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

Author 1 Author 2 . . .

Last updated: Sep 16, 2019

Abstract

Report results fo open innovation competition aimed at solving a gene-related deconvolution problem.

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

Contents

1 Introduction **3**

2 Methods **4**

2.1 L1000 Experimental Scheme 4
2.2 Statistical deconvolution of gene-specific expression profiles. 4
2.3 Data generation for contest 6
2.4 Scoring configuation 6

3 Results **8**

3.1 Participation 8
3.2 Overall accuracy and speed 9
3.3 Clustering submissions 10
3.4 Ensemble approaches 11
3.5 Minors: 11

4 Discussion **11**

5 Figures **12**

5.1 Scoring accuracy 12
5.2 Accuracy vs. Speed 12
5.3 Accuracy vs speed (AB) 12
5.4 Gene knockdowns 12
5.5 High / Low bead discrepancy 12
5.6 Clustering of solutions 19
5.7 To read 20

6 References **20**

TODOS:

- Check length requirements for Nat. Met.
- Intro [AB]
- Methdos TN [DONE]
- Clustering
- Add disaggregated data

1 Introduction

Many recent examples have shown significant benefits for drug discovery from the systematic analysis of large repositories of gene-expression profiles [Refs]. However, traditional gene-expression high-throughput profiling technologies that are based multianalyte methods, such as Luminex profiling technology, are limited by the type and number of available analytes [Refs]. Therefore, the cost of big data generation in biology remains prohibitive.

Using an assay called L1000, The Connectivity Map (CMap) group at the Broad Institute has developed a novel approach that matches pairs of genes to the same Luminex beads to double the count of profiled genes per bead, thus lowering costs [Subramanian 2017]. A central component of this approach is to quantify gene-type frequency of beads, and then statistically deconvolve and compare gene type-specific average expression profiles for pairs of mixed gene samples.

This type of deconvolution problems are ubiquitous and have a long history in biology [refs]. For example, deconvolution problems arise when trying to identify cell type–specific gene expression differences in complex tissues [Shen-orr et al., 2010]; or in the discovery of target proteins of small molecules [Jung, 2015].

Common approaches are parametric (mixture models) and non-parametric. CMap's current solution to this problem is an algorithm, called "D-Peak," based on a K-means clustering [Refs]. This method automatically partitions a set of gene-expression measurements into k clusters, taking the mean of the two largest clusters, assigning the mean value of the largest cluster to the gene in higher proportion and the mean value of the second largest cluster to the gene in . This works well but has several problems as well. [List problems.] Including time. The trade-off between accuracy and computation time is currently unknown.

Alternative methods are well known, such as Gaussian etc. But it would have required substantial resources to experiment with these alternative approaches (more than what already done) and to adapt new to our data. Moreover, impossible an exhaustive search for all available approaches to try; and the combination of these different approaches.

Instead, we used an open innovation competition as a research tool to engage a variety of computer scientist, software developers and bioinformatics in the problem. This approach allows simultaneous exploration of competing approaches tailored to our problem, at no cost.

2 Methods

In biomedical research, our focus here, deconvolution problems are common in multianalyte assay methods. These methods are widely used to do X, Y and Z. In general terms, multianalyte assay methods are based on microspheres with different fluorescence decay times. This feature can be used to do X, Y, and Z. [EXPLAIN BRIEFLY CMap PROBLEM]. One problem with existing approaches is that they [. . . .]

To identify accurate methods we launched an open challenge that allowed a rapid exploration of different approaches. Key ingredients of there challenges are: training and testing dataset benchmark solution to improve

2.1 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.

2.2 Statistical deconvolution of gene-specific expression profiles.

In each sample, assume fluorescent-intensity values X_{ij} for beads $i = 1, 2, \ldots, n$ and analytes $j = 1, 2, \ldots, J$, and gene-specific proportions w_{ik} for beads $i = 1, 2, \ldots, n$ and genes $k = 1, 2, \ldots, K$.

Our model of analyte fluourescent intensity is:

$$X_{ij} = \sum_{k=1}^{K} w_{ik} h_{kj} + e_{ij}.$$

where h_{ik} is the gene-expression value for genes $k = 1, 2, \ldots, K$ and analytes $j = 1, 2, \ldots, J$.

For the UNI detection method, the gene-specific proportions are such that each analyte has only one gene. Hence, $w_{ik}^{\text{uni}} = 1$ when $j = k$, and it is zero otherwise. This implies that each sample can detect at most J different genes under the UNI method.

For the DUO detection method, the gene-specific proportions are such that each analyte is paired with two genes in 1:2 ratio. Hence, pick an element $g \in G^2$ from the set G^2 of all non-overlapping subsets of size two of the gene set G. For each pair of genes in g associated with an analyte j, we have: $w_{i1}^{\text{duo}} = 2/3$, $w_{i2} = 1/3$ and is zero otherwise.

2.2.1 Benchmark k-means solution

CMap's current solution to this problem is based on a k-means clustering algorithm called $dpeak$ that works as follows:

For each measurement, the dpeak partitions the list of realizations into $k >= 2$ distinct clusters and identifies two of the clusters whose ratio of membership is as close as possible to 2:1. The algorithm then takes the median intensity of each of the two clusters, assigning these values to the appropriate gene (i.e., matching clusters with more observations to the gene mixed in higher proportion).

After deconvoluting each sample on a plate, dpeak then uses the plate-wide distributions to perform adjustments on a per-well basis, correcting peaks that may have been misassigned (see Appendix).

Known problems with the current approach are that k-means is generally a biased and inconsistent estimator of the peaks of a bimodal distribution [ref]. It also sometimes fails to detect peaks with few observations or it incorrectly identifies these peaks as extraneous and disregards them. Another limitation is that it is computationally expensive (the current Matlab implementation takes about 30 minutes on a 12-core server to process one set of 384 experiments).

Additionally, because half the landmark genes are measured using two-fold less bead than the other half, these low-proportion genes are subject to increased variability, which the benchmark k-means solution does not mitigate. Hence, there is an unequal noise distribution in the benchmark's deconvoluted expression profiles.

2.3 Data generation for contest

To generate data for this contest, we profiled six 384-well perturbagen plates, each containing mutually exclusive sets of compound and shRNA treatments. 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 2017). The amplicon was then split and detected in both UNI and DUO detection modes. The three pairs of data were arbitrarily assigned to training, testing, and holdout categories, where in each case the UNI data served as the ground truth. Training data were made available for all the contestants to develop and validate their solutions offline. The testing data were used to evaluate solutions during the contestn and populate the live leaderboard. Holdout data were used evaluate competitors' final submissions and guard against over-fitting. Prizes were awareded based on performance on the holdout dataset.

Table 1: Data generated

Category	Type
training	Compounds
testing	Compounds
holdout	Compounds
training	shRNA
testing	shRNA
holdout	shRNA

2.4 Scoring configuation

Contest submissions were scored based on accuracy and speed.

2.4.1 Accuracy

TODO: update equations to use Latex

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 (See figure XX).

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. 2017[ref].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.

2.4.1.1 Accuracy based on Spearman correlation

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 MDUO, p and MUNI, 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 between the rows of MDUO, p and the rows of MUNI, p; 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:

CORp = median(diag(spearman(MDUO, p, MUNI, p))

2.4.1.2 Accuracy based on AUC of extreme modulations

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 bead type a in a given dataset p, let AUCp, c denote the corresponding area under the curve where c = { high | low }, where high means UNIDE, p >= 2, and low means UNIDE, p <= -2; then, compute the arithmetic mean of the area under the curve per class to obtain the corresponding score per dataset:

AUCp = (AUCp, hight + AUCp, low) / 2

2.4.2 Speed

For a given dataset p, the speed component of the score is computed as the run time in seconds for deconvoluting the data in both test cases: Runtimep

Notice that multi-theading is allowed in this match, and thus highly recommended. Your submission will be tested on multi-core machines (presumably, 8 cores). The maximum execution time of your solution will be capped by 30 minutes.

2.4.3 Aggregegated score

The accuracy and speed components were integrated into a single aggregate scores as follow:

SCORE = MAX_SCORE * (MAX(CORp, 0))2 * AUCp2 * exp(- Tsolution / (3 * Tbenchmark)) where Tbenchmark is the deconvolution time required by the reference DPeak implementation

3 Results

3.1 Participation

The contest attracted 294 participants, who made 820 code submissions, an averge of 18.2 submissions per participant. The top finishers in the contest employed a variety of different analysis approaches, including descision tree regressors (DTR), Gaussian mixture models (GMM), convolutional neural networks (CNN), and customized versions of k-means, all with notably improved performance relative to the benchmark. Table xx lists the top 9 finishers and the languages and algorithms each used.

Table 2: 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

Fig. 1. Participation stats (Submission counts)

3.2 Overall accuracy and speed

Using the holdout dataset, we computed the average AUC (on level-four data), the average rank correlation (level-two data), as well as the fraction of knocked-down genes correctly predicted for each of the top solutions and the benchmark. The benchmark achieved an AUC of xxx and xxx, a correlation of xxx and xxx for plate 1 and 2, respectively, as well as it successfully predicted xxx knocked-down genes in plate 2. The corresponding runtime for the benchmark in each plate was of xxx and xxx seconds.

Fig. 2 shows that all top-nine solutions improved upon the performance of the benchmark in nearly all measures. Overall, the average AUC improvement was of 2% (min = xxx, max =xxxx), the average correlation improvement was of 3% (min = xxx, max = xxx) and the average improvement in correct kd genes predictions was of 3% (min = xxx, max =xxx). Given a theoretical maximum improvement of xxx, xxx, and xxx, for the AUC, correlation and KD metrics, we observe that competitors achieves larger improvements (relative to the maximum achievable) in the AUC metric.

Only two submissions (out of nine) a performance that was better in AUC and correlation but worse than the benchmark on the KD success metric. These submissions, both based on k-means, ranked sixth and eighth in the final leaderboard.

Improvement with respect to speed were substantial. The average improvement was 300% ranging between 100% and 500%. We observe no accuracy-vs-speed trade-off. The empirical relationship between improvements in accuracy and improvements in speed is essentially flat.

Further examination based on the underlying methods used by the top solutions suggests that parametric models, such as mixture models, are both fast and relatively accurate, although non-parametric models, such as regression forests and k-means, tend to be more accurate. The fastest algorithm was indeed based on a parametric model, i.e., gaussian mixture model, which achieves a good level of accuracy on these data. This solution ranked second overall and achieved the greatest speed improvmenet in both plates.

The most accurate algorithm was instead based on a random forest. This solution ranked first overall and achieved the best performance in both AUC and correlation measures, as well as it was among the top three approaches in the KD measure.

We observe no significant difference in performance between the plates, with all the submissions achieving similar scores in both plates.

Todos:

- In Methods section, explain accuracy as measured in the coontest (slide p. 124). And then explain, KD additional test of accuracy (slides p. 128). Results are good on both.
- (How far froom the max achievable improvement in accuracy (down-sampling uni)?)
- Discrepancy between genes with high/low bead counts.
- Update equations to use Latex

3.2.1 Knockdown predictions

In addition to the Spearman correlation and extreme modulation AUC metrics used in the contest, we were also able to asses each algorithms ability to correctly predict the successful knockdown (KD) of landmark genes. The shRNA experiments used to generate the contest data were constructed such that each shRNA specifically targeted one of the landmark genes. Hence, the expectation in each of those experiments is that the targeted landmark gene should exhibit a greatly reduced expression level, which should manifest in a very low z-score in DE data. Additionally, we expect that the targeted landmark gene should be most dramatically down-regulated in the samples in which it was directly targeted. To assess this, we compute a gene-wise rank by DE z-score across all samples in a given plate. Criteria for indicating a successful KD are $zs <= -2ANDrank <= -10$. For each algorith, we compouted the KD success frequency as the fraction of the 376 landmark-targeting shRNA experiments in the holdout data for which the predictions met these success criteria. We used the UNI ground-truth data to estimate the maximum achievable KD success 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.* **??**). These results suggest that the algorithm improvements, as assessed by the accuracy metrics used in the contest, translate to improvements in biologically relevent metrics used in common applications of L1000 data.

TODO: 1) compute precision/recall for KD predictions

3.2.2 Reduction of difference between low and high bead proportions

One of the issues with the benchmark k-means solution is that it does little to mitigate the discrepancy in prediction accuracy between the genes measued with high and low bead proportions. We observe that the contestant solutions reduce this difference, as measured

TODO: 1) global distributions of DECONV values split by hi/low prop 2) include UNI for comparison 3) show that variance is higher for low prop but difference is reduced w/ new algos

3.3 Clustering submissions

3.3.1 2D projection

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 contenstant predictions for both $DECONV$ and DE data using t-distributed stochastic neighbor embedding (t-SNE, van der Maaten et al [ref]). Each point represents a single ground truth or predicted sample.

We observe that in $DECONV$ data the samples primarily cluster by pertubagen type, with the exeption of the ground truth UNI data, which appears distinct from the deconvoluted samples (*Fig.* **??** A and B). Separating the samples by algorithm reveals commonalities in the predictions generated by similar algorithms (*Fig.* **??** 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 affects have been greatly reduced (*Fig.* **??** D). This is reassuring, given that in production mode 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.

3.3.2 Clustering by gene-Level performance

We observed that the global structure of DE data seemed independent of algorithm type in general. We additionally sought to understand whether there were differences between the algorithms at the individual gene level. To assess this, we clustered the algorithms in the space of their gene-level accuracy metrics (Figure XX). We observe that for a large proportion of genes, all algorithms perform about equally well. There are subsets of genes for which accuracy is a bit more algorithm-dependent (ex. foo bar).These differences suggest complementarity between the algorithms which could potentially be leveraged using an ensemble approach.

3.4 Ensemble approaches

Figure 3. (A) Scatterplot runtime vs accuracy for ensamble (slides p. 163)

Speed vs accuarcy trade-off. Integration one or multiple methods?

3.5 Minors:

- signs of overfitting (compare traing vs testing)

4 Discussion

Summary of the results presented in the methods section.

Discussion generality of the solutions

- Novel? Have any of these solutions previously been applied to deconvolution problems?
- Specific to this problem or general to others?

Discuss implications of these methods for CMap production

- Preliminary results on past data conversion
- Directions for pipeline integration and generation of future data
- Cost savings
- Implementation strategy and outcomes
- Increase in data processing throughput

5 Figures

```
## Loading required package: methods
```

```
## Loading required package: rhdf5
```

5.1 Scoring accuracy

5.2 Accuracy vs. Speed

5.3 Accuracy vs speed (AB)

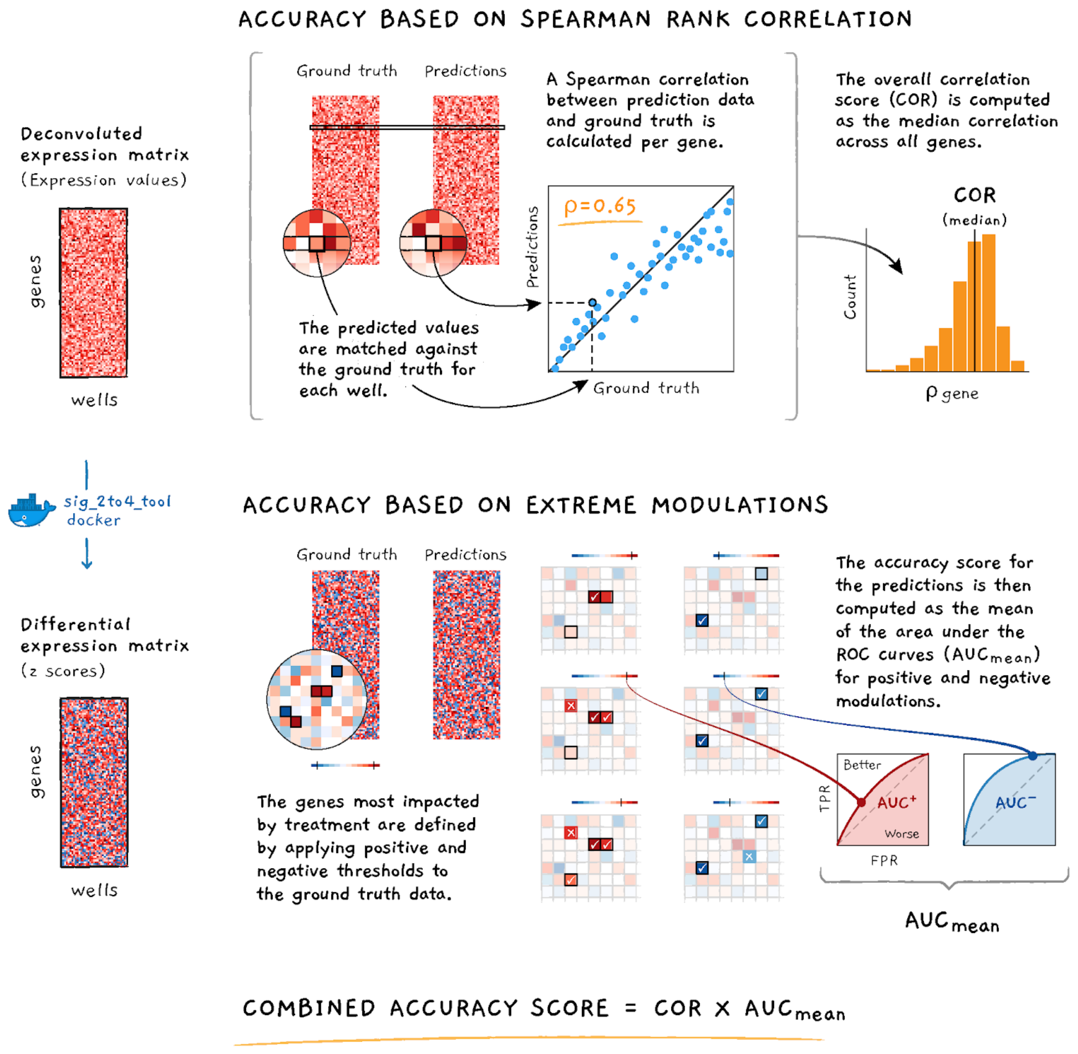
5.4 Gene knockdowns

5.5 High / Low bead discrepancy

```
## reading data/DECONV_holdout_n8228x976.gctx
```

```
## done
```

```
## Formal class 'GCT' [package "cmapR"] with 7 slots
##   ..@ mat    : num [1:976, 1:8228] 4722 6557 735 1462 1430 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:976] "55847" "7416" "10174" "25803" ...
##   .. .. ..$ : chr [1:8228] "ground-truth:DPK.CP003_PC3_24H_X1_B42:A03" "AliGebily:DPK.CP003_PC3_2
##   ..@ rid    : chr [1:976] "55847" "7416" "10174" "25803" ...
##   ..@ cid    : chr [1:8228] "ground-truth:DPK.CP003_PC3_24H_X1_B42:A03" "AliGebily:DPK.CP003_PC3_
##   ..@ rdesc  :'data.frame':  976 obs. of  1 variable:
##   .. ..$ id: chr [1:976] "55847" "7416" "10174" "25803" ...
##   ..@ cdesc  :'data.frame':  8228 obs. of  1 variable:
```


ACCURACY BASED ON SPEARMAN RANK CORRELATION
Deconvoluted expression matrix
(Expression values)
genes
wells
Ground truth
Predictions
Predictions
ρ=0.65
Ground truth
COR
(median)
Count
ρ gene
sig_2to4_tool docker
Differential expression matrix
(z scores)
genes
wells
ACCURACY BASED ON EXTREME MODULATIONS
Ground truth
Predictions
Better
Worse
AUC+
AUC−
TPR
FPR
AUCmean

A Spearman correlation between prediction data and ground truth is calculated per gene.

The overall correlation score (COR) is computed as the median correlation across all genes.

The predicted values are matched against the ground truth for each well.

The genes most impacted by treatment are defined by applying positive and negative thresholds to the ground truth data.

The accuracy score for the predictions is then computed as the mean of the area under the ROC curves (AUC_{mean}) for positive and negative modulations.

COMBINED ACCURACY SCORE = COR X AUC_{mean}

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.

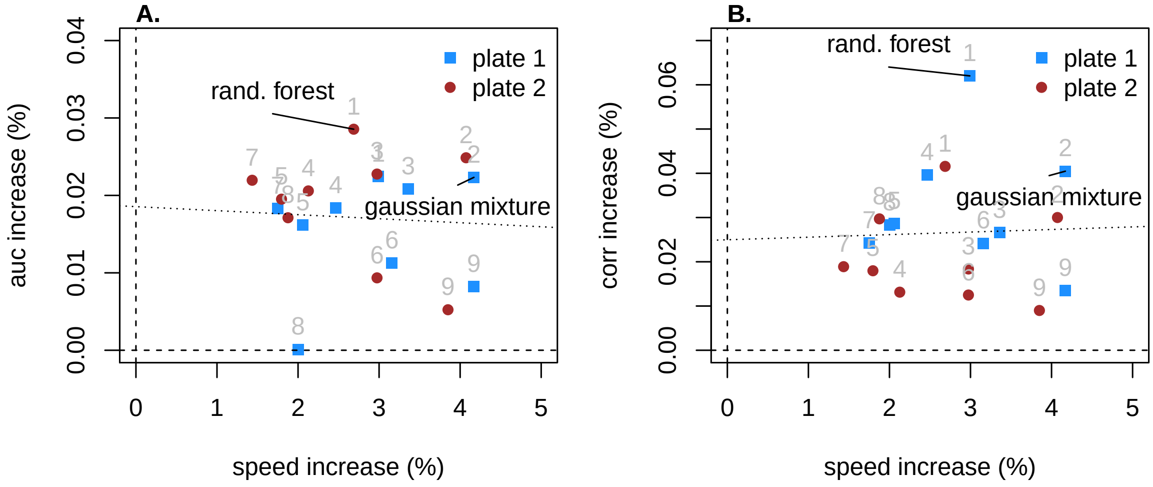
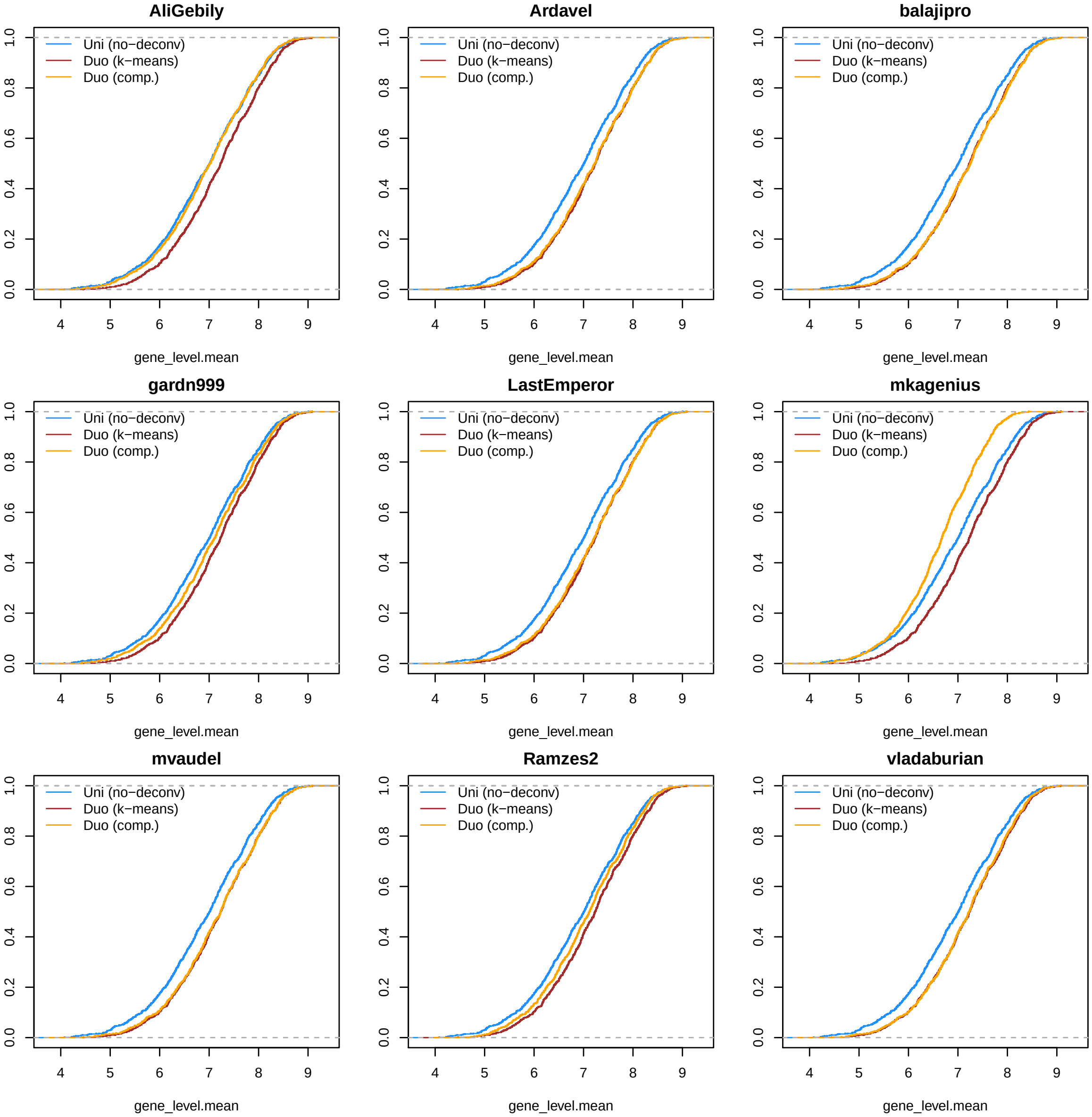
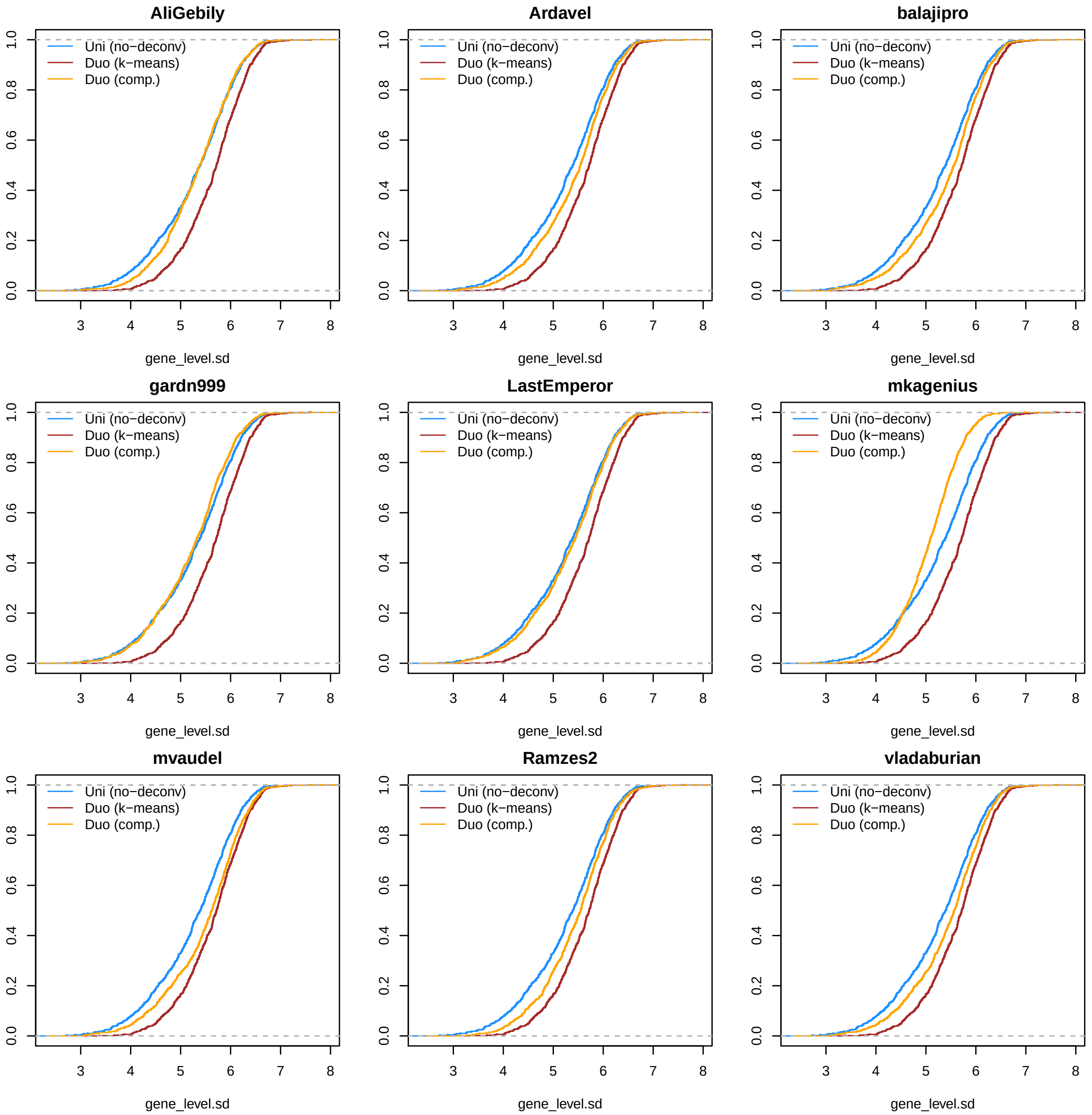


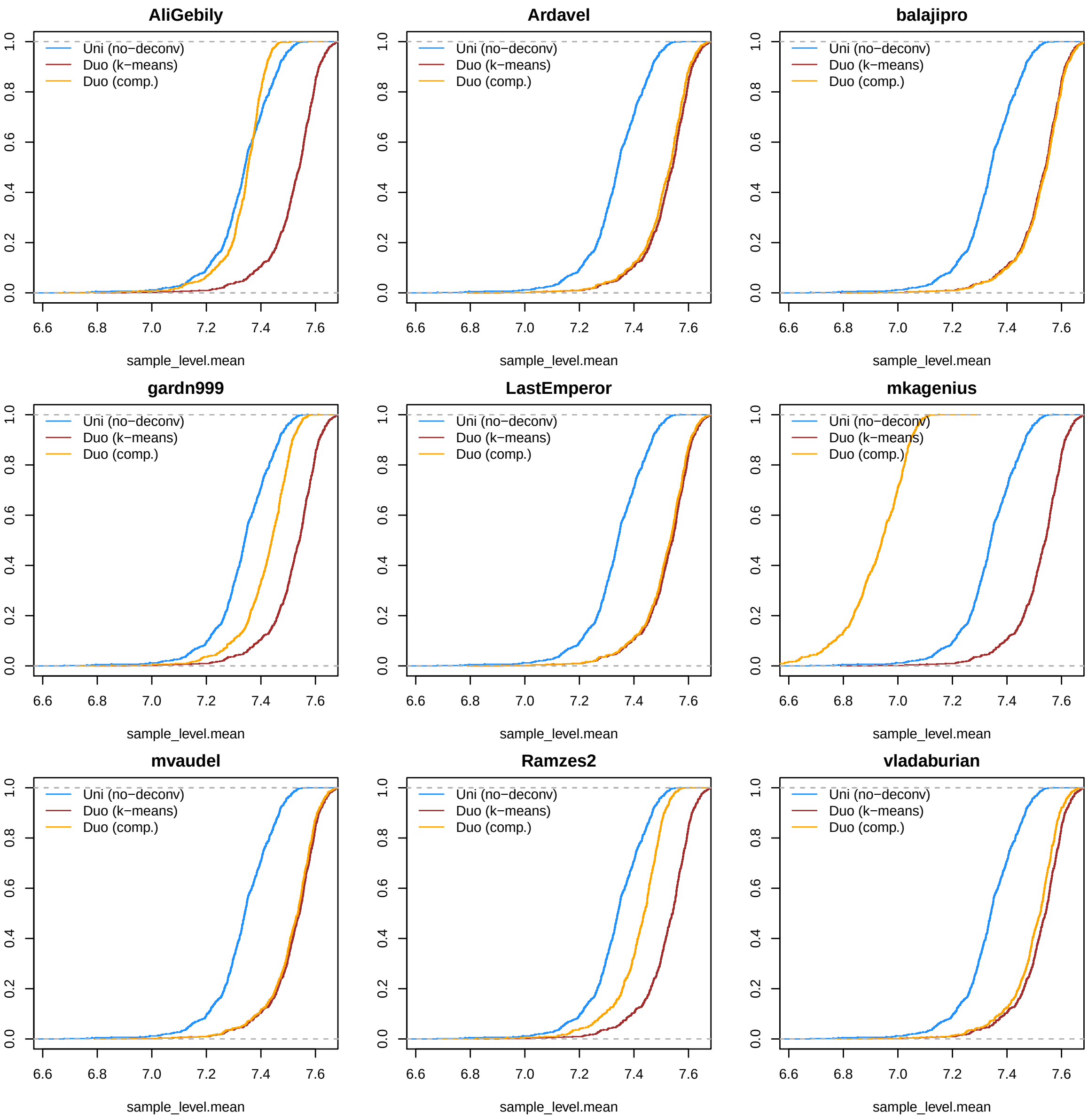
Figure 2: **Accuracy vs speed.** This figure shows the relationship between improvements in AUC (A.), correlation (B.) and fraction of correctly predicted knocked-down genes(C.) and improvements in speed for the top-nine solutions. These measures are computed for each of the plates on the holdout dataset. Labels at the dots indicate the rank of the submission in the final leaderboard (1 = first best, 2 = second best, etc.). Dashed lines indicate levels with no improvement over the benchmark. Dotted lines indicate (ols) association between accuracy and speed improvements. The name of the most accurate and fastest approach are highlighted in each panel.

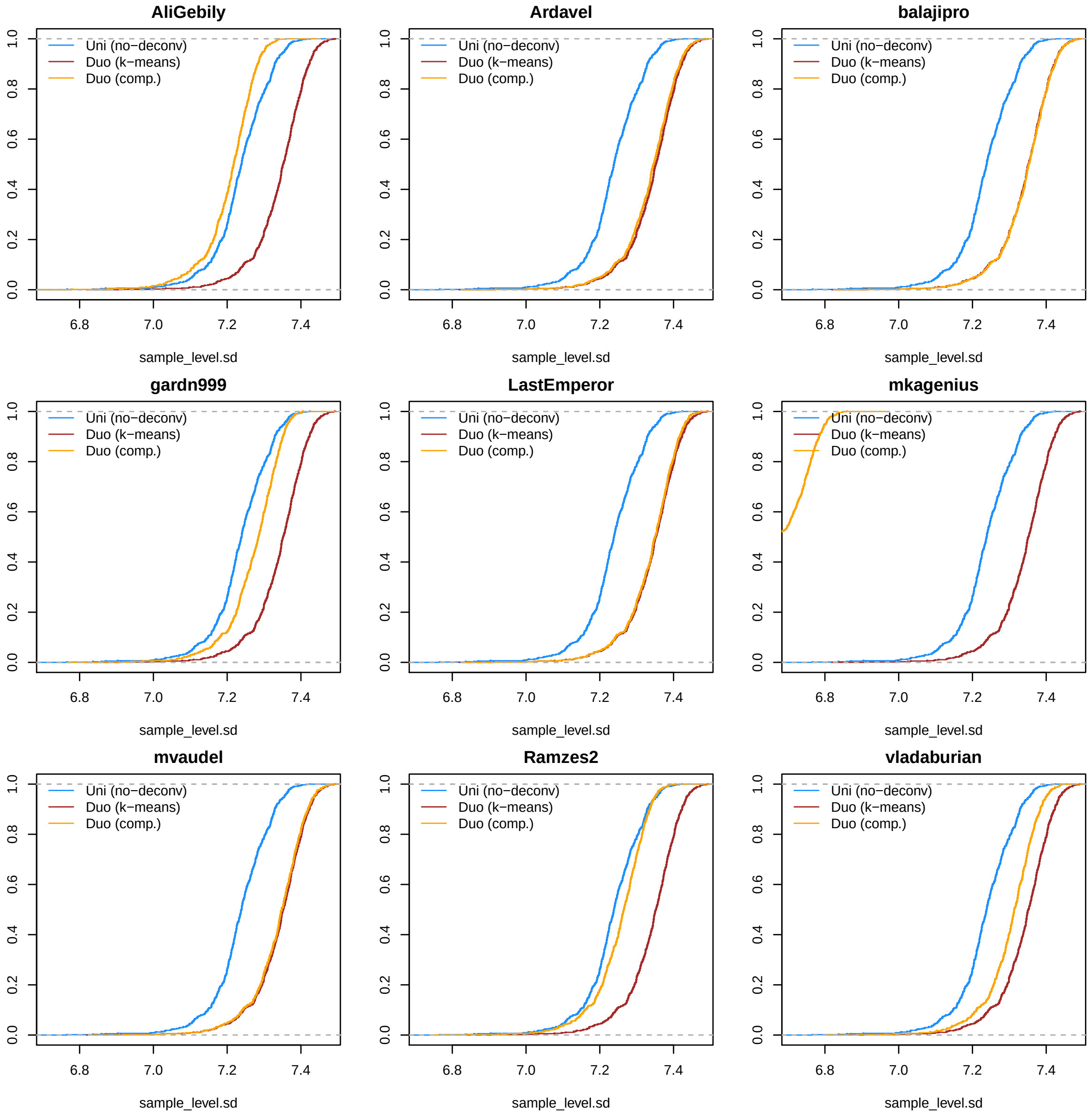
```
##   .. ..$ id: chr [1:8228] "ground-truth:DPK.CP003_PC3_24H_X1_B42:A03" "AliGebily:DPK.CP003_PC3_24
##   ..@ version: chr(0)
##   ..@ src    : chr "data/DECONV_holdout_n8228x976.gctx"
```

	55847	7416	10174	25803	466
Est	4692	6159	630	1099	1310
SE	415	441	104	346	250
Est	5436	5569	1018	1153	1486
SE	597	625	293	374	282

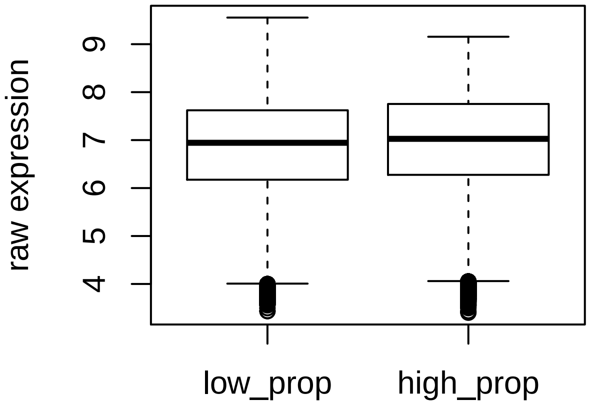

AliGebily
Ardavel
balajipro
gardn999
LastEmperor
mkagenius
mvaudel
Ramzes2
vladaburian
Uni (no−deconv)
Duo (k−means)
Duo (comp.)
0.0
0.2
0.4
0.6
0.8
1.0
4
5
6
7
8
9
gene_level.mean


AliGebily
Ardavel
balajipro
gardn999
LastEmperor
mkagenius
mvaudel
Ramzes2
vladaburian
Uni (no−deconv)
Duo (k−means)
Duo (comp.)
0.0
0.2
0.4
0.6
0.8
1.0
3
4
5
6
7
8
gene_level.sd


AliGebily
Ardavel
balajipro
gardn999
LastEmperor
mkagenius
mvaudel
Ramzes2
vladaburian
Uni (no−deconv)
Duo (k−means)
Duo (comp.)
0.0
0.2
0.4
0.6
0.8
1.0
6.6
6.8
7.0
7.2
7.4
7.6
sample_level.mean



```r
# example - plot raw expression values by high vs low bead in ground truth
row_groups <- with(gt_ds@rdesc, split(1:nrow(gt_ds), high_prop))
boxplot(list(
  low_prop = log(gt_ds@mat[row_groups[[1]], ]),
  high_prop = log(gt_ds@mat[row_groups[[2]], ])
), ylab="raw expression")
```



5.6 Clustering of solutions

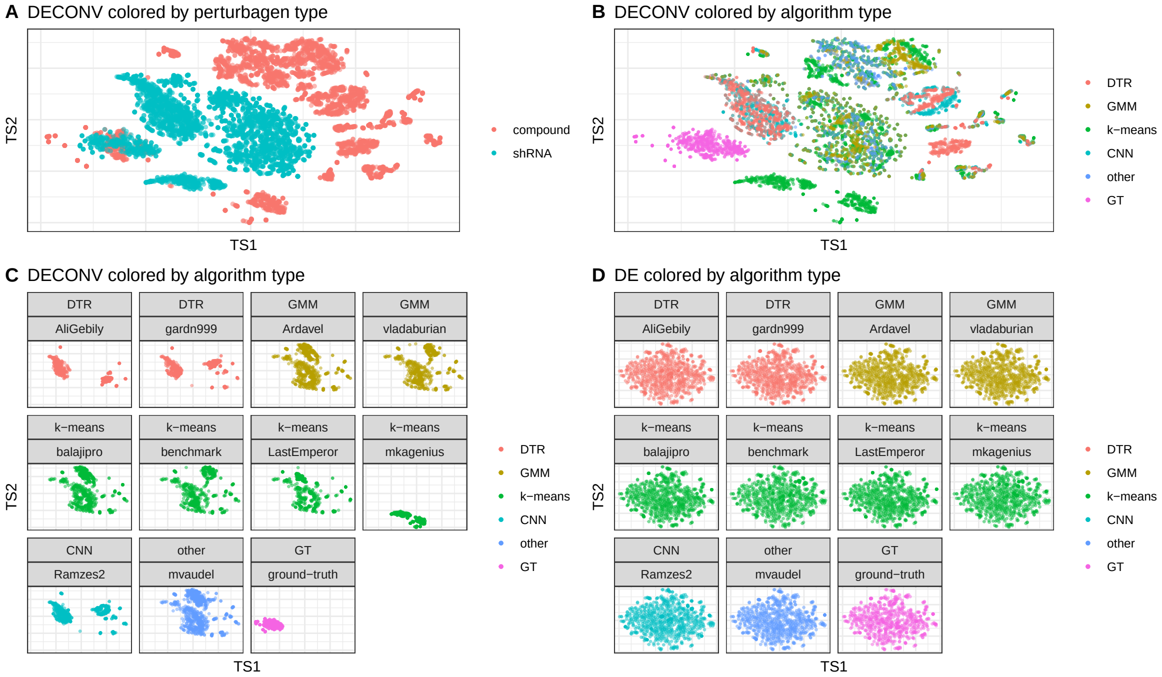


Figure 3: **t-SNE projection of deconvoluted data.** Each point represents the 2D projection of a sample generated by UNI ground truth (GT) or by applying a deconvolution algorithm to DUO data. t-SNE was run on the 2 plates of holdout data, one each containing compound and shRNA treatmens. DECONV data 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.

5.7 To read

- Compound signature detection on LINCS L1000 big data used a fuzzy c-means Gaussian Mixture Model (GMM) to process raw L1000 data, showing better performance compared to KNN. This method is described below:

  > To deconvolute such overlapped peaks, we assumed that the fluorophore intensities of each analyte type (corresponding to a specific mRNA type) had a Gaussian distribution. The distribution of the mixture of analytes GeneH(i) and GeneL(i) corresponding to the expression levels of GeneH and GeneL, respectively, should be subject to a bimodal Gaussian distribution, with the proportion of 1.25 to 0.75. We initialized the estimations of the two Gaussian distributions using buzzy c-means clustering [11] and estimated the GMM parameters using the Nelder-Mead method [12]. Thus, the overlapped peaks were deconvoluted as the two estimated Gaussian peaks and the expression levels of the two genes sharing the same analyte were extracted. Mathematical details are included in the Supplementary Methods (the GMM model).

- Deconvolution of linear systems by constrained regression and its relationship to the Wiener theory
- Efficient Bayesian-based multiview deconvolution
- A Bayesian deconvolution strategy for immunoprecipitation-based DNA methylome analysis
- Gene expression deconvolution in linear space
- Cell type–specific gene expression differences in complex tissues

6 References