Improving Deconvolution Methods in Biology through Open Innovation Competitions: an Application to the Connectivity Map

[INCOMPLETE, ORDER TBD] Andrea Blasco (Harvard)

Ted Natoli (Broad) Rinat A. Sergeev (Harvard) Jin H. Paik (Harvard)

Max Macaluso (Broad) Rajiv Narayan (Broad) ...

Karim R. Lakhani (Harvard) Aravind Subramanian (Broad)

Last updated: Oct 24, 2019

Abstract

Deconvolution methods are ubiquitous and have a long history in many areas of science and engineering. A common problem of deconvolution algorithms is the validation of results using realistic datasets for testing, and the costs involved in tailoring different approaches to such datasets. We describe the deconvolution problem of an assay, L1000, that measures mRNA transcript abundance of approximately 1000 genes from human cells using 500 Luminex beads, such that two transcripts are identified by a single bead color. Using L1000, we generated a novel dataset of transcripts for various perturbagens (shRNA and compounds) where genes were measured both in tandem and individually. We then ran an open competition to elicit different deconvolution algorithms. The competition produced a wide variety of techniques including machine-learning algorithms, such as Convolutional Neural Networks and Random Forests, and more traditional approaches, such as Gaussian mixtures and k-means. All top solutions achieved notable improvements over the benchmark.

Keywords: biology; open innoation competitions; crowdsourcing; deconvolution; gene expressions; cell lines.

Contents

TODOS:

- Intro: Write paragraph 1 for the actual journal submission.
- Results: Describe motivation for each metric
  - assessing “global similarity” b/w predictions and ground truth.
  - assessubg EM detection
- Figures: drop panel C
- Address comments at the end of manuscript

1 Introduction

Multiplex[ed] assays frequently use a single analyte to measure multiple populations ...

RNA profiling is an essential means to learn about the state of cells and tissues in response to perturbations; it helps detecting similarities between different cellular states that, in turn, provide clues about therapeutic intervention, help the elucidation of moas, etc. However, gaining an understanding of genetic transcriptome perturbations will require extensive experimentation, which can be costly to generate at scale.

A central problem Over the last decade, technological advances have made it possible to generate large perturbational datasets which in turn have helped accelerate therapeutic discovery. A large proportion of these technologies rely on using multiple analytes to measure biological signals. These assays are extremely powerful but in some applications are limited by the number of unique analytes they can detect.[1]

This limitation is more prominent for gene-expression profiling, given the high number of genes involved. Within this context, the Connectivity Map (CMap) has developed an assay, called L1000, that integrates traditional multianalyte detection methods and statistical analyses to simultaneously measure the expression of multiple genes using the same analyte type, thus multiplying the capacity of existing technologies [1].

Cleary et al. [2] proposes to use "compressed sensing to use low-level measurements or composite signals (from multiple genes) to recover high-level... Deng et al. [3] Hie, Bryson, and Berger [4] such as to measure differential gene expression for each gene in mixed cell-type samples [5],

A central component of this approach is the computational deconvolution of gene expression signals from multiple genes measured using the same analyte. Similar deconvolution problems are ubiquitous in biology and several solutions have been proposed in a variety of different contexts. Examples include algorithms to identify cell type–specific gene expression differences in complex tissues [5] or dynamic changes in cell populations [6], or ways to discover the target proteins of small molecules [7].

[1] For example, Luminex’s most advanced platform, called FLEXMAP 3D, can measure simultaneously a maximum of 500 bead types for an equal number of genes or proteins from a small sample.

CMap's current deconvolution approach, called dpeak, is based on the k-means clustering algorithm, which is a flexible unsupervised learning procedure widely used in biology, as well as in many other fields. This approach automatically partitions a set of gene-expression measurements into k clusters by minimizing the within-cluster sum of squares. It then assigns the clusters to each gene within the sample (see Subramanian et al. [1] for the details). This approach works well in practice but has several limitations; it tends to split clusters incorrectly when the distributions are not reasonably well separated, it is sensitive to outliers, and it is computationally demanding.[2]

Motivated by recent successes with machine learning techniques in biology, we hypothesized better deconvolution approaches that could likely be developed. To test this hypothesis, we generated a novel experimental dataset of the response of approximately 1,000 genes to 122 different perturbagens (shRNA and compounds) within 4 to 10 replicates. Using the L1000 platform, we varied the detection mode for acquiring the data between two genes per analyte type (DUO) and one gene per analyte type (UNI). We then used the data to run an open innovation competition (as described in Lakhani et al. [8] and Blasco et al. [9]) where we offered cash incentives for different solutions for the deconvolution problem. We assessed the accuracy of these different approaches using the UNI data as the ground truth.

The top approaches included very popular machine-learning methods, such as Random Forests and Convolutional Neural Networks (CNNs), as well as more traditional Expectation-Maximization (EM) approaches based on Gaussian mixtures. Overall, the results of the challenge enabled us to perform a cost-effective comparison of different methods under nearly identical conditions, which we report below.

2 Methods

We ran the competition on the Topcoder platform (Wipro, India) for a total of 21 days. A prize purse of $23,000 in cash was offered to competitors as incentive to be divided among the top 9 submissions ($8000, $6000, $4000, $2000, $1000, $800, $600, $400, $100).

To generate data for this contest, we profiled six 384-well perturbagen plates, each containing mutually exclusive sets of compound and short-hairpin (shRNA) treatments (see S1 Table and S2 Table for a complete list of the perturbagens). Multiple treatment types were used to avoid potentially over-fitting to any one. The compound and shRNA perturbagen plates were arbitrarily grouped into pairs, and to avoid any potential 'information leakage' each pair was profiled in a different cell line. The resulting lysates were amplified by Ligation Mediated Amplification (LMA, Subramanian et al. [1]). The amplicon was then split and detected in both UNI and DUO detection modes, resulting in three pairs of data generated under comparable circumstances.

Next, we randomly split the three pairs of data, as described above, into training, testing, and

[2] An efficient k-means algorithm, such as Lloyd's, has a running time that scales as $O(kn)$ where n is the number of observations and k the number of clusters.

holdout subsets. The training data were available for all the contestants to develop and validate their solutions offline. The testing data were used to evaluate solutions during the contest and populate the live leaderboard. The holdout data were used to evaluate competitors' final submissions and to guard against over-fitting. Prizes were awarded based on performance on the holdout dataset. In each category, the UNI data served as the ground truth.

We developed a scoring function to evaluate the submissions of the competitors. The function combined two different accuracy metrics and a measure of computational speed. The accuracy metrics were the Spearman rank correlation between the values and the ground truth; and the area under the receiver operating curve (ROC) as a measure of aggregated classification performance in the prediction of extreme modulated genes (See S1 Appendix for the details).

3 Results

The contest attracted 294 participants, who made 820 code submissions with an average of about 18 submissions per participant. The submissions were narrowed down to the top nine scoring submissions in the contest. We illustrate the variety of different analysis methods by outlining the top four below (while S3 Table provides brief information on the others).

The winning approach (developed by `gardn999`, a competitor from the United States with a degree in Physics from the University of Kansas) was a random forest algorithm that combined the predictions of 10 different trees. Each tree was trained using 60 different features of the data. These features included the actual measurements for each bead type (binned into 50 different variables), as well as aggregate variables of associations at the plate and experiment level.

The second best (developed by `Ardavel`, a competitor from Poland with a Master's degree in Computer Science from The Lodz University of Technology) classified the clusters using a Gaussian mixture model fitted with the Expectation-Maximization (EM) algorithm for each gene pair. Next, it improved the predictions using the clustered data to fit a plate-wide distribution of the estimated expression levels and cluster sizes for each gene. The fitted distribution was then used to adjust the peak assignments.

The third best (developed by `mkagenius`, a competitor from India with a bachelor's degree in Computer Science) was a k-means algorithm with fine adjustments to avoid local minima, minimize the impact of extreme outliers, and improve speed.

Finally, the fourth best (developed by `Ramzes2`, a competitor from Ukraine with a bachelor's degree in Computer Science from the Cherkasy National University) was based on a Convolutional Neural Network (CNN). The algorithm first filtered the data to remove extreme outlier measurements. The filtered data were then transformed into a 32-bin histogram for each pair of genes and served as input to a CNN, which consisted of two parts. The first part used the classic U-net architecture [10] (a contracting path to capture context and a symmetric expanding path) to provide adequate

representation of the data. The output of this network was then used to label bins into one of the two genes for each pair, and to predict the exact value within the bin. This step was achieved using two subnetworks with the same architecture but weights trained separately (using a mean squared error loss function).

Despite the wide diversity of the approaches described above, all top solutions included ways to automatically remove extreme outlier measurements and to incorporate plate level information to refine the predictions.

3.1 Accuracy and speed

3.1.1 Correlation

We tested the accuracy of the solutions by comparing the distribution of the genewise spearman correlation (ρ) between the ground-truth gene-expressions (as detected by UNI) and values of the competitors and of the benchmark that were obtained through the deconvolution of DUO data. The Spearman rank correlation was high overall ($\rho > 0.56$). However, the empirical distributions of the competitor soutions were right-shifted compared to the k-means benchmark, indicating more accurate predictions in both shRNA and compound experiments (*Fig.* **??**, A and B). On average, nearly all competitors achieved significant improvements compared to the benchmark (*Fig.* **??**, C and D), with the random forest approach of the contest winner raising the average correlation by 3 to 4 percentage points (approximately 5% increase over the benchmark).

To understand whether there were differences between the algorithms at the individual gene level, we identified the best performing algorithm (by the correlation metric) for each of the 976 landmark genes (*Fig.* **??**, E and F). The random forest approach was the best performer for about a third (30%) of the genes. The gaussian mixture model was the best performer for about one fourth (20%). Thus, the top two contest submissions achieved the highest correlation in more than half of the genes. Even so, all but a few algorithms were the best performers for at least 5% of the genes, suggesting some complementarity between the algorithms, which could potentially be leveraged by an ensemble approach, as discussed below.

3.1.2 Extreme modulations

We further tested the accuracy of the competitors' solutions on the detection of, so called, "extreme modulations." These are genes that are notably up- or down-regulated by perturbation, and hence exhibit exceedingly high (or low) differential expression (DE) values. We obtained the DE values by using a robust z-score procedure (as described by Subramanian et al. [1]) and evaluated the detection accuracy of each solution by computing the corresponding area under the curve (AUC) using UNI DE data as ground truth.

All the solutions achieved a good detection accuracy (AUC > 0.87). As before, the competitors achieved significant improvements relative to the benchmark in both the shRNA and compound experiments (*Fig. 2*, A, B and C).

We further tested differences in the probability of detection of the extreme modulations for targeted gene knockdowns (KD). These are experiments involving an shRNA targeting one of the 976 landmark genes. Hence, we expected the targeted gene should exhibit a very low DE value (highly negative). For each such experiment, we define a successful KD as one in which the DE value is less than -2 and the gene-wise rank is less than 10, meaning that the targeted gene achieves one of its lowest DE values in the experiment where it was targeted. We computed the KD success frequency for each competitor algorithm as well as the benchmark and UNI data. We used the UNI ground-truth data to estimate the maximum achievable frequency, which in this case was 0.8. We observe that all but 2 of the top 9 contestant algorithms achieve a higher KD success frequency than the benchmark solution (*Fig. 2*, D). These results suggest that the algorithm improvements, as assessed by the accuracy metrics used in the contest, translate to improvements in biologically relevant metrics used in common applications of L1000 data.

Given the variety of methods represented amongst the prize-winning solutions, we sought to assess whether there were notable differences in the predictions generated. Using the holdout dataset, we generated a two-dimensional projection of the UNI ground truth data and DUO-derived benchmark and contestant predictions for both DECONV and DE data using t-distributed stochastic neighbor embedding (t-SNE, Maaten and Hinton [11]). We observe that in DECONV data the samples primarily cluster by pertubagen type, except for the ground truth UNI data, which appears distinct from the deconvoluted samples (*Fig. 5*, A and B). Separating the samples by algorithm reveals commonalities in the predictions generated by similar approaches (*Fig. 5*, C). For example, the decision tree regressor (DTR) algorithms have similar 'footprints' in the projection, as do the k-means and Gaussian mixture model (GMM) algorithms. This suggests that in general similar algorithms generate predictions with similar properties. After the standard transformation to DE data, we observe that the t-SNE projection is much more homogenous, indicating that perturbagen type and algorithm-specific effects have been greatly reduced (*Fig. 5* D). This is reassuring, given that in production mode the downstream analysis of this data will be based on DE. It also suggests that integrating multiple deconvolution algorithms into an ensemble method might be feasible.

3.1.3 Reduction variation across replicate samples

Our data contain several replicates which enabled us to study improvements in inter-replicate variation of gene expression-quantities. This is of crucial practical importance to biologists because a lower variance means higher reproducibility. And one of the issues with the benchmark k-means solution was that it tends to do little to mitigate the discrepancy in variability between the genes measured with high and low bead proportions. To test this hypothesis, we computed the distribution of the sample variance (s^2) of the measurements of each gene-perturbagen combination for each

approach. The sample size ranged between 4 to 11 replicates of the same perturbagen experiment on a given plate. Of particular interest was the difference between distributions for genes in high versus low bead proportions, as we expected those in low bead proportions to have a higher variance. Consistent with our expectations, the distribution of sample variances of the benchmark was right-shifted compared to the distribution of the competitors (*Fig. 3*) and the shift was more visible for genes in low bead proportions. This evidence indicates significant improvements in the inter-replicate variance of the competitors over the benchmark.

3.1.4 Computational speed

Speed improvements over the benchmark were substantial (*Fig.* **??**). The benchmark took about 4 minutes per plates. In contrast, the fastest algorithm took as little as 4 seconds per plate (a 60x speedup compared to the benchmark), and the slowest was well below one minute per plate. We observed no particular trade-off between speed and accuracy. The fastest algorithm (Ardavel), that was based a gaussian mixture model, achieved a good level of accuracy as well, and ranked second overall. On the other hand, the algorithm with the best performance in terms of accuracy (gardn999), which was based on a decision tree regression, also achieved a decent speed performance compared to the benchmark. Thus, at least within the context of the implemented solutions, we found a negligible trade-off between speed and accuracy.

3.1.5 Ensembles

Next, to assess the complementarity of the algorithms, we assessed the performance of ensembles composed of different number of models. For the present analysis, we focused on the subset of the data with shRNA experiments (ignoring the data with compound experiments). For this subset, the competitors achieved a significantly lower correlation and AUC than the data with compound experiments. We used a basic approach to aggregate the predictions of each solution by using the median prediction. Then, we tested the performance using the Spearman correlation and the AUC metrics computed on the holdout dataset for that plate. Results (*Fig. 6*) showed that the performance of the ensemble tends to increase with the number of models involved. However, the maximum performance in both metrics tends to plateau (or even decrease) after the ensamble reaches a size equal to 3 models.

3.2 Concluding remarks

In summary, we have created a novel dataset of over 120 shRNA and compound experiments with measurements for about 1,000 genes. This dataset constitutes a public resource to all the researchers in this area who are interested in testing their deconvolution appproaches. Using an open innovation competition, we collected and evaluated multiple and diverse deconvolution methods. The best

approach was based on a random forest, which achieved consistently the highest level of accuracy compared to all other solutions. Next, we will apply these resutls to over one million experiments that constitute the Connectivity Map, and explore cost svings achieved by having a lower number of replicates. Another potential application is to leverage these algorithms to enable detection of three or more genes using the same analyte type.

4 Figures

4.1 Scoring accuracy

4.2 Spearman rank correlation

4.3 AUC plots

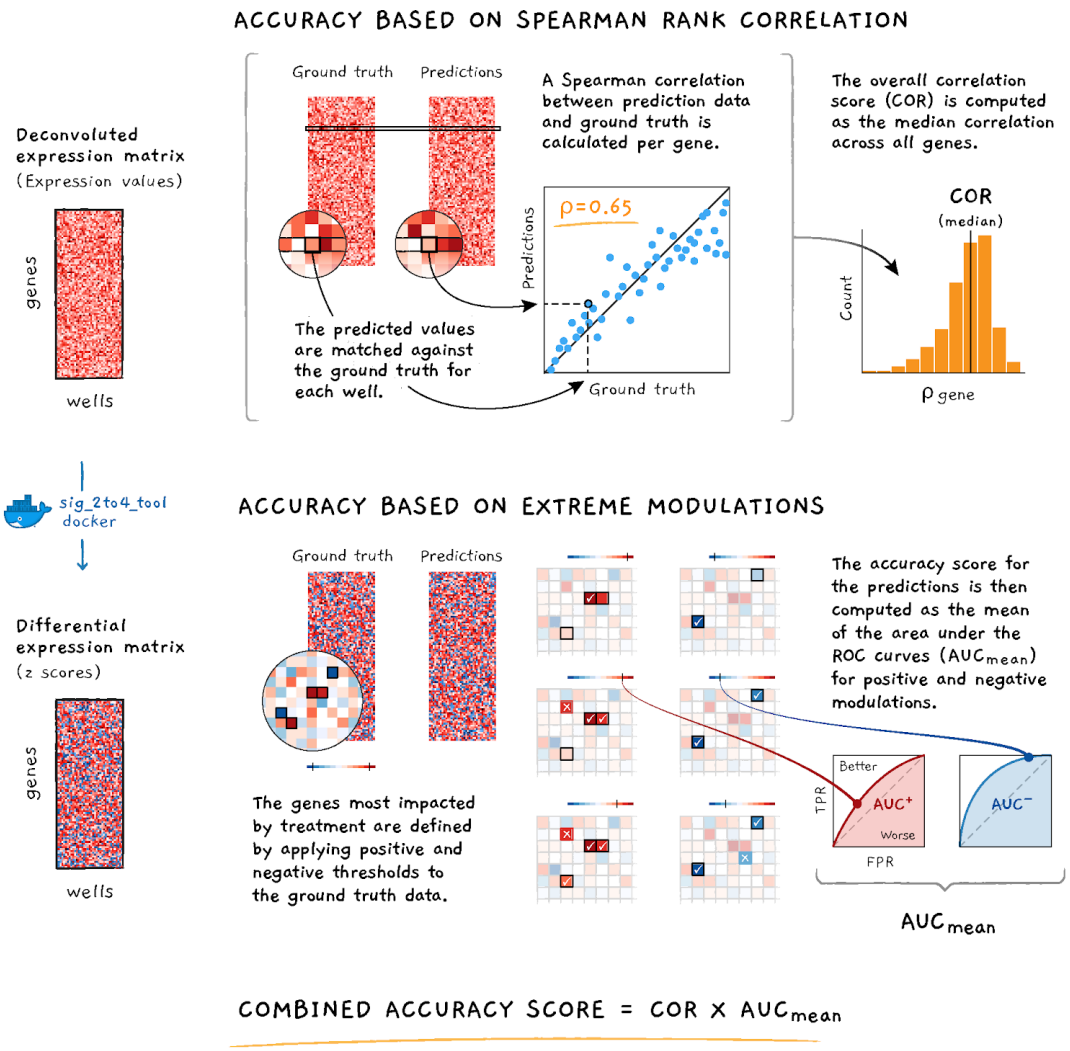

Figure 1: **Schematic illustrating accuracy components of scoring function**. The accuracy component is computed as the product of gene-wise Spearman correlations with ground truth and the area under the curve AUC of extreme modulations.

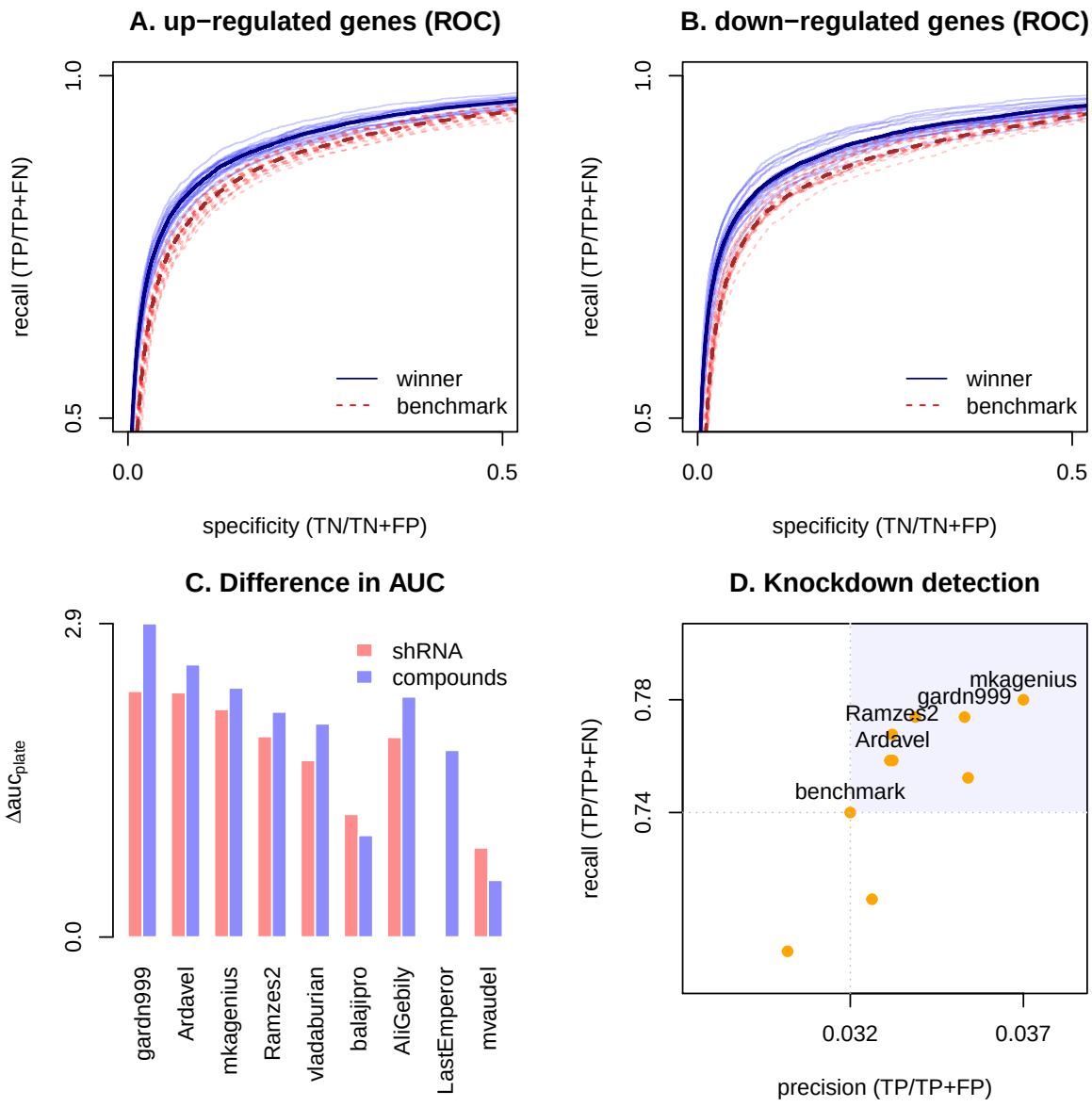

Figure 2: **AUC.** Top panels show the receiver-operator characteristic (ROC) curves for the levels of (**A**) up- and (**B**) down-regulated genes (based on UNI data) detected by the top competitor and the benchmark. The winner's ROC curve is higher compared to the one of the benchmark, indicating a more accurate detection. Curves based on random subsamples of 200 genes are used to visualize uncertainty around the ROC curves based on all the 976 genes. Panel **C** shows the difference in AUC, area under the curve, values between the benchmark and each of the sumbmissions, stratified by experiment type. Panel **D** shows the levels of recall (true positive rate) and precision for the detection of knockdown genes in the shRNA experiments. All top solutions exhibit higher precision and recall compared to the benchmark. (Note that the very low level precision of the algorithms should raise no concerns because the count of false positive is likely inflated by possible downstream effects of the knockdown).

4.4 Inter-replicate variance

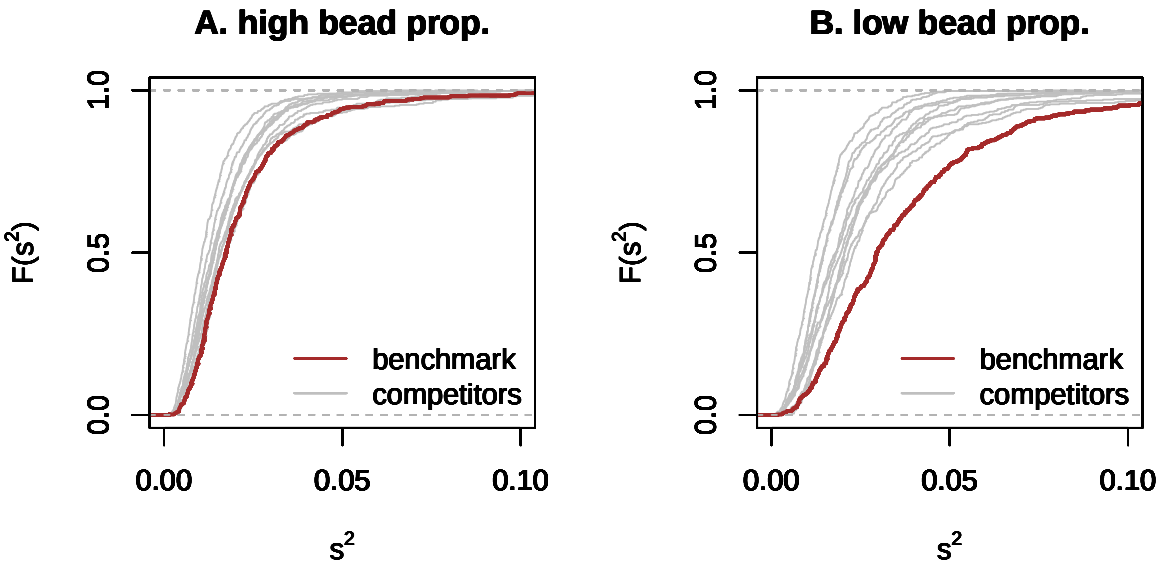

Figure 3: Figure shows empirical CDF of the distribution of the sample variance (s^2) of the logarithm of the gene values for each combination of gene and perturbagen. The gene-perturbagen variance is based on a sample that ranges between 4 to 11 replicates of the same perturbagen experiment on a given plate. Data are then stratified for genes with (**A**) high and (**B**) low bead proportions. The CDF of the benchmark is on the right, indicating higher inter-replicate variance compared to the distribution of the competitors.

5 Runtime and speedups

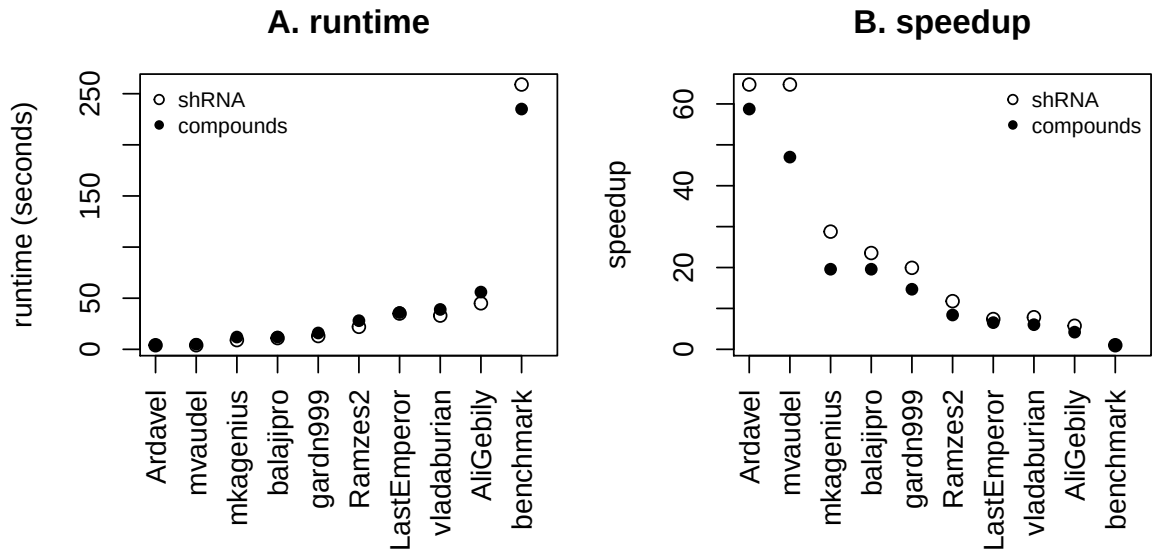

Figure 4: **Speed improvements.** Distribution of the per-plate runtime (in seconds) and speedups over the benchmark ($t_{\text{benchmark}}/t_{\text{competitor}}$) for each of the competitors' algorithms

5.1 Clustering of solutions

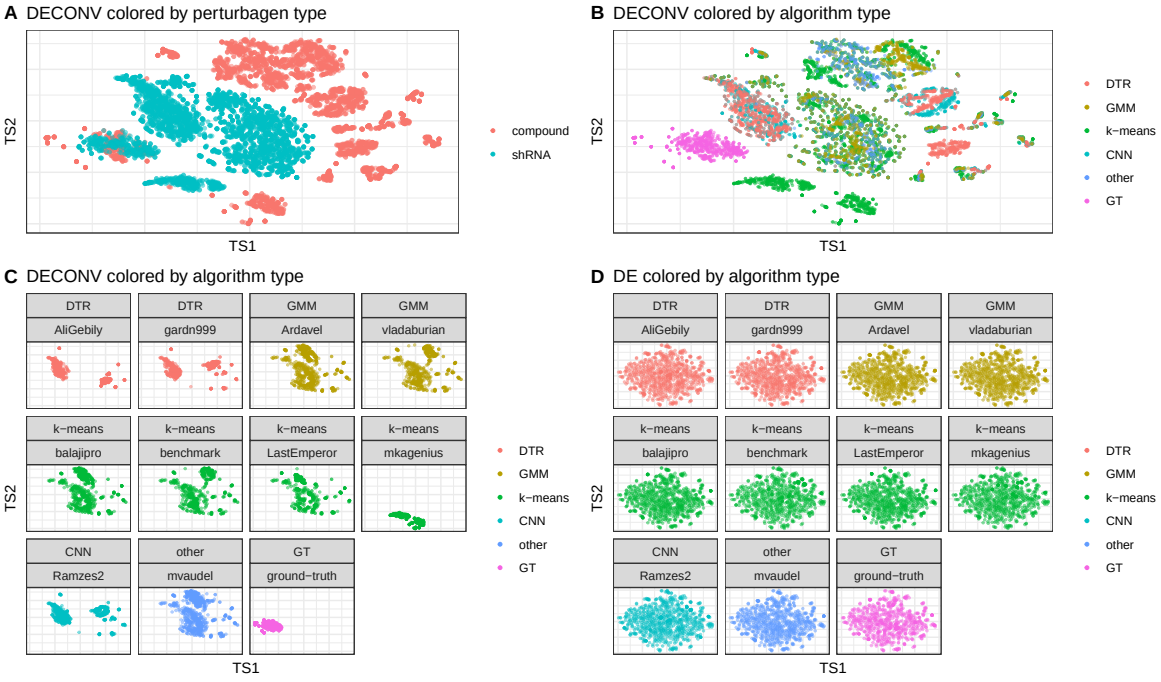

Figure 5: **t-SNE projection of deconvoluted data.** Each point represents the 2D projection of a sample generated by UNI ground truth (GT) or by applying a deconvolution algorithm to DUO data. t-SNE was run on the 2 plates of holdout data, one each containing compound and shRNA treatmens. DECONV data colored by perturbagen type (A) and algorithm type (B). DECONV (C) and DE (D) data colored by algorithm type and stratified by each individual implementation.

5.2 Ensembles

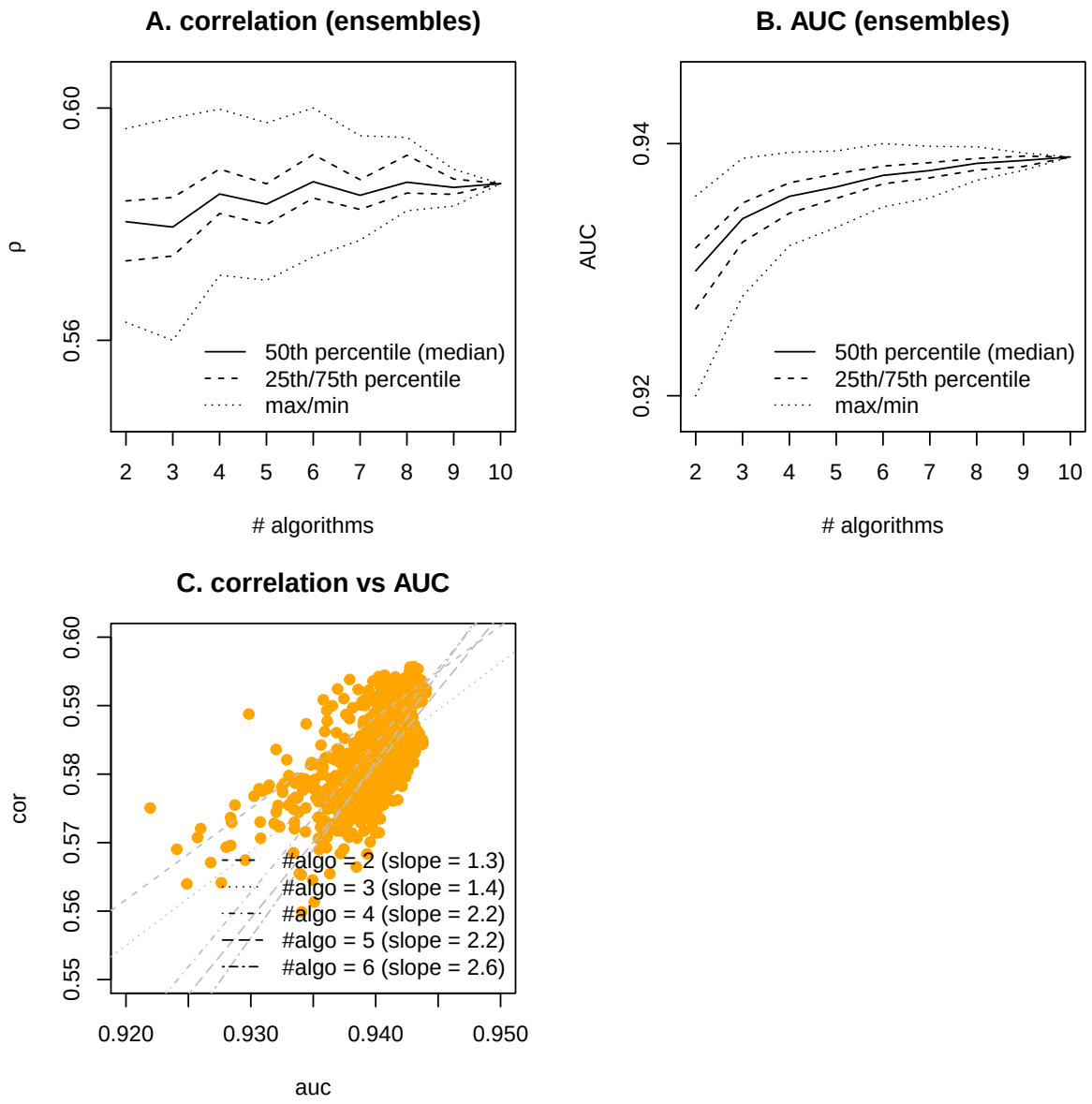

Figure 6: **Ensemble** Top panels show the performance in the **A** correlation metric and **B** the AUC metric of the ensemble based on the median prediction of all possible combinations of a given size of the top 10 algorithms (including the benchmark). The median performance of the ensemble tends to increase with its size. However, the maximum performance in both metrics tends to plateau (or even decrease) after the ensamble reaches a size equal to 3. Panel **C** shows the association between the performance metrics for all the ensembles. The metrics are highly correlated. We reported the estimated slopes of a linea regression, indicating that the correlation is stronger for ensembles with size 4 or higher.

Supporting information

S1 Table.

Compound perturbagens descriptives. This table shows componud perturbagen names (`pert_iname`), unique id (`pert_id`), time of treatment (`pert_itime`), dose (`pert_idose`), and number of replicates (`num_replicates`).

pert_iname	pert_id	pert_itime	pert_idose	num_replicates
abiraterone(cb-7598)	BRD-K50071428	24 h	10 um	11
acalabrutinib	BRD-K64034691	24 h	10 um	11
afatinib	BRD-K66175015	24 h	10 um	11
artesunate	BRD-K54634444	24 h	10 um	11
azithromycin	BRD-K74501079	24 h	10 um	11
betamethasone dipropionate (diprolene)	BRD-K58148589	24 h	10 um	11
CGS-21680	BRD-A81866333	24 h	10 um	11
chelidonine	BRD-K32828673	24 h	10 um	11
clobetasol	BRD-K84443303	24 h	10 um	11
digoxin	BRD-A91712064	24 h	10 um	11
disulfiram	BRD-K32744045	24 h	10 um	10
emetine hcl	BRD-A77414132	24 h	10 um	10
eplerenone	BRD-K19761926	24 h	10 um	11
epothilone-a	BRD-K71823332	24 h	10 um	9
flumetasone	BRD-K61496577	24 h	10 um	11
fluocinolone	BRD-K94353609	24 h	10 um	11
genipin	BRD-K28824103	24 h	10 um	11
hydrocortisone	BRD-K93568044	24 h	10 um	10
hyoscyamine	BRD-K40530731	24 h	10 um	11
indirubin	BRD-K17894950	24 h	10 um	10
L-745870	BRD-K05528470	24 h	10 um	10
nTZDpa	BRD-K54708045	24 h	10 um	11
oligomycin-a	BRD-A81541225	24 h	10 um	11
PRIMA1	BRD-K15318909	24 h	10 um	11
RITA	BRD-K00317371	24 h	10 um	11
spironolactone	BRD-K90027355	24 h	10 um	11

pert_iname	pert_id	pert_itime	pert_idose	num_replicates
tanespimycin	BRD-K81473043	24 h	10 um	11
tretinoin	BRD-K71879491	24 h	10 um	10
UB-165	BRD-A14574269	24 h	10 um	11
ursolic-acid	BRD-K68185022	24 h	10 um	11
WAY-161503	BRD-A62021152	24 h	10 um	11
ZM-39923	BRD-K40624912	24 h	10 um	11

S2 Table.

Non-compound perturbagens descriptives. This table shows non-compound perturbagen names (`pert_iname`), unique id (`pert_id`), and number of replicates (`num_replicates`).

pert_iname	pert_id	num_replicates
ABCB6	TRCN0000060320	4
ADI1	TRCN0000052275	4
ALDOA	TRCN0000052504	4
ANXA7	TRCN0000056304	4
ARHGAP1	TRCN0000307776	4
ASAH1	TRCN0000029402	4
ATMIN	TRCN0000141397	4
ATP2C1	TRCN0000043279	4
B3GNT1	TRCN0000035909	4
BAX	TRCN0000033471	4
BIRC5	TRCN0000073718	4
BLCAP	TRCN0000161355	4
BLVRA	TRCN0000046391	4
BNIP3L	TRCN0000007847	4
CALU	TRCN0000053792	4
CCDC85B	TRCN0000242754	4
CCND1	TRCN0000040038	4
CD97	TRCN0000008234	4
CHMP4A	TRCN0000150154	4
CNOT4	TRCN0000015216	4

pert_iname	pert_id	num_replicates
DDR1	TRCN0000000618	4
DDX10	TRCN0000218747	4
DECR1	TRCN0000046516	4
DNM1L	TRCN0000001097	3
ECH1	TRCN0000052455	4
EIF4EBP1	TRCN0000040206	4
EMPTY_VECTOR	TRCN0000208001	15
ETFB	TRCN0000064432	4
FDFT1	TRCN0000036327	4
GALE	TRCN0000049461	4
GFP	TRCN0000072181	16
GRN	TRCN0000115978	4
GTPBP8	TRCN0000343727	4
HDGFRP3	TRCN0000107348	4
HIST1H2BK	TRCN0000106710	4
IKBKAP	TRCN0000037871	4
INPP4B	TRCN0000230838	4
INSIG1	TRCN0000134159	4
ITFG1	TRCN0000343702	3
JMJD6	TRCN0000063340	4
LBR	TRCN0000060460	4
LGMN	TRCN0000029255	4
LPGAT1	TRCN0000116066	4
LSM6	TRCN0000074719	4
MAPKAPK2	TRCN0000002285	4
MAPKAPK3	TRCN0000006154	4
MAPKAPK5	TRCN0000000684	4
MIF	TRCN0000056818	4
MRPL12	TRCN0000072655	4
NT5DC2	TRCN0000350758	4
NUP88	TRCN0000145079	4
PARP2	TRCN0000007933	4

pert_iname	pert_id	num_replicates
PLCB3	TRCN0000000431	4
POLE2	TRCN0000233181	4
PPIE	TRCN0000049371	4
PRKAG2	TRCN0000003146	4
PSMB10	TRCN0000010833	4
PTPN6	TRCN0000011052	4
RAB11FIP2	TRCN0000322640	4
RALB	TRCN0000072956	4
RHEB	TRCN0000010425	3
RNF167	TRCN0000004100	4
RPN1	TRCN0000072588	4
SLC25A4	TRCN0000044967	4
SNX11	TRCN0000127684	4
STK25	TRCN0000006270	4
STUB1	TRCN0000007525	4
STXBP1	TRCN0000147480	4
SYPL1	TRCN0000059926	4
TATDN2	TRCN0000049828	4
TM9SF3	TRCN0000059371	4
TMEM110	TRCN0000127960	4
TMEM50A	TRCN0000129223	4
trcn0000014632	TRCN0000014632	4
trcn0000040123	TRCN0000040123	4
trcn0000220641	TRCN0000220641	4
trcn0000221408	TRCN0000221408	4
trcn0000221644	TRCN0000221644	4
TSKU	TRCN0000005222	4
UGDH	TRCN0000028108	4
USP14	TRCN0000007428	4
USP6NL	TRCN0000253832	4
VAT1	TRCN0000038193	4
VDAC1	TRCN0000029126	4

pert_iname	pert_id	num_replicates
WIPF2	TRCN0000029833	4
YME1L1	TRCN0000073864	4
ZW10	TRCN0000155335	4

S3 Table

Top finishers. This table lists the top 9 finishers and the languages and algorithms each used.

Table 3: Summary of contestant solutions

rank	handle	language	method	category
1	gardn999	Java	random forest regressor	DTR
2	Ardavel	C++	Gaussian mixture model	GMM
3	mkagenius	C++	modified k-means	k-means
4	Ramzes2	Python/C++	ConvNet	CNN
5	vladaburian	Python/C++	Gaussian mixture model	GMM
6	balajipro	Python/C++	modified k-means	k-means
7	AliGebily	Python	boosted tree regressor	DTR
8	LastEmperor	Python	modified k-means	k-means
9	mvaudel	Java	other	other

S1 Appendix

Scoring function. This appendix describes the scoring function used in the contest to evaluate the performance of the competitors' submissions.

Submissions were scored based on a scoring function that combines measures of accuracy and computational speed. Accuracy measures were obtained by comparing the contestant's predictions, which were derived from DUO data, to the equivalent UNI ground truth data generated from the same samples.

The scoring function combines two measures of accuracy: correlation and AUC, which are applied to deconvoluted ($DECONV$) data and one to differential expression (DE) data, respectively.

DE is derived from DECONV by applying a series of transformations (parametric scaling, quantile

normalization, and robust z-scoring) that are described in detail in Subramanian et al. [1]. The motivation for scoring DE data in addition to $DECONV$ is because it is at this level where the most bilogically interesting gene expression changes are observed. Of particular interest is obtaining significant improvement in the detection of, so called, "extreme modulations." These are genes that notably up- or down-regulated by pertubation and hence exhibit an exceedingly high (or low) DE values relative to a fixed threshold.

The first accuracy component is based on the Spearman rank correlation between the predicted $DECONV$ data and the corresponding UNI ground truth data.

For a given dataset p, let $M_{\text{DUO},p}$ and $M_{\text{UNI},p}$ denote the matrices of the estimated gene intensities for $G = 976$ genes (rows) and $S = 384$ experiments (columns) under DUO and UNI detection. Compute the Spearman rank correlation matrix, ρ, between the rows of these matrices and take the median of the diagonal elements of the resulting matrix (i.e., the values corresponding to the matched experiments between UNI and DUO) to compute the median correlation per dataset,

$$\text{COR}_p = median(diag(\rho(M_{\text{DUO},p}, M_{\text{UNI},p})).$$

The second component of the scoring function is based on the Area Under the receiver operating characteristic Curve (AUC) that uses the competitor's DE values at various thresholds to predict the UNI's DE values being higher than 2 ("high") or lower than -2 ("low").

For a given dataset p, let $\text{AUC}_{p,c}$ denote the corresponding area under the curve where $c = \{\text{high}, \text{low}\}$ (either higher than 2 or less than -2); then, compute the arithmetic mean of the area under the curve per class to obtain the corresponding score per dataset:

$$\text{AUC}_p = (\text{AUC}_{p,\text{high}} + \text{AUC}_{p,\text{high}}/2.$$

These accuracy components were integrated into a single aggregate scores:

$$\text{SCORE} = \text{SCORE}_{\text{max}} \cdot (\max(\text{COR}_p, 0))2 \cdot \text{AUC}_p \cdot \exp(-T_{\text{solution}}/(3 \cdot T_{\text{benchmark}})),$$

where T_{solution} is the run time in seconds for deconvoluting the data in each plate, and $T_{\text{benchmark}}$ is the deconvolution time required by the benchmark D-Peak implementation.

S2 Appendix

L1000 Experimental Scheme The L1000 assay uses Luminex bead-based fluorescent scanners to detect gene expression changes resulting from treating cultured human cells with chemical or genetic perturbations [Subramanian 2017]. Experiments are performed in 384-well plate format, where each well contains an independent sample. The Luminex scanner is able to distinguish between 500

different bead types, or colors, which CMap uses to measure the expression levels of 978 landmark genes using two detection approaches.

In the first detection mode, called *UNI*, each of the 978 landmark genes is measured individually on one of the 500 Luminex bead colors. In order to capture all 978 genes, two detection plates are used, each measuring 489 landmarks. The two detection plates' worth of data are then computationally combined to reconstruct the full 978-gene expression profile for each sample.

By contrast, in the *DUO* detection scheme two genes are measured using the same bead color. Each bead color produces an intensity histogram which characterizes the expression of the two distinct genes. In the ideal case, each histogram consists of two peaks each corresponding to a single gene. The genes are mixed in 2:1 ratio, thus the areas under the peaks have 2:1 ratio (see Figure 1), which enables the association of each peak with the specific gene. **The practical advantage of the DUO detection mode is that it uses half of the laboratory reagents as UNI mode, and hence *DUO* is and has been the main detection mode used by CMap.**

After *DUO* detection, the expression values of the two genes are computationally extracted in a process called 'peak deconvolution,' described in the next section.

.1 Comments

.1.1 Rinat's

1. There is some odd figure ordering in the draft. Figure 1 is referenced only at the very last in Appendix (thus, should be an appendix figure); Then there are Figs 3, 6, 4, ??=5, and 7. Easy fix - just not to forget
2. Similarly, the table of contents of the section 4. Figures does not match the list of figures. It is also better to use Figure 1, Figure 2 in that list - instead of 4.1, 4.2, 4.3...
3. As for the figures, I wonder if there is a better way to distinguish the algorithm performances than comparing their CDFs which are currently barely distinguishable on the plots. May be, showing the difference: F_benchmark[p] - F_algorithm[p]? Or even: P_algorithm[F[p]]-P_benchmark[F[p]]? Just an idea - but it may over-complicate things, so we may just want to keep it as it is.
4. As for the ensemble, more accurate way of estimating the complementarity of the algos would be to use a vector of weights (w_i), compute the ensemble predictions as a sum: P_ensemble = Sum[w_i * P_i], then derive w_i that maximize the Score[P_ensemble] on testing dataset, and then compute the score on the holdout one. The advantage of that is that:

a) you get realistic ensemble performance, that is expected to always increase (saturating -> diminishing returns) with more algorithms included.
b) you obtain the explicit values of w_i - weights of the algos, that would represent their add-on value. Then you can state something smart about w_i decreasing with i, and possibly higher

paired w_ij for those algorithms i and j that are produced by different approaches (to strengthen the narrative around Table S3 - category). Again - just an idea. We can drop it and keep as it is.

5. I wonder if on Figure 5 it is better to sort the contestants by their placement rather then by their runtime - that would better illustrate the statement of low correlation between performance and runtime. It would also make sense to add a category (Table S3) mark to this plot - as the runtimes may be heavily clustered by the method. Also, I would probably use the log-scale for Y-axes on those plots. Again - the draft looks really great! Great job!!!

References

[1] Aravind Subramanian et al. "A next generation connectivity map: L1000 platform and the first 1,000,000 profiles". In: *Cell* 171.6 (2017), pp. 1437–1452.

[2] Brian Cleary et al. "Efficient generation of transcriptomic profiles by random composite measurements". In: *Cell* 171.6 (2017), pp. 1424–1436.

[3] Yue Deng et al. "Scalable analysis of cell-type composition from single-cell transcriptomics using deep recurrent learning". In: *Nature methods* 16.4 (2019), p. 311.

[4] Brian Hie, Bryan Bryson, and Bonnie Berger. "Efficient integration of heterogeneous single-cell transcriptomes using Scanorama". In: *Nature biotechnology* 37.6 (2019), p. 685.

[5] Shai S Shen-Orr et al. "Cell type–specific gene expression differences in complex tissues". In: *Nature methods* 7.4 (2010), p. 287.

[6] Peng Lu, Aleksey Nakorchevskiy, and Edward M Marcotte. "Expression deconvolution: a reinterpretation of DNA microarray data reveals dynamic changes in cell populations". In: *Proceedings of the National Academy of Sciences* 100.18 (2003), pp. 10370–10375.

[7] Georg C Terstappen et al. "Target deconvolution strategies in drug discovery". In: *Nature Reviews Drug Discovery* 6.11 (2007), p. 891.

[8] Karim R Lakhani et al. "Prize-based contests can provide solutions to computational biology problems". In: *Nature biotechnology* 31.2 (2013), p. 108.

[9] Andrea Blasco et al. "Advancing Computational Biology and Bioinformatics Research Through Open Innovation Competitions". In: *bioRxiv* (2019), p. 565481.

[10] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation". In: *International Conference on Medical image computing and computer-assisted intervention*. Springer. 2015, pp. 234–241.

[11] Laurens van der Maaten and Geoffrey Hinton. "Visualizing data using t-SNE". In: *Journal of machine learning research* 9.Nov (2008), pp. 2579–2605.