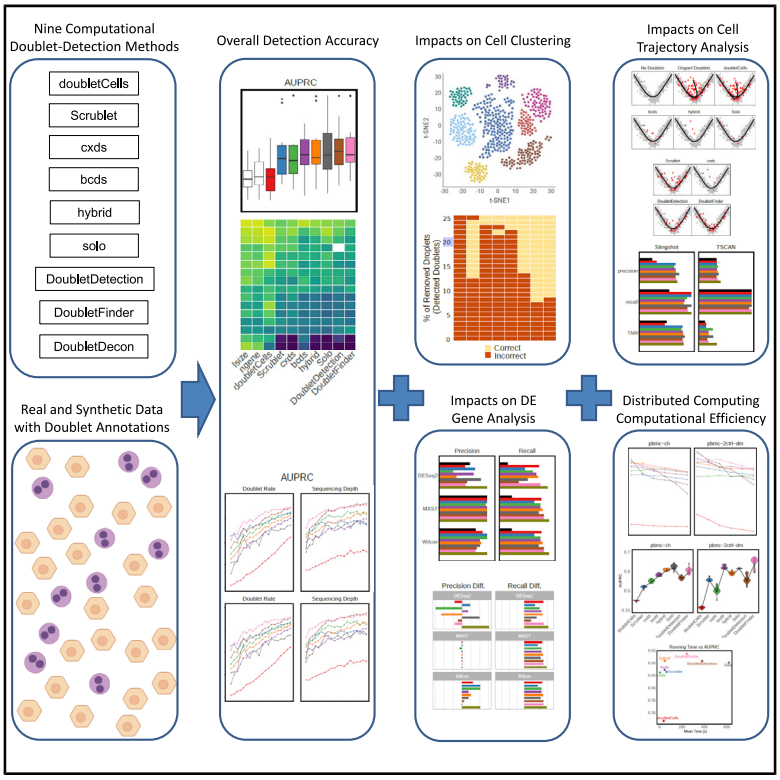
Article

Cell Systems

Benchmarking Computational Doublet-Detection Methods for Single-Cell RNA Sequencing Data

Graphical Abstract



Authors

Nan Miles Xi, Jingyi Jessica Li

Correspondence

jli@stat.ucla.edu

In Brief

We conduct a systematic benchmark study of nine cutting-edge computational doublet-detection methods. We evaluate the methods' detection accuracy, impacts on downstream analyses, and computational efficiency, using a comprehensive set of real and synthetic data. Although no method dominates in all aspects, the DoubletFinder and cxds methods have the best detection accuracy and computational efficiency, respectively.

Highlights

- Benchmark of nine computational doublet-detection methods for scRNA-seq data
- Evaluation based on the most comprehensive set of real data with labeled doublets
- Diverse performance and distinct advantages of different doublet-detection methods
- DoubletFinder excels in detection accuracy; cxds leads in computational efficiency

Xi & Li, 2021, Cell Systems *12*, 1–19
February 17, 2021
https://doi.org/10.1016/j.cels.2020.11.008

CellPress

Please cite this article in press as: Xi and Li, Benchmarking Computational Doublet-Detection Methods for Single-Cell RNA Sequencing Data, Cell Systems (2020), https://doi.org/10.1016/j.cels.2020.11.008

Cell Systems

CellPress

Article

Benchmarking Computational Doublet-Detection Methods for Single-Cell RNA Sequencing Data

Nan Miles Xi[1] and Jingyi Jessica Li[1,2,3,4,*]
[1]Department of Statistics, University of California, Los Angeles, CA 90095-1554, USA
[2]Department of Human Genetics, University of California, Los Angeles, CA 90095-7088, USA
[3]Department of Computational Medicine, University of California, Los Angeles, CA 90095-1766, USA
[4]Lead Contact
*Correspondence: jli@stat.ucla.edu
https://doi.org/10.1016/j.cels.2020.11.008

SUMMARY

In single-cell RNA sequencing (scRNA-seq), doublets form when two cells are encapsulated into one reaction volume. The existence of doublets, which appear to be—but are not—real cells, is a key confounder in scRNA-seq data analysis. Computational methods have been developed to detect doublets in scRNA-seq data; however, the scRNA-seq field lacks a comprehensive benchmarking of these methods, making it difficult for researchers to choose an appropriate method for specific analyses. We conducted a systematic benchmark study of nine cutting-edge computational doublet-detection methods. Our study included 16 real datasets, which contained experimentally annotated doublets, and 112 realistic synthetic datasets. We compared doublet-detection methods regarding detection accuracy under various experimental settings, impacts on downstream analyses, and computational efficiencies. Our results show that existing methods exhibited diverse performance and distinct advantages in different aspects. Overall, the DoubletFinder method has the best detection accuracy, and the cxds method has the highest computational efficiency. A record of this paper's transparent peer review process is included in the Supplemental Information.

INTRODUCTION

Single-cell RNA sequencing (scRNA-seq) is a family of emerging sequencing technologies that have revolutionized biomedical sciences by revealing genome-wide gene expression levels within each of thousands to millions of individual cells (Saliba et al., 2014; Kolodziejczyk et al., 2015; Vallejos et al., 2015). Since its invention, scRNA-seq has become an essential experimental approach to investigate cell-to-cell heterogeneity, distinguish cell types and subtypes, identify cell-type-specific genes, and reveal cellular dynamic processes (Liu and Trapnell, 2016; Luecken and Theis, 2019). Among various scRNA-seq experimental protocols, two major types—droplet microfluidics and well-based protocols—have gained popularity because of their high throughput, low cost per cell, and ability to detect unique mRNA transcripts via unique molecular identifiers (UMIs) (Hwang et al., 2018; Chen et al., 2019). Both types of protocols distribute a cell suspension into reaction volumes (droplets or wells) to hopefully encapsulate one cell per volume (i.e., a singlet), and then mRNA molecules in each volume are labeled by a unique droplet barcode. For simplicity, we will refer to a reaction volume as a droplet in the following text.

During the distribution step of an scRNA-seq experiment, however, one droplet may encapsulate more than one cell, creating a so-called doublet that is disguised as a single cell (Luecken and Theis, 2019). The doublet rate (i.e., the proportion of doublets) in a scRNA-seq experiment depends on the throughput and protocol, and doublets may constitute as many as 40% of droplets (Bernstein et al., 2020). There are two major classes of doublets: homotypic doublets, which are formed by transcriptionally similar cells; and heterotypic doublets, which are formed by cells of distinct types, lineages, or states (McGinnis et al., 2019a, 2019b; Wolock et al., 2019). Compared with homotypic doublets, heterotypic doublets are generally easier to detect due to their distinct gene expression profiles unlike those of singlets (McGinnis et al., 2019a, 2019b).

The existence of doublets, especially heterotypic doublets, in scRNA-seq datasets may confound downstream analysis; for example, doublets can form spurious cell clusters, interfere with differentially expressed (DE) gene analysis, and obscure the inference of cell developmental trajectories (Luecken and Theis, 2019; Wolock et al., 2019). Several experimental techniques have been developed to detect doublets in scRNA-seq using droplet barcodes. Example techniques include cell hashing (doublets are the droplets whose barcodes are associated with more than one oligo-tagged antibody) (Stoeckius et al., 2018), species mixture (doublets are the droplets whose barcodes are associated with more than one species) (Wolock et al., 2019), demuxlet (doublets are the droplets whose barcodes are associated with mutually exclusive sets of SNPs) (Kang et al., 2018), and MULTI-seq (doublets are the droplets whose barcodes are associated with more than one lipid-tagged index)

Please cite this article in press as: Xi and Li, Benchmarking Computational Doublet-Detection Methods for Single-Cell RNA Sequencing Data, Cell Systems (2020), https://doi.org/10.1016/j.cels.2020.11.008

(McGinnis et al., 2019a, 2019b). However, these techniques require special experimental preparation, extra costs, and time, and they are not guaranteed to remove all doublets, e.g., demuxlet cannot detect the doublets formed by cells from the same individual. Moreover, they cannot remove doublets from existing scRNA-seq data.

Realizing the limitations of experimental strategies, researchers have attempted to tackle this doublet challenge from an alternative perspective: developing computational methods to detect doublets from already-generated scRNA-seq data (Luecken and Theis, 2019). So far, nine doublet-detection methods have been developed (with software packages and full-text manuscripts) based on distinct algorithm designs (Lun et al., 2016; Gayoso and Shor, 2018; Bais and Kostka, 2020; DePasquale et al., 2019; McGinnis et al., 2019a, 2019b; Wolock et al., 2019; Bernstein et al., 2020) (Table 1). Here is a brief summary of these methods except hybrid, which is a combination of two methods: bcds and cxds. Seven out of the eight methods (with cxds as the only exception) first generate artificial doublets by combining gene expression profiles of two randomly selected droplets. Except DoubletDecon, the other six methods subsequently define a doublet score for each original droplet as the level of similarity the droplet has to those artificial doublets; next, with a pre-defined or user-specified threshold, they detect doublets as the original droplets whose doublet scores exceed the threshold. The key difference of the seven artificial-doublet-based methods is how they distinguish original droplets from artificial doublets: Five of them use classification algorithms (Scrublet, doubletCells, and DoubletFinder use *k*-nearest neighbors [kNN]; bcds uses gradient boosting; Solo uses neural networks), DoubletDetection uses the hypergeometric test, and DoubletDecon decides whether an original droplet resembles an artificial doublets based on its deconvolution algorithm (unlike other methods, DoubletDecon identifies doublets without providing doublet scores). As the only method that does not generate artificial doublets, cxds defines doublet scores based on gene co-expression, and similar to the other six doublet-score-based methods, it subsequently thresholds doublet scores to identify doublets.

While each computational doublet-detection method was shown to perform well under certain metrics by its developers, currently there is no systematic, third-party benchmarking of these methods' doublet-detection accuracy, effects on downstream analysis, or computation efficiency. As a result, users lack guidelines to choose an appropriate doublet-detection method for their analysis task. Hence, a detailed assessment of existing doublet-detection methods is in great demand. In addition to assisting users, it will provide useful guidance for computationalists to improve existing methods or develop new methods.

Here, we conducted a comprehensive benchmark study of computational methods for doublet detection. We evaluated nine cutting-edge methods—doubletCells (Lun et al., 2016), Scrublet (Wolock et al., 2019), cxds (Bais and Kostka, 2020), bcds (Bais and Kostka, 2020), hybrid (Bais and Kostka, 2020), Solo (Bernstein et al., 2020), DoubletDetection (Gayoso and Shor, 2018), DoubletFinder (McGinnis et al., 2019a, 2019b), and DoubletDecon (DePasquale et al., 2019)—in three aspects. First, we compared their overall doublet-detection accuracy using two criteria: the area under the precision-recall curve (AUPRC) and the area under the receiver operating characteristic curve (AUROC), on a collection of 16 real scRNA-seq datasets containing experimentally annotated doublets. To further evaluate the performance of these methods under various experimental settings, we simulated 80 realistic scRNA-seq datasets and evaluated the AUPRC and AUROC of each method under a wide range of doublet rates, sequencing depths, numbers of cell types, and cell-type heterogeneity levels. Second, considering that the ultimate goal of doublet detection is to improve the accuracy of downstream scRNA-seq data analyses, we compared these nine doublet-detection methods in terms of their impacts on four downstream analyses: DE gene analysis, highly variable gene identification, cell clustering, and cell trajectory inference. We simulated seven doublet-containing scRNA-seq datasets with pre-defined cell types, DE genes, and cell trajectories. Then, we evaluated the accuracy of the four downstream analyses by their state-of-the-art computational methods before and after doublets were removed by each doublet-detection method. The rationale is that a good doublet-detection method should improve the accuracy of downstream analyses after its use. Third, we compared the computational efficiency of doublet-detection methods in aspects, including distributed computing, speed, scalability, stability, and usability.

In summary, the nine doublet-detection methods exhibited a large variation in their performance under each evaluation criterion. First, the benchmarking result of detection accuracy shows that there is still room for improvement: The best method DoubletFinder achieved a mean AUPRC value of 0.537 on 16 real datasets. On simulated datasets, most methods performed better on datasets with higher doublet rates, larger sequencing depths, more cell types, or greater heterogeneity between cell types. Second, we observed that doublet removal by most methods indeed improved the identification of DE genes and highly variable genes, the elimination of spurious cell clusters, and the inference of cell trajectories; yet, the degree of improvement varied from method to method. Third, most methods except cxds had deteriorated performance under distributed computing, because global data information was lost in each distributed data batch. The cxds method also performed the best in terms of speed and scalability. Overall, DoubletFinder is highlighted as the best computational doublet-detection method for its highest detection accuracy and largest improvement on downstream analyses, while cxds is found as the most computationally efficient method in our benchmark.

RESULTS

Doublet-Detection Accuracy on Real scRNA-Seq Datasets

To evaluate the overall doublet-detection accuracy of the nine methods, we collected 16 public scRNA-seq datasets with doublets annotated by experimental techniques (Kang et al., 2018; Stoeckius et al., 2018; McGinnis et al., 2019a, 2019b; Wolock et al., 2019) (STAR Methods). Our collection covers a variety of cell types, droplet and gene numbers, doublet rates, and sequencing depths, thus representing varying levels of difficulty in detecting doublets from scRNA-seq data (Table S1). To the

Please cite this article in press as: Xi and Li, Benchmarking Computational Doublet-Detection Methods for Single-Cell RNA Sequencing Data, Cell Systems (2020), https://doi.org/10.1016/j.cels.2020.11.008

Table 1. An Overview of Nine Computational Doublet-Detection Methods Evaluated in this Study

Method	Programming Language	Version	Artificial Doublets	Dimension Reduction	Guidance on Threshold Selection	Algorithm Description
Scrublet (Wolock et al., 2019)	Python	0.2.1	yes	principal component analysis (PCA)	yes	It generates artificial doublets by adding two randomly selected droplets' gene expression profiles. The doublet score of each droplet is defined as the proportion of artificial doublets among its *k*-nearest neighboring droplets in the principal component (PC) space, whose number of dimensions is specified by users.
doubletCells (Lun et al., 2016)	R	1.16.0	yes	PCA	no	It generates artificial doublets by adding two randomly selected droplets' gene expression profiles. For each droplet, it calculates the proportion of artificial doublets, p_A, in a neighborhood in the PC space, whose number of dimensions is specified by users. The radius of the neighborhood is set to be the median distance from the droplet to its 50th nearest neighbor. The doublet score of each droplet is defined as $p_A/(1-p_A)^2$.
cxds (Bais and Kostka, 2020)	R	1.2.0	no	highly variable genes	no	It calculates a p value for each pair of genes under the null hypothesis that the number of droplets where exactly one of the two genes is expressed follows a binomial distribution. The doublet score of each droplet is defined as the sum of negative (natural) log p values of co-expressed gene pairs, where two genes in each pair both have non-zero expression levels in this droplet.
bcds (Bais and Kostka, 2020)	R	1.2.0	yes	highly variable genes	no	It generates artificial doublets by adding two randomly selected droplets' gene expression profiles and pools these artificial doublets with the original droplets. Then it trains a gradient boosting classifier to classify the pooled droplets into original droplets and artificial doublets. The doublet score of each droplet is defined as the predicted probability of being an artificial doublet.
Hybrid (Bais and Kostka, 2020)	R	1.2.0	–	–	no	It normalizes the doublet scores of cxds and bcds to values between 0 and 1. The doublet score of each droplet is defined as the sum of the two normalized doublet scores.

(*Continued on next page*)

Table 1. ***Continued***

Method	Programming Language	Version	Artificial Doublets	Dimension Reduction	Guidance on Threshold Selection	Algorithm Description
DoubletDetection (Gayoso and Shor, 2018)	Python	2.5.2	yes	PCA	no	It generates artificial doublets by adding two randomly selected droplets' gene expression profiles and pools these artificial doublets with the original droplets. Then it conducts Louvain clustering on the pooled droplets. For each droplet cluster, it performs a hypergeometric test and computes p value = 1 - hypergeom.cdf(N, K, n, k), where N is the number of droplets, K is the number of artificial doublets, n is the number of droplets in this cluster, and k is the number of artificial doublets in this cluster. All droplets in this cluster will have the same p value. It repeats the above steps (starting from artificial doublet generation) for a user-specified number of runs. The doublet score of each droplet is defined as its average p value across all runs.
DoubletFinder (McGinnis et al., 2019a, 2019b)	R	2.0.3	yes	PCA	yes	It generates artificial doublets by averaging two randomly selected droplets' gene expression profiles. The doublet score of each droplet is defined as the proportion of artificial doublets among its *k*-nearest neighboring droplets in the PC space, whose number of dimensions is specified by users. The number of neighbors, k, is selected by maximizing the mean-variance normalized bimodality coefficient (Pfister et al., 2013) of the distribution of doublet scores.
Solo (Bernstein et al., 2020)	Linux command	0.5	yes	variational autoencoder	0.5 by default	For a randomly selected droplet pair, it estimates a multinomial distribution whose number of trials equals the sum of total counts in these two droplets and whose event probabilities equal the gene proportions calculated from the mean gene expression profile of these two droplets. Then it generates artificial doublets by randomly sampling a gene expression profile from this multinomial distribution. That is, the number of artificial doublets equals the number of randomly selected droplet pairs. These artificial doublets are pooled with the original droplets. Then it trains a neural network to classify the pooled droplets into original droplets and artificial doublets. The doublet score of each droplet is defined as the predicted probability of being an artificial doublet.

(*Continued on next page*)

Please cite this article in press as: Xi and Li, Benchmarking Computational Doublet-Detection Methods for Single-Cell RNA Sequencing Data, Cell Systems (2020), https://doi.org/10.1016/j.cels.2020.11.008

Table 1. *Continued*

Method	Programming Language	Version	Artificial Doublets	Dimension Reduction	Guidance on Threshold Selection	Algorithm Description
DoubletDecon (DePasquale et al., 2019)	R	1.1.5	yes	deconvolution	doublet detection without doublet scores	It generates artificial doublets by taking a weighted average of two randomly selected droplets' gene expression profiles (the default weights are 0.7 and 0.3). Putative doublets are defined as those droplets whose gene expression profiles after deconvolution (Gong and Szustakowski, 2013) are concentrated on the centroids of artificial doublet clusters. Finally, it defines doublets as those putative doublets whose gene expression profiles are dissimilar to those of original droplet clusters.

best of our knowledge, our collection is by far the most comprehensive set of scRNA-seq data that contains experimentally validated doublets, and it can serve as a benchmark standard for future method development.

To benchmark the nine methods, we included two baseline methods, which simply use the library size (lsize) and the number of expressed genes (ngene) of each droplet as their respective doublet-detection criterion (Luecken and Theis, 2019; Wolock et al., 2019). Except for DoubletDecon, all the methods output a doublet score for each droplet (Table 1; The two baseline methods have lsize and ngene as their doublet scores; a droplet with a larger score is more likely a doublet), and we defined their detection accuracy as their AUPRC and AUROC values (STAR Methods). We found that all the methods successfully output their identified doublets from all the 16 datasets except DoubletDetection, which could not run on the pdx-MULTI dataset. Across the 16 datasets, each method exhibited a large variance in its detection accuracy, and no method consistently achieved the top performance (Figures 1A and 1B; Tables S2 and S3). Compared with the two baseline methods, doubletCells was the only method that did not outperform them on a majority of datasets, while Solo and hybrid were the only two methods that consistently outperformed them on all datasets (Table S4). Overall, DoubletFinder and Solo achieved the highest mean AUPRC and AUROC values across datasets, respectively (Tables S2 and S3). DoubletFinder was also the top-performing method on the most datasets in terms of both AUPRC and AUROC (Table S4). We note that all the methods had AUPRC values much lower than their AUROC values on every dataset, an expected phenomenon given the imbalance between the number of singlets and doublets. Since AUROC is an overly optimistic measure of accuracy under such imbalanced scenarios (Branco et al., 2016), we will focus on AUPRC in the following discussion.

The highest AUPRC value on each dataset ranged from 0.239 to 1.000, with a mean of 0.570 across the 16 datasets (Table S2). This large discrepancy between datasets is further exemplified by the fact that several methods achieved almost perfect AUPRC values on two datasets: hm-12k and hm-6k, while all the methods performed poorly on another two datasets: pbmc-1B-dm and J293t-dm (with AUPRC values under 0.335). A likely reason for this discrepancy is the annotation of doublets in these real datasets. In hm-12k and hm-6k, doublets were annotated as the droplets that contain cells of two species, so all annotated doublets were heterotypic and easy to identify (Bais and Kostka, 2020; McGinnis et al., 2019a, 2019b; Wolock et al., 2019; Bernstein et al., 2020). In contrast, doublets annotated in the other datasets might have included homotypic doublets that are difficult to identify, posing a challenge to doublet-detection methods; or they might have missed certain heterotypic doublets (e.g., if doublets are defined as the droplets that contain cells from two individuals, then heterotypic doublets formed by cells of different types within an individual would be missed), creating a downward bias in the calculation of detection accuracy (see further discussion in the STAR Methods). In addition, varied data quality and cell heterogeneity pose different levels of difficulty to doublet detection. The highest mean AUPRC value, which was achieved by DoubletFinder, was only 0.537. These results demonstrate the general difficulty in

Please cite this article in press as: Xi and Li, Benchmarking Computational Doublet-Detection Methods for Single-Cell RNA Sequencing Data, Cell Systems (2020), https://doi.org/10.1016/j.cels.2020.11.008

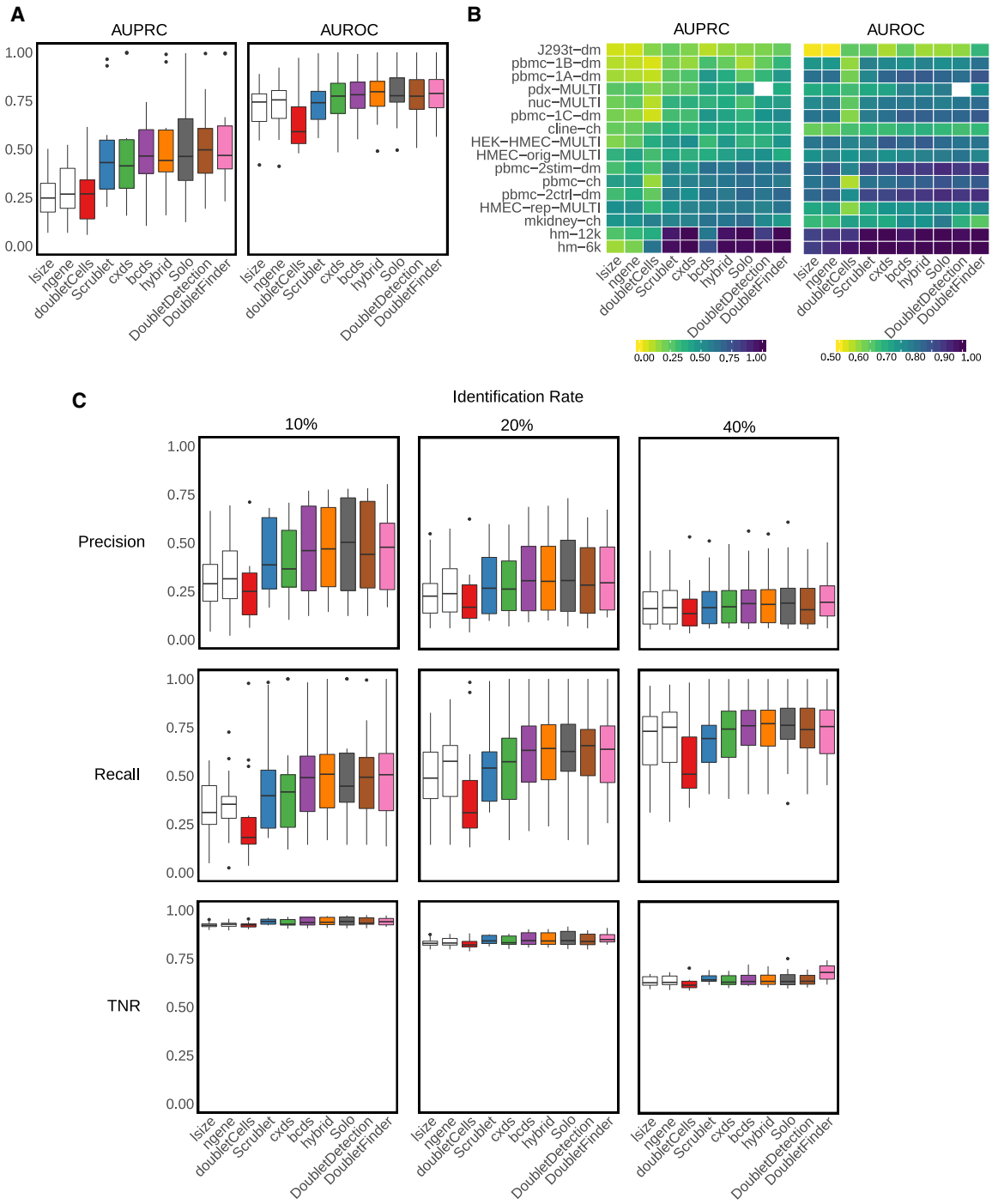


Figure 1. Evaluation of the Eight Doublet Detection Methods (Except DoubletDecon) Using 16 Benchmark scRNA-seq Datasets

(A and B) Performance (AUPRC and AUROC values) of each method applied to benchmark datasets, with (A) showing the distributions and (B) showing the values per dataset (white squares indicating failed runs); two baseline methods (lsize and ngene) are included in the comparison.

(C) Precision, recall, and TNRs of each method under the 10%, 20%, or 40% identification rate, which is the percentage of droplets that received the highest doublet scores and were identified as doublets.

detecting doublets from scRNA-seq data and suggest possible room for improvement by future method development.

Motivated by the fact that doublets are identified based on a single threshold in practice, we further examined the detection accuracy of doublet-detection methods under a specific identification rate, i.e., the percentage of droplets identified as doublets. For each method, the top 10%, 20%, and 40% droplets with the highest doublet scores were identified as doublets,

Please cite this article in press as: Xi and Li, Benchmarking Computational Doublet-Detection Methods for Single-Cell RNA Sequencing Data, Cell Systems (2020), https://doi.org/10.1016/j.cels.2020.11.008

and the corresponding precision, recall, and true negative rates (TNRs) were calculated (Figure 1C; Table S5). As expected, higher identification rates led to higher recall and lower TNR values. Interestingly, the precision decreased as the identification rate increased, a phenomenon suggesting that all doublet-detection methods tend to assign higher doublet scores to annotated doublets and are thus desirable (Figure 1C). The comparison of doublet-detection methods gave a result consistent with that based on the overall detection accuracy measures, AUPRC and AUROC. DoubletFinder and Solo were still the top two methods in terms of the mean precision, recall, and TNR, where the mean was calculated across the 16 datasets (Table S5).

Since DoubletDecon cannot output doublet scores, we could not calculate its AUPRC or AUROC on a dataset, and thus, excluded it from the previous comparison. To fairly compare DoubletDecon with other methods, we ran DoubletDecon on every dataset and recorded its number of identified doublets if successful; then, we thresholded the doublet scores of other methods so that they identified the same number of doublets as DoubletDecon did. Based on the resulting doublets identified by each method from every dataset, we calculated the precision, recall, and TNR (STAR Methods). By these three criteria, DoubletDecon and doubletCells did not outperform the baseline methods lsize and ngene. Among the other seven methods, Solo and DoubletFinder achieved the highest precision and TNRs, while Solo and hybrid obtained the highest recall rates (Figure S1A; Tables S6–S8). Moreover, we observed that DoubletDecon failed to run on four datasets (hm-12k, pbmc-2ctrl-dm, J293t-dm, and nuc-MULTI) and tended to overestimate the number of doublets (Table S9). Our results suggest that DoubletDecon needs improvement in its accuracy and robustness. Adding the functionality that outputs doublet scores will also enhance the usability of DoubletDecon, because users can then have the flexibility to decide the number of doublets to be detected and removed based on their preference and knowledge (Bloom, 2018).

Doublet-Detection Accuracy on Synthetic scRNA-Seq Data under Various Experimental Settings and Biological Conditions

To thoroughly evaluate the performance of doublet-detection methods under a wide range of experimental settings and biological conditions, we utilized scDesign (Li and Li, 2019), a statistical simulator that generates realistic scRNA-seq datasets well mimicking real data generated by a variety of scRNA-seq experimental protocols. It is advantageous to use synthetic data to benchmark doublet-detection methods, because we would have the access to ground-truth doublets and the flexibility to vary experimental settings and biological conditions in a comprehensive way. Specifically, we generated 80 scRNA-seq datasets with varying doublet rates (i.e., percentages of doublets), sequencing depths, cell types, and between-cell-type heterogeneity levels (STAR Methods). Except for DoubletDecon, we applied every doublet-detection method to all these synthetic datasets and calculated its AUPRC values to measure its accuracy. Figure 2A shows how the performance of every method changed as we varied the doublet rate, the sequencing depth, the number of cell types, or the between-cell-type heterogeneity level.

First, all the eight methods had improved accuracy as the doublet rate increased. This result is not surprising, as these methods all formulated the doublet-detection problem, explicitly or implicitly, as a binary classification problem where the two classes are singlets and doublets. The more balanced the two classes are in size, the easier the binary classification is, in general. Given the fact that under both droplet microfluidics and well-based scRNA-seq protocols, doublets are more likely to form as the number of cells increases (Zheng et al., 2017; Luecken and Theis, 2019; Wolock et al., 2019), our result suggests that doublet-detection methods would work more effectively on scRNA-seq datasets with more cells (or droplets). This finding agrees with our previous result that all the methods performed the worst on the J293t-dm dataset, which contains only 500 droplets, the fewest among all the 16 datasets.

Second, we found that the performance of these methods consistently benefited from a larger sequencing depth. This is in line with the expectation that deeper sequencing creates a higher data resolution, making doublet-detection methods more capable of differentiating doublets from singlets.

Third, we evaluated the impact of the number of cell types on the accuracy of doublet-detection methods. It is expected that a cell mixture with more cell types would result in more heterotypic doublets, which are formed by cells of different types. Thanks to their distinct gene expression profiles that do not resemble those of any cell types, heterotypic doublets are, in general, easier to detect than homotypic doublets, which are formed by cells of the same type (Wolock et al., 2019). As expected, most methods exhibited improved accuracy as the number of cell types increased, with cxds, bcds, and hybrid (a combination of cxds and bcds) as the only three exceptions.

Fourth, we investigated how the between-cell-type heterogeneity level—the extent to which gene expression profiles differ between cell types—would affect the accuracy of doublet detection. In theory, the greater the heterogeneity, the more distinct heterotypic doublets are from singlets. Again, all the methods fit this theory except cxds, bcds, and hybrid. Hence, we saw consistent results about the effects of the number of cell types and the between-cell-type heterogeneity level on doublet detection.

We also compared the AUROC values of the eight doublet-detection methods on the same synthetic scRNA-seq datasets as above (Figure S1B). Consistent with our AUPRC results, most methods performed better on the datasets with a higher doublet rate, a larger sequencing depth, more cell types, or a greater level of between-cell-type heterogeneity, though the improvement in AUROC was less significant than in AUPRC. This is expected as AUPRC is a better accuracy measure than AUROC for imbalanced binary classification (Saito and Rehmsmeier, 2015). Combining our AUPRC and AUROC results, we found DoubletFinder as the top-performing method across all the experimental settings and biological conditions we studied. DoubletDetection and Scrublet also demonstrated strong performance compared with the rest of methods. We excluded DoubletDecon from this comparison and the following DE gene identification, highly variable gene identification, cell clustering, and cell trajectory inference analyses, because it failed to run on most of our synthetic datasets, likely due to its software implementation issue (Github, 2020).

Please cite this article in press as: Xi and Li, Benchmarking Computational Doublet-Detection Methods for Single-Cell RNA Sequencing Data, Cell Systems (2020), https://doi.org/10.1016/j.cels.2020.11.008

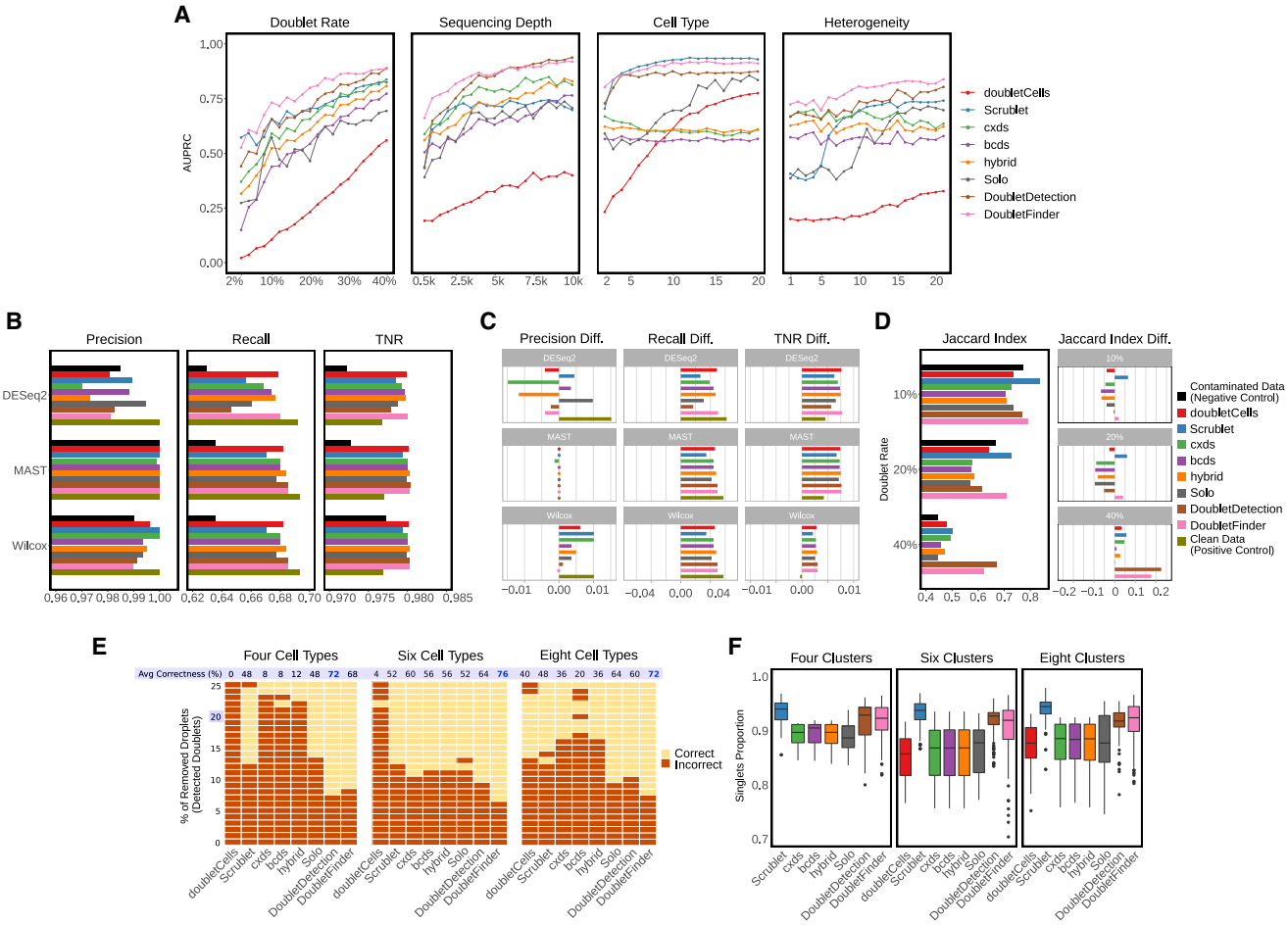


Figure 2. Evaluation of the Eight Doublet-Detection Methods (Except DoubletDecon) Using Four Simulation Studies, and the Effects of Doublet Detection on DE Analysis, HVGs Identification, and Cell Clustering

(A) Performance (AUPRC values) of each method in four simulation settings: varying doublet rates (from 2% to 40% with a step size of 2%), varying sequencing depths (from 500 to 10,000 UMI counts per cell, with a step size of 500 counts), varying numbers of cell types (from 2 to 20 with a step size of 1), and 20 heterogeneity levels, which specify the extent to which genes are differentiated between two cell types (STAR Methods).

(B) Precision, recall, and TNR by each of three differential expression (DE) methods: DESeq2, MAST, and the Wilcoxon rank-sum test (Wilcox), after each of the eight doublet-detection methods was applied to a simulated dataset; for negative and positive controls, we included the DE accuracies on the contaminated data with 40% doublets and the clean data without doublets.

(C) We re-illustrate the results in (B) by showing the improved DE accuracy in each metric (precision, recall, and TNR) after removing detected doublets from the contaminated data; the results on the clean data without doublets are shown as a positive control.

(D) Left panel: the Jaccard index between the post-doublet-detection HVGs of each doublet-detection method and the clean HVGs under the 10%, 20%, or 40% doublet rate. The Jaccard index between the contaminated HVGs and the clean HVGs was used as negative control for each doublet rate. Right panel: illustration of the left panel; the improved Jaccard indices upon the negative controls (i.e., Jaccard index differences) after the detected doublets by each method were removed from the contaminated data.

(E) Cell clustering result by the Louvain algorithm after each of the eight doublet-detection method was applied to remove a varying percentage of droplets as the identified doublets (y axis, from 0% to 25% with step size of 1%); the true numbers of cell clusters are four, six, and eight under three simulation settings, each containing 20% true doublets; the yellow color indicates that the correct number of clusters was identified, while the red color indicates otherwise. The true percentage of doublets, 20%, is highlighted in blue. For each method, its average correctness (i.e., the percent of yellow colors across all the removal percentages) is also highlighted in blue.

(F) Under the same three simulation settings as in (A), the distributions of the singlet proportions are shown after doublet removal by each method, if the remaining droplets led to the correct number of cell clusters in (A); doubletCells is not shown for the four-cluster setting, because it did not lead to the correct number of cell clusters in (A).

Effects of Doublet Detection on DE Gene Analysis

The existence of doublets in scRNA-seq datasets is expected to confound the downstream DE gene analysis by violating the necessary "identical distribution" assumption (i.e., cells of the same type follow the same distribution of gene expression levels) in statistical tests (Luecken and Theis, 2019). As a result, if a doublet-detection method is effective, its doublet removal should improve the accuracy of DE gene analysis. To evaluate the eight doublet-detection methods from this perspective, we used scDesign to generate a synthetic scRNA-seq dataset

Please cite this article in press as: Xi and Li, Benchmarking Computational Doublet-Detection Methods for Single-Cell RNA Sequencing Data, Cell Systems (2020), https://doi.org/10.1016/j.cels.2020.11.008

with two cell types and 1,126 between-cell-type DE genes (6% of a total of 18,760 genes; STAR Methods). We referred to this dataset as the "clean data." We then mixed each cell type with randomly forming doublets by targeting a 40% doublet rate, and the resulting dataset was referred to as the "contaminated data." Next, we applied each doublet-detection method to the dataset and removed 40% droplets (with the highest doublet scores assigned by each method) from the contaminated data. Finally, we conducted DE gene analysis using three methods—DESeq2 (Love et al., 2014), MAST (Finak et al., 2015), and Wilcoxon rank-sum test (Fay and Proschan, 2010)—on the clean data, the contaminated data, and the dataset after each doublet-detection method was applied. The DE gene analysis result was summarized in three accuracy measures: precision, recall, and TNR, all of which were calculated under the Bonferroni-corrected p value threshold of 0.05, the default threshold used by DESeq2 and MAST (Wang et al., 2019).

We benchmarked the accuracy resulted from each doublet-detection method against the negative control (the accuracy based on the contaminated data) and the positive control (the accuracy based on the clean data). Figure 2B shows that all the three DE methods achieved extremely high precision (> 98%) and TNRs (> 97%) even on the contaminated data, an expected result because these DE methods all utilize statistical tests and are inherently conservative in their identification of DE genes. Such conservativeness makes these DE methods only identify the genes that are highly likely DE, leading to high precision (the percentage of true DE genes among the identified genes) and TNR (the percentage of non-identified genes among the true non-DE genes). Although the TNR result seems counterintuitive as the TNR values after doublet detection and removal even exceeded the TNR values of the clean data by around 0.005, this difference was merely due to the statistical uncertainty of these TNR values and thus is not conclusive. On the other hand, recall (the percentage of identified genes among the true DE genes) is an informative measure that reflects the negative influence of doublets: For all the three DE methods, their recall dropped from ~70% on the clean data to ~63% on the contaminated data. Pleasantly, all the eight doublet-detection methods were effective in improving the recall (Figure 2C). In particular, DoubletFinder, doubletCells, bcds, and hybrid consistently had top performance, regardless of the choice of DE methods. This result confirms that removing doublets is indeed beneficial for DE gene analysis.

Effects of Doublet Detection on Highly Variable Gene Identification

The identification of highly variable genes (HVGs) is an essential step that precedes cell dimension reduction, cell clustering, and cell trajectory inference in scRNA-seq data analysis (Yip et al., 2019). The goal of this step is to identify HVGs, i.e., the informative genes that exhibit strong cell-to-cell variations and thus can distinguish cells, so that the dimensions of each cell can be reduced from tens of thousands of genes to thousands, or even hundreds of genes, to facilitate those downstream analyses. Considering the importance of HVG identification, we evaluated the extent to which the identification would be negatively affected by doublets (Amezquita et al., 2020) and how much the eight doublet-detection methods could alleviate such negative impacts.

For this purpose, we simulated a clean scRNA-seq dataset without doublets by scDesign, and then we added randomly formed doublets to generate three contaminated datasets with 10%, 20%, and 40% doublet rates. For each contaminated dataset, we applied the eight doublet-detection methods to remove a percentage of droplets that received the highest doublet scores, and the percentage was set as the dataset's doublet rate. As a result, each contaminated dataset corresponds to eight post-doublet-detection datasets. Then, we used Seurat (Butler et al., 2018; Stuart et al., 2019) to identify HVGs from the clean dataset, the three contaminated datasets, and the 24 post-doublet-detection datasets. We refer to the identification results as a set of clean HVGs, three sets of contaminated HVGs, and 24 sets of post-doublet-detection HVGs. An effective doublet-detection method is expected to result in post-doublet-detection HVGs that agree better with the clean HVGs than the corresponding contaminated HVGs do. To measure the agreement between two sets of HVGs, we used the Jaccard index, which is the ratio of the size of the intersection to the size of the union of the two sets. The larger the Jaccard index, the better agreement the two sets have. In our evaluation, for each doublet rate, the Jaccard index between the contaminated HVGs and the clean HVGs served as the negative control.

Figure 2D shows that the negative control Jaccard index decreased from 0.772 to 0.447 as the doublet rate increased from 10% to 40%, matching our expectation. Among the eight doublet-detection methods, DoubletFinder and Scrublet were the only two methods whose post-doublet-detection HVGs consistently led to better Jaccard indices than the negative controls under all three doublet rates. Notably, the benefit of doublet detection on HVG identification was most obvious at the 40% doublet rate, under which all the doublet-detection methods outperformed the negative control.

Effects of Doublet Detection on Cell Clustering

Another major motivation to remove doublets from scRNA-seq data is to avoid the misinterpretation of spurious cell clusters (i.e., droplet clusters) formed by heterotypic doublets as novel cell types (Luecken and Theis, 2019; Wolock et al., 2019). To evaluate the capacity of doublet-detection methods for removing spurious cell clusters, we used scDesign to simulate realistic scRNA-seq datasets composed of four, six, or eight cell types and mixed with 20% randomly forming doublets (i.e., the true doublet rate is 20%). We performed cell clustering on each of these datasets after applying every doublet-detection method and removing a certain percent of droplets that received the highest doublet scores from that method (STAR Methods). Considering that the true doublet rate is unknown and difficult to estimate in practice, we varied this removal percentage from 0% to 25%, with a step size of 1%. For the subsequent cell clustering, we followed the most popular Seurat method to apply the Louvain clustering algorithm (Blondel et al., 2008), which automatically determines the number of cell clusters in a data-driven way. Then for each dataset, every doublet-detection method, and each removal percentage, we compared the number of cell clusters with the number of cell types.

Figure 2E shows that, under the ideal scenario that the removal percentage was set to the true doublet rate 20%, four methods (Scrublet, Solo, DoubletDetection, and

Please cite this article in press as: Xi and Li, Benchmarking Computational Doublet-Detection Methods for Single-Cell RNA Sequencing Data, Cell Systems (2020), https://doi.org/10.1016/j.cels.2020.11.008

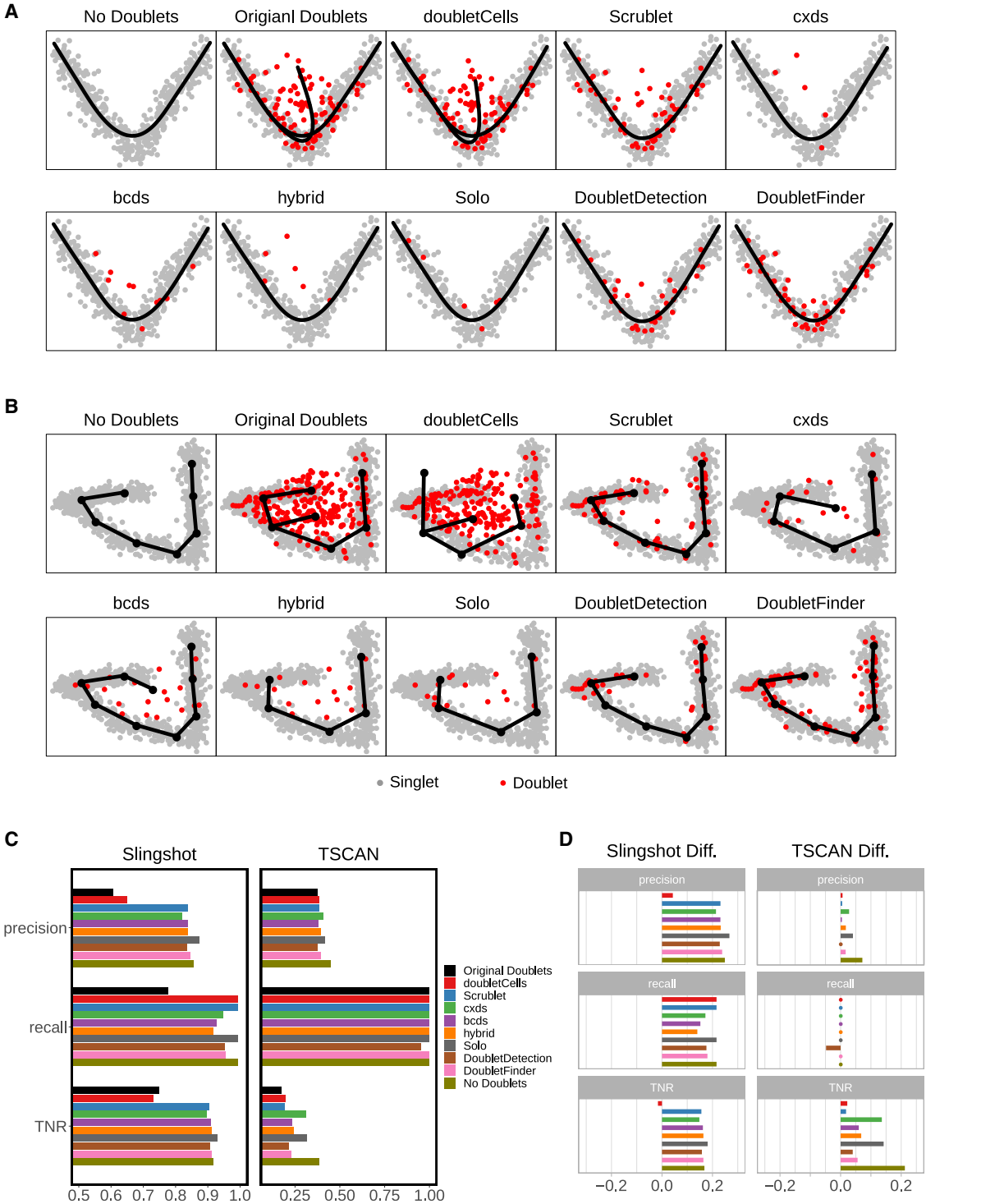


Figure 3. Effects of Doublet Detection on Cell Trajectory Inference

(A) Trajectories constructed by Slingshot after each of the eight doublet-detection methods were applied to remove the identified doublets, whose percentage among all the droplets was set to 20%, the percentage of true doublets in the simulated dataset. The true cell topology is bifurcating. For negative and positive controls, we included the trajectories constructed on the original dataset with 20% doublets and its cleaned version without doublets.

(B) Trajectories constructed by MST after each of the eight doublet-detection methods were applied to remove the identified doublets, whose percentage among all the droplets was set to 20%, the percentage of true doublets in the simulated dataset. The true cell topology is a conjunction of three trajectories. For negative and positive controls, we included the trajectories constructed on the original dataset with 20% doublets and its cleaned version without doublets.

(C) Precision, recall, and TNR of temporally DE genes identified by the GAM applied to trajectories constructed by Slingshot and TSCAN, after each of the eight doublet-detection method was applied to remove the identified doublets, whose percentage among all the droplets was set to 20%, the percentage of true

(legend continued on next page)

Please cite this article in press as: Xi and Li, Benchmarking Computational Doublet-Detection Methods for Single-Cell RNA Sequencing Data, Cell Systems (2020), https://doi.org/10.1016/j.cels.2020.11.008

DoubletFinder) consistently removed spurious cell clusters and led to the correct numbers of cell types. Among the eight methods, DoubletDetection and DoubletFinder exhibited the most robust performance, as they successfully led to the correct numbers of cell types under the widest range of removal percentages. Scrublet and Solo also exhibited good performance in removing spurious cell clusters. In contrast, doubletCells, cxds, bcds, and hybrid all had unstable performance, and they did not always remove spurious cell clusters even under the ideal scenario (when the removal percentage was set to 20%). Overall, this result supports the use of DoubletDetection and DoubletFinder to remove doublets before the application of cell clustering to identify novel cell types.

Unlike heterotypic doublets, homotypic doublets do not form spurious clusters because of their similar gene expression profiles to those of singlets of the same cell type (Wolock et al., 2019). In other words, homotypic doublets tend to cluster together with singlets. Even though the existence of homotypic doublets does not much affect cell clustering, it may potentially bias the identification of cell-type-specific genes by DE gene analysis because homotypic doublets are not real cells. To evaluate the capacity of doublet-detection methods in eliminating homotypic doublets, we calculated the proportion of singlets in each identified cell cluster when the number of cell clusters matched the number of cell types in Figure 2E (STAR Methods). Figure 2F shows that Scrublet led to cell clusters with the highest proportions of singlets. DoubletDetection and DoubletFinder also had excellent performance, and these three methods all clearly outperformed the rest of the methods. Combining the results in Figures 2E and 2F, we conclude that Scrublet, DoubletDetection, and DoubletFinder demonstrated the best capacity in removing heterotypic and homotypic doublets.

To examine how robust the above results are to the choice of clustering algorithms, we repeated the above analyses using a second clustering algorithm: the density-based spatial clustering of applications with noise (DBSCAN) (Ester et al., 1996). Compared with the Louvain clustering algorithm, the DBSCAN algorithm led to the correct numbers of cell clusters under fewer and more sporadic removal percentages for all the doublet-detection methods (Figure S2A). This result suggests that the DBSCAN algorithm works less effectively than the Louvain algorithm for clustering cells in scRNA-seq data (Duò et al., 2018; Feng et al., 2020). Nevertheless, with the DBSCAN algorithm, Scrublet, DoubletDetection, and DoubletFinder still achieved the top performance in removing spurious cell clusters and homotypic doublets (Figures S2A and S2B). In summary, based on the results of two clustering algorithms, we would recommend DoubletDetection and DoubletFinder as the top two choices for removing spurious cell clusters in cell clustering analysis, and we identified Scrublet and DoubletFinder as the best-performing algorithms for removing homotypic doublets before the identification of cell-type-specific genes.

Effects of Doublet Detection on Cell Trajectory Inference

Another important scRNA-seq data analysis is to infer a cell trajectory, which corresponds to a cellular process, such as cell differentiation, immune responses, and carcinogenesis, based on the similarity of cells in terms of gene expression profiles (Saelens et al., 2019). An inferred cell trajectory is called pseudotime, an ordering of cells in a path or a tree (Trapnell et al., 2014). The accuracy of cell trajectory inference depends on both the inference methods and the scRNA-seq data quality. Similar to cell clustering, cell trajectory inference is also biased by the existence of doublets (Tian et al., 2019). In particular, heterotypic doublets may result in spurious branches in an inferred trajectory. We expect that doublet-detection methods, if effective, should increase the accuracy of cell trajectory inference.

To evaluate the eight doublet-detection methods from this perspective, we used Splatter (Zappia et al., 2017) to generate two scRNA-seq datasets: one including a bifurcating trajectory and the other containing a conjunction of three sequential trajectories (STAR Methods). We referred to them as "clean data." We then mixed the two datasets with randomly forming doublets by targeting a 20% doublet rate, and the resulting datasets were referred to as the "contaminated data." Similar to our DE gene analysis, we used each doublet-detection method to remove 20% droplets (with the highest doublet scores assigned by that method) from each contaminated dataset. As a result, we obtained two suites of datasets corresponding to a bifurcating trajectory and a conjunction of three sequential trajectories, with each suite containing the clean data, the contaminated data, and the data cleaned by each doublet-detection method. For cell trajectory inference, we applied Slingshot (Street et al., 2018) to the first suite of datasets (Figure 3A) and minimum spanning tree (MST) (Herring et al., 2018) to the second suite of datasets (Figure 3B). We chose Slingshot and MST because they were the top-performing methods in previous benchmark studies (Saelens et al., 2019; Tian et al., 2019). We considered the cell trajectories inferred from the clean data and the contaminated data as the positive and negative controls, respectively.

Figures 3A and 3B show that the doublets in the contaminated data indeed led to spurious branches that did not exist in the inferred trajectories from the clean data. Except for doubletCells, all the doublet-detection methods effectively removed doublets such that spurious branches no longer existed in the inferred cell trajectories. In particular, in the second task of inferring a conjunction of three sequential trajectories (Figure 3B), Scrublet, DoubletDetection, and DoubletFinder led to inferred trajectories that most resembled the trajectory inferred from the clean data. Figures 3A and 3B also show that DoubletDetection and DoubletFinder are the best two methods for removing the "outlier" doublets whose gene expression profiles do not resemble those of any singlets.

Following cell trajectory inference, a typical next step is to explore gene expression dynamics along the inferred trajectory

doublets in the simulated dataset. The true cell topology is a single lineage. For negative and positive controls, we included the accuracy of temporally DE genes identified from the contaminated data with 20% doublets and the clean data without doublets.

(D) We re-illustrate the results in (C) by showing the improved accuracy in each metric (precision, recall, and TNR) after removing detected doublets from the contaminated data; the results on the clean data without doublets are shown as a positive control.

Please cite this article in press as: Xi and Li, Benchmarking Computational Doublet-Detection Methods for Single-Cell RNA Sequencing Data, Cell Systems (2020), https://doi.org/10.1016/j.cels.2020.11.008

and to identify temporally DE genes (Luecken and Theis, 2019; Saelens et al., 2019). Hence, the accuracy of cell trajectory inference largely determines the accuracy of temporally DE gene identification. Beyond checking the inferred cell trajectories after doublet removal, as shown in Figures 3A and 3B, we evaluated the effects of doublet removal on the identification of temporally DE genes. We used Splatter to simulate a scRNA-seq dataset with a single lineage and 250 temporally DE genes out of a total of 750 genes (STAR Methods). We referred to this dataset as the "clean data." We then mixed the data with randomly forming doublets by targeting a 20% doublet rate, and the resulting dataset was referred to as the "contaminated data." Next, we used eight doublet-detection methods to remove 20% droplets (with the highest doublet scores assigned by each method) from the contaminated data. Finally, we employed a general additive model (GAM) (Hastie and Tibshirani, 1990) to regress each gene's expression levels on the corresponding cell/droplet pseudotime inferred by Slingshot or TSCAN (Ji and Ji, 2016) on the clean data, the contaminated data, and the dataset after each doublet-detection method was applied. Note that we replaced MST by TSCAN because MST does not output pseudotime values for droplets and TSCAN is built upon the MST algorithm.

The temporally DE gene analysis result was summarized in three accuracy measures: precision, recall, and TNR, all of which were calculated under the Bonferroni-corrected p value threshold of 0.05. Again, we used the accuracy obtained from the clean data and the contaminated data as the positive and negative controls, respectively. Doublet removal made a more significant improvement on the identification of temporally DE genes when Slingshot was used for trajectory inference (Figures 3C and 3D). With Slingshot, all the eight doublet-detection methods except doubletCells successfully restored the precision, recall, and TNR from low values on the contaminated data to values as high as those on the clean data. With TSCAN, however, the restoration effects were only obvious in precision and TNR by Solo and cxds. In summary, doublet removal is beneficial for cell trajectory inference and the subsequent identification of temporally DE genes, and we observed strong beneficial effects when Slingshot was used for trajectory inference.

Performance of Doublet-Detection Methods under Distributed Computing

A grand challenge in single-cell data sciences is the skyrocketing demand for computational and storage resources due to the rapidly increasing data sizes (Lähnemann et al., 2020). For example, a scRNA-seq dataset may contain up to millions of droplets, each of which has expression levels of tens of thousands of genes (Regev et al., 2017). Analyzing such huge datasets is often beyond the capacity of a single computer but requires distributed computing, which analyzes data subsets in parallel. Specific to the doublet-detection task, distributed computing means that droplets are divided into batches, one batch per computer node; then a doublet-detection method would be applied in parallel to all batches, and it would assign doublet scores to droplets in each batch. After this parallelization step, doublet scores would be pooled from multiple batches, and a threshold would be set on the pooled doublet scores to detect doublets. Compared with the centralized computing that uses all the droplets together, distributed computing may have deteriorated doublet-detection accuracy due to the limited data information within each droplet batch. Hence, how a doublet-detection method performs under distributed computing is an important evaluation criterion for the scalability and flexibility of the method.

To investigate the performance of doublet-detection methods under distributed computing, we randomly divided two large real scRNA-seq datasets—pbmc-ch and pbmc-2ctrl-dm—into a varying number of batches with equal numbers of droplets, and we evaluated how the doublet-detection accuracy of each method changed with the number of batches. It is expected that the more the batches, the worse the accuracy, and our results confirmed this. Figures 4A and 4B show the AUPRC and AUROC values of each method under each number of batches, which varied from one to ten. The AUPRC and AUROC values were calculated based on the pooled doublet scores as described above. We excluded DoubletDecon from this comparison, because it failed to run for most numbers of batches, again suggesting its software implementation issue (Github, 2020). With only one batch, distributed computing is reduced to centralized computing, and the corresponding accuracy is supposedly the performance ceiling of every method. As expected, most doublet-detection methods had decreasing accuracy, which is clearer in AUPRC (Figure 4A) than AUROC (Figure 4B), as the number of batches increased. Among the eight methods, doubletCells was an underperforming outlier with the lowest overall accuracy. DoubletDetection and Solo were among the top-performing methods under centralized computing; however, they exhibited the largest accuracy decrease under distributed computing. In contrast, DoubletFinder was consistently a top performer, demonstrating its superior accuracy again and its robustness under distributed computing.

Computational Efficiency, Scalability, Stability, and Software Implementation of Doublet-Detection Methods

In addition to the above evaluation that focused on the effects of doublet removal on various scRNA-seq data analyses, we also compared doublet-detection methods in four computational aspects: efficiency, scalability, stability, and software implementation.

First, we summarized the running time of the nine doublet-detection methods (including their required data preprocessing steps; STAR Methods) on the 16 real scRNA-seq datasets in Table S1. Figure 4C shows that cxds is the fastest method, while Solo, DoubletDecon, DoubletDetection, and DoubletFinder are significantly slower than the other methods. Figure 4D shows that there was no straightforward relationship between the mean AUPRC and the mean running time of eight doublet-detection methods (with the mean calculated across the 16 real datasets). Nevertheless, the three most computationally intensive methods—Solo, DoubletDetection, and DoubletFinder—had better accuracy than the other methods except hybrid did. Interestingly, the hybrid method, an ensemble of cxds and bcds, largely improved on both base methods without much running time increase. Among all methods, DoubletFinder achieved the highest mean AUPRC, while not being the most computationally intensive method. Normalizing the mean running time by the mean AUPRC value for every method, we found cxds as the most resource-efficient method (Table S10).

Please cite this article in press as: Xi and Li, Benchmarking Computational Doublet-Detection Methods for Single-Cell RNA Sequencing Data, Cell Systems (2020), https://doi.org/10.1016/j.cels.2020.11.008

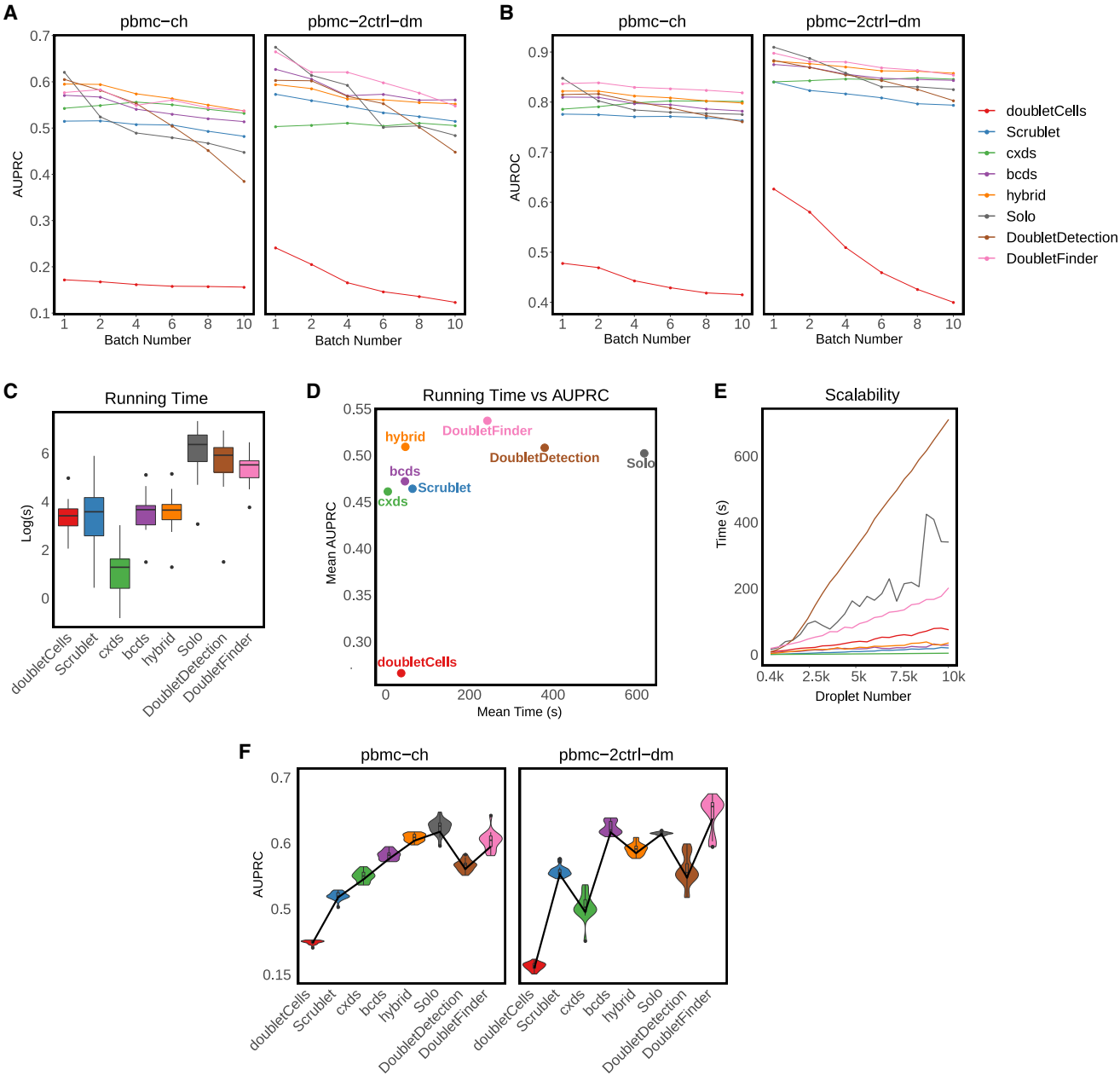


Figure 4. Comparison of Doublet-Detection Methods in Terms of Distributed Computing, Running Time, Scalability, and Stability

(A and B) Distributed computing performance of each method on two real datasets pbmc-ch and pmc-2ctrl-dm. We first divided the original datasets into varying numbers of batches with equal sizes; then we applied each method to individual batches separately to identify and remove doublets; finally, we pooled batches together to assess the detection accuracy (AUPRC and AUROC values) of each method. The legend on the right applies to both panels (A) and (B).

(C) Distribution of running time in (natural log) seconds of each method across 16 real datasets.

(D) Mean AUPRC versus mean running time (across 16 real datasets) of eight doublet-detection methods.

(E) Scalability of each method. We calculated the relationship between running time and droplet number for each method on simulated datasets with varying droplet numbers.

(F) Stability of each method. We generated 20 datasets by randomly subsampling 90% droplets and 90% genes from the real datasets pbmc-ch and pbmc-2ctrl-dm, and we applied each method to all the subsampled datasets. For each real dataset, the distribution of AUPRC values of each method across subsampling is shown, with 25% quantiles connected. We use the variance of the distribution to measure the stability of each method.

Second, we examined the scalability of doublet-detection methods by how fast their running time increases as the number of droplets grows. We used scDesign to generate 25 synthetic scRNA-seq datasets with the number of droplets ranging from 400 to 10,000 (STAR Methods). Then we applied each doublet-detection method to these datasets and recorded its running time (DoubletDecon was excluded, because it failed to run on most synthetic data.) As shown in Figure 4E, all methods except

Please cite this article in press as: Xi and Li, Benchmarking Computational Doublet-Detection Methods for Single-Cell RNA Sequencing Data, Cell Systems (2020), https://doi.org/10.1016/j.cels.2020.11.008

Solo had running time scaled linearly with the number of droplets. The reason that Solo exhibited an erratic relationship between its running time and the number of droplets is probably due to its neural-network design. Among the other seven methods, cxds and DoubletDetection demonstrated the best and worst scalability, respectively.

Third, we evaluated doublet-detection methods in terms of the statistical stability, i.e., how much their AUPRC and AUROC values varied across subsets of droplets and genes. The smaller the variation, the larger the statistical stability. We randomly downsampled two large real scRNA-seq datasets—pbmc-ch and pbmc-2ctrl-dm—into 20 data subsets with 90% droplets and 90% genes. Then we applied each doublet-detection method to these data subsets and recorded the resulting AUPRC and AUROC values (DoubletDecon was excluded, because we were unable to calculate its AUPRC and AUROC values, as explained before.) Figures 4F and S2C show the distributions of AUPRC and AUROC values of each method when applied to the subsets generated from each original dataset. Interestingly, we observed a roughly inverse relationship between the overall doublet-detection accuracy and the statistical stability. For example, DoubletFinder had the best overall accuracy in terms of both AUPRC and AUROC, yet, its variation across data subsets was much greater than that of Scrublet, which had a much lower overall accuracy. Despite its suboptimal stability, we still found DoubletFinder as a top performer if we compare the lower-quartile accuracy (i.e., the 25th percentile of AUPRC and AUROC values) of these methods. To summarize, even though statistical stability is an important criterion, in practice, it is often overruled by the overall accuracy reflected by the mean, median, or lower-quartile accuracy value. In terms of the overall accuracy, we found DoubletFinder, Solo, and hybrid as the top three methods.

Fourth, we evaluated the software implementation of doublet-detection methods, because user-friendliness, software quality, and active maintenance are crucial to the success of bioinformatics tools (Mangul et al., 2019). We scored each method in four aspects: software quality, execution convenience, publication, and documentation and support (STAR Methods). Table 2 lists our score reasoning and the overall usability score of each method. In particular, DoubletDetection and DoubletDecon did not successfully run on one or more datasets. Regarding user support, Solo, DoubletDetection, DoubletFinder, and DoubletDecon have active Q&As on their software webpages for collecting users' feedback and answering users' questions. Among the nine methods, DoubletFinder achieved the highest usability score thanks to its excellent implementation.

DISCUSSION

With the rapid development of scRNA-seq technologies, a skyrocketing number of computational methods have been developed for various scRNA-seq data analyses (Zappia et al., 2018). For example, since 2018, more than 45 imputation methods have already been developed to recover missing gene expression (commonly referred to as "dropouts") in scRNA-seq data (Li and Li, 2018; Lopez et al., 2018; Risso et al., 2018; van Dijk et al., 2018; Lähnemann et al., 2020). Such richness of computational methods is a double-sided blade. On the one hand, scRNA-seq researchers have more blocks to build analysis pipelines that accommodate their scientific investigation needs; on the other hand, it becomes increasingly difficult for researchers to choose the method, from dozens of methods developed for the same purpose, that best fits each step of their pipeline. Unlike in experimental sciences where new technologies often replace old ones, there are usually no clear-cut or universal choices of computational methods. An appropriate choice of computational method is case by case, depending on data characteristics and scientific questions at hand. Inappropriate method choices would, to varying extents, bias data analysis (such as by introducing artificial, non-biological signals) and ultimately lead to false discoveries (Andrews and Hemberg, 2018; Weber et al., 2019). To avoid this issue, the scRNA-seq field and the broad biomedical science community yearn for comprehensive benchmark studies that independently and fairly evaluate computational methods (Lähnemann et al., 2020). A well-designed benchmark study should offer users objective, accurate, and informative guidance on selecting the appropriate method(s) for a specific analysis task.

To provide the first, comprehensive benchmark of computational doublet-detection methods, in this study, we evaluated nine existing methods using 16 real and 112 synthetic scRNA-seq datasets from three perspectives: overall detection accuracy, impacts on downstream analyses, and computational efficiency. We further categorized our benchmark results in nine aspects, including four related to doublet-detection accuracy and five associated with software implementation (Figure 5, which does not include DoubletDecon because it failed to run in most evaluations). In summary, DoubletFinder was the best method in terms of accuracy, yet, its computational efficiency and stability were not among the best. The cxds method was the opposite: It had the best computational efficiency, excellent stability, but medium accuracy. Our summary is consistent with the aforementioned principle of computational methods that no method is universally the best, so a fair comparison of computational methods should be multifaceted.

Although our benchmark study has collected all the available scRNA-seq datasets to date that contain doublet annotations, we note that none of the annotations is utterly accurate due to experimental limitations. For example, the two species-mixture datasets, hm-12k and hm-6k, only labeled the heterotypic doublets formed by a human cell and a mouse cell; the six demuxlet datasets only labeled the doublets formed by cells of two individuals; many homotypic doublets were unlabeled in all these datasets. As a result, the incompleteness of doublet annotations would have inflated the false negative rates and reduced the precision of computational doublet-detection methods in our benchmark. To overcome this limitation, we designed extensive simulations to benchmark computational doublet-detection methods in a fair and comprehensive manner. Yet, how to generate accurate doublet annotations by experimental techniques remains an open question to experimental scientists.

Regarding the future development and benchmark of computational doublet-detection methods, here we list five open questions we deem important for computational scientists:

(1) How to estimate the unknown doublet rate in a scRNA-seq dataset? Some methods provide heuristic guidance

Please cite this article in press as: Xi and Li, Benchmarking Computational Doublet-Detection Methods for Single-Cell RNA Sequencing Data, Cell Systems (2020), https://doi.org/10.1016/j.cels.2020.11.008

Table 2. Usability of the Nine Doublet-Detection Methods

	Software Quality	Execution Convenience	Publication	Documentation & Support	Usability Score
doubletCells	excellent (success on all datasets)	excellent (R package)	good (published as a part of a research paper in a peer-reviewed journal)	good (documentation, custom webpage, but no Q&A)	6
Scrublet		excellent (Python module)	excellent (published as an independent research paper in a peer-reviewed journal)	good (documentation, GitHub webpage, but no Q&A)	7
cxds		excellent (R package)			7
bcds					7
Hybrid					7
Solo		good (Linux command-line with a stringent requirement on input data format: loom/hd5)	excellent (published as an independent research paper in a peer-reviewed journal)	excellent (documentation, GitHub webpage, and active Q&A)	7
DoubletDetection	good (failure on one real dataset)	excellent (Python module)	fair (GitHub webpage, manuscript with algorithm description)		5
DoubletFinder	excellent (success on all datasets)	excellent (R package)	excellent (published as an independent research paper in a peer-reviewed journal)		8
DoubletDecon	fair (failure on four real datasets and the majority of synthetic datasets)	excellent (R package)	excellent (published as an independent research paper in a peer-reviewed journal)	excellent (documentation, GitHub webpage, and active Q&A)	6

We measured the usability of each method in four aspects: software quality, execution convenience, publication, and documentation and support. Each aspect has three levels: excellent, good, and fair, which correspond to scores 2, 1, and 0, respectively. The usability score of a method is the sum of its four scores under the four aspects.

to estimate the doublet rates or select the threshold on doublet scores. For example, DoubletFinder suggests using the rates of heterotypic doublets and Poisson doublet formation as the respective lower and upper bounds of the expected doublet rate (Bloom, 2018; McGinnis et al., 2019a, 2019b); Scrublet recommends setting the doublet-score threshold in the middle of the two modes, which it expects to appear, in the doublet-score distribution (Wolock et al., 2019); Solo sets the doublet-score threshold to 0.5 by default (Bernstein et al., 2020). However, there lacks consensus or direct estimation of the doublet rate from scRNA-seq data. To address this issue, we suggest estimating the null distribution of doublet scores (of singlets) as a preceding step; with a reliable null distribution estimate, estimating the doublet rate would then become feasible (Efron and Hastie, 2016).

(2) How to distinguish homotypic doublets from singlets? Existing computational doublet-detection methods cannot well identify the homotypic doublets that have similar transcriptome profiles to those of singlets, likely due to the ways they generate artificial doublets (Lun et al., 2016; Gayoso and Shor, 2018; Bais and Kostka, 2020; DePasquale et al., 2019; McGinnis et al., 2019a, 2019b; Wolock et al., 2019; Bernstein et al., 2020). A possible direction is to extract and incorporate features that can distinguish homotypic doublets from singlets, such as the droplet library size.

(3) How to distinguish doublets from droplets contaminated by ambient mRNA? Ambient mRNA molecules are released from lysed cells into the cell suspension; they may enter droplets and contaminate the measured transcriptome profiles of those droplets. Similar to doublets, contaminated droplets by ambient mRNA also confound scRNA-seq data analysis (Luecken and Theis, 2019). Existing computational doublet-detection methods do not distinguish these two types of non-singlet droplets; instead, computational methods have been developed separately to detect contaminated droplets (Yang et al., 2020; Young and Behjati, 2020). Ideally, the single-cell field desires a computational method that can simultaneously remove all non-singlet droplets, including doublets, contaminated droplets, and empty droplets, from scRNA-seq data.

(4) How to improve doublet-detection algorithms regarding the use of artificial doublets? The majority of existing computational methods tackle the doublet-detection task as a binary classification problem (Table 1). To train a classification algorithm, they use original droplets in data and artificial doublets they simulate to represent "singlets" and "doublets," respectively. However, not all original droplets are singlets, because otherwise we would not need doublet detection. By neglecting differences between original droplets and singlets, existing methods do not supply their classification algorithms with quality training data, and a likely consequence is that their post-training classifiers would

Please cite this article in press as: Xi and Li, Benchmarking Computational Doublet-Detection Methods for Single-Cell RNA Sequencing Data, Cell Systems (2020), https://doi.org/10.1016/j.cels.2020.11.008

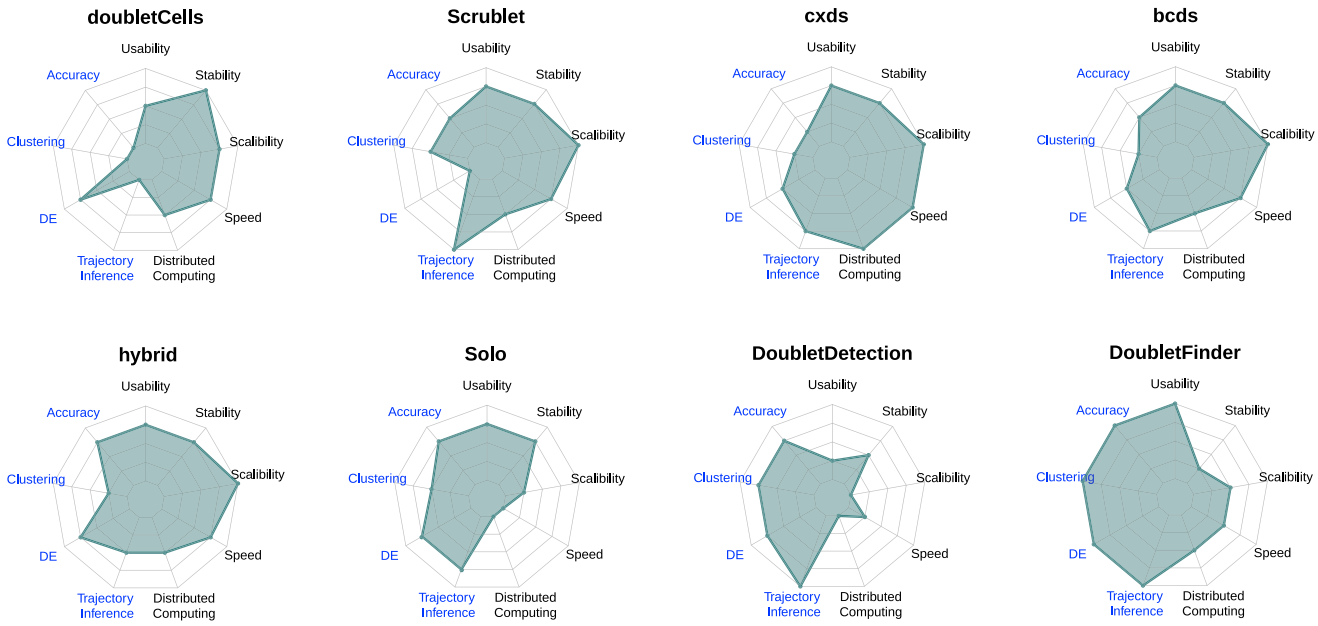


Figure 5. A Graphical Summary of Benchmark Results

The four aspects related to doublet-detection accuracy are marked in blue, while the other five aspects related to software implementation are marked in black.

be biased (Nettleton et al., 2010), and thus, miss a substantial number of doublets among original droplets. A possible remedy for this drawback is to filter out the likely doublets from the original droplets, e.g., by applying outlier detection methods (Domingues et al., 2018), before simulating artificial doublets and subsequently training a classification algorithm. An alternative remedy is to keep the training data but train a classification algorithm under the ''learning with noise labels'' machine-learning framework (Nettleton et al., 2010; Natarajan et al., 2013). Moreover, there are possible improvements to be made in the generation of artificial doublets. Instead of simply adding or averaging the gene expression profiles of two random droplets as done in existing methods, finer adjustments can be made to the mixing of two droplets so as to generate more realistic artificial doublets.

(5) How to ensemble doublet-detection methods? As a multifaceted problem, doublet detection can hardly be solved by one single computational method. This is due to the diversity of scRNA-seq datasets. The success of the method hybrid, an ensemble of two methods bcds and cxds, motivated us to think that ensembling reasonable and complementary methods, a technique widely used in machine learning (Dietterich, 2000; Hastie et al., 2009), may boost the accuracy of doublet detection. Tables S12 and S13 show the pairwise similarities of doublet-detection methods in terms of their doublet scores and identified doublets in the 16 real datasets. Seeing that the top-performing methods exhibited noticeable differences, we expect that there is room for using the ensemble technique to develop a more accurate doublet-detection method (see further discussion in the STAR Methods).

By dissecting existing doublet-detection methods, we found method performance highly dependent on the values of hyperparameters (also known as tuning parameters), if any. For example, DoubletFinder, Scrublet, and doubletCells all use the kNN algorithm to distinguish doublets from singlets; however, surprisingly, DoubletFinder outperformed the other two methods in most of our comparisons. A probable reason is that DoubletFinder optimizes several key hyperparameters of the kNN algorithm in a reasonable and data-driven way. For example, DoubletFinder selects the number of nearest neighbors k by maximizing the bimodality of the doublet score distribution. This advantage makes DoubletFinder adaptable to scRNA-seq datasets with distinct characteristics (Lun et al., 2016; McGinnis et al., 2019a, 2019b; Wolock et al., 2019). In contrast, Scrublet and doubletCells each assign a fixed default value to k, restricting their flexibility and generalizability (Lun et al., 2016; McGinnis et al., 2019a, 2019b; Wolock et al., 2019) (see further discussion in the STAR Methods). The choice of hyperparameter values is especially important for methods built upon complex algorithms. For example, bcds uses the gradient boosting algorithm (Bais and Kostka, 2020), a leading classification algorithm that has more hyperparameters than the simple kNN algorithm does (Chen and Guestrin, 2016); however, the additional complexity did not make bcds outperform DoubletFinder, probably due to the lack of hyperparameter optimization. This phenomenon emphasizes the importance for bioinformatics tools to optimize hyperparameter values in a scientific, data-driven way (Feurer and Hutter, 2019; Waring et al., 2020).

Ideally, doublet removal requires both experimental techniques and computational methods. If permitted, researchers may use an experimental technique and a computational method sequentially. That is, they first use an experimental technique, such as multiplexing, to filter out obvious doublets (e.g.,

the doublets formed by cells of different samples) and then apply a computational method to further screening for the remaining droplets that are likely doublets. Or they may combine the doublet scores assigned to each droplet by an experimental technique and a computational method, as proposed by the method Solo. This second approach requires the experimental technique to have a doublet scoring system (Bernstein et al., 2020).

In summary, computational doublet detection is critical for the quality control of scRNA-seq data analysis (Luecken and Theis, 2019). Our study is a comprehensive benchmark of currently available doublet-detection methods under a wide variety of biological and technical settings. Our study provides much-needed guidance to researchers in choosing appropriate doublet-detection methods for scRNA-seq data analysis. Our results also point out directions for further methodological development and improvement in computational doublet detection, an active area of bioinformatics research (Pierre-Luc, 2020).

STAR★METHODS

Detailed methods are provided in the online version of this paper and include the following:

- KEY RESOURCES TABLES
- RESOURCE AVAILABILITY
  - Lead Contact
  - Materials Availability
  - Data and Code Availability
- METHOD DETAILS
  - Real Data Preprocessing
  - Benchmark Environment and Parameter Settings
  - Measures of Doublet-Detection Accuracy
  - Simulation of scRNA-seq Datasets Containing Doublets
  - Experimental Settings Used in Benchmarking Simulations
  - DE Gene Analysis
  - Identification of Highly Variable Genes
  - Cell Clustering Analysis
  - Cell Trajectory Inference
  - Distributed Computing
  - Scalability, Stability, and Usability
  - Accuracy of Computational Doublet Detection in Relation to Experimental Techniques for Doublet Labeling
  - Pairwise Similarities of Computational Doublet-Detection Methods
  - Comparison of Hyperparameter Selection in kNN-Base Methods

SUPPLEMENTAL INFORMATION

Supplemental Information can be found online at https://doi.org/10.1016/j.cels.2020.11.008.

ACKNOWLEDGMENTS

We thank Dr. Bo Li at the University of Texas Southwestern Medical Center (https://www.lilab-utsw.org/research) for bringing our attention to the doublet-detection problem. We also appreciate the comments and feedback from our group members in the Junction of Statistics and Biology at UCLA (http://jsb.ucla.edu). This work was supported by NIH/NIGMS R01GM120507, NSF DBI-1846216, Sloan Research Fellowship, Johnson & Johnson WiSTEM2D Award, and UCLA DGSOM W.M. Keck Foundation Junior Faculty Award. MSC 2010 subject classifications: 62H20

AUTHOR CONTRIBUTIONS

N.M.X. and J.J.L. conceived the idea. N.M.X. conducted the bioinformatics analysis and wrote the paper. J.J.L. thoroughly reviewed and edited the paper.

DECLARATION OF INTERESTS

The authors declare no competing interests.

Received: June 28, 2020
Revised: October 6, 2020
Accepted: November 19, 2020
Published: December 17, 2020

REFERENCES

Allaire, J.J., et al. (2018). Reticulate: interface to Python. R Package Version *1*. https://rstudio.github.io/reticulate/.

Amezquita, R.A., Lun, A.T.L., Becht, E., Carey, V.J., Carpp, L.N., Geistlinger, L., Marini, F., Rue-Albrecht, K., Risso, D., Soneson, C., et al. (2020). Orchestrating single-cell analysis with Bioconductor. Nat. Methods *17*, 137–145, https://doi.org/10.1038/s41592-019-0654-x.

Andrews, T.S., and Hemberg, M. (2018). False signals induced by single-cell imputation. F1000Res *7*, 1740, https://doi.org/10.12688/f1000research.16613.2.

Bais, A.S., and Kostka, D. (2020). scds: computational annotation of doublets in single-cell RNA sequencing data. Bioinformatics *36*, 1150–1158, https://doi.org/10.1093/bioinformatics/btz698.

Bernstein, N.J., Fong, N.L., Lam, I., Roy, M.A., Hendrickson, D.G., and Kelley, D.R. (2020). Solo: doublet identification in single-cell RNA-Seq via semi-supervised deep learning. Cell Syst. *11*, 95–101.e5.

Blondel, V.D., Guillaume, J., Lambiotte, R., and Lefebvre, E. (2008). Fast unfolding of communities in large networks. J. Stat. Mech. *2008*, 10008, https://doi.org/10.1088/1742-5468/2008/10/P10008.

Bloom, J.D. (2018). Estimating the frequency of multiplets in single-cell RNA sequencing from cell-mixing experiments. PeerJ *6*, e5578.

Branco, P., Torgo, L., and Ribeiro, R.P. (2016). A survey of predictive modeling on imbalanced domains. ACM Comput. Surv. *49*, 1–50, https://doi.org/10.1145/2907070.

Butler, A., Hoffman, P., Smibert, P., Papalexi, E., and Satija, R. (2018). Integrating single-cell transcriptomic data across different conditions, technologies, and species. Nat. Biotechnol. *36*, 411–420.

Chen, G., Ning, B., and Shi, T. (2019). Single-cell RNA-Seq technologies and related computational data analysis. Front. Genet. *10*, 317.

Chen, T., and Guestrin, C. (2016). XGBoost: a scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794.

DePasquale, E.A.K., Schnell, D.J., Van Camp, P.J., Valiente-Alandí, Í., Blaxall, B.C., Grimes, H.L., Singh, H., and Salomonis, N. (2019). DoubletDecon: deconvoluting doublets from single-cell RNA-sequencing data. Cell Rep. *29*, 1718–1727.e8.

Dietterich, T.G. (2000). Ensemble methods in machine learning. In Lecture Notes in Computer Science (Springer), pp. 1–15.

Domingues, R., Filippone, M., Michiardi, P., and Zouaoui, J. (2018). A comparative evaluation of outlier detection algorithms: experiments and analyses. Pattern Recognit. *74*, 406–421.

Duò, A., Robinson, M.D., and Soneson, C. (2018). A systematic performance evaluation of clustering methods for single-cell RNA-seq data. F1000Res *7*, 1141.

Durinck, S., Spellman, P.T., Birney, E., and Huber, W. (2009). Mapping identifiers for the integration of genomic datasets with the R/Bioconductor package biomaRt. Nat. Protoc. *4*, 1184–1191.

Edgar, R., Domrachev, M., and Lash, A.E. (2002). Gene Expression Omnibus: NCBI gene expression and hybridization array data repository. Nucleic Acids Res. *30*, 207–210.

Efron, B., and Hastie, T. (2016). Computer Age Statistical Inference (Cambridge University Press).

Ester, M., Kriegel, H.-P., Sander, J., and Xiaowei, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. KDD'96: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, pp. 226–231.

Fay, M.P., and Proschan, M.A. (2010). Wilcoxon-Mann-Whitney or t-test? On assumptions for hypothesis tests and multiple interpretations of decision rules. Stat. Surv. *4*, 1–39.

Feng, C., Liu, S., Zhang, H., Guan, R., Li, D., Zhou, F., Liang, Y., and Feng, X. (2020). Dimension reduction and clustering models for single-cell RNA sequencing data: a comparative study. Int. J. Mol. Sci. *21*, https://doi.org/10.3390/ijms21062181.

Feurer, M., and Hutter, F. (2019). Hyperparameter optimization. In Automated Machine Learning: Methods, Systems, Challenges, F. Hutter, L. Kotthoff, and J. Vanschoren, eds. (Springer International Publishing), pp. 3–33.

Finak, G., McDavid, A., Yajima, M., Deng, J., Gersuk, V., Shalek, A.K., Slichter, C.K., Miller, H.W., McElrath, M.J., Prlic, M., et al. (2015). MAST: a flexible statistical framework for assessing transcriptional changes and characterizing heterogeneity in single-cell RNA sequencing data. Genome Biol. *16*, 278.

Gayoso, A., and Shor, J. (2018). DoubletDetection (Zenodo). https://doi.org/10.5281/zenodo.2678042.

Github. (2020). DoubletDecon. https://github.com/EDePasquale/DoubletDecon/issues.

Gong, T., and Szustakowski, J.D. (2013). DeconRNASeq: a statistical framework for deconvolution of heterogeneous tissue samples based on mRNA-Seq data. Bioinformatics *29*, 1083–1085.

Grau, J., Grosse, I., and Keilwagen, J. (2015). PRROC: computing and visualizing precision-recall and receiver operating characteristic curves in R. Bioinformatics *31*, 2595–2597.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition (Springer Science and Business Media).

Hastie, T.J., and Tibshirani, R.J. (1990). Generalized Additive Models (CRC Press).

Herring, C.A., Chen, B., McKinley, E.T., and Lau, K.S. (2018). Single-cell computational strategies for lineage reconstruction in tissue systems. Cell. Mol. Gastroenterol. Hepatol. *5*, 539–548.

Hwang, B., Lee, J.H., and Bang, D. (2018). Single-cell RNA sequencing technologies and bioinformatics pipelines. Exp. Mol. Med. *50*, 96.

Ji, Z., and Ji, H. (2016). TSCAN: pseudo-time reconstruction and evaluation in single-cell RNA-seq analysis. Nucleic Acids Res. *44*, e117.

Kang, H.M., Subramaniam, M., Targ, S., Nguyen, M., Maliskova, L., McCarthy, E., Wan, E., Wong, S., Byrnes, L., Lanata, C.M., et al. (2018). Multiplexed droplet single-cell RNA-sequencing using natural genetic variation. Nat. Biotechnol. *36*, 89–94.

Kolodziejczyk, A.A., Kim, J.K., Svensson, V., Marioni, J.C., and Teichmann, S.A. (2015). The technology and biology of single-cell RNA sequencing. Mol. Cell *58*, 610–620.

Lähnemann, D., Köster, J., Szczurek, E., McCarthy, D.J., Hicks, S.C., Robinson, M.D., Vallejos, C.A., Campbell, K.R., Beerenwinkel, N., Mahfouz, A., et al. (2020). Eleven grand challenges in single-cell data science. Genome Biol. *21*, 31.

Li, W.V., and Li, J.J. (2018). An accurate and robust imputation method scImpute for single-cell RNA-seq data. Nat. Commun. *9*, 997.

Li, W.V., and Li, J.J. (2019). A statistical simulator scDesign for rational scRNA-seq experimental design. Bioinformatics *35*, i41–i50.

Liu, S., and Trapnell, C. (2016). Single-cell transcriptome sequencing: recent advances and remaining challenges. F1000Res *5*, https://doi.org/10.12688/f1000research.7223.1.

Lopez, R., Regier, J., Cole, M.B., Jordan, M.I., and Yosef, N. (2018). Deep generative modeling for single-cell transcriptomics. Nat. Methods *15*, 1053–1058.

Love, M.I., Huber, W., and Anders, S. (2014). Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. Genome Biol. *15*, 550.

Luecken, M.D., and Theis, F.J. (2019). Current best practices in single-cell RNA-seq analysis: a tutorial. Mol. Syst. Biol. *15*, e8746.

Lun, A.T.L., McCarthy, D.J., and Marioni, J.C. (2016). A step-by-step workflow for low-level analysis of single-cell RNA-seq data with Bioconductor. F1000Res *5*, 2122, https://doi.org/10.12688/f1000research.9501.2.

Mangul, S., Martin, L.S., Eskin, E., and Blekhman, R. (2019). Improving the usability and archival stability of bioinformatics software. Genome Biol. *20*, 47.

McGinnis, C.S., Murrow, L.M., and Gartner, Z.J. (2019a). DoubletFinder: doublet detection in single-cell RNA sequencing data using artificial nearest neighbors. Cell Syst. *8*, 329–337.e4.

McGinnis, C.S., Patterson, D.M., Winkler, J., Conrad, D.N., Hein, M.Y., Srivastava, V., Hu, J.L., Murrow, L.M., Weissman, J.S., Werb, Z., et al. (2019b). Multi-seq: sample multiplexing for single-cell RNA sequencing using lipid-tagged indices. Nat. Methods *16*, 619–626, https://doi.org/10.1038/s41592-019-0433-8.

Natarajan, N., et al. (2013). Learning with noisy labels. In Adv. Neural Inf. Process. Syst. 26, C.J.C. Burges, et al., eds. (Curran Associates, Inc.), pp. 1196–1204.

Nettleton, D.F., Orriols-Puig, A., and Fornells, A. (2010). A study of the effect of different types of noise on the precision of supervised learning techniques. Artif. Intell. Rev. *33*, 275–306.

Pfister, R., Schwarz, K.A., Janczyk, M., Dale, R., and Freeman, J.B. (2013). Good things peak in pairs: a note on the bimodality coefficient. Front. Psychol. *4*, 700.

Pierre-Luc. (2020). scDblFinder. (GitHub). https://github.com/plger/scDblFinder.

Regev, A., Teichmann, S.A., Lander, E.S., Amit, I., Benoist, C., Birney, E., Bodenmiller, B., Campbell, P., Carninci, P., Clatworthy, M., et al. (2017). The human cell atlas. eLife *6*, e27041.

Risso, D., Perraudeau, F., Gribkova, S., Dudoit, S., and Vert, J.P. (2018). A general and flexible method for signal extraction from single-cell RNA-seq data. Nat. Commun. *9*, 284.

Saelens, W., Cannoodt, R., Todorov, H., Saeys, Y., et al. (2019). A comparison of single-cell trajectory inference methods: towards more accurate and robust tools. bioRXiv. https://doi.org/10.1101/276907.

Saito, T., and Rehmsmeier, M. (2015). The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets. PLoS One *10*, e0118432.

Saliba, A.E., Westermann, A.J., Gorski, S.A., and Vogel, J. (2014). Single-cell RNA-seq: advances and future challenges. Nucleic Acids Res. *42*, 8845–8860, https://doi.org/10.1093/nar/gku555.

Stoeckius, M., Zheng, S., Houck-Loomis, B., Hao, S., Yeung, B.Z., Mauck, W.M., Smibert, P., and Satija, R. (2018). Cell Hashing with barcoded antibodies enables multiplexing and doublet detection for single cell genomics. Genome Biol. *19*, 224.

Street, K., Risso, D., Fletcher, R.B., Das, D., Ngai, J., Yosef, N., Purdom, E., and Dudoit, S. (2018). Slingshot: cell lineage and pseudotime inference for single-cell transcriptomics. BMC Genomics *19*, 477, https://doi.org/10.1186/s12864-018-4772-0.

Stuart, T., Butler, A., Hoffman, P., Hafemeister, C., Papalexi, E., Mauck, W.M., Hao, Y., Stoeckius, M., Smibert, P., and Satija, R. (2019). Comprehensive integration of single-cell data. Cell *177*, 1888–1902.e21.

Tian, L., Dong, X., Freytag, S., Lê Cao, K.A., Su, S., JalalAbadi, A., Amann-Zalcenstein, D., Weber, T.S., Seidi, A., Jabbari, J.S., et al. (2019). Benchmarking single cell RNA-sequencing analysis pipelines using mixture control experiments. Nat. Methods *16*, 479–487.

Trapnell, C., Cacchiarelli, D., Grimsby, J., Pokharel, P., Li, S., Morse, M., Lennon, N.J., Livak, K.J., Mikkelsen, T.S., and Rinn, J.L. (2014). The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single cells. Nat. Biotechnol. *32*, 381–386.

Vallejos, C.A., Marioni, J.C., and Richardson, S. (2015). BASiCS: bayesian analysis of single-cell sequencing data. PLoS Comp. Biol. *11*, e1004333.

van Dijk, D., Sharma, R., Nainys, J., Yim, K., Kathail, P., Carr, A.J., Burdziak, C., Moon, K.R., Chaffer, C.L., Pattabiraman, D., et al. (2018). Recovering gene interactions from single-cell data using data diffusion. Cell *174*, 716–729.e27, https://doi.org/10.1016/j.cell.2018.05.061.

Wang, T., Li, B., Nelson, C.E., and Nabavi, S. (2019). Comparative analysis of differential gene expression analysis tools for single-cell RNA sequencing data. BMC Bioinformatics *20*, 40.

Waring, J., Lindvall, C., and Umeton, R. (2020). Automated machine learning: review of the state-of-the-art and opportunities for healthcare. Artif. Intell. Med. *104*, 101822.

Weber, L.M., Saelens, W., Cannoodt, R., Soneson, C., Hapfelmeier, A., Gardner, P.P., Boulesteix, A.L., Saeys, Y., and Robinson, M.D. (2019). Essential guidelines for computational method benchmarking. Genome Biol. *20*, 125.

Wolock, S.L., Lopez, R., and Klein, A.M. (2019). Scrublet: computational identification of cell doublets in single-cell transcriptomic data. Cell Syst. *8*, 281–291.e9.

Yang, S., Corbett, S.E., Koga, Y., Wang, Z., Johnson, W.E., Yajima, M., and Campbell, J.D. (2020). Decontamination of ambient RNA in single-cell RNA-seq with DecontX. Genome Biol. *21*, 57, https://doi.org/10.1186/s13059-020-1950-6.

Yip, S.H., Sham, P.C., and Wang, J. (2019). Evaluation of tools for highly variable gene discovery from single-cell RNA-seq data. Brief. Bioinform. *20*, 1583–1589.

Young, M.D., and Behjati, S. (2020). SoupX removes ambient RNA contamination from droplet based single cell RNA sequencing data. bioRxiv https://www.biorxiv.org/content/10.1101/303727v2.abstract.

Zappia, L., Phipson, B., and Oshlack, A. (2017). Splatter: simulation of single-cell RNA sequencing data. Genome Biol. *18*, 174.

Zappia, L., Phipson, B., and Oshlack, A. (2018). Exploring the single-cell RNA-seq analysis landscape with the scRNA-tools database. PLoS Comp. Biol. *14*, e1006245.

Zheng, G.X.Y., Terry, J.M., Belgrader, P., Ryvkin, P., Bent, Z.W., Wilson, R., Ziraldo, S.B., Wheeler, T.D., McDermott, G.P., Zhu, J., et al. (2017). Massively parallel digital transcriptional profiling of single cells. Nat. Commun. *8*, 14049.

Please cite this article in press as: Xi and Li, Benchmarking Computational Doublet-Detection Methods for Single-Cell RNA Sequencing Data, Cell Systems (2020), https://doi.org/10.1016/j.cels.2020.11.008

STAR★METHODS

KEY RESOURCES TABLES

REAGENT or RESOURCE	SOURCE	IDENTIFIER
Software and Algorithms		
scDesign	(Li and Li, 2019)	https://github.com/Vivianstats/scDesign
Seurat	(Butler et al., 2018; Stuart et al., 2019)	https://satijalab.org/seurat/
Splatter	(Zappia et al., 2017)	https://github.com/Oshlack/splatter
biomaRt	(Durinck et al., 2009)	https://github.com/grimbough/biomaRt
reticulate	(Allaire, 2018)	https://github.com/rstudio/reticulate
PRROC	(Grau et al., 2015)	https://cran.r-project.org/web/packages/PRROC/index.html
proxy	https://cran.r-project.org/web/packages/proxy/proxy.pdf	https://cran.r-project.org/web/packages/proxy/index.html
dbscan	(Ester et al., 1996)	https://github.com/mhahsler/dbscan
Slingshot	(Street et al., 2018)	https://github.com/kstreet13/slingshot
Computational doublet-detection methods	Table 1	https://github.com/xnnba1984/Doublet-Detection-Benchmark
Benchmark of computational doublet-detection methods	This paper	https://github.com/xnnba1984/Doublet-Detection-Benchmark
Deposited Data		
Real datasets	Table S1	https://zenodo.org/record/4062232#.X6GordD0laQ
Simulation datasets	This paper	https://zenodo.org/record/4062232#.X6GordD0laQ

RESOURCE AVAILABILITY

Lead Contact

Further information and requests for resources and reagents should be directed to and will be fulfilled by the Lead Contact, Jingyi Jessica Li (jli@stat.ucla.edu).

Materials Availability

This study did not generate new materials.

Data and Code Availability

- The real and synthetic scRNA-seq datasets used in this study have been deposited at Zenodo repository and are publicly available at https://zenodo.org/record/4062232#.X3YR9Hn0kuU%E3%80%82.
- The source code used in this study is publicly available at https://github.com/xnnba1984/Doublet-Detection-Benchmark.
- The scripts used to generate the figures reported in this paper are available at https://github.com/xnnba1984/Doublet-Detection-Benchmark.
- Any additional information required to reproduce this work is available from the Lead Contact.

METHOD DETAILS

Real Data Preprocessing

Whenever preprocessed datasets were available, they were directly used in this study. Otherwise, datasets were preprocessed in the same way as in the original studies in which they were generated. In every dataset, genes and droplets were removed if they had no reads in any droplets and any genes, respectively. Below is the preprocessing detail for every dataset.

pbmc-ch (Stoeckius et al., 2018): human peripheral blood mononuclear cells (PBMCs) from eight donors. Doublets were annotated by cell hashing with CD45 as the hashing antibody. This dataset is available at https://www.dropbox.com/sh/ntc33ium7cg1za1/AAD_8XIDmu4F7lJ-5sp-rGFYa?dl=0 in files *pbmc_hto_mtx.rds* and *pbmc_umi_mtx.rds*. Its preprocessing pipeline is available at

https://satijalab.org/seurat/v3.1/hashing_vignette.html, including an instruction about how to extract the doublet annotation.

cline-ch (Stoeckius et al., 2018): four human cell lines HEK, K562, KG1, and THP1. Doublets were annotated by cell hashing with CD29 and CD45 as the hashing antibodies. The access URL and preprocessing pipeline of this dataset are the same as those of the pbmc-ch dataset. The dataset is in files *hto12_hto_mtx.rds* and *hto12_umi_mtx.rds*.

Mkidney-ch (Bernstein et al., 2020): dissociated mouse kidney cells. Doublets were annotated by cell hashing with cholesterol modified oligos (CMOs) as the hashing antibodies. The raw count matrix and doublet annotations were downloaded from the Gene Expression Omnibus (GEO) (Edgar et al., 2002) with the accession GSE140262.

hm-12k and **hm-6k** (Zheng et al., 2017): two mixtures of human HEK293T and mouse NIH3T3 cells with 12,000 and 6000 droplets respectively. The raw count matrices were downloaded from
https://support.10xgenomics.com/single-cell-gene-expression/datasets/2.1.0/hgmm_12k and https://support.10xgenomics.com/single-cell-gene-expression/datasets/2.1.0/hgmm_6k

A droplet was annotated as a doublet if its barcode was associated with both human and mouse. Mouse genes were mapped into their human orthologs using R package *biomaRt (*Durinck et al., 2009) (v 2.44.1). Then each pair of human and mouse count matrices was concatenated into each of the two datasets.

pbmc-1A-dm, **pbmc-1B-dm**, and **pbmc-1C-dm** (Kang et al., 2018): three samples of PBMCs from systemic lupus erythematosus (SLE) patients. Droplets were sequenced immediately after thawing. Doublets were annotated by demuxlet (Kang et al., 2018). The raw count matrix and doublet annotations were downloaded from the GEO with the accession GSE96583.

pbmc-2ctrl-dm and **pbmc-2stiml-dm** (Kang et al., 2018): two samples of PBMCs from SLE patients. Droplets were sequenced after being cultured for six hours following thawing, with (pbmc-2stiml-dm) or without (pbmc-2ctrl-dm) IFN-beta stimulation. Doublets were annotated by demuxlet. The raw count matrix and doublet annotations were downloaded from the GEO with the accession GSE96583.

J293t-dm (Kang et al., 2018): a mixture of human Jurkat and HEK293T cell lines. Doublets were annotated by demuxlet. The raw count matrix was downloaded from
https://ucsf.app.box.com/s/vg1bycvsjgyg63gkqsputprq5rxzjl6k/file/220975201845.
Doublet annotations were obtained from
https://ucsf.app.box.com/s/vg1bycvsjgyg63gkqsputprq5rxzjl6k/file/220974993609

pdx-MULTI (McGinnis et al., 2019a, 2019b): a mixture of human breast cancer cells and mouse immune cells from a patient-derived xenograft (PDX) mouse model. Doublets were annotated by MULTI-seq (McGinnis et al., 2019a, 2019b). The dataset was downloaded from the GEO with the accession GSE129578. Doublet were annotated by following the data processing pipeline available at https://github.com/chris-mcginnis-ucsf/MULTI-seq.

HMEC-orig-MULTI and **HMEC-rep-MULTI** (McGinnis et al., 2019a, 2019b): human primary mammary epithelial cells (HMECs) with HMEC-orig-MULTI as the original sample and HMEC-rep-MULTI as a technical replica. The GEO accession and preprocessing pipeline of this dataset are the same as those of the pdx-MULTI dataset.

HEK-HMEC-MULTI (McGinnis et al., 2019a, 2019b): a mixture of human HEK293Ts and HMECs. The GEO accession and preprocessing pipeline of this dataset are the same as those of the pdx-MULTI dataset.

nuc-MULTI (McGinnis et al., 2019a, 2019b): a mixture of purified nuclei from human HEK293Ts, Jurkats, and mouse embryonic fibroblasts (MEFs). The GEO accession and preprocessing pipeline of this dataset are the same as those of the pdx-MULTI dataset. Mouse genes were mapped into their human orthologs using R package *biomaRt* (v 2.44.1).

Benchmark Environment and Parameter Settings

All doublet-detection methods were executed on a server with two Intel(R) Xeon(R) E5-2687W v4 CPUs, 256GB memory, and Ubuntu 18.04 system. An Nvidia(R) Geforce(R) RTX 2080 Ti GPU was used to accelerate the execution of the Solo method as suggested (Bernstein et al., 2020). The parameters of doublet-detection methods were set to their recommended values or default values if no recommendation was available. The latest version of each method (by September 2020; Table 1) was used. Random seeds were fixed and saved in our code to ensure reproducibility. The detailed configuration for each method is summarized below.

doubletCells: The method was executed by following the instruction at
https://bioconductor.statistik.tu-dortmund.de/packages/3.8/workflows/vignettes/simpleSingleCell/inst/doc/work-6-doublet.html.

Doublet scores were obtained from the *dblCells* function in R package *scran* (v 1.16.0) with parameters set to default.

Scrublet: R package *reticulate* (v 1.16) was used to execute the python module *scrublet* (v 0.2.1). The parameters were set by following the instruction at
https://github.com/AllonKleinLab/scrublet/blob/master/examples/scrublet_basics.ipynb.

Doublet scores were obtained from the function *Scrublet.scrub_doublets*. cxds, bcds and hybrid: These three methods were executed by following the instructions at https://github.com/kostkalab/scds.

Doublet scores were obtained from the functions *cxds*, *bcds* and *cxds_bcds_hybrid* in R package *scds* (v 1.2.0) with parameters set to default.

DoubletDetection: R package *reticulate* (v 1.16) was used to execute the python module *doubletdetection*. The parameters were set by following the instruction at
https://nbviewer.jupyter.org/github/JonathanShor/DoubletDetection/blob/master/tests/notebooks/PBMC_8k_vignette.ipynb.

Please cite this article in press as: Xi and Li, Benchmarking Computational Doublet-Detection Methods for Single-Cell RNA Sequencing Data, Cell Systems (2020), https://doi.org/10.1016/j.cels.2020.11.008

The parameter *n_iters* was set to 5, as larger values were found to increase the running time significantly, but with little improvement in performance. Doublet scores were obtained from the function *doubletdetection.BoostClassifier.fit*.

DoubletFinder: The method was executed by following the instruction at https://github.com/chris-mcginnis-ucsf/DoubletFinder.

Doublet scores were obtained from the function *doubletFinder_v3* in R package *DoubletFinder* (v 2.0.3) with parameters set to default.

DoubletDecon: The method was executed by following the instruction at https://github.com/EDePasquale/DoubletDecon.

Doublet predictions were obtained from the function *Main_Doublet_Decon* in R package *DoubletDecon* (v 1.1.5) with parameters set to default.

Solo: The method was executed by following the instruction at the GitHub repositoryhttps://github.com/calico/Solo

Every scRNA-seq count matrix was transformed into the *loom* format as required by the method. The parameters were set the same as those in the file *Solo_params_example.json*, which was downloaded from the GitHub repository. Doublet scores were obtained from the file *softmax_scores.npy*.

Measures of Doublet-Detection Accuracy

Methodologically, computational doublet-detection methods employ binary classification algorithms to distinguish between two classes: singlets and doublets. AUPRC and AUROC, two measures of the overall accuracy of a binary classification algorithm, were used to evaluate the overall doublet-detection accuracy of each method. These two measures were calculated using the functions *pr.curve* and *roc.curve* in R package *PRROC* (v 1.3.1). Both functions input two vectors: the predicted doublet scores of true singlets and those of true doublets, and they output AUPRC and AUROC, one value each.

Simulation of scRNA-seq Datasets Containing Doublets

All synthetic scRNA-seq datasets used in this study were generated in two steps. In Step 1, singlets in each dataset were generated by scDesign (Li and Li, 2019), which estimated a generative model of gene expression profiles from a real scRNA-seq dataset (cell type: HEK293t; protocol: 10x Genomics; gene number: 18760). The detailed experimental settings are described in the next subsection. In Step 2, given the number of singlets and a pre-specified doublet rate (i.e., the proportion of doublets among all droplets), the corresponding number of doublets were generated by random pairing of singlets. In detail, two randomly sampled singlets had their gene expression profiles (in UMI counts) averaged by gene, and that averaged profile is called a prototype doublet. For each of the 16 real scRNA-seq datasets, a doublet-to-singlet size ratio, defined as (average doublet library size)/(average singlet library size), was calculated. Then the library size of each prototype doublet was multiplied by a factor sampled from a normal distribution, whose mean and standard division were set to the mean and standard deviation of the 16 doublet-to-singlet size ratios. This scaling step turned prototype doublets into doublets, so that the doublet-to-singlet size ratios in the synthetic data were similar to those in the real data. Finally, the singlets used to generate doublets were removed. In mathematical terms, if X singlets were generated in Step 1 and the doublet rate was Y (a value between 0 and 1), then after Step 2 the numbers of doublets and singlets would be XY/(1+Y) and X(1-Y)/(1+Y), respectively, both rounded to the nearest integers. For example, if 1000 singlets were generated in Step 1 and the doublet rate was 20%, the numbers of doublets and singlets in the final dataset would be 167 and 667, respectively, making a total number of 834 droplets.

Experimental Settings Used in Benchmarking Simulations

80 scRNA-seq datasets were generated by scDesign to benchmark doublet-detection methods in four aspects: varying doublet rates, sequencing depths (i.e., per-cell library sizes), cell types, and between-cell-type heterogeneity levels.

- 20 synthetic datasets were generated with doublet rates increasing from 2% to 40% by a step size of 2%. The per-cell library size was set to 2000 UMI counts. All datasets contained two cell types. Based on the data generation scheme described in the last subsection, 500 singlets were generated for each cell type in Step 1. In Step 2, doublets were introduced based on each doublet rate, and the singlets used to generate doublets were removed.
- 20 synthetic datasets were generated with per-cell library sizes increasing from 500 to 10,000 UMI counts by a step size of 500 counts. All datasets contained two cell types. Based on the data generation scheme described in the last subsection, 500 singlets were generated for each cell type in Step 1. In Step 2, doublets were introduced based on a 20% doublet rate, and the singlets used to generate doublets were removed.
- 19 synthetic datasets were generated with numbers of cell types increasing from 2 to 20 by a step size of 1. The per-cell library size was set to 2000 UMI counts. Based on the data generation scheme described in the last subsection, 500 singlets were generated for each cell type in Step 1. In Step 2, doublets were introduced based on a 20% doublet rate, and the singlets used to generate doublets were removed.
- 21 synthetic datasets were generated with varying heterogeneity levels between two cell types. The heterogeneity level was controlled by four parameters (pUp, pDown, fU, and fL) in scDesign. Specifically, pUp and pDown denote the proportions of up- and down-regulated genes, and fU and fL define the upper and lower bounds of fold changes in the expression levels of DE genes. The following parameter combinations were used to generate 21 heterogeneity levels:

Level 1: pUp = 0.010, pDown = 0.010, fU = 1.0, and fL = 0.5;

Please cite this article in press as: Xi and Li, Benchmarking Computational Doublet-Detection Methods for Single-Cell RNA Sequencing Data, Cell Systems (2020), https://doi.org/10.1016/j.cels.2020.11.008

Level 2: pUp = 0.012, pDown = 0.012, fU = 1.2, and fL = 0.6;

…

Level 21: pUp = 0.050, pDown = 0.050, fU = 5.0, and fL = 2.5.

At all heterogeneity levels, the per-cell library size was set to 2000 UMI counts. Based on the data generation scheme described in the last subsection, 500 singlets were generated for each cell type in Step 1. In Step 2, doublets were introduced based on a 20% doublet rate, and the singlets used to generate doublets were removed.

DE Gene Analysis

One synthetic scRNA-seq dataset was generated by scDesign to have two cell types. The per-cell library size was 10,000 UMI counts. The pUp and pDown parameters in scDesign were both set to 0.03, suggesting that a total of 6% of genes were DE between the two cell types (3% up-expressed and 3% down-expressed). The fU and fL parameters in scDesign (i.e., the upper and lower bound of fold changes for DE genes) were set to 3 and 1.5, respectively. Based on the data generation scheme described in the Subsection "Simulation of scRNA-seq datasets containing doublets," 500 singlets were generated for each cell type in Step 1. In Step 2, doublets were introduced based on the 40% doublet rate, and the singlets used to generate doublets were removed. Three DE methods—DESeq2 (Love et al., 2014), MAST (Finak et al., 2015), and the Wilcoxon rank-sum test (Fay and Proschan, 2010) implemented in the R package Seurat (v 3.1.5) (Butler et al., 2018; Stuart et al., 2019)—were applied to this dataset ("contaminated dataset" containing both singlets and doublets), its clean version without doublets ("clean dataset" only containing singlets), and its post-doublet-detection version after each doublet-detection method was applied (the top 40% droplets that received the highest doublet scores were removed). After each DE method was applied to every dataset, genes whose Bonferroni-corrected p-values did not exceed 0.05 were identified as DE. Three accuracy measures—precision, recall, and TNR—were calculated for every set of identified DE genes. For each DE method, its accuracy on the contaminated dataset and the clean dataset were used as the negative and positive controls, respectively, for benchmarking its accuracy on the post-doublet-detection datasets (Figures 2B and 2C).

Identification of Highly Variable Genes

Three synthetic datasets were generated with 10%, 20%, and 40% doublet rates, respectively. The per-cell library size was set to 2000 UMI counts. All datasets contained two cell types. Based on the data generation scheme described in the Subsection "Simulation of scRNA-seq datasets containing doublets," 500 singlets were generated for each cell type in Step 1. In Step 2, doublets were introduced based on each doublet rate, and the singlets used to generate doublets were removed. To identify the highly variable genes (HVGs), we applied the function *FindVariableFeatures* in R package *Seurat* (v 3.1.5) with default parameters to the three datasets ("contaminated datasets" containing both singlets and doublets; one dataset per doublet rate), their clean versions without doublets ("clean datasets" only containing singlets), and their post-doublet-detection version after each doublet-detection method was applied (the top 10%, 20%, or 40% droplets that received the highest doublet scores were removed, and the removal percentage was set to the doublet rate). We refer to the identified HVGs as contaminated HVGs, clean HVGs, and post-doublet-detection HVGs, respectively. The Jaccard index between two sets of HVGs was calculated by the function *simi* in R package *proxy* (v 0.4-24) (Figure 2D).

Cell Clustering Analysis

Three synthetic scRNA-seq datasets were generated by scDesign to have four, six, and eight cell types. The per-cell library size was 2000 UMI counts. Based on the data generation scheme described in the Subsection "Simulation of scRNA-seq datasets containing doublets," 500 singlets were generated for each cell type in Step 1. In Step 2, doublets were introduced based on a 20% doublet rate, and the singlets used to generate doublets were removed. The heterogeneity between cell types was determined by the default pUp, pDown, fU, and fL parameters in scDesign. After each doublet-detection method was applied to each dataset, the top x% of droplets, which received the highest doublet scores (with the removal percentage x% ranging from 0% to 25% by a step size of 1%), were removed; then two clustering algorithms—Louvain clustering implemented in R package *Seurat* (v 3.1.5) and DBSCAN (Ester et al., 1996) implemented in R package *dbscan* (v 1.1–5)—were used to identify cell clusters. Finally, the numbers of cell clusters were compared with the numbers of cell types to evaluate the effectiveness of doublet removal (Figures 2E and S2A). Whenever the number of cell clusters matched the number of cell types, the proportion of singlets among the remaining droplets was used to measure each doublet-detection method's capacity for removing homotypic doublets (Figures 2F and S2B). In the example of four cell types, if a doublet-detection method (given a clustering algorithm) correctly led to four cell clusters under six removal percentages, then a proportion of singlets was calculated for each of the 24 clusters (four clusters times six removal percentages), resulting in 24 proportions.

Cell Trajectory Inference

Two scRNA-seq datasets were generated by Splatter (Zappia et al., 2017) to have cell trajectories. Both datasets contained 1000 genes. In Step 1 of the data generation scheme described in the Subsection "Simulation of scRNA-seq datasets containing doublets," the first dataset had 500 singlets following a bifurcating trajectory, whose two branches had 250 singlets each, and the second dataset had 1000 singlets from a conjunction of three sequential trajectories, two of which had 333 singlets and the other had 334 singlets. In Step 2 for both datasets, doublets were introduced based on a 20% doublet rate, and the singlets used to generate dou-

Please cite this article in press as: Xi and Li, Benchmarking Computational Doublet-Detection Methods for Single-Cell RNA Sequencing Data, Cell Systems (2020), https://doi.org/10.1016/j.cels.2020.11.008

blets were removed. Parameters in Splatter were set to default except for de.prob and de.facLoc, which were set to 0.5 and 0.2, respectively. Each dataset was expanded into a suite, including its original version ("contaminated dataset"), clean version without doublets ("clean dataset"), and its post-doublet-detection version after each doublet-detection method was applied (the top 20% droplets that received the highest doublet scores were removed). For the first suite of datasets, cell trajectories were constructed by Slingshot (Street et al., 2018) based on the pipeline available at https://github.com/kstreet13/slingshot/blob/master/vignettes/vignette.Rmd.

For the second suite of datasets, the minimum spanning tree (MST) algorithm implemented in R package *slingshot* (v 1.6.1) was used to construct cell trajectories. The trajectories constructed from the contaminated dataset and the clean dataset were used as the negative and positive controls, respectively, for benchmarking the trajectories inferred from the post-doublet-detection datasets (Figures 3A and 3B).

In the temporally DE genes analysis, a scRNA-seq dataset with a single trajectory was generated by following the Slingshot pipeline available at https://github.com/kstreet13/slingshot/blob/master/vignettes/vignette.Rmd.

This dataset contained 750 genes, whose temporal expression dynamics were categorized into four types: 500 stable genes with unchanged mean expression levels, 100 activated genes with increasing mean expression levels, 100 deactivated genes with decreasing mean expression levels, and 50 transient genes with mean expression levels first increasing and then decreasing, along the trajectory. The genes of the latter three types were defined as temporally DE genes. The mean expression levels of all 750 genes were specified by following the Slingshot pipeline. The per-cell library sizes were sampled from a negative binomial distribution with mean 1875 and dispersion 4. In the generation of a singlet, the 750 gene expression levels were sampled from a multinomial distribution with the number of trials as the (randomly sampled) per-cell library size and the probability of success as the 750 genes' normalized mean expression levels (summing up to 1). Following this, 300 singlets were generated in Step 1 of the data generation scheme described in the Subsection "Simulation of scRNA-seq datasets containing doublets." In Step 2, doublets were introduced based on a 20% doublet rate, and the singlets used to generate doublets were removed.

After data generation, the pseudotime of each droplet was inferred by Slingshot and TSCAN on this dataset ("contaminated data"), its clean version without doublets ("clean data"), and its post-doublet-detection version after each doublet-detection method was applied (the top 20% droplets that received the highest doublet scores were removed). Then for each dataset, we regressed each gene's expression levels in all droplets on the inferred pseudotime of the same droplets by the general additive model (GAM), which was implemented in the R function *gam*, and obtained a p-value. As a result, the genes with Bonferroni-corrected p-values under 0.05 were identified as temporally DE genes. Three accuracy measures—precision, recall, and TNR—were calculated for every set of identified temporally DE genes. The accuracy on the contaminated data and the clean data were used as the negative and positive controls, respectively, for benchmarking the accuracy on the post-doublet-detection data obtained by each doublet-detection method (Figures 3C and 3D).

Distributed Computing

We used two real scRNA-seq datasets pbmc-ch and pbmc-2ctrl-dm to compare the performance of doublet-detection methods under distributed computing. These two datasets are relatively large in our real data collection, containing 15,272 and 13,913 droplets (Table 1). For each doublet-detection method, its accuracy (AUPRC and AUROC) on the original datasets were used as the baselines. Next, the original dataset was randomly split into two, four, six, eight, and ten equally-sized batches for distributed computing. For every number of batches, each doublet-detection method was executed on each batch separately, the resulting doublet scores were concatenated across batches, and AUPRC and AUROC were calculated for the concatenated doublet scores and compared with the baselines (Figures 4A and 4B).

Scalability, Stability, and Usability

25 synthetic scRNA-seq datasets with varying numbers of droplets were generated by scDesign to examine the scalability of doublet-detection methods. Specifically, the number of genes was fixed to 5000, and the number of droplets increased from 400 to 10,000, with a step size of 400. Each doublet-detection method was executed on the 25 datasets, and the relationship between its running time and the number of droplets was plotted in Figure 4E.

Two real datasets, pbmc-ch and pbmc-2ctrl-dm, were used to evaluate the stability of doublet-detection methods. From each dataset, 20 subsets were generated by randomly subsampling 90% of droplets and 90% of genes. Each doublet-detection method was executed on all these subsets, and its stability was shown by the distributions of the resulting AUPRC and AUROC across subsets (Figure 4F).

Four criteria were defined for doublet-detection methods' usability: software quality, execution convenience, publication, and documentation & support. The software quality criterion indicates whether a doublet-detection method can be executed on all real and synthetic datasets used in this study. The execution convenience criterion is related to the popularity of the computational platform required to run a method. Methods written in R and Python packages are preferred because of the popularity of these two languages. The publication criterion is regarding whether a doublet-detection method has been published in a peer-reviewed journal. The documentation & support criterion evaluates a method's user-support resources, such as open-source code, tutorials, and active Q&As. Each criterion has three levels: excellent, good, and fair, corresponding to a score of 2, 1, and 0, respectively. The final usability score of a method was defined as the sum of the method's scores in these four criteria.

Please cite this article in press as: Xi and Li, Benchmarking Computational Doublet-Detection Methods for Single-Cell RNA Sequencing Data, Cell Systems (2020), https://doi.org/10.1016/j.cels.2020.11.008

Accuracy of Computational Doublet Detection in Relation to Experimental Techniques for Doublet Labeling

Four experimental techniques were used to label doublets in the 16 real datasets used in this study: cell hashing (Stoeckius et al., 2018), species mixture (Wolock et al., 2019), demuxlet (Kang et al., 2018), and MULTI-seq (McGinnis et al., 2019a, 2019b). To examine the relationship between the accuracy of computational doublet-detection methods and the use of experimental techniques for doublet labeling, we calculated the mean AUPRC of each computational method across the datasets labeled by each experimental technique (Figure S2D; Table S11). Overall, all computational doublet-detection methods achieved the highest accuracy on the species-mixture datasets, followed by the cell-hashing, MULTI-seq, and demuxlet datasets. This is an expected result since doublet-detection methods are more capable of identifying heterotypic doublets than homotypic doublets by design (Lun et al., 2016; Gayoso and Shor, 2018; Bais and Kostka, 2020; DePasquale et al., 2019; McGinnis et al., 2019a, 2019b; Wolock et al., 2019; Bernstein et al., 2020), and all the labeled doublets in the species-mixture datasets are heterotypic (i.e., formed by cells of two species); meanwhile, the cell-hashing, MULTI-seq, and demuxlet datasets contain labeled doublets that are both heterotypic and homotypic (e.g., formed by cells of the same type from two samples or individuals), and they miss certain heterotypic doublets (e.g., formed by cells of different types from the same sample or individual). Among the eight doublet-detection methods (excluding DoubletDecon which cannot generate doublet scores), DoubletFinder, cxds, and Solo achieved the highest detection accuracy on the species-mixture datasets, demonstrating their strength of identifying heterotypic doublets. DoubletFinder was also the top performer on the MULTI-seq and demuxlet datasets in terms of mean AUPRC, while Solo excelled on the cell-hashing datasets. Interestingly, cxds exhibited the largest performance discrepancy between the species-mixture datasets and the other three types of datasets, highlighting its stronger priority towards identifying heterotypic doublets than other methods'.

Pairwise Similarities of Computational Doublet-Detection Methods

First, we calculated the Pearson correlation coefficient between every two doublet-detection methods (except hybrid, which is an ensemble of bcds and cxds, and DoubletDecon, which cannot generate doublet scores) in terms of their doublet scores in each of the 16 benchmark datasets; for every pair of methods, we averaged their 16 Pearson correlation coefficients (Table S12). Among the 21 pairs of methods, DoubletFinder-DoubletDetection, Solo-bcds, and DoubletFinder-bcds have the largest mean correlations. Second, we calculated the Jaccard index between every two doublet-detection methods (except hybrid and DoubletDecon) in terms of their identified doublets, whose numbers are set equal to the number of labeled doublets, in each of the 16 benchmark datasets; for every pair of methods, we averaged their 16 Jaccard indices (Table S13). Among the 21 pairs of methods, DoubletFinder-DoubletDetection, DoubletDetection-Solo, and DoubletFinder-Solo have the largest mean Jaccard indices, which reflect the large overlaps of their identified doublets. These two similarity analyses indicate the possibility of developing an ensemble method to combine the top-performing methods that are not too similar (Hastie et al., 2009). Given the high accuracy of DoubletFinder and the distinctive algorithm design of cxds (the only method without artificial doublets), these two methods may serve as good candidates to be combined into an ensemble method.

Comparison of Hyperparameter Selection in kNN-Base Methods

The algorithm designs of Scrublet and DoubletFinder are similar because they both define each droplet's doublet score as the proportion of artificial doublets among the *k*-nearest neighbors of this droplet in the principalcomponent (PC) space. The major difference between Scrublet and DoubletFinder is how they select hyperparameters, including the number of artificial doublets to generate, the number of genes used to perform the principal component analysis, the number of PCs to define nearest neighbors, and the number of nearest neighbors *k*. Table S14 summarizes the default hyperparameter settings of Scrublet and DoubletFinder. In particular, DoubletFinder automatically selects *k* by maximizing the mean-variance normalized bimodality coefficient (Pfister et al., 2013) of the distribution of doublet scores. To examine the effect of hyperparameter selection on the method performance, we selected four real datasets on which DoubletFinder outperformed Scrublet, and replaced the hyperparameters of Scrublet by those of DoubletFinder, including the *k*s selected by DoubletFinder for those datasets. Figure S2E summarizes the AUPRC values of three methods—DoubletFinder, Scrublet with default hyperparameters, and Scrublet with the same hyperparameters as DoubletFinder—on each of the four datasets. With the hyperparameters of DoubletFinder, Scrublet improved its detection accuracy on two datasets, nuc-MULTI and pbmc-1C-dm, but it still underperformed DoubletFinder. On the other two datasets, cline-ch and pbmc-1A-dim, Scrublet performed similarly or even worse, respectively, with the hyperparameters of DoubletFinder. This result suggests that hyperparameter selection is an important but not the only factor that determines the performance of doublet-detection methods. Other aspects of algorithm design, including the generation of artificial doublets and algorithm implementation, also play critical rules.