much of the information that is used to relate queries to gene lists is based on human studies, and this may explain some inconsistencies.

The primary use case for scfind is to query large collections of previously annotated single-cell datasets to identify the group of cells that provide the best match. As a second use case, scfind can facilitate the annotation of newly collected datasets. Once the cells have been grouped through unsupervised clustering, determining the biological significance of each of the clusters is typically a difficult and time-consuming process [50]. It typically requires expert knowledge and the researcher must search the literature to match marker genes with known pathways, processes or cell types. Scfind has the potential to speed up this process in several ways. For example, the methods for marker gene identification will make it possible to quickly identify relevant gene sets and the search functions allow the user to find the cluster that best corresponds to a gene set that has been derived from other experiments from the literature.

Online Methods

Compression and decompression

Compression is carried out separately for each row (typically representing a gene or a peak), i, and cell-type, c, and it is split into two steps. First, we use the Elias-Fano coding [19] to store the array containing the indexes of cells with non-zero elements. The Elias-Fano code uses two bit strings, H_{ic} and L_{ic} to store an array of ascending integers. The sparsity of the array automatically determines how each value is stored in H_{ic} and L_{ic} to provide asymptotically optimal compression. The total number of cells, n_c, and the number of cells with non-zero expression, n_{ic}, are also stored to allow for decoding. Second, for all cells the non-zero expression values, g_{ij}, are fit to a log-normal distribution and the mean, m_{ic}, and the variance, v_{ic}, are stored. Given a user-specified number of bits, b, each value g_{ij} is assigned to one of the 2^b quantiles. The bit vectors are then concatenated and stored as B_{ic}. The bit strings B_{ic}, H_{ic} and L_{ic} along with n_{ic}, m_{ic}, and v_{ic} are stored as entries in a hash table with the row label as key. Since each cell-type is stored separately, it is easy to filter results by excluding cell-types, tissues or experiments.

When decoding, the indexes of the non-zero cells are first obtained from H_{ic} and L_{ic}. Since the decoded lists are sorted, intersection can be carried out in linear time. By decoding the lists with the smallest number of cells first, the search is further sped up. An approximation of the original expression value, h_{ij}, can be obtained by identifying the midpoint of each quantile for the log-normal distribution with mean m_{ic} and variance v_{ic}.

Compression ratios and file sizes

The compression ratios in Figure 1b and Figure S1 were calculated by the cumulative size of index / cumulative size of raw data and object size / raw data size, respectively, using the command `obj_size` of the R package `lobstr`, comparing the size of the original expression matrix and the compressed scfind index. The index sizes reported in Fig 1c correspond to the file-sizes when saved to disk.

Search times

For Fig 1e,f we used the `hyperQueryCellTypes` function to search for cells expressing the randomly selected genes. For the comparison, we used the `counts` function applied to each of the `SingleCellExperiment` objects used to represent the tissues from the TM FACS dataset in order to obtain the count matrix. To create SingleCellExperiment, Seurat and Loom objects, `SingleCellExperiment` function of the SingleCellExperiment package, `CreateSeuratObject` function of the Seurat package and `build_loom` function of the SCopeLoomR package are used respectively. The parameter `chunk.size` was set as 1000 and 5000 for Loom-1000 and Loom-5000, respectively. The search times in Fig 1e were calculated as the average of 100 queries involving random sets of genes. For both panels we used rejection sampling to ensure that only queries returning a non-empty set of cells were considered. All analyses are performed on the Jupyterhub platform built with 100GB RAM and 4 Intel Xeon 2.50 GHz CPUs.

Quantization accuracy

The Spearman correlation was computed using the built-in R function by comparing the quantized gene expression values, q, with the original values, g. The quantized gene expression values can be retrieved using the command `getCellTypeExpression`.

Logical search operators

By adding '*' in front of a gene name it will be used as OR, and by adding '-' in front, it will be used as NOT. It is also possible to combine the two operators for a NOT OR query.

Identification of T cell subsets

We defined the different non-overlapping subsets of T cells using combinations of *Il2ra*, *Ptprc*, *Il7r* and *Ctla4*[21]. The queries used were "-*Il2ra*, *Ptprc*, *Il7r*" (naive T cells), "*Il2ra*, -*Il7r*" (Effector T cells), "-**Il2ra*, -**Ptprc*, *Il7r*" (Effector memory T cells), "*Il2ra*, -*Ptprc*, *Il7r*" (Central memory T cells), and "*Il2ra*, *Ptprc*, *Il7r*, -*Ctla4*" (Resting T reg cells).

Cell-type enrichment

To determine if cell-type i is enriched if m_i cells are observed out of a total of m cells returned for the query, a hypergeometric test is used. Here, the total number of balls is given by N, the number of white balls in the urn is given by n_i, the number of white balls drawn is given by m_i, and the total number of draws is given by m. The reported p-value is further adjusted using the Benjamini-Hochberg procedure [51] to correct for multiple testing.

Visualization

To aid in the visualization of the results, scfind calculates a UMAP [52] projection of the cells during index construction. Cells that match a query are automatically identified in the UMAP projection. This feature can help the user identify patterns, e.g. sub-clusters, formed by the cells matching the query.

Identification of marker genes

We ran the scfind and Seurat commands `cellTypeMarkers` and `FindMarkers` respectively for each cell type from heart, kidney, liver, lung, mammary, marrow, muscle, pancreas, spleen and thymus of the TM FACS datasets. For each cell type present in the CellMarkers database we extracted the same number of markers as was listed in the database. The marker genes of cell types with matched names from the Seurat and CellMarkers database were evaluated using the `evaluateMarkers` command.

Evaluation of precision and recall for marker gene search

When searching for marker genes, all genes are evaluated and ranked based on either precision, recall or F1 (default). When evaluating multiple markers in AND/OR fashion, scfind employs a greedy strategy by considering the cells expressing all/at least one of the top m markers to calculate the precision and recall as described above.

Identification of maximal marker gene sets

By default scfind includes genes with a recall greater than 0.9, but the threshold may need to be adjusted depending on sequencing depth.

Similar genes

For the query terms, we first identify all corresponding cell indices, C. For each term i not in the query, we extract the list of indices, D_i. The terms are then ranked based on the Jaccard similarity, $|C \cap D_i|/|C \cup D_i|$. For the cardiomyocyte example, we used the command `findSimilarGenes`.

Frequent pattern growth search and TF-IDF scores

To identify subsets of the original query that are guaranteed to return cell hits we use a modified version of the frequent pattern growth (FPGrowth) algorithm. The FPGrowth algorithm is popular in data mining where it is used to identify groups of items that co-occur in transactions. In the context of scfind one may think of cells as transactions and genes or peaks as items.

We first modify the original query so that all terms are related by OR logic to retrieve all relevant cells. For each cell c_j, an inverted cell index is constructed, containing a list of the subset of query terms found. The next step is to determine a minimum support threshold, s, i.e. the minimum number of cells required to consider a query. To speed up this process, we use a heuristic to ensure that the number of subqueries considered remains bounded. For each term in the query, g_i, scfind estimates the overlap with other terms in the query, and for n terms, s is defined as the median of the $n(n-1)/2$ values.

The FPGrowth algorithm takes a set of cells and builds a data structure called an FP-tree. In the FP-tree each node represents a term from the query, making it a more compact representation than a matrix as each node contains a counter to keep track of the number of cells where the term was present. To construct the tree, we use the inverted index to iterate across all cells. As the terms are inserted in order of decreasing frequency, the most widely used terms will be found near the root, and widely supported subqueries can be extracted by traversing the tree. To generate subqueries the tree is traversed depth first, but the search is interrupted when the value of a node is below s. Note that the heuristic is chosen to ensure that the number of subqueries is limited, even for very long queries (>50 genes). If a low number subqueries are returned initially, the threshold will be lowered to ensure that a sufficient number of subqueries is considered.

To help the user decide which subquery to choose, scfind ranks them using a score inspired by term frequency-inverse document frequency (TF-IDF). The score is calculated for each element of the expression matrix as $t_{ij} = \log(e_{ij} / \Sigma_i \, e_{ij}) - \log(N_i / N)$, where e_{ij} is the quantized expression values, N_i is the total number of cells with non-zero values for term i and N is the total number of cells in the reference. For a query involving n terms the score for each cell is given by $\Sigma_i^{\,n} \, t_{ij}/n^{1/2}$.

To illustrate the subquery optimization we consider the genes *Acta1*, *Actc1*, *Atp2a2*, *Myh6*, *Nppa*, *Ryr2*, *Tnnc1* and *Tpm*, which can distinguish cardiac muscle cells in the heart [30] (Figure 3a). To verify that the subquery optimization routine provides meaningful results we carried out a positive control experiment. For each cell type c_j in each of the five scRNA-seq atlases we randomly selected five genes from the top 20 marker genes as defined by the F1 score. We calculated the top TF-IDF score and whether or not c_j came up as the top ranked cell type for the best subquery. As a comparison, we also selected five genes from the top 100 marker genes. The results show that the former query results in significantly higher TF-IDF scores and is more likely to return the expected cell type as the top hit (Supplementary Figure 13, 14, 15).

Robustness analysis of TF-IDF search

We also investigated negative controls to test the robustness of scfind to changes in the expression matrix. We carried out an experiment whereby each element in the TM FACS data is multiplied by a random number which is uniformly distributed in the interval [.5, 2]. We then ran 1,000 queries containing gene lists from the GO database, and in 95% of cases the same cell type was ranked at the top for the best subquery as for the unperturbed data. Scfind is more sensitive to changes in the sparsity pattern and when a random selection of 10% of the elements in the expression matrix were either set to zero or given a non-zero value, the same cell type was chosen in 76% of cases. We also randomly permuted the cell labels for each tissue, and in this case only 51% of the queries returned the same cell type as the original query. Furthermore, when we permute all cell labels regardless of tissue, only 0.3% of queries return the same result.

An important challenge in combining datasets into a cell atlas is to account for systematic technical differences between experiments, i.e. batch effects. Batch driven variation can be corrected *post-hoc* through computational means, but how these corrections can best be carried out remains an open problem. Several methods have been published in recent years [53], but some of these are incompatible with scfind as they do not operate on the expression matrix, but rather on another representation of the data, e.g. a selection of principal components or the distance matrix [54]. The methods that do modify the expression matrix, e.g. Combat [55] and Seurat v3 [56], typically do not preserve zeros and may even introduce negative values. Consequently, the batch corrected expression matrix may violate one of the central assumptions that scfind makes about the data. An important constraint when applying batch correction methods prior to building an index is to ensure that zeros are preserved. To test the impact of batch correction methods directly, we implemented modified versions of Combat, Limma and Seurat v3 where only non-zero elements are adjusted and we used them to combine the two Tabula Muris datasets. The results show that batch correction with Combat, Limma and Seurat v3 has little effect on the F1 score of the top marker genes as 75% change by <0.1. (Supplementary Figure 18 and Supplementary Table 12). Taken together, we conclude that scaling and batch correction have limited impact on the results for scfind.

Processing of PubTator and PubMed Phrases

A custom Julia script was used to build indexes mapping variants and phrases to mouse and human gene names. The indexes were also inverted to allow gene names to be mapped to phrases for the word cloud visualization. Processed dictionaries are available as additional files for the Homo sapiens and Mus musculus genes.

Word2vec analysis

We used the Julia package `Word2Vec` for the calculation of cosine similarity between the tokens of the arbitrary query and the tokens of the similar phrases subsetted from the manually generated phrases to gene names dictionaries. To identify the best match phrase, the cosine distances between each pair of query-to-phrase tokens is calculated using the PubMed word2vec models. The distance of each phrase is averaged by the number of tokens. The mean distances are used to rank the best matched phrase.

Wordcloud visualization

We use the R package `wordcloud`. As an example, we consider a query for *Irx4*, *Myl2*, *Xdh*, *Dlk1*, *Hyal2*, *Tmem190*, *Cpne4*, *Cyp1a1* and *Irx5* which are markers discriminating between atrial and ventricular heart tissue [30]. Subquery optimization for the TM FACS datasets suggests *Hyal2* and *Myl2* as the best query, and it shows cardiomyocytes and heart epithelial cells as the most enriched cell types in the resulting word cloud (Figure 4b).

Cell type specificity of super enhancers

For each locus from dbSUPER we used its mm9 coordinates to search for enriched cell types using an OR combination of the peaks at each loci. A locus is reported if it has a p-value<10^{-10}, is found in at least 10 cells and >10% of the cells from that cell type.

Cell type specific genes and peaks

A gene is considered cell type specific if it is found in at least 25 cells and enriched at a p-value <10^{-5} with respect to the other cell types in the tissue. For each cell type specific gene we considered all peaks within 100 kb of the 5' most TSS. A peak is considered cell type specific if it is present in at least 10 cells and at least 10% of the cells from one cell type. Moreover, we require that the peak does not meet the above criteria for any other cell type in the same tissue.

We manually selected motifs from the JASPAR 2018 Core collection [57]. We then used the R packages `TFBStools` [58] and the `motifmatchr` package to identify motif instances in the cell type specific peaks.

Analysis of SNARE-seq data

Indexes for the RNA-seq and ATAC-seq datasets were constructed separately and then merged according to corresponding cells. For each matrix which columns represent the cell of each cell type and rows represent both gene names and peaks loci, is organised as a `SingleCellExperiment` object. An scfind object is built by merging indexes of all cell types using the function `buildCellTypeIndex` and `mergeDataset`.

A gene is considered cell type specific if it is found in at least 25 cells with a p-value <10^{-6} compared to other cell types in the tissue. For each cell type specific gene we considered all peaks within 1 Mb of the 5' most TSS. A peak is considered cell type specific if it is present in at least 10 cells and at least 10% of the cells from the cell type of interest. Moreover, we require that the peak does not meet the above criteria for any other cell type in the same tissue. Motif enrichment was calculated in the same way as before.

Datasets used

Mouse atlases

The data for the MCA was downloaded from https://figshare.com/s/865e694ad06d5857db4b, the Tabula Muris data from https://figshare.com/projects/Tabula_Muris_Transcriptomic_characterization_of_20_organs_and_tissues_from_Mus_musculus_at_single_cell_resolution/27733, the BCA data from http://linnarssonlab.org/data/, the

MOCA data from https://oncoscape.v3.sttrcancer.org/atlas.gs.washington.edu.mouse.rna/downloads and the sciATAC-seq data from http://atlas.gs.washington.edu/mouse-atac/data/, SNARE-seq data from https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE126074. For all datasets, custom R scripts were used to combine the expression and annotation files to generate `SingleCellExperiment` objects. The `SingleCellExperiment` objects were used to build the scfind indexes.

GO annotation

The GO annotation was downloaded from Ensembl Biomart website.

PubTator, Pubmed Phrases and word2vec

The PubTator resources covering PubMed abstracts until 2018 was downloaded from ftp://ftp.ncbi.nlm.nih.gov/pub/lu/PubTator/ and the PubMed phrases dataset including abstracts until 2017 was downloaded from ftp://ftp.ncbi.nlm.nih.gov/pub/lu/PubMedPhrase/PubMed_Phrases.tar.gz. The word2vec model trained on PubMed abstracts was downloaded from http://bio.nlplab.org/.

Super enhancers

We downloaded the .bed files corresponding to mm9 for Bone Marrow, Cerebellum, Heart, Kidney, Liver, Lung, Spleen and Thymus from the dbSUPER website. We also downloaded the .bed file containing hg19 coordinates for GM12878 cells.

Reporting Summary. Further information on research design is available in the Nature Research Reporting Summary linked to this article.

Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

Acknowledgements

JL, NP, VK and MH were supported by a core grant from the Wellcome Trust. JL was also supported by a grant "Search tools for scRNA-seq data" from the Chan Zuckerberg Initiative. We would like to thank members of the Hemberg group, Yong Liu, Tobias Bergmann and Anais Meziani for assisting with beta-testing of the software and Luz Garcia-Alonso, Vanessa Jane Hall, Alessandro Ori, Sarah Teichmann and Roser Vento for feedback on the manuscript. We would like to thank Jana Eliasova for assistance with figure 1a

Code availability

The code for scfind is available at github.com/hemberg-lab/scfind and the code for generating the figures in this manuscript is available at https://github.com/hemberg-lab/scfind-paper-figures A Code Ocean capsule of the tool is provided (https://doi.org/10.24433/CO.2453077.v1) [62].

Data availability

The index `LinnarssonAtlas.rds` for the the BCA data from http://linnarssonlab.org/data/ can be downloaded from https://scfind.cog.sanger.ac.uk/indexes/LinnarssonAtlas.rds. The index `mca.rds` for the the MCA data from https://figshare.com/s/865e694ad06d5857db4b can be downloaded from https://scfind.cog.sanger.ac.uk/indexes/mca.rds. The index `tm_10x.rds` for the the TM, 10X and TM, FACS data from https://figshare.com/projects/Tabula_Muris_Transcriptomic_characterization_of_20_organs_and_tissues_from_Mus_musculus_at_single_cell_resolution/27733 can be downloaded from https://scfind.cog.sanger.ac.uk/indexes/tm_10X.rds and https://scfind.cog.sanger.ac.uk/indexes/tm_facs.rds respectively. The index `atacseq.rds` for the the sciATACseq data from http://atlas.gs.washington.edu/mouse-atac/data/ can be downloaded from https://scfind.cog.sanger.ac.uk/indexes/atacseq.rds. The source data underlying Figs 1, 2, 3 and 5 are provided as a Source Data file.

References

1. Tabula Muris Consortium. et al. Single-cell transcriptomics of 20 mouse organs creates a Tabula Muris. Nature. 2018; 562:367–372. [PubMed: 30283141]
2. Han X, et al. Mapping the Mouse Cell Atlas by Microwell-Seq. Cell. 2018; 172:1091–1107.e17. [PubMed: 29474909]
3. Saunders A, et al. Molecular Diversity and Specializations among the Cells of the Adult Mouse Brain. Cell. 2018; 174:1015–1030.e16. [PubMed: 30096299]
4. Zeisel A, et al. Molecular architecture of the mouse nervous system. Cell. 2018; 174:999–1014.e22. [PubMed: 30096314]
5. Cusanovich DA, et al. A Single-Cell Atlas of In Vivo Mammalian Chromatin Accessibility. Cell. 2018; 174:1309–1324.e18. [PubMed: 30078704]
6. Cao J, et al. The single-cell transcriptional landscape of mammalian organogenesis. Nature. 2019; 566:496–502. [PubMed: 30787437]
7. Regev A, et al. The human cell atlas. elife. 2017; 6
8. Howick VM, et al. The Malaria Cell Atlas: Single parasite transcriptomes across the complete Plasmodium life cycle. Science. 2019; 365
9. MacArthur J, et al. The new NHGRI-EBI Catalog of published genome-wide association studies (GWAS Catalog). Nucleic Acids Res. 2017; 45:D896–D901. [PubMed: 27899670]
10. The Gene Ontology Consortium. Expansion of the Gene Ontology knowledgebase and resources. Nucleic Acids Res. 2017; 45:D331–D338. [PubMed: 27899567]
11. Sewell W. Medical Subject Headings in MEDLARS. Bull Assoc Med Libr. 1964; 52:164–170.
12. Landrum MJ, et al. ClinVar: public archive of interpretations of clinically relevant variants. Nucleic Acids Res. 2016; 44:D862–8. [PubMed: 26582918]
13. Cariaso M, Lennon G. SNPedia: a wiki supporting personal genome annotation, interpretation and analysis. Nucleic Acids Res. 2012; 40:D1308–12. [PubMed: 22140107]
14. Franzén O, Gan L-M, Björkegren JLM. PanglaoDB: a web server for exploration of mouse and human single-cell RNA sequencing data. Database (Oxford) 2019. 2019
15. Haeussler M, et al. The UCSC Genome Browser database: 2019 update. Nucleic Acids Res. 2019; 47:D853–D858. [PubMed: 30407534]
16. Athar A, et al. ArrayExpress update - from bulk to single-cell expression data. Nucleic Acids Res. 2019; 47:D711–D715. [PubMed: 30357387]
17. Srivastava D, Iyer A, Kumar V, Sengupta D. CellAtlasSearch: a scalable search engine for single cells. Nucleic Acids Res. 2018; 46:W141–W147. [PubMed: 29788498]
18. Sato K, Tsuyuzaki K, Shimizu K, Nikaido I. CellFishing.jl: an ultrafast and scalable cell search method for single-cell RNA sequencing. Genome Biol. 2019; 20:31. [PubMed: 30744683]
19. Vigna, S. Quasi-succinct indices. Proceedings of the sixth ACM international conference on Web search and data mining - WSDM ’13 83; ACM Press. 2013.
20. Schaum N, et al. Single-cell transcriptomic characterization of 20 organs and tissues from individual mice creates a Tabula Muris. BioRxiv. 2017; doi: 10.1101/237446

21. Golubovskaya V, Wu L. Different Subsets of T Cells, Memory, Effector Functions, and CAR-T Immunotherapy. Cancers (Basel). 2016; 8
22. Zhang X, et al. CellMarker: a manually curated resource of cell markers in human and mouse. Nucleic Acids Res. 2019; 47:D721–D728. [PubMed: 30289549]
23. Bausch-Fluck D, et al. A mass spectrometric-derived cell surface protein atlas. PLoS ONE. 2015; 10:e0121314. [PubMed: 25894527]
24. Lambert SA, et al. The human transcription factors. Cell. 2018; 172:650–665. [PubMed: 29425488]
25. Kanehisa M, Sato Y, Kawashima M, Furumichi M, Tanabe M. KEGG as a reference resource for gene and protein annotation. Nucleic Acids Res. 2016; 44:D457–62. [PubMed: 26476454]
26. Ju W, et al. Defining cell-type specificity at the transcriptional level in human disease. Genome Res. 2013; 23:1862–1873. [PubMed: 23950145]
27. Eisenberg E, Levanon EY. Human housekeeping genes, revisited. Trends Genet. 2013; 29:569–574. [PubMed: 23810203]
28. Han J, Pei J, Yin Y, Mao R. Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach. Data Min Knowl Discov. 2004; 8:53–87.
29. Sparck Jones K. A statistical interpretation of term specificity and its application in retrieval. Journal of Documentation. 1972; 28:11–21.
30. Piccini I, Rao J, Seebohm G, Greber B. Human pluripotent stem cell-derived cardiomyocytes: Genome-wide expression profiling of long-term in vitro maturation in comparison to human heart tissue. Genom Data. 2015; 4:69–72. [PubMed: 26484180]
31. Sayers EW, et al. Database resources of the National Center for Biotechnology Information. Nucleic Acids Res. 2019; 47:D23–D28. [PubMed: 30395293]
32. Wei C-H, Kao H-Y, Lu Z. PubTator: a web-based text mining tool for assisting biocuration. Nucleic Acids Res. 2013; 41:W518–22. [PubMed: 23703206]
33. Pers TH, et al. Biological interpretation of genome-wide association studies using predicted gene functions. Nat Commun. 2015; 6:5890. [PubMed: 25597830]
34. Manica M, Mathis R, Cadow J, Rodríguez Martínez M. Context-specific interaction networks from vector representation of words. Nat Mach Intell. 2019; 1:181–190.
35. Hastings J, et al. The ChEBI reference database and ontology for biologically relevant chemistry: enhancements for 2013. Nucleic Acids Res. 2013; 41:D456–63. [PubMed: 23180789]
36. Kim S, Yeganova L, Comeau DC, Wilbur WJ, Lu Z. PubMed Phrases, an open set of coherent phrases for searching biomedical literature. Sci Data. 2018; 5:180104. [PubMed: 29893755]
37. Mikolov T, Sutskever I, Chen K, Corrado G, Dean J. Distributed representations of words and phrases and their compositionality. NIPS. 2013:3111–3119.
38. Pyysalo S, Ginter F, Moen H, Salakoski T, Ananiadou S. Distributional Semantics Resources for Biomedical Text Processing. Languages in Biology and Medicine. 2013
39. Alfares AA, et al. Results of clinical genetic testing of 2,912 probands with hypertrophic cardiomyopathy: expanded panels offer limited additional sensitivity. Genet Med. 2015; 17:880–888. [PubMed: 25611685]
40. Flavigny J, et al. Identification of two novel mutations in the ventricular regulatory myosin light chain gene (MYL2) associated with familial and classical forms of hypertrophic cardiomyopathy. J Mol Med. 1998; 76:208–214. [PubMed: 9535554]
41. Hnisz D, et al. Super-enhancers in the control of cell identity and disease. Cell. 2013; 155:934–947. [PubMed: 24119843]
42. Parker SCJ, et al. Chromatin stretch enhancer states drive cell-specific gene regulation and harbor human disease risk variants. Proc Natl Acad Sci USA. 2013; 110:17921–17926. [PubMed: 24127591]
43. Khan A, Zhang X. dbSUPER: a database of super-enhancers in mouse and human genome. Nucleic Acids Res. 2016; 44:D164–71. [PubMed: 26438538]
44. Joo MS, Koo JH, Kim TH, Kim YS, Kim SG. LRH1-driven transcription factor circuitry for hepatocyte identity: Super-enhancer cistromic analysis. EBioMedicine. 2019; 40:488–503. [PubMed: 30638865]

45. Thomas GD, et al. Deleting an Nr4a1 Super-Enhancer Subdomain Ablates Ly6Clow Monocytes while Preserving Macrophage Gene Function. Immunity. 2016; 45:975–987. [PubMed: 27814941]

46. Kleftogiannis D, Kalnis P, Bajic VB. Progress and challenges in bioinformatics approaches for enhancer identification. Brief Bioinformatics. 2016; 17:967–979. [PubMed: 26634919]

47. Chen S, Lake BB, Zhang K. High-throughput sequencing of the transcriptome and chromatin accessibility in the same cell. Nat Biotechnol. 2019; 37:1452–1457. [PubMed: 31611697]

48. Bradshaw AD, Sage EH. SPARC, a matricellular protein that functions in cellular differentiation and tissue response to injury. J Clin Invest. 2001; 107:1049–1054. [PubMed: 11342565]

49. Callaham ML, Wears RL, Weber EJ, Barton C, Young G. Positive-outcome bias and other limitations in the outcome of research abstracts submitted to a scientific meeting. JAMA. 1998; 280:254–257. [PubMed: 9676673]

50. Kiselev VY, Andrews TS, Hemberg M. Challenges in unsupervised clustering of single-cell RNA-seq data. Nat Rev Genet. 2019; 20:273–282. [PubMed: 30617341]

51. Benjamini Y, Hochberg Y. Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing. Journal of the Royal Statistical Society Series B (Methodological). 1995; 57:289–300.

52. McInnes L, Healy J, Saul N, Großberger L. UMAP: uniform manifold approximation and projection. JOSS. 2018; 3:861.

53. Tian L, et al. Benchmarking single cell RNA-sequencing analysis pipelines using mixture control experiments. Nat Methods. 2019; 16:479–487. [PubMed: 31133762]

54. Chazarra-Gil R, Hemberg M, Kiselev VY, van Dongen S. Flexible comparison of batch correction methods for single-cell RNA-seq using BatchBench. BioRxiv. 2020; doi: 10.1101/2020.05.22.111211

55. Johnson WE, Li C, Rabinovic A. Adjusting batch effects in microarray expression data using empirical Bayes methods. Biostatistics. 2007; 8:118–127. [PubMed: 16632515]

56. Stuart T, et al. Comprehensive Integration of Single-Cell Data. Cell. 2019; 177:1888–1902.e21. [PubMed: 31178118]

57. Khan A, et al. JASPAR 2018: update of the open-access database of transcription factor binding profiles and its web framework. Nucleic Acids Res. 2018; 46:D260–D266. [PubMed: 29140473]

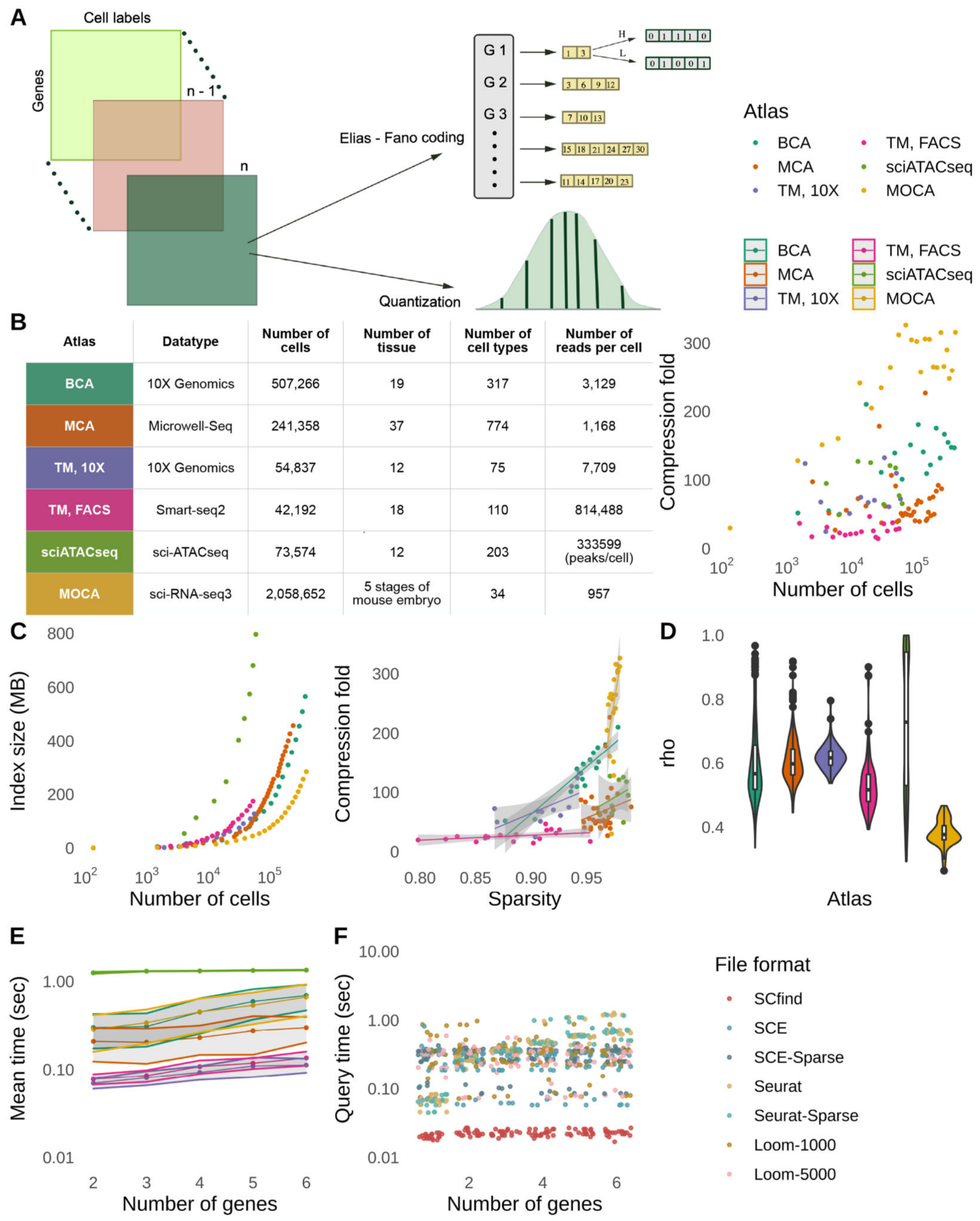
58. Tan G, Lenhard B. TFBSTools: an R/bioconductor package for transcription factor binding site analysis. Bioinformatics. 2016; 32:1555–1556. [PubMed: 26794315]

59. Matthay MA, Thompson BT. Dexamethasone in hospitalised patients with COVID-19: addressing uncertainties. Lancet Respir Med. 2020; doi: 10.1016/S2213-2600(20)30503-8

60. Travaglini KJ, et al. A molecular cell atlas of the human lung from single-cell RNA sequencing. Nature. 2020; doi: 10.1038/s41586-020-2922-4

61. Sungnak W, et al. SARS-CoV-2 entry factors are highly expressed in nasal epithelial cells together with innate immune genes. Nat Med. 2020; 26:681–687. [PubMed: 32327758]

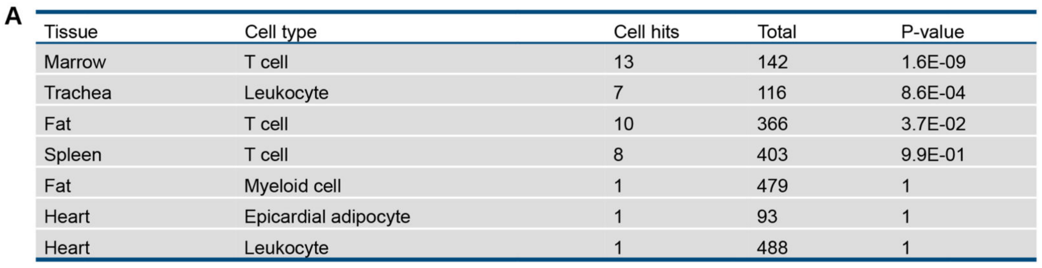
62. Lee, JTH, Patikas, N, Kiselev, VY, Hemberg, M. Fast searches of large collections of single cell data using scfind. Code Ocean; 2021.



Atlas	Datatype	Number of cells	Number of tissue	Number of cell types	Number of reads per cell
BCA	10X Genomics	507,266	19	317	3,129
MCA	Microwell-Seq	241,358	37	774	1,168
TM, 10X	10X Genomics	54,837	12	75	7,709
TM, FACS	Smart-seq2	42,192	18	110	814,488
sciATACseq	sci-ATACseq	73,574	12	203	333599 (peaks/cell)
MOCA	sci-RNA-seq3	2,058,652	5 stages of mouse embryo	34	957

Figure 1. Compression and search times.

(a) Schematic illustration of scfind’s compression strategy. Each square represents an expression matrix from an individual experiment. Each matrix is compressed and the resulting vectors are then concatenated. (b) Summary statistics for the six atlases considered in this manuscript along with compression ratios where each point represents one tissue. (c) Cumulative size of the resulting index when merging the different tissues (left) and fraction of zeros compared to compression ratios (right). Errors are presented as 95% confidence interval around the fitted lines. For legend, see (b). (d) Accuracy of the reconstructed expression values when using two bits for the quantization. The violin plots show Spearman correlations for each gene and cell type (BCA n=317, MCA n=774, TM, 10X n=75, TM, FACS n=110, sciATACseq n=203 and MOCA n=34 cell types). Violin plots show the density (width), median (center line), interquartile range (hinges) and 1.5 times the interquartile range (adjacent lines); outlier data beyond this range are plotted as individual points. (e) Mean search times for AND queries involving up to six genes (n=100 independent experiments). Data are presented as mean values +/− SEM" as appropriate. (f) Comparison of query times with three other file formats.



Tissue	Cell type	Cell hits	Total	P-value
Marrow	T cell	13	142	1.6E-09
Trachea	Leukocyte	7	116	8.6E-04
Fat	T cell	10	366	3.7E-02
Spleen	T cell	8	403	9.9E-01
Fat	Myeloid cell	1	479	1
Heart	Epicardial adipocyte	1	93	1
Heart	Leukocyte	1	488	1

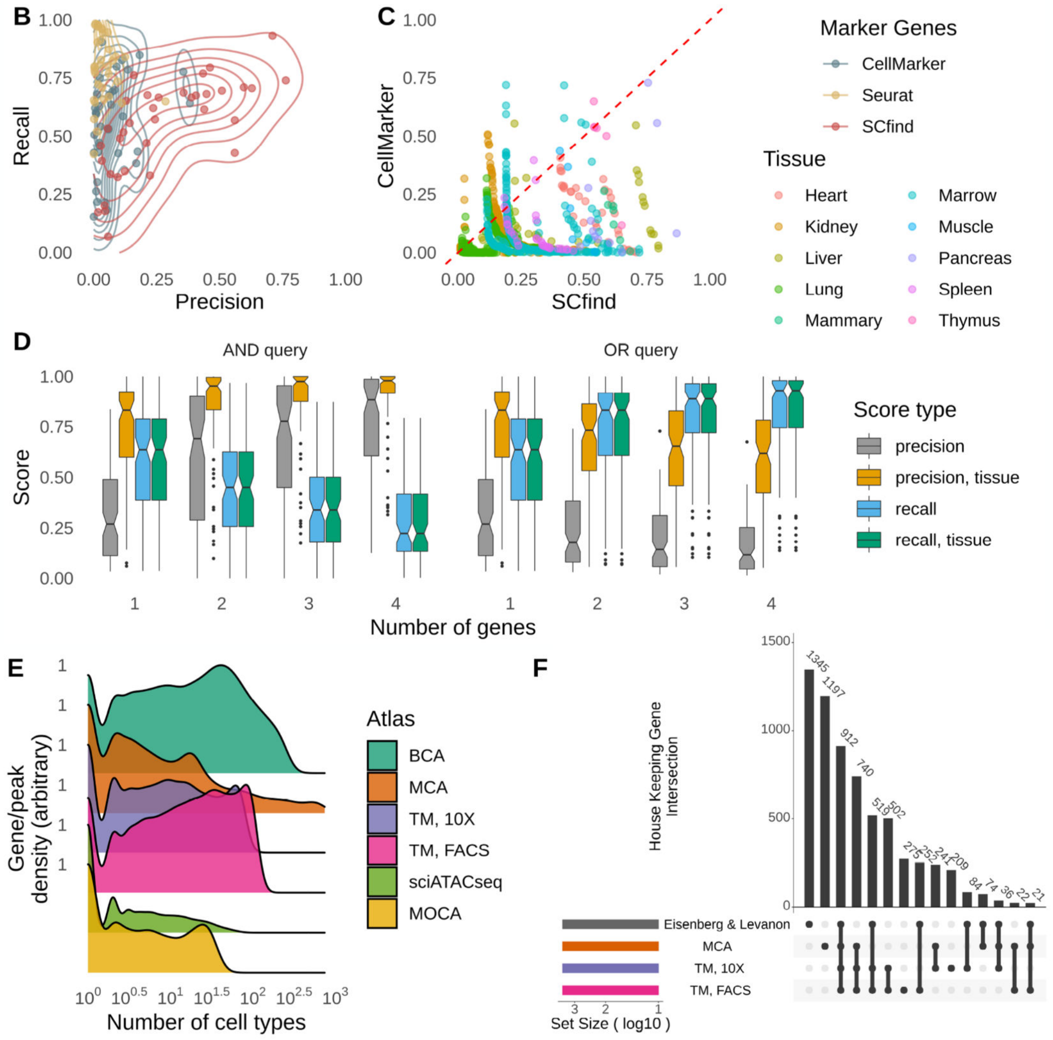
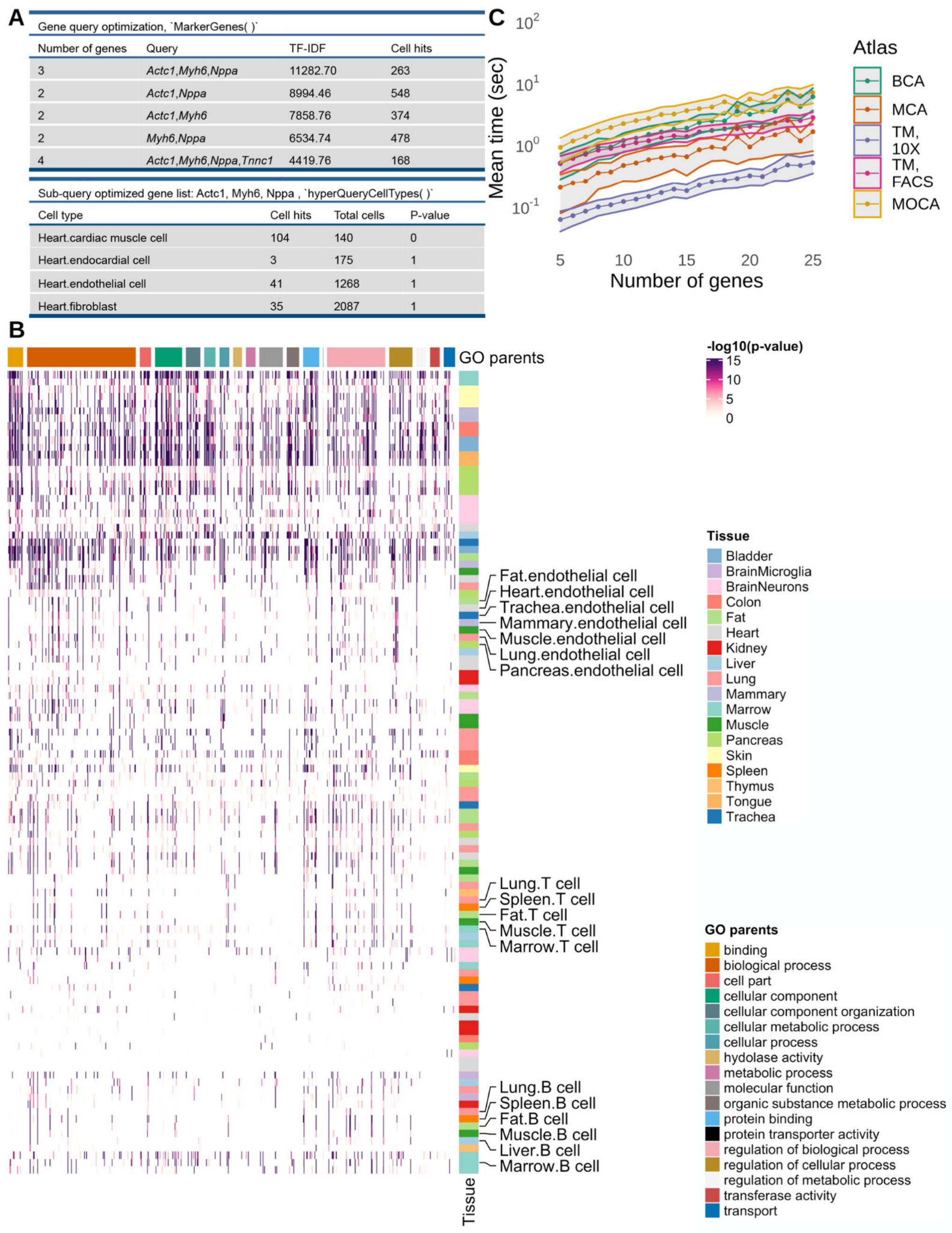


Figure 2. Basic search, identification of cell type specific and housekeeping genes

(a) Sample query showing some of the results for a search of the TM FACS dataset for *Il2ra*, *Ptprc*, *Il7r* and *Ctla4*. One-tailed hypergeometric test with Holm adjustment for multiple comparison was used. (b) Precision vs Recall for the marker genes for 37 cell types from the TM FACS dataset as determined by scfind or the CellMarker database. The overlaid density curves help the reader identify where the majority of points are found, and they indicate that scfind in general has higher scores than CellMarker. (c) Distribution of F1 scores (harmonic mean of the precision and recall scores) of marker genes identified in 37 cell types from the

TM FACS dataset as determined by scfind or CellMarker (d) Precision and recall values for combinations of 1-4 marker genes using AND or OR logic were calculated for all cell types in the TM FACS data (n=110 cell types). The boxplots show the spread of scores when compared either to all cell types or just the ones from the same tissue. Box plots show the median (center line), interquartile range (hinges) and 1.5 times the interquartile range (whiskers); outlier data beyond this range are plotted as individual points. One-tailed hypergeometric test with Holm adjustment for multiple comparison was used. (e) Histogram showing the distribution of cell type specificities for genes and peaks from the six atlases. (f) Upset plot showing the overlap of 3,144 housekeeping genes from the literature and three mouse cell atlases.



Gene query optimization, `MarkerGenes( )`

Number of genes	Query	TF-IDF	Cell hits
3	*Actc1,Myh6,Nppa*	11282.70	263
2	*Actc1,Nppa*	8994.46	548
2	*Actc1,Myh6*	7858.76	374
2	*Myh6,Nppa*	6534.74	478
4	*Actc1,Myh6,Nppa,Tnnc1*	4419.76	168

Sub-query optimized gene list: Actc1, Myh6, Nppa , `hyperQueryCellTypes( )`

Cell type	Cell hits	Total cells	P-value
Heart.cardiac muscle cell	104	140	0
Heart.endocardial cell	3	175	1
Heart.endothelial cell	41	1268	1
Heart.fibroblast	35	2087	1

Figure 3. Subquery optimization

(a) Example for optimization of the query *Acta1*, *Actc1*, *Atp2a2*, *Myh6*, *Nppa*, *Ryr2*, *Tnnc1* and *Tpm* for the TM FACS dataset. One-tailed hypergeometric test with Holm adjustment for multiple comparison was used. (b) Heatmap showing the p-values for the best subquery for each GO term containing between 5 and 25 genes for the TM FACS data. Rows represent cell types and columns represent GO terms. For the result of each gene set, one-tailed hypergeometric test with Holm adjustment for multiple comparison was used. (c) Mean run times for subquery optimization for lists from the GO annotation containing between 5 and 25 genes. data are presented as mean values +/− SEM.

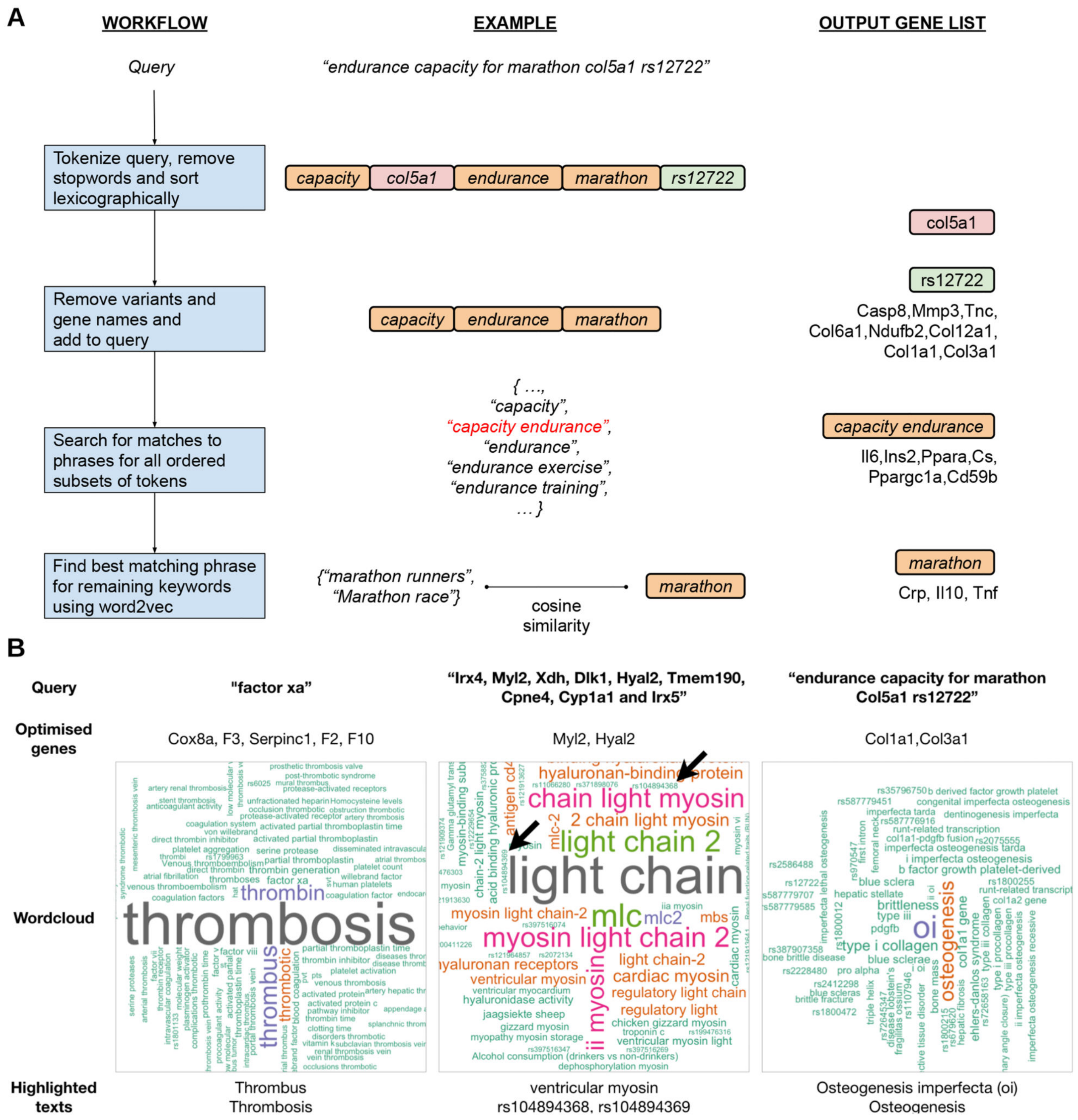

A
WORKFLOW
EXAMPLE
OUTPUT GENE LIST
Query
"endurance capacity for marathon col5a1 rs12722"
Tokenize query, remove stopwords and sort lexicographically
capacity
col5a1
endurance
marathon
rs12722
col5a1
rs12722
Casp8,Mmp3,Tnc, Col6a1,Ndufb2,Col12a1, Col1a1,Col3a1
Remove variants and gene names and add to query
capacity
endurance
marathon
{ ..., "capacity", "capacity endurance", "endurance", "endurance exercise", "endurance training", ... }
Search for matches to phrases for all ordered subsets of tokens
capacity endurance
Il6,Ins2,Ppara,Cs, Ppargc1a,Cd59b
Find best matching phrase for remaining keywords using word2vec
{"marathon runners", "Marathon race"}
cosine similarity
marathon
Crp, Il10, Tnf
B
Query
"factor xa"
"Irx4, Myl2, Xdh, Dlk1, Hyal2, Tmem190, Cpne4, Cyp1a1 and Irx5"
"endurance capacity for marathon Col5a1 rs12722"
Optimised genes
Cox8a, F3, Serpinc1, F2, F10
Myl2, Hyal2
Col1a1,Col3a1
Wordcloud
thrombosis
thrombin
thrombus
thrombotic
light chain
myosin light chain 2
chain light myosin
light chain 2
mlc
myosin
hyaluronan-binding protein
oi
osteogenesis
type i collagen
brittleness
Highlighted texts
Thrombus
Thrombosis
ventricular myosin
rs104894368, rs104894369
Osteogenesis imperfecta (oi)
Osteogenesis

Figure 4. Free text searches and visualization of results

(a) Flow chart showing how a query is processed by scfind. (b) Examples of word cloud associations following query optimization. Arrows in the middle panel show variant names (see main text).

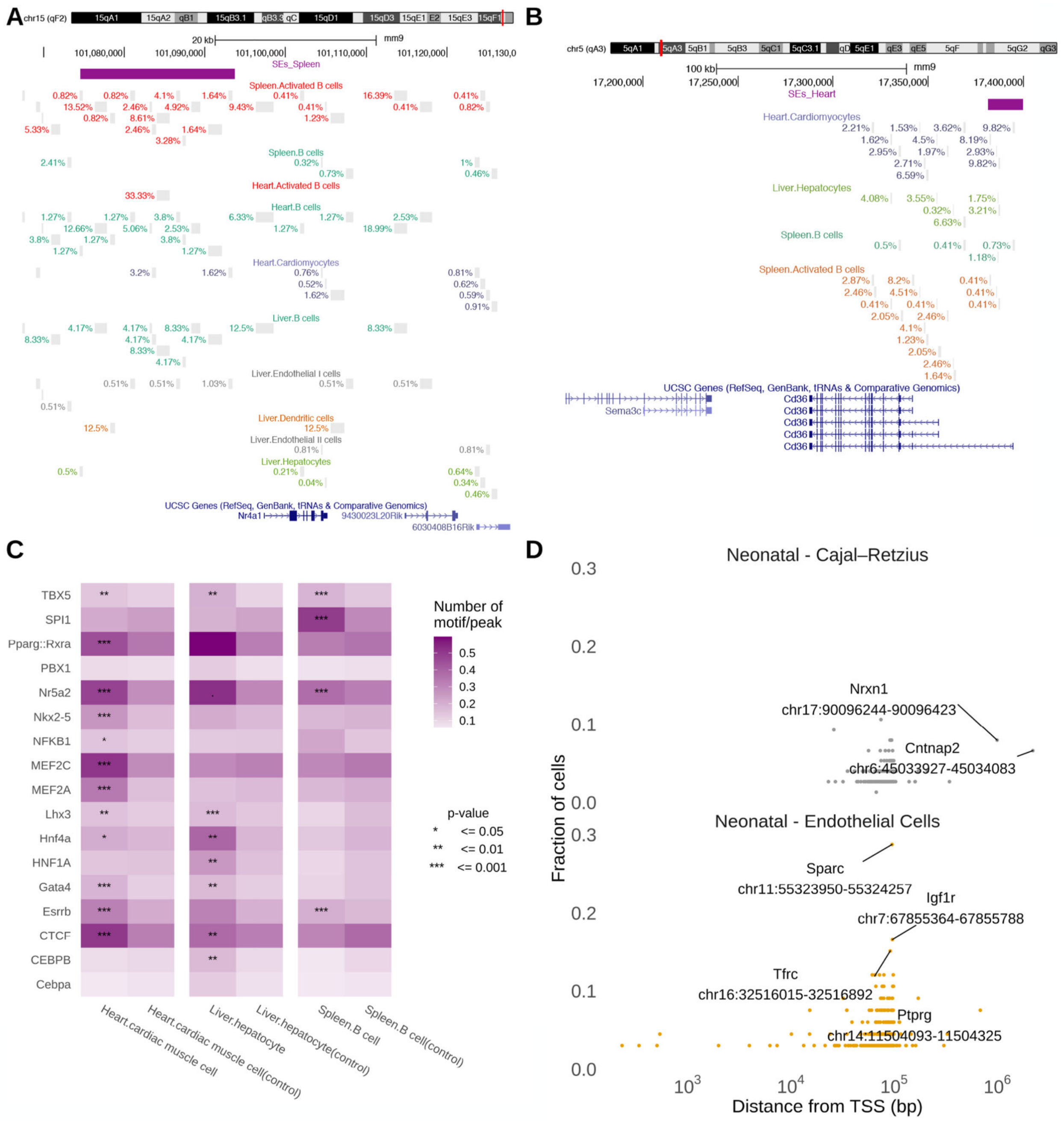


Figure 5. Determining cell type specificity of distal enhancers using RNA-seq and ATAC-seq data

(a) UCSC genome browser screenshot showing the *Nr4a1* locus which has a super enhancer in pro-B cells and monocytes. Grey marks represent open chromatin and the numbers show the fraction of cells containing each peak. (b) UCSC genome browser screenshot of the *Cd36* locus which harbors a cardiomyocyte specific enhancer [43] showing open chromatin in several cell types (c) Motif enrichment in putative distal enhancers that are specific to cardiac muscle cells, hepatocyte and B cells. Well-known regulators, e.g. *Mef2* for heart muscles, *Hnf4* for hepatocytes and *Spi1* for B cells have high enrichments. Two-tailed

Fisher's exact test with Benjamini and Hochberg adjustment for multiple comparison was used. (***p <0.001; **p <0.01; *p <0.05). (d) Cell type specific open chromatin-gene pairs for Cajal-Retzius neurons and endothelial cells from neonatal cerebral cortex in mouse. Instances where the open chromatin is >500 kb from the TSS are highlighted.

Table 1
Queries involving keywords mapped to gene lists

Summary statistics of the number of keywords ang genes for different categories.

Dictionary to genes	**Number of entries**	**Number of genes**
Variant ID	Human: 12911, Mouse: 11617	Human: 77648, Mouse: 68240
GWAS traits terms	Human: 958, Mouse: 958	Human: 14922, Mouse: 14922
MeSH/OMIM ID	Human: 171162, Mouse: 171162	Human: 83297, Mouse: 81418
MeSH/chem ID	Human: 69816, Mouse: 69816	Human: 152965, Mouse: 145284
PubMed Phrase	Human: 70858, Mouse: 67229	Human: 789896, Mouse: 750844