Cell Type Deconvolution and Transformation of Microenvironment Microarray Data

by

Gregory J. Hunt

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Statistics)
in the University of Michigan
2018

Doctoral Committee:

Assistant Professor Johann A. Gagnon-Bartsch, Chair
Professor Jun Li
Professor Kerby Shedden
Professor Naisyin Wang

Gregory J. Hunt
gjhunt@umich.edu
ORCID ID: 0000-0001-7794-8404

To my friends, family, and especially Robin.

ACKNOWLEDGMENTS

I would like to thank Johann, Julie, Laura, Mark, Melanie and Saskia. Without their advice and guidance this work would not be what it is today.

Table of Contents

List of Figures

List of Tables

Abstract

Transformations are an important aspect of data analysis. In this work we explore the impact of data transformation on the analysis of high-throughput -omics data. Specifically, we explore two applications where data transformation plays an important role. The first application is estimating cell types using gene expression data. Here we develop dtangle, a method that carefully considers scale transformations when estimating cell type proportion estimates. This method broadly outperforms existing deconvolution methods in a comprehensive meta-analysis. Secondly, we explore the role of simple data transformations for the analysis of microenvironment microarray data. In this section we look at simple data transformations and how they interact with visualization, discovery of latent effects, and data integration. We find that simple transformations applied alone or in sequence can make salient important aspects of the data.

Chapter 1

Introduction

1.1 Background, Motivation, and Overview of What is Accomplished in This Work

Recent and widespread adoption of high-throughput bio-technology has produced an abundance of data in fields like genomics, metabolomics, proteomics, and others. Collectively known as high-throughput "-omics" all of these fields use high-throughput experiments to investigate the role of bio-molecules in tissue and cell-level processes. These experiments are called "high-throughput" because they allow a large number of simultaneous measurements. For example, RNA-sequencing is used to simultaneously assay tens-of-thousands of mRNA transcripts. Similarly, microenvironment microarrays can study the interaction between a cell line and thousands of microenvironments in parallel.

An important consideration in the analysis of such high-throughput data is the choice of scale. Often, transforming data to another scale can benefit analysis. For example, a scale transformation might make application of methods more robust. Similarly, it might help emphasize the important variation in the data. In this work we will explore two cases where the careful application of scale transformations can enhance the analysis of high-throughput -omics data. Those two cases are

1. estimating cell types from genomic data,
2. discovery of latent effects in microenvironment microarray data.

These two applications encompass Chapters 2 and 3, respectively. In the remainder of this section we will briefly introduce these topics and summarize our findings.

In Chapter 2 we introduce dtangle, a method for estimating cell type proportions from microarray and bulk RNA-seq data. This problem is known as cell type deconvolution. While data scale has generally been explored in the context of genomic data this discussion has not been fully imported into the literature of estimating cell type proportions. Our method dtangle approaches estimating unknown cell types through the lens of data scale. The method proposes a mixing model

of the biological process on a linear scale but then fits the model on a logarithmic scale. These dual scales combine a plausible biological model on the linear scale with a robust fitting procedure on the log scale. Broadly, we find that dtangle out-performs existing deconvolution methods in a comprehensive meta-analysis of methods over real and simulated data. Finally, in an application on gene expressions from patients with Lyme disease, we demonstrate that dtangle's estimates are consistent with previous findings. Supplementary information for dtangle is contained in Chapter 5.

In Chapter 3 we consider scale transformations of microenvironment microarray (MEMA) data. Our aim is to explore how these transformations enhance visualization and discovery of latent technical and biological effects. We focus our exploration on three transformations: (1) a Gaussianizining non-linear scale change, (2) a robust z-score transformation, and (3) a transformation to remove outliers. We find that the first and third transformations help ameliorate misleading effects of skewness and outliers. They consequently make prominent other important effects in the data. We also find that the second transformation, a robust z-score, makes integration of features simple. Altogether, These three transformations individually and in sequence help make salient important latent effects. Supplementary information for this work is contained in Chapter 6.

Chapter 2

dtangle: Accurate and Robust Cell Type Deconvolution

2.1 Abstract

Motivation: Cell type composition of tissues is important in many biological processes. To help understand cell type composition using gene expression data, methods of estimating (deconvolving) cell type proportions have been developed. Such estimates are often used to adjust for confounding effects of cell type in differential expression analysis (DEA).
Results: We propose dtangle, a new cell type deconvolution method. dtangle works on a range of DNA microarray and bulk RNA-seq platforms. It estimates cell type proportions using publicly available, often cross-platform, reference data. We evaluate dtangle on eleven benchmark data sets showing that dtangle is competitive with published deconvolution methods, is robust to outliers and selection of tuning parameters, and is fast. As a case study, we investigate the human immune response to Lyme disease. dtangle's estimates reveal a temporal trend consistent with previous findings and are important covariates for DEA across disease status.
Availability: dtangle is on CRAN (cran.r-project.org/package=dtangle) or github (dtangle.github.io).

2.2 Introduction

Complex organisms have a vast collection of specialized cell types. The presence and interaction of these cell types is important to understanding many biological processes. For example, shifts in the relative composition of cell types is important to developmental processes of organisms including embryogenesis, morphogensis, cell differentiation and growth [Lu *et al.*, 2003]. Likewise, understanding the presence or absence of cell types is of direct etiological interest for many diseases and dysfunctions [Newman *et al.*, 2015; Abbas *et al.*, 2009; Altboum *et al.*, 2014; Lu *et al.*, 2003]. For example, changes in glial populations in brain tissue are characteristic of Alzheimer's disease [Mohammadi *et al.*, 2015]. Similarly, white blood cell composition can be indicative of

acute cellular rejection of transplanted kidneys [Shen-Orr *et al.*, 2010]. Cell type composition is also important in tumorigenic processes. It has been shown that heterogeneity of tumors cells is implicated in the metastatic potential of cancer [Marusyk and Polyak, 2011; Lu *et al.*, 2003].

Given the importance of understanding cell type composition, several methods to estimate cell type proportions using high-throughput gene profiling experiments have been developed. Known as "cell type deconvolution", these methods have been successfully employed in a variety of applications. Deconvolution algorithms have been used to study cell type compositional changes in patients in clinical studies [Newman *et al.*, 2015; Abbas *et al.*, 2009; Gong *et al.*, 2011; Altboum *et al.*, 2014; Bowling *et al.*, 2017]. In these studies, estimating constituent cell types of carefully selected tissues reveals important cell type compositional dynamics of diseases. Similarly, such gene expression deconvolution has been posited as useful for clinical cell type monitoring, for example, by tracking patients' leukocytes [Newman *et al.*, 2015]. Finally, estimating cell type proportions is important for deconfounding differential expression analysis. In differential expression studies detecting gene expression differences within each cell type is confounded by changes in the cell type composition across the factor of interest. For example, diseases will simultaneously affect changes in gene expression within each cell type and through compositional changes in the tissues. Including estimated proportions of cell types to account for this confounding has been shown to improve differential expression analysis [Capurro *et al.*, 2015; Hagenauer *et al.*, 2016].

We present dtangle, a new deconvolution method that is accurate, robust, and simple to compute. It estimates cell type proportions using biologically plausible models of high throughput profiling technology. We compare dtangle to other methods on 11 benchmark data sets. These data sets include many different cell types, profiling technologies, and cover realistic scenarios like batch effects, mixed technologies, and third party references. Analysis of this data shows that dtangle out-competes existing methods in a broad range of applications.

2.3 Materials and Methods

dtangle requires two pieces of external knowledge: (1) reference data and (2) marker genes. First, dtangle requires auxiliary gene expression reference data for each cell type (e.g. from GEO [Edgar, 2002]). Second, dtangle requires marker genes for each cell type. A gene is defined as marker of a cell type if it is predominantly expressed by that type. dtangle can determine marker genes using the reference data or they may be specified by the user.

dtangle's approach is built on a biologically appropriate linear mixing model of linear-scale expressions but robustly fitting the model using log-transformed data and thus sets it apart from other deconvolution methods.

2.3.1 The dtangle Estimator

In this section we describe the mathematical form of dtangle's estimator. Intuition for the estimator follows in subsequent sections. Assume we have a mixture sample of K cell types. Let $Y \in \mathbb{R}^N$ be the (base-2) log-scale expression measurements of this mixture sample and $p_1, \ldots, p_K$ be the mixing proportions of the cell types. For $k = 1, \ldots, K$ assume that there are ν_k reference samples of cell type k and let $Z_{kr} \in \mathbb{R}^N$ be the log-scale expressions of the r^{th} type k reference. Furthermore, let $G_k \subset \{1, \ldots, N\}$ be the set of type k marker genes. These marker gene sets are mutually disjoint.

Let $g_k = |G_k|$ and define $\overline{Y_{G_k}} = \frac{1}{g_k} \sum_{n \in G_k} Y_n$ and $\overline{Z_{G_k}} = \frac{1}{g_k \nu_k} \sum_{n \in G_k} \sum_{r=1}^{\nu_k} Z_{krn}$ to be the average of all type k marker genes across the mixture and reference samples, respectively. Define $D_{kt} = \frac{1}{\gamma} \left(\left(\overline{Y_{G_k}} - \overline{Y_{G_t}} \right) - \left(\overline{Z_{G_k}} - \overline{Z_{G_t}} \right) \right)$ and $D_k = (D_{k1}, \ldots, D_{kK})$. The value D_{kt} is a normalized measure of the type k marker genes' expression over the type t markers' expressions in the mixture. Precisely, D_{kt} is the average difference of marker expressions, $\overline{Y_{G_k}} - \overline{Y_{G_t}}$, baseline normalized by their average difference across the references, $\overline{Z_{G_k}} - \overline{Z_{G_t}}$, and adjusted by γ, a term we discuss in detail later. We estimate p_k by mapping $D_k \in \mathbb{R}^K$ into the unit interval $[0, 1]$ by a multivariate logistic function $L_k : \mathbb{R}^K \to [0, 1]$. Precisely, for $x \in \mathbb{R}^K$ let $L_k(x) = 1/\left(1 + \sum_{t \neq k} 2^{-x_t}\right)$ and estimate p_k as

$$\widehat{p_k} = L_k(D_k) \tag{2.1}$$

(see Supplementary section 5.1 for details). This definition ensures that $\widehat{p_k} \geq 0$ and $\sum_{k=1}^{K} \widehat{p_k} = 1$.

2.3.2 Motivation and Model

Let us first define some terminology. Measured expressions are determined by a gene expression profiling (GEP) technology by measuring the amount of mRNA transcribed from each gene. Typically these measured expressions are further summarized, e.g. by MAS or RMA, and normalized, e.g. quantile or TPM normalization. We call these processed measurements the "measured gene expressions." Often, they are transformed by a logarithm to produce "log-scale" measured expressions, otherwise, they are "linear-scale". We call the true, yet unobserved, amount of mRNA transcribed from each gene the "actual expression" of the gene. This actual gene expression can also be considered on the linear-scale or the log-scale. (See Supplementary Figure 5.1 for a graphical representation of these relationships.) Given these definitions, the statistical modeling that yields the dtangle estimator (Equation 2.1) is as follows.

First we posit that actual expressions mix linearly on the linear-scale. If η_{kn} is the actual linear-scale expression of the n^{th} gene in a sample of type k cells and η_n is the actual linear-scale

expression in the mixture, then dtangle assumes

$$\eta_n = \sum_{k=1}^{K} p_k \eta_{kn}. \tag{2.2}$$

This assumption is simply that the total amount of mRNA in a mixture is the sum amount from each cell type.

Second, dtangle assumes that log-scale *measured* expressions are well modeled as linear in log-scale *actual* expressions. Statistically,

$$\begin{aligned} Y_n &= \mu + \theta_n + \gamma \log_2(\eta_n) + \varepsilon_n \\ Z_{krn} &= \alpha + \theta_n + \gamma \log_2(\eta_{kn}) + \varepsilon_{krn} \end{aligned} \tag{2.3}$$

for $n = 1, \ldots, N, r = 1, \ldots, \nu_k$, $k = 1, \ldots, K$. (Recall the $Y's$ and $Z's$ are on the log-scale and the $\eta's$ are not.) We assume uncorrelated errors ε with zero mean and finite variance.

Equation 2.3 models several important features of the transformation from actual to measured expressions by the GEP technology. First, μ and α model the samples' and references' mean measured expressions. This accounts for experimental features like quantity of mRNA or sequencing depth (for RNA-seq). We assume the references have been normalized (e.g. quantile normalized or mean centered) so that they share an intercept α. Second, θ_n accounts for gene-specific effects like length biases in RNA-seq or probe affinities in microarrays. Intuitively, γ is a factor to account for imperfect mRNA quantification. Ideally, $\gamma = 1$ meaning, on the linear-scale, increasing actual expression always leads to a proportional increase in measured expression. For RNA-seq we find $\gamma \approx 1$, however for microarray technology a γ slightly smaller than 1 helps account for saturation and attenuation of the intensity measurements for lowly and highly expressed genes (see Supplementary section 5.1). While such measuring imperfections are well-known, dtangle is the only existing method to account for them.

Finally, dtangle assumes marker genes are (approximately) expressed by only one cell type. If n is a marker gene for cell type k ($n \in G_k$), this implies

$$\eta_{\ell n} = 0 \text{ for all } \ell \neq k. \tag{2.4}$$

(This is an approximation. See Supplementary section 5.2.3 for further discussion.)

Combining Equation 2.2 with Equation 2.3 and Equation 2.4 we have

$$\begin{aligned} D_{kt} &= \frac{1}{\gamma}\left(\left(\overline{Y_{G_k}} - \overline{Y_{G_t}}\right) - \left(\overline{Z_{G_k}} - \overline{Z_{G_t}}\right)\right) \\ &= \log_2\left({}^{p_k}/_{p_t}\right) + \delta \\ &\approx \log_2\left({}^{p_k}/_{p_t}\right) \end{aligned} \tag{2.5}$$

where δ is a function of the ε's and $\delta \to 0$ as $g_k, g_t \to \infty$ (for details see Supplementary Section 5.3). Thus assuming the approximation in Equation 2.5 holds for all t then

$$D_k \approx \left(\log_2\left({}^{p_k}/_{p_1}\right), \ldots, \log_2\left({}^{p_k}/_{p_K}\right)\right)$$

and so $L_k(D_k) \approx p_k$.

2.3.3 Relationship of dtangle to other deconvolution methods

The area of "cell type deconvolution" encompasses several related inference problems. However, every deconvolution problem includes three main components: (1) measured expressions from mixture samples, (2) measured expressions from reference samples of each cell type, and (3) the proportion each mixture sample is comprised of each cell type. Typically it is always assumed that (1) is known. The deconvolution problem is then estimating either: (a) the mixing proportions, given the reference expressions, (b) the reference expressions given the mixing proportions, or (c) the proportions and the references jointly. All three problems are considered instances of deconvolution. dtangle most closely resembles problem (a), of estimating unknown mixing proportions given measured expressions from the mixture and references. In Section 2.4 we compare dtangle to methods solving both (a) and (c) since they both estimate the proportions. Problem (a), called "partial deconvolution" [Gaujoux, 2013], is typically solved as a regression or penalized regression problem [Abbas *et al.*, 2009; Gong *et al.*, 2011; Lu *et al.*, 2003; Wang *et al.*, 2006; Qiao *et al.*, 2012; Altboum *et al.*, 2014; Newman *et al.*, 2015; Valencia *et al.*, 2017], problem (c), called "full deconvolution", is usually accomplished by non-negative matrix factorization [Venet *et al.*, 2001; Repsilber *et al.*, 2010; Gaujoux and Seoighe, 2012; Zhong *et al.*, 2013].

2.3.3.1 Scale: Interpretability, Robustness, and Efficiency

Existing methods to solve problems (a), (b) or (c) are based on a common linear mixing model. Let $X \in \mathbb{R}^{S \times N}$ be the S mixture samples' N linear measured expressions, $M \in \mathbb{R}^{S \times K}$ so that M_{sk} is the percentage of type k cells in sample s, and $U \in \mathbb{R}^{K \times N}$ so that the K rows of U are reference expressions of the K cell types. Existing methods presume a linear mixing model on either the

linear scale,

$$X \approx MU, \tag{2.6a}$$

or the logarithmic scale,

$$\log(X) \approx M \log(U). \tag{2.6b}$$

They then solve for (a) M, (b) U (equiv. $\log(U)$) or (c) both, presuming the other components are known.

Both Equation 2.6a and Equation 2.6b have advantages and drawbacks. Equation 2.6a is a physically plausible linear mixing model of linear measured expressions. It posits that mRNA from a sample of cells is the sum of the mRNA from each cell. While plausible, fitting this model on the linear-scale is non-robust and statistically inefficient. The highly-skewed data means the fit is unduly influenced by data in the tail of the distribution [Li *et al.*, 2016]. Furthermore, since the variance of gene expressions typically scales with their mean, regression approaches are sub-optimal [Li *et al.*, 2016]. In contrast, Equation 2.6b models a linear mixture of log expressions. This approach is more robust since the log transformation ameliorates the skewness and heteroskedasticity. However Equation 2.6b is not physically plausible. It implicitly assumes that the mRNA in a mixture sample is the product (not sum) of the mRNA from each cell.

dtangle's approach is to take advantage of the beneficial aspects of each scale while avoiding their problems. Firstly, dtangle is based on a biologically plausible linear mixing model of linear-scale actual expressions (Equation 2.2). Second, dtangle's linear model between actual and measured expression (Equation 2.3) and definition of D_{kt} (Equation 2.5) are on the log-scale. This makes dtangle robust and statistically efficient. dtangle only transforms into the linear-scale in its final step robustly exponentiating after averaging, not before.

Similar to Equation 2.6a dtangle uses a plausible and interpretable physical model of mixing (Equation 2.2). However dtangle robustly averages log-scale expressions (Equation 2.5) and thus has robust character similar to fitting using Equation 2.6b . Supplementary section 5.2.2 uses simulations to explore these points in more depth.

2.4 Results

2.4.1 Benchmarking

To evaluate dtangle we compare it to eight other deconvolution algorithms (Supplementary Table 5.1). Six methods are accessed through the CellMix R package [Gaujoux, 2013]. We also compare to CIBERSORT and EPIC as they are recent and powerful methods [Newman *et al.*, 2015; Valencia *et al.*, 2017]. We only compare dtangle against methods that estimate cell type

proportions from gene expression data for arbitrary cell types. We do not compare to methods like xCell [Aran *et al.*, 2017] which produce enrichment scores and not percentages. We also do not compare against the many deconvolution methods for methylation data or fully-unsupervised methods whose cell types have to be inferred with further post-hoc analysis e.g. CAM [Wang *et al.*, 2016]. Furthermore, we do not compare against methods that only estimate cell type proportions from a very specific subset of cells or only in the context of a specific problem, for example, immune cell infiltration of tumors by methods like TIMER [Li *et al.*, 2016].

Like dtangle, all methods require marker genes. However four "full" deconvolution methods we analyze require only marker genes and do not explicitly require reference data. Nonetheless, we find marker genes through DEA on the reference data and so, in one way or another, all methods use reference data. There are several "completely unsupervised" deconvolution methods in the literature (e.g. Wang *et al.* [2016]) that require neither markers nor references. However their estimates are difficult to interpret biologically unless reference data is used post-hoc to map proportions to cell types. For this reason we do not compare to such methods. Finally, while full deconvolution algorithms also estimate type-specific expressions profiles, we only compare dtangle to their estimated mixing proportions as this is what dtangle estimates.

We choose marker genes for deconvolution following Abbas *et al.* [2009]. First we restrict analysis to genes in the the highest quartile of variance. We then rank genes by p-value using a t-test between the reference expressions of the two most highly expressed cell types. For each cell type, the 10% of genes with lowest p-values are designated markers.

Note that many genes selected as markers using this approach do not exactly satisfy Equation 2.4 . Further filtering the set of marker genes to attempt to ensure they satisfy Equation 2.4 could potentially improve the performance of dtangle. However, in our analysis we nonetheless follow the method of Abbas *et al.* [2009] without any further filtering to ensure that the method of marker selection is not biased in favor of dtangle. The exact same set of marker genes are used for each algorithm.

2.4.2 Data Sets Compared

We compare dtangle to the eight other algorithms across eleven benchmarking data sets (Supplementary Table 5.2). The true mixing proportions are known for each data set either because the experiment was conducted by mixing each cell type in known proportions or because an independent physical sorting technique, like flow cytometry, was used to estimate the proportions. Most data sets include their own cell type references.

For the RNA-seq data we TPM normalize, transform as one plus the read count. For the microarray data we quantile normalize on a logarithmic scale. All data is re-exponentiated so it is

on the linear-scale for algorithms that require it. Pre-processing code is available in the dtangle.data R package available at dtangle.github.io.

2.4.3 Microarray Data

2.4.3.1 Mixture Experiments With References

We consider five microarray mixture experiments: data sets Abbas, Kuhn, Gong, Shi and Shen-Orr (Supplementary Table 5.2). For each algorithm we estimate the mixing proportions in each data set. We evaluate the algorithms' accuracy in terms of absolute error of estimated proportions from true proportions and by Pearson correlation and R^2 of the estimates against the truth for each cell type. dtangle has the lowest median error, second lowest mean error, and the highest mean and median correlation and R^2 across the data sets (Supplementary Figure 5.4). Furthermore dtangle has the lowest variability for both absolute error, correlation, and R^2. This meta-analysis shows that for the microarray mixture experiments dtangle is the most accurate algorithm but it is also one of the most consistently accurate algorithms. For each data set supplementary boxplots of error, correlation, R^2, as well as scatter plots may be found in Supplementary Figures 5.13, 5.15, 5.16, 5.22, 5.23.

We highlight comparisons between dtangle, CIBERSORT, and EPIC on two data sets where dtangle performs worst and best relative to other algorithms (Figure 2.1a and Figure 2.1b). For the Gong data blood and breast tissue were mixed in known proportions. While dtangle does poorly relative to other deconvolution algorithms it still performs quite well. The estimated mixing proportions are still highly correlated with the truth (see Supplementary Figure 5.15). Conversely, the Shen-Orr data is from a microarray mixture experiment where rat liver, brain and lung cDNA were mixed in known proportions. Here, dtangle performs as well or better than the other algorithms (Figure 2.1b, Supplementary Figure 5.22). dtangle performs on par with a strong algorithm like CIBERSORT and out-performs a method like EPIC.

2.4.3.2 Mixtures Without References

In practice pure reference samples of each cell type are not typically generated along with the mixed samples to be deconvolved. In this case existing reference data for each of the cell types to deconvolved must be procured. Typically these pure reference samples are collected from repositories like GEO.

The Becht data set is a mixture experiment where cDNA from the HCT116 colorectal carcinoma line and various leukocytes (NK, B, neutrophils, T, and monocytes) were mixed in known quantities and analyzed with an Affymetrix microarray. Unlike previous data sets no reference data was produced as part of the mixture experiment. Like the authors we use publicly available

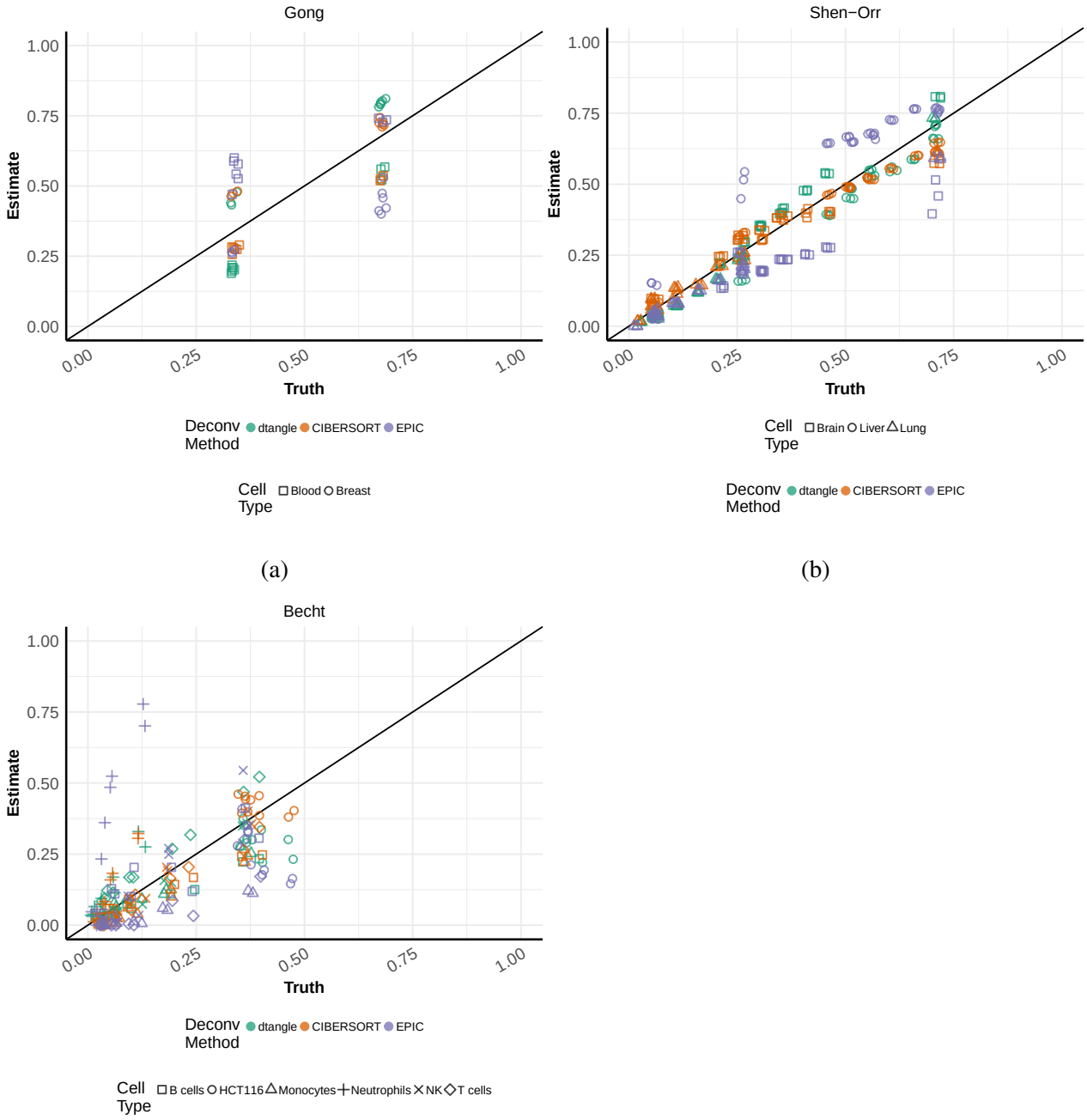


Figure 2.1: Scatter plots of dtangle, CIBERSORT, and EPIC on the Kuhn, Shen-Orr and Becht datasets. Each point is a particular cell type in a sample.

expression data from GEO as references for each cell type. In total there are 776 samples gathered from GEO which we use to create reference profiles for the six cell types. On this data dtangle performs as well or better than strong methods like CIBERSORT and EPIC (Figure 2.1c, Supple-

mentary Figure 5.14). dtangle has commensurate mean/median error, correlation and R^2 as these methods.

2.4.3.3 Performance evaluation with flow cytometry based cell sorting

Mixture experiments are only a surrogate for cell mixtures found in organisms. Realistically, deconvolution methodology is applied to complex tissue extracted from an organism. Such tissue will be a mixture of many cell types (more types than in a typical mixture experiment) and the cell types will have complex inter-cellular interactions modifying their gene expressions. The difficulties in estimating cell type proportions from such complex tissue is likely only partially explored by a mixture experiment.

The Newman follicular lymphoma (FL) data was generated by taking lymph node biopsy samples and enumerating immune cell sub-types using flow cytometry [Newman *et al.*, 2015]. This process identified 3 leukocyte types (B, CD4 T and CD8 T) in various proportions across samples from 14 patients. As cell type expression reference data we use the same reference data used to create the LM22 reference by Newman *et al.* [2015]. It contains gene expressions of 22 white blood cell types as references. Similar to Newman *et al.* [2015] we group these 22 types into 12.

The Newman peripheral blood mononuclear cells (PBMC) data was generated from blood samples from twenty adults where the proportions of nine types of leukocytes were determined by flow cytometry. We again use the same data to create references as used to create the LM22 data set [Newman *et al.*, 2015]. dtangle compares well with other deconvolution methods on these two data sets (Supplementary Figure 5.19, 5.20). For the Newman PBMC data set dtangle has the highest average correlation and lowest average error. For the Newman FL data dtangle has the highest average correlation however the overall accuracy suffers somewhat because of biases in the CD4T and B cell types. This may be due to the large number of cell types making it difficult for our markers to distinguish among them.

To investigate the effect of marker gene selection we re-analyze both the Newman FL and PBMC data sets using the exact LM22 signature matrix used in Newman *et al.* [2015] (see Supplementary Figure 5.25, 5.26). The LM22 signature matrix is a highly curated set of marker genes for 22 PBMCs developed by Newman *et al.* [2015]. The results largely remain the same however the biases largely disappear for dtangle. In particular, dtangle is across the board the best performing method on the Newman PBMC data and dtangle has the highest average correlation and R^2 for the Newman FL data. This further underlines the fact that choosing references and markers is an important component of deconvolution and needs to be considered carefully.

2.4.4 RNA-seq

We also investigate the performance of deconvolution methods on RNA-seq mixture experiments (Supplementary Figure 5.5). The Liu and Parsons data sets are RNA-seq mixture experiments with internal reference data. The Linsley data set is a realistic data set of leukocytes extracted from patients where the true proportions are determined by flow-cytometry and external references are used. dtangle, CIBERSORT, EPIC and LS Fit seem to be the best algorithms across the RNA-seq data sets. For each data set supplementary boxplots of error, correlation, R^2, as well as scatter plots may be found in Supplementary Figures 5.17, 5.18, 5.21.

2.4.5 Meta-analysis

We compare dtangle to the other algorithms in a meta-analysis (Figure 2.2). dtangle has the lowest median error and second lowest mean error of all methods. Similarly dtangle has the highest mean and median correlation and R^2 across datasets. Thus dtangle's approach is a general purpose cell type deconvolution algorithm that works well across many technologies and tissue types. Even if we first logarithmically transform the data and then modify the other methods so that they fit using log-scale expressions dtangle still performs strongly. Indeed, after this transformation dtangle does better, it has the lowest mean/median error, and highest mean/median correlation and R^2 (see Supplementary Figure 5.3). This shows that even if we make robust enhancements to existing deconvolution methods dtangle still broadly out-performs existing approaches. (Note that using log-scale data for the other methods is for illustrative purposes only. We do not generally recommend it as the models and associated software of many of the other methods like CIBERSORT and EPIC explicitly require linear scale expressions. For our analysis we had to modify their code to allow log-scale expressions.)

2.4.6 Robustness to marker selection

Thus far we have been selecting marker genes by, among the top 25% most variable genes in the references, ranking marker genes following Abbas *et al.* [2009] with a t-test p-value between the top two most expressed cell types for each gene and selecting the top 10% of differentially expressed genes.. To analyze the sensitivity of dtangle to how the markers are ranked we consider another way of ranking marker genes. This second method looks, for each gene, at the ratio of the mean expression for each cell type to the sum of the mean expressions of the gene by all other cell types. We call the Abbas method "p-value" and call this latter approach "Ratio." While we recommend this latter method for dtangle, thus far we have used the p-value ranking so as to be conservatively fair in our comparison.

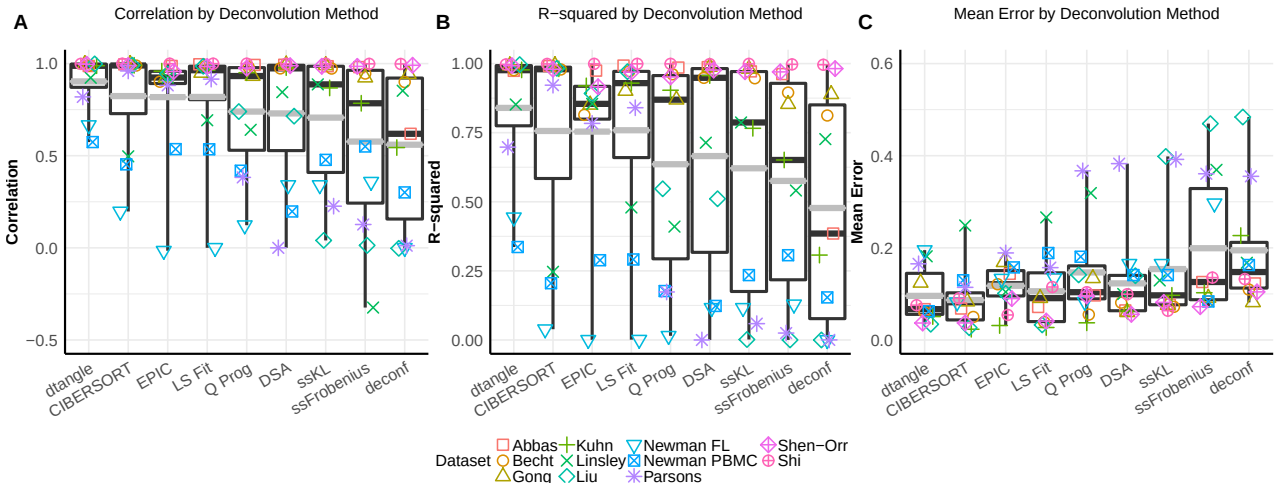


Figure 2.2: Meta-analysis of deconvolution algorithms. Side-by-side box plots of the mean errors, correlations, and R^2 across the algorithms. The bold black line is median, the grey line is mean. Overlapping are jittered points of the metric for each data set.

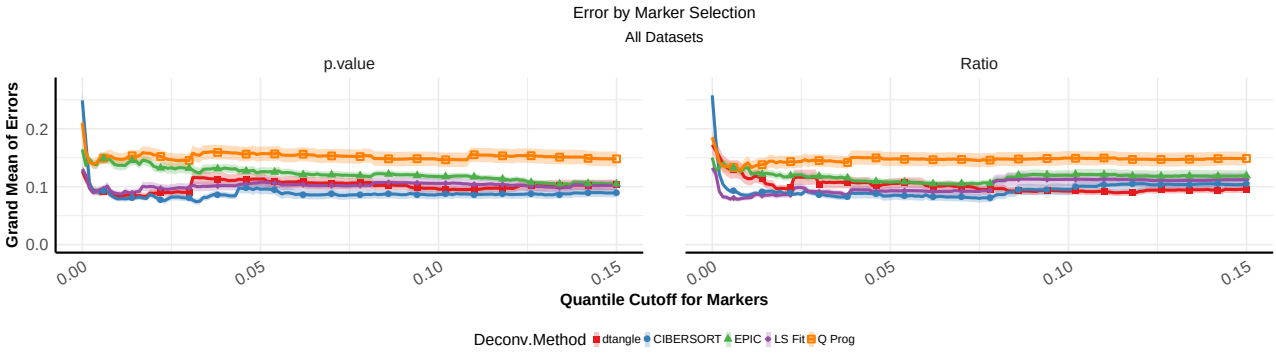


Figure 2.3: Grand error means across marker ranking methods (p-value and Ratio) and number of markers. 95% confidence bands included.

In Figure 2.3 we look at the grand error mean of each algorithm across all data sets for a range of marker tuning parameters. We compare partial deconvolution algorithms as they are the most competitive with dtangle. dtangle is robust to the way markers are ranked (p-value or Ratio) and ranking threshold (quantile cutoff) determining the number of markers to use. In Supplementary Figure 5.6, 5.7, and 5.8 we include similar plots looking at the median error, and mean and median correlation and R^2 for all data sets, microarray data sets, and RNA-seq data sets. Looking these various metrics, we see that Q. Prog and EPIC (and sometimes LS Fit and CIBERSORT) appear sensitive to the quantile cutoff and to which ranking method is chosen (p-value or Ratio). Across all metrics, dtangle is consistently competitive with existing methods and is not as sensitive to small changes in such tuning parameters.

Marker gene selection also influences computational time. For each data set we timed all

algorithms across a range of quantile cutoffs using p-value ranking (Supplementary Figure 5.9). dtangle is consistently the fastest algorithm. It is between one and four orders of magnitude faster than other algorithms regardless of what quantile cutoff is used.

2.4.7 Application To Lyme Disease

To demonstrate dtangle on a biological problem we consider RNA-seq data of PBMCs from Lyme disease patients [Bouquet *et al.*, 2016]. To better understand persistent Lyme symptoms (e.g. fatigue or arthritis) it is of interest to understand the progression of the human immune response to Lyme [Bouquet *et al.*, 2016]. To this end Bouquet *et al.* measure gene expression in a subset of white blood cells (PBMCs). PBMCs of 28 patients were collected at the point of diagnosis (V1), after a 3-week course of doxycycline (V2) and 6 months later (V5). PBMCs from 13 matched controls were also collected (C).

We use dtangle to estimate, for each sample, the cell type proportions of nine types of PBMCs (B, dendritic, macrophages, mast, monocytes, NK, CD4 T, CD8 T and gamma-delta T). We use as reference the LM22 data set from Newman *et al.* [2015], choosing the top 10% of differentially expressed genes for each cell type as markers. We find that the phagocytes (dendritic, macrophages, mast and monocytes) make up a larger percentage of the patients' PBMCs earlier, rather than later, in the infection (Supplementary Figure 5.27). We see a large difference between the control group and V1 and decreasing differences between the controls and V2 and V5. Natural killer (NK) cells follow this same pattern.

The estimated cell type percentages agree with the current understanding of Lyme. The initial infection induces an immune response where fast-acting phagocytes are recruited to attack the foreign bacteria [Dame *et al.*, 2007]. This agrees with dtangle's estimates of a relatively large percentage phagocytes early in the infection that decreases with time. Phagocytes decrease in numbers once the bacteria has been cleared and they are no longer needed. Furthermore, NK cells follow the same pattern. This agrees with work from Horowitz *et al.* [2012] showing NK cells are rapidly activated by cytokines after a bacterial infection.

In Bouquet *et al.* [2016] the authors seek to find genes that are differentially expressed among the groups (V1, V2, V5 and C). Following Bouquet *et al.* [2016] we compare the control group to V1, V2 and V5 and find that there are 399 genes that are differentially expressed in the intersection of each of the three comparisons. This was done controlling for a FDR of 0.05 by the Benjamini-Hochberg procedure.

As this previous differential expression analysis was not corrected for cell type proportions we expect to find genes that are correlated with cell type. We add in covariates to account for composition of fast-acting cell types (phagocyte and NK). After doing so we only find 158 genes

differentially expressed in the same comparison. Thus the cell type composition changes the results of the analysis greatly. dtangle is one tool practitioners can use to help tease apart histological changes in cell composition from changes in gene expression within particular cell types.

2.5 Discussion

dtangle is a simple and robust deconvolution estimator. It is a closed-form estimator deriving from plausible biological modeling. Our meta-analyses show that dtangle is a robust and accurate, typically performing better than eight of the best existing methods across eleven diverse data sets. It can accurately deconvolve cell types using microarray and RNA-seq technology and is very fast to compute where other methods are not. Furthermore it is consistent with standard physical sorting methods like flow cytometry on realistic complex clinical tissue. Finally, dtangle has competitive accuracy when dealing with realistic data sets where the reference samples are obtained from publicly available repositories. dtangle works well even when these reference data sets were created using a different profiling technology. This points to scRNA-seq data as a promising source for references.

dtangle has some of the same limitations as other algorithms. Primarily, it is necessary that the cell types comprising each sample be known in advance and that reference data is available. Furthermore dtangle needs to find marker genes for each cell type. This can be potentially difficult if there are many cell types or the cell types are closely related. Nonetheless, dtangle seems to perform well in many situations. We will continue to develop dtangle to overcome some of these challenges to broaden its utility.

Chapter 3

Transformations of Microenvironment Microarray Data Improves Discovery and Integration of Latent Effects

3.1 Abstract

Motivation: The microenvironment of cells is broadly defined as their immediate physical and bio-chemical surroundings. This microenvironment is an important component of many fundamental cell and tissue level processes and is implicated in many diseases and dysfunctions. Thus understanding the interaction of cells with their microenvironment can further not only basic research but also aid the discovery of therapeutic agents. To study the microenvironment of cells, a new image-based cell-profiling technology called the microenvironment microarray (MEMA) has seen recent success. The relatively new nature of this technology calls for a detailed exploration of appropriate transformations for processing MEMA data.

Results: We study several simple ways of transforming MEMA data. We find that Gaussianizing the data and removing outliers can enhance visualization and discovery of latent technical and biological effects. Furthermore, a robust z-score transformation allows recovery common latent effects through an equitable integration of features. In summary, we see that the individual and sequential application of these transformations has the potential to benefit exploratory analyses.

Availability: All results and code used for analysis is available at umich.box.com/v/mematransformation

3.2 Introduction

The microenvironment of a cell encompasses its immediate physical and bio-chemical surroundings. This includes, for example, the adjacent extra cellular matrix (ECM), surrounding cells, ligands like hormones, cytokines, chemokines, growth factors, and much more. These microenvironmental components modify cellular behavior through a host of different mechanisms. Accordingly, the interaction of cells with their microenvironment is a component of many cell and tissue level processes [Lin *et al.*, 2012]. For example, the extra-cellular matrix has been long known to

regulate cellular functions like adhesion, migration, proliferation and differentiation [Teti, 1992]. The microenvironment is also implicated in the development, progression, and ultimately treatment of many diseases and dysfunctions. For example, it has been posited that communication between B-cells and their proximate stromal cells can promote malignant B-cell growth and drug resistance [Burger *et al.*, 2009]. Similarly, towards the goal of understanding therapeutic efficacy, it has recently been shown that the microenvironment of HER2-positive breast cancer cells is implicated in drug response [Watson *et al.*, 2018]. Thus a better understanding of the microenvironment benefits not only basic research but also furthers an understanding of the interaction between therapeutic agents and regulatory behavior.

To study the microenvironment, a powerful technology called the Microenvironment Microarray (MEMA) has seen recent success. The technology, first developed by Mark LaBarge at Lawrence Berkeley National Laboratory, allows the study of several thousand combinations of microenvironmental factors on molecular and biological endpoints. This is done via high-throughput image-based cell profiling technology. Specifically, a MEMA consists of a plastic substrate divided into several partitioned "wells." Each well contains an array of several hundred $\sim 300 \mu m$ "spots." Added to each spot is a collection of several hundred cells and a pair of microenvironmental perturbagens. This perturbagen pair consists of an insoluble extra-cellular matrix protein (ECMp) and a soluble ligand. The ECMps are added specifically to spots, while the ligands are added generally to the buffer solutions in wells. (The soluble ligands cannot be localized to a single spot.) Thus the cells in a spot interact with an ECMp specific to their spot and a ligand common to their well. After adding these perturbagens the cells are allowed to grow for 72 hours. Subsequently, the cells are immunofluorescently stained and imaged with high-content fluorescent microscopy.

The fluorescent microscopy images are used to quantify biological endpoints of interest like cell proliferation, differentiation, or apoptosis. Specifically, we quantify the endpoints using features we extract from the images. Typically, several hundred features are extracted from each image. These features cover a wide range of cellular aspects like stain intensity, cell count, morphological characteristics, and many more. The number of features extracted is largely a product of the sophistication of the image analysis software and the level of detail requested. Thus it is relatively easy to generate large amounts of data from MEMAs. While this plethora of data presents new opportunities for discovery it also necessitates a fresh methodological discussion. Towards this goal, this paper will systematically explore simple and robust methods for processing MEMA data. Our goal is to share the processing steps for MEMA data that we have found to enhance visualization, integration, and the discovery of important biological and technical effects.

3.3 Materials and Methods

In this section we will briefly describe the structure of MEMA data, outline our steps for processing the data, and motivate why these steps enhance visualization, integration, and discovery of latent biological and technical effects.

3.3.1 Structure of MEMA Data

In this paper we work with microenvironment microarray data from the Microenvironment Perturbagen (MEP) LINCS Center at the Oregon Health and Science University. The data is accessible through Synapse with identifiers syn10155286, syn10155292 and syn10155282 [Syn, 2018]. In total we analyze 24 MEMAs of human epithelial mammary tissue (MCF10A). The 24 MEMAs come in three batches of eight plates. Each MEMA plate is divided evenly into eight wells. Each well contains 700 spots in a 20 by 35 grid. Cells are added to the spots along with a spot-specific ECMp. Afterwards, a buffer solution containing a specific ligand is added to each well. The pattern of ECMps is identical across all wells however a (potentially) different ligand is added to each well. After incubating the cells for 72 hours they are fluorescently stained, imaged, and cell-level features are extracted with image analysis software. For the analysis in this paper, we work with spot-level features (median summarized cell-level features). For each image feature we have a data matrix of 192 wells (3 batches $\times$ 8 plates $\times$ 8 wells) by 700 spots.

3.3.2 Robust Re-scaling

To process these feature matrices we follow three sequential steps:

Three-step Robust Re-scaling (RR)
Step 1: (G) robustly "Gaussianize" the data, Step 2: (Z) convert the data to robust z-scores, Step 3: (O) remove outliers.

We will briefly look at these steps in more detail. The (G) step transforms the data using a Box-Cox-like procedure. It first estimates a Gaussianizing transformation column-wise across the feature matrix. It separately optimizes over families of power and inverse hyperbolic sine transformations to make each column of the feature matrix as normal as possible. The procedure then chooses the median transformation across columns and applies this transformation element-wise to the feature matrix. The second step (Z) is basically a z-score transformation. The (Z) step subtracts (element-wise) a global mean from the feature matrix and divides the feature matrix (element-wise) by a global estimate of the standard deviation. Finally, after the (Z) step, the outlier

removal procedure (O) simply thresholds the z-scores and marks as missing anything beyond four standard deviations. A full mathematical description of these steps is found in Section 6.1. In the remainder of this section we will motivate (1) why these steps help discovery of important latent effects in the data and (2) how this processing improves data integration.

3.3.2.1 Discovery of Important Latent Effects

An important component in the analysis of MEMA data is the discovery of latent technical and biological effects. We may be interested in such latent effects for their own sake or may be interested in removing them if they are unwanted. Examples of latent technical effects include batch across plates or wells and spatial effects within wells. Biological effects include, for example, differences in biological endpoints due to ECMps or ligands. Discovery of latent effects is typically done through visual inspection of plots or quantitative analysis like PCA. Unfortunately, such methods are often misled by prominent yet uninteresting aspects of the data.

As an example of how analyses like PCA can be misled, consider using PCA to identify groups in skewed data. Assume we have data that is the union of two highly skewed groups. If the group means are separated by a small distance (relative to their tail lengths) then the group difference will likely be over-shadowed by the long tails. In this case, PCA will identify the tail skewness, not the group difference, as the most prominent variation. To illustrate this point, in Figure 3.1 we display two plots from simulated data comprised of two log-normal groups. While we can see from Figure 3.1 (A) that the (un-transformed) log-normal data is well described the first several PCs, Figure 3.1 (B) shows that these same PCs do not well capture the group effect. However these plots show the converse that if we first log-transform the data. While we might need more PCs to describe the over-all data after a log transformation these first several PCs capture the group effect quite well. Broadly, these results are a product of the data skewness. Capturing the group effect is easier after a log transformation because the data no longer has a long distracting tail. The PCs from the skewed data are capturing skewness, not group effects. After log-transformation, this is no longer a problem.

As motivated by the previous example, we want to attenuate the influence of prominent, yet uninteresting, variation. Our processing steps described in Section 3.3.2 attempt to ameliorate the effects of two commonly encountered, and potentially misleading, aspects of MEMA data. Those aspects are (1) skewness in measurement scales and (2) anomalous outliers. By anomalous outliers we mean extremely unusual data points that are not informative of much beyond their own uniqueness. Often, these outliers are errors in the data collection or processing pipeline. For example, several blank spots are typically included on a MEMA for alignment purposes. If these spots are accidentally included in analysis they will almost certainly be anomalous.

To guard our analysis against un-interesting variation we follow the three robust re-scaling

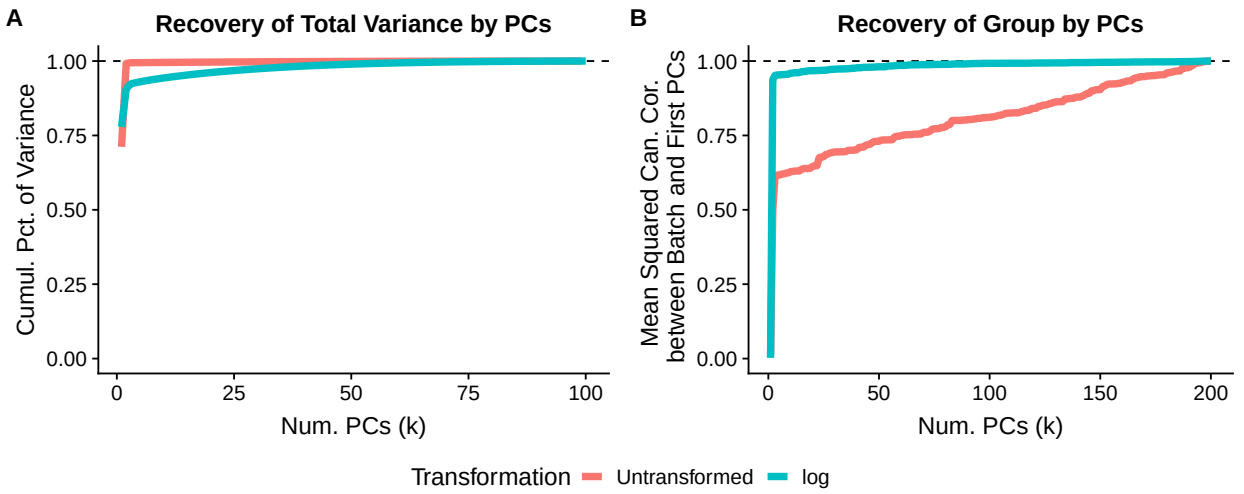


Figure 3.1: (A) The percentage of cumulative variance captured by first k principal components for both un-transformed data and log-transformed data. (B) The mean squared canonical correlations between the grouping factor and the first k principal components.

steps (G), (Z) and (O) outlined in Section 3.3.2. The (G) step is used to prevent a feature's naturally long-tailed measurement scale from dominating analysis. This is done by monotonically transforming the data to reduce skewness. Specifically, we apply a robust Box-Cox-like procedure to "Gaussianize" the data and make the feature's distribution approximately bell-shaped. This is what the log transformation does in Figure 3.1. It takes the skewed log-normal distribution and makes it approximately normal. Unfortunately, processing MEMA data is not as simple as log transforming all features. MEMAs produce hundreds of features with many different distributions. The appropriate Gaussianizing transformation will potentially be very different from feature to feature. It might even be best to use transformations in different families from one feature to the next. Since there are too many features to determine by-hand an appropriate Gaussianizing transformation, our (G) step automatically chooses one which works well.

We call this (G) procedure "robust" because it distinguishes between a distribution that is fundamentally skewed and one which simply has a few anomalous outliers. It attempts to find a transformation to reduce the skewness of the bulk of the data without being influenced by extreme points. These extreme outliers are instead removed by the combined (Z) and (O) steps. To remove the outliers, first the (Z) step converts the data to z-scores using robust estimates of the mean and standard deviation. Subsequently, the (O) step designates any entry of the feature matrix bigger in magnitude than four an outlier and marks it as missing (effectively removing it).

The (G), (Z), and (O) steps attempt to enhance analysis while minimizing changes to the data. The Gaussianizing transformation is made only to rectify a fundamentally skewed distribution, not to reign in outliers. Conversely, outliers are identified only after a de-skewing transformation has

been made. Thus points are removed only if they are truly anomalous. In turn, this enhances the prominence of informative points that are large but otherwise obscured by uninformative outliers.

3.3.2.2 Integrating Features

In addition to improving the recovery of latent effects in individual features, we are also interested in integrating information across features to recover common latent effects. To extract a common set of latent effects from a collection of feature matrices we use a PCA-like approach. This method captures common latent effects across a collection of features using the eigenvectors of the features' average left and right Gram matrices. We call these eigenvectors the left and right average singular vectors (ASVs). A precise mathematical description of this is contained in Section 6.2. An important component of integration in this way is a careful consideration of the different features' scales. Here, the (G), (Z) and (O) processing steps are helpful. These steps (especially (G) and (Z)) robustly transform the data so that all the features are on a commensurate scale. This allows us to use a simple arithmetic mean of Gram matrices to equitably integrate information across features.

3.4 Results

3.4.1 Features and Transformations Considered

The MEMA plates we analyze are grown, stained and imaged in three separate processing batches. A different set of stains is used in each batch. Those sets are: (1) "SS1" (containing stains DAPI, Actin, CellMask and MitoTracker), (2) "SS2noH3" (containing stains DAPI, Fibrillarin and EdU), and (3) "SS3" (containing stains DAPI, KRT5, KRT19 and CellMask). Because each of these three processing batches use a different staining set we will refer to these batches as the "staining batches" and identify them with the staining set used in each batch. However it should be noted that these three batches are three separate experiments run at three different times. Nonetheless, aside from the staining set, the experimental conditions were made as identical as possible across the three experiments.

In total there are 108 different image features extracted from the MEMAs. Since different staining sets are used for different MEMAs not all features are extracted for all MEMAs. There are 55 features extracted in at least two of the staining batches and 21 features that are extracted from all three batches. We will primarily focus on four features:

1. cell area ("Cells_CP_AreaShape_Area")

2. cell compactness ("Cells_CP_AreaShape_Compactness")

3. spot cell count ("Spot_PA_SpotCellCount")

4. total cytoplasm DAPI intensity ("Cytoplasm_CP_Intensity_IntegratedIntensity_Dapi").

We choose these four example features because they represent several different feature types. The first two example features are morphological traits of cells, the third feature is the number of cells, and the last feature is an intensity measurement in the cytoplasm of the cells. Where possible, we will include compact summaries of the results for all features. We have deposited the full results for all features at umich.box.com/v/mematransformation

To explore the effects of our the three processing steps (G), (Z) and (O), we will consider five different transformations of our feature matrices. These transformations are:

1. no transformation, denoted (NT),

2. the (G) step only

3. the (Z) step only

4. the (O) step only

5. the full three-step sequential application of (G), (Z), and (O) denoted (RR) for "robust re-scaling."

3.4.2 Visualization

3.4.2.1 Density Plots

A typical first step in exploratory analysis is data visualization. Simple data visualizations can succinctly summarize the broad nature of the data and inform qualitative analyses. In Figure 3.2 we plot the density of cell area for our five transformations. The colored densities correspond to staining batches. The black line is the density of all data combined. Notice in Figure 3.2 that the density of the un-transformed data (NT) largely just reflects that the data has a long tail. The same can be said for the z-score transformation (Z). Conversely, we see that the other processing steps make the staining batches obvious. Both removing outliers (O) and Gaussianization (G) nicely highlight the two staining batches. These transformations de-emphasize the data's long tail and favor the group difference. Notice however that while removing outliers does help, the remaining data is still fundamentally skewed. Similarly, while the (G) step makes the bulk of the data normal it leaves several low outliers. Both of these issues partially obscure the groups we wish to discover. We can ameliorate these issues by combining the (G) and (O) steps in the (RR) transformation. We can see in the (RR) plot that the staining batches have been separated into two groups that are

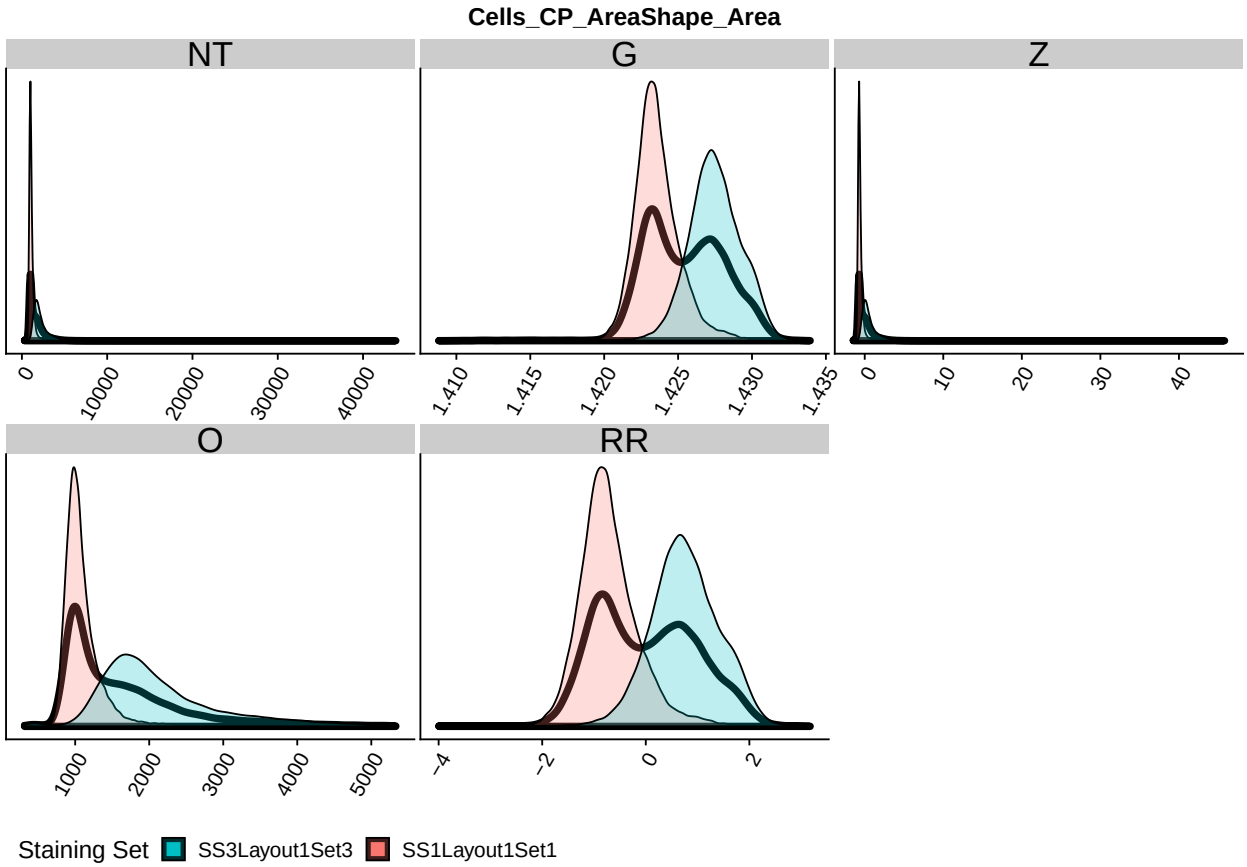


Figure 3.2: Density of elements of cell area feature matrix. Black density is all elements combined. Colored densities are the densities for the two staining batches. Subplots are for five processing transformations of this matrix: (NT) no transformation, (G) Gaussianization, (Z) z-score, (O) outlier removal, (RR) the three-step (G), (Z), and (O), robust re-scaling.

commensurately shaped and approximately normal. Thus while the (G) and (O) transformations are individually useful, their combination separates the groups more nicely.

In Figure 6.1 we display similar plots to Figure 3.2 but for the other three example features. Largely we see the same behavior of the five transformations. The plots of the un-transformed data (NT) and z-transformed data (Z) largely highlight the distributions' tails. However the (G) and (O) steps help recover the staining batches. The combination of these steps in the (RR) transformation highlights them further. In Figures 6.2, 6.3 and 6.4 we display density plots similar to Figure 6.1 but highlight the densities of the different wells, plates, and ligands instead of the staining batches. These plots exhibit similar, albeit more attenuated, behavior. In summary, the (G) and (O) steps and their combination in the (RR) transformation focuses the data on aspects like staining batch, plate or ligand rather than just picking up the distribution's tail.

3.4.2.2 Heat-maps

Another way to visualize the MEMA data is through heat-maps. Heat-maps can be useful for discovering spatial effects and assessing the quality of data. As an example, consider visualizing cell area by plotting a heat-map of all spots over all plates. In Figure 3.3 we display a single well from this heat-map across the five transformations. (The full heat-map may be found in Figure 6.5.) The colors in the heat-map are selected so that spots are more blue if they are close to the minimum cell area, red if they are close to the maximum, and white if they are close to half-way between. Dark grey spots are omitted according to the MEMA design. The green spots in these plots are missing. These spots are missing either due to experimental error or because they have been removed as part of analysis.

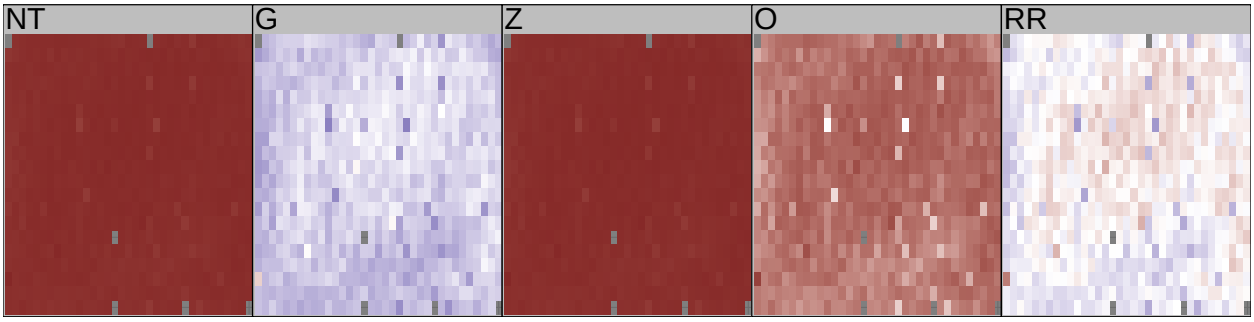


Figure 3.3: Heat map of a single well across the five transformations (NT), (G), (Z), (O), (RR).

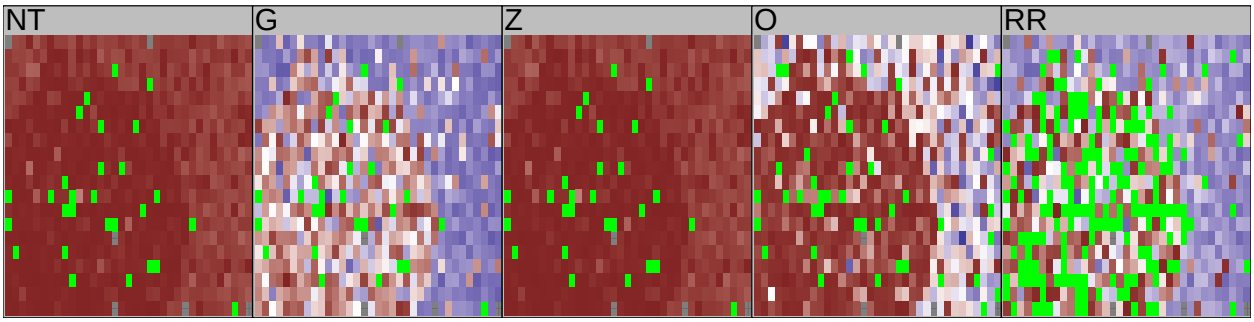


Figure 3.4: Similar to Figure 3.3 but focusing on a different well.

This figure shows that producing the heat-map using un-transformed data (NT) or z-transformed data (Z) is not very informative. The skewness and outliers ensure that a the bulk of data points are assigned a tiny range of colors. Thus the plots are essentially a single color. Conversely, the heat-maps of the (G) and (O) transformed data is much more informative. We can start to see a slight spatial effect between the left and bottom edges and the rest of the well. We can also see a non-spatial effect where certain spots are much different than their surroundings. We will see

in Section 3.4.5 that this is a biological effect of the ECMps NID1 and ELN. Finally, when we combine the (G) and (O) steps with the (Z) step we get the (RR) plot. The combination of these steps really highlights the spatial effect within the well. It also highlights the non-spatial effect manifested as several deep blue spots in the mostly red upper-right of the well.

In Figure 3.4 we display a similar heat-map but focus on a different well. We see similar behavior when looking at this well as the previous. The (G) and (O) transformations reveal a big spatial difference between the lower left and the rest of the well. This spatial difference is also seen in the (RR) transformation. Notice however, that the number of points removed (in green) using the (RR) transformation is much different than the number removing using just the (O) step. The difference here is driven by the scale at which outliers are thresholded. The (O) step thresholds outliers on the original scale, the (RR) transformation removes outliers on a Gaussianized scale. Notice that the (RR) transformation removes points according to the spatial pattern. Essentially, this transformation highlights the spatial pattern and identifies these points as being poor quality and thus fit for removal. On the other hand, while the (O) transformation highlights the spatial pattern it does not remove the points. It is likely that this spatial pattern is not biological, but an unwanted technical effect and thus its removal with the (RR) transformation is prudent. It follows that the (RR) transformation (which identifies outliers on a Gaussianized scale) is better able to detect points that should be removed. This suggests that a simple z-score thresholding procedure for removing outliers works better on a Gaussianized scale than not.

In addition to revealing within-well effects these transformations can also highlight batch effects between, for example, plates, wells, or staining sets. In Figure 3.5 we show the heat-map of cell area for eight wells across the (NT), (G), (O) and (RR) transformations. (The (Z) transformation looks identical to (NT).) The top four wells in each sub-plot are from the first staining batch, the bottom four wells are from the second. Nonetheless, we see little indication of batch in the (NT) plot. It is solidly red. We start to see a hint of a batch effect when looking at the (G) and (O) heat-maps. The bottom of the (G) heat-map is lighter blue than the top, and the top of the (O) heat-map is lighter red than the bottom. This batch effect is even more prominent in the heat-map made from the (RR) transformed data. In this sub-plot see solid-blue in the top batch and mostly red in the bottom batch. In addition to highlighting the batch, we again plainly see spatial patterns within the wells. This is visible in the (O) and (G) plots too, but most prominent in the (RR) plot.

Finally, let's revisit the interaction between outlier removal and scale. Notice that the number of removed points (in green) in the (O) plot is quite high. Conversely, if we Gaussianize first before removing outliers (as in the (RR) plot) we remove many fewer points using the same outlier thresholding procedure. This indicates that many points that are identified as outliers by the (O) step alone probably aren't points that should be removed. Instead, these points are symptomatic of a fundamentally skewed distribution of data as we saw in the (O) plot in Figure 3.2. Thus we

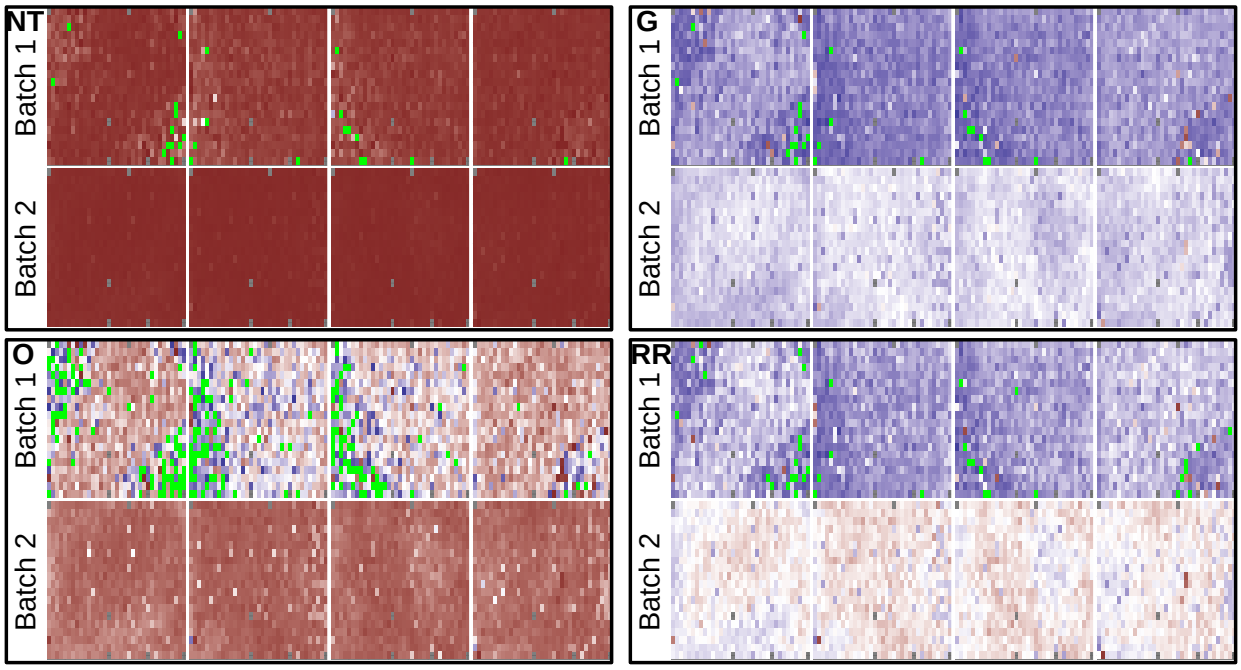


Figure 3.5: Heat map of a eight wells across the five transformations (NT), (G), (Z), (O), (RR). Top row of each subplot is from first staining batch. Bottom row is from second staining batch. Colors are more blue if they are close to the minimum, red if they are close to the maximum, and white if they are close to half-way between. Green spots are missing. Dark grey spots are omitted according to the MEMA design.

suggest (as in our three-step (RR) transformation) to first Gaussianize the data and then remove outliers. We can see from Figure 3.5 that this allows us to recover the spatial and batch effects without removing too many points. This in conjunction with Figure 3.4 shows that thresholding outliers on a Gaussianized scale does a better job at removing points if and only if they should be removed.

3.4.3 Recovering Technical Effects Across Wells

An important class of analyses for MEMA data are those seeking to discover latent effects. Latent effects are important for several reasons. First, they may be of direct scientific interest. For example, we may find effects relating to ECMps or ligands. Often, however, the latent effects are unwanted. These effects may be, for example, a technical artifact of the data collection or processing. Indeed we have already seen that the staining batch in our MEMA data can be qualitatively identified through heat-maps (see Figure 3.5). Unfortunately, such batch effects are common in high-throughput biological experiments like MEMAs. Furthermore, these kinds of effects are often large and obscure the biological variation in which we are interested. Indeed it is not uncom-

mon for the first order effects in high-throughput data to be technical batches.

In the case of unwanted latent effects, identifying them helps us attenuate their influence on analysis. For example, we can project them out of the data. Typically latent effects are discovered using the singular value decomposition (SVD) also known as principal components analysis (PCA). In this section we will explore how our three-step (RR) transformation helps in the recovery of interesting latent effects using the SVD. To aid this exploration, we will focus on the very prominent staining batch effect. We saw in Figure 3.5 that the staining batch was visible by eye using the (G), (O) and (RR) transformations. To assess how well we can recover the staining batch using the SVD, we will mask the true (known) staining batches from analysis and use the SVD to re-discover them. We can then measure the efficacy of this re-discovery by correlating the discovered latent effects with the true known batches.

In Figure 3.6 we plot the mean of the squared canonical correlations (CCs) between the first k principal components (left singular vectors) and the staining batch dummy variables. We vary k across the x-axis from 1 to 192. This is done for our four example features and our five transformed versions of those features. From this figure we can see that simple data transformations have the potential greatly to enhance discovery of the staining batch. Consider the CC plots of cell area feature and total DAPI intensity feature in Figure 3.6. As compared with no transformation (NT), these plots shows that the (G) and (O) steps increase how much of the staining batch is captured by the first several PCs. The (G) and (O) steps attenuate the non-informative tails of the distributions and focus the PCs on the differences across the staining batches. Unfortunately, the (G) and (O) steps alone are not a universal solution across features. For example, in the cell count plot we don't see much improvement over (NT) by removing outliers (O). Presumably this is because there are not many true outliers in the data. Instead the data is probably just a bit skewed. Thus removing extreme points doesn't fundamentally have much effect on the distribution of the data. Worse, as seen in the cell compactness feature, spuriously removing extreme points can actually be detrimental to recovering the batch effect. This likely happens because the (O) step is removing informative points. This is a very real danger when removing outliers by thresholding skewed data. Similarly, the (G) step is not always the best approach by itself (see the area and compactness plots). The Gaussianization method used in the (G) step is robust to outliers. Thus this step will attenuate the effects of some truly misleading outliers.

Nonetheless, when we combine the (G) and (O) steps sequentially we see excellent performance across the board. In all four panels of Figure 3.6 we see that the three-step (RR) transformation performs as well or better than any of the other transformations. Indeed the (RR) transformation always identifies the staining batch much more quickly than no transformation (NT). Even for the area feature, where the un-transformed data already recovers the batch well, the (RR) processing steps still slightly improves batch discovery. These improvements come from the (RR)

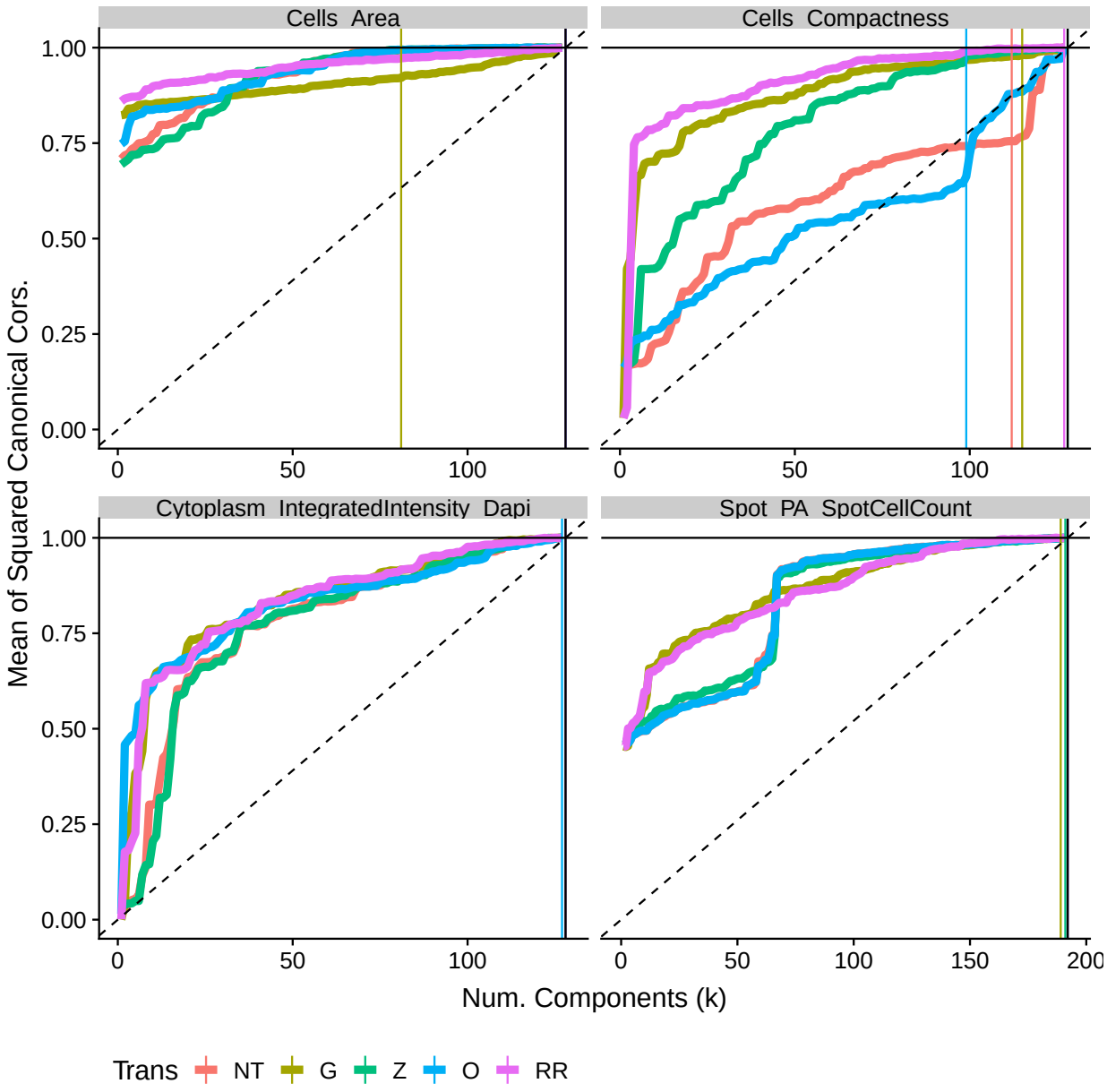


Figure 3.6: Mean of the squared canonical correlations between the first k principal components and the staining batch dummy variables.

transformation's attenuation of skewness and outliers in the data. The (RR) transformation reduces the effects this misleading variation using a carefully selected sequence of the (G), (Z), and (O) steps that first reduce skewness, if necessary, and then removes outliers, if any.

To show that the (RR) transformation generally improves results, we summarize batch recovery for all features and all transformations in Figure 3.7. In this plot, we calculate the area under the CC curves (AUC) for each feature. We order the features left to right by the difference in the AUC

between (RR) and (NT). Broadly, we see the same behavior in Figure 3.7 as displayed in Figure 3.6. While we can find individual cases where the (G) and (O) steps individually might be beneficial there are also cases where they are detrimental. However combining these steps in (RR) seems to generally improve recovery of the staining batch. Sometimes we see a substantial improvement using the (RR) transformation. Rarely do we see that (RR) is detrimental. In the rare cases where (RR) is not optimal, it is only marginally worse than any of the alternative transformations.

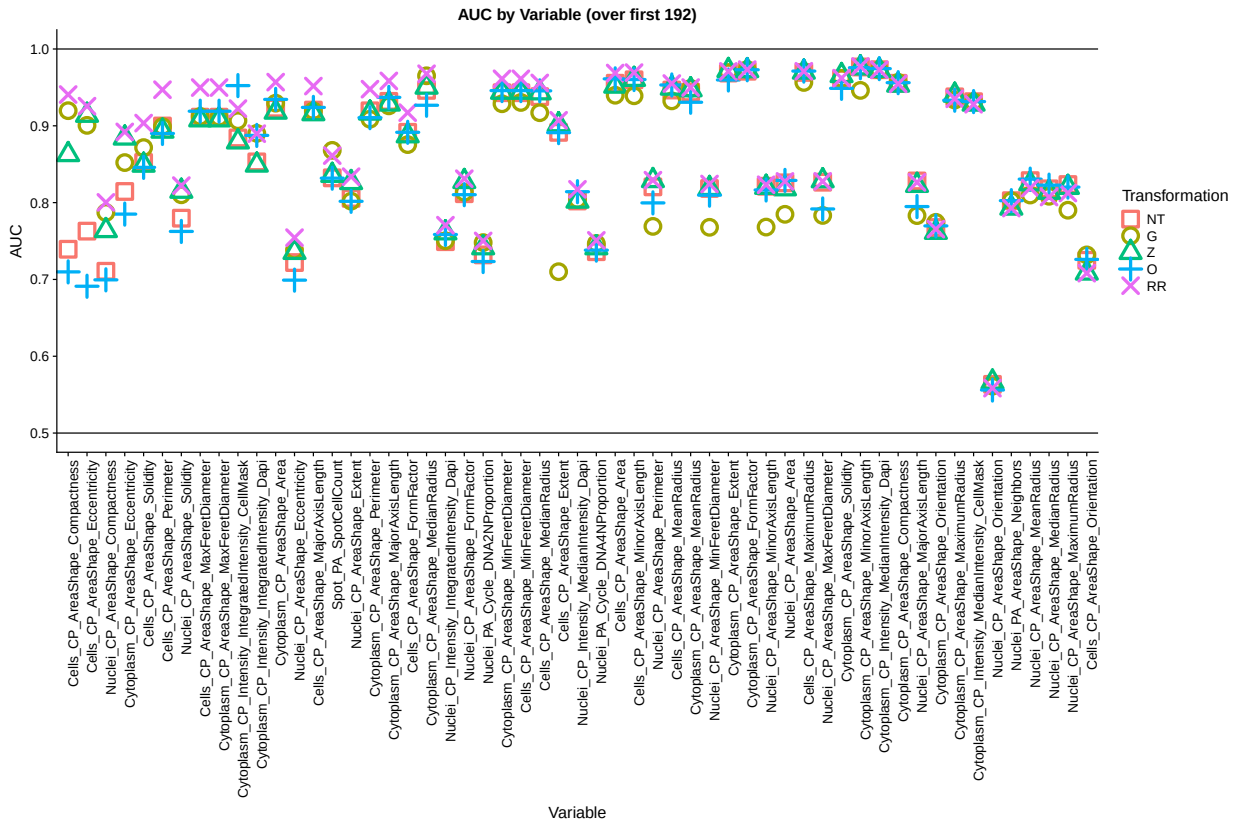


Figure 3.7: Grand mean of the squared canonical correlations across number of components (k). Canonical correlation is calculated between the first k principal components and the staining batch dummy variables.

To explore recovery of other latent effects besides stain, in Figures 6.9 - 6.14 we display similar CC and AUC plots for the recovery of plate, well, and ligand effects. While these effects are not as prominent, we still see that the (RR) transformation improves recovery of these latent effects without being detrimental.

3.4.4 Data Integration for Discovering Between-Well Effects

The plethora of features that can be extracted from MEMA images presents a good opportunity for data integration. One way we are interested in integrating features is by combining them to better recover common latent effects. In this section we will explore recovering latent effect through data integration. We will do this using the left average singular vectors (ASVs) as described in Section 3.3.2.2.

In the left panel of Figure 3.8 we plot the mean squared canonical correlations between the first k left ASVs and the staining batch. We vary k from 1 to 192. These ASVs are calculated using the 21 features that are measured across all MEMAs. As previously, we include lines for all five processing transformations (NT), (G), (Z), (O) and (RR).

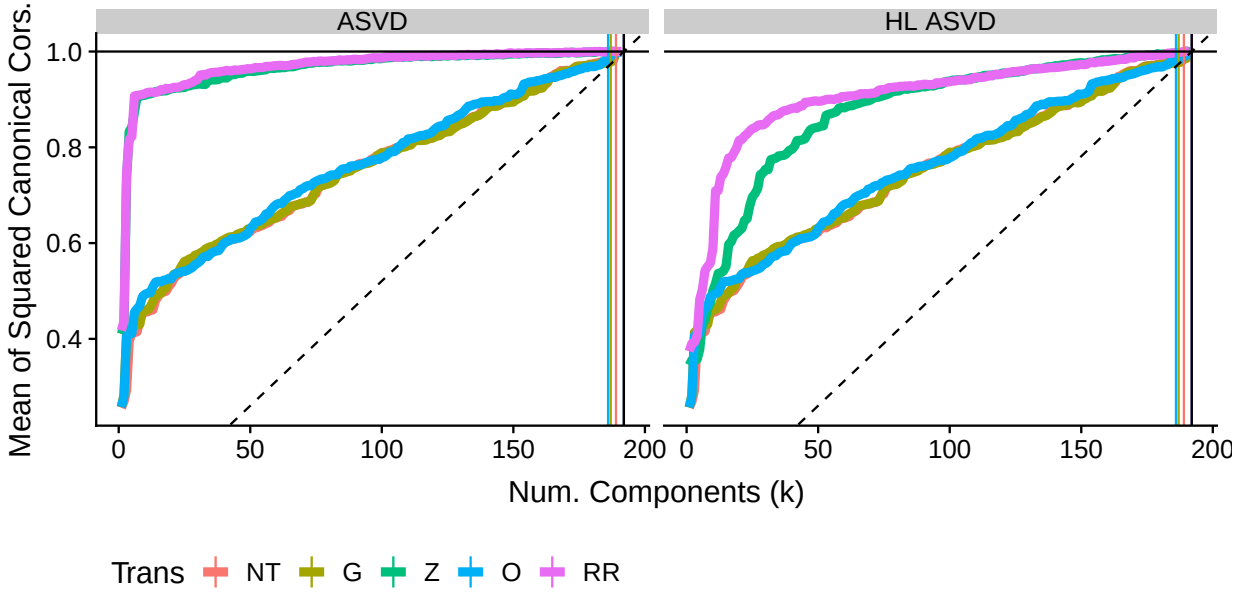


Figure 3.8: Mean of the squared canonical correlations between the first k principal components and the staining batch dummy variables. Principal components come integration of (Left) the 21 features that are measured across all MEMAs, and, (Right) among those 21, the five with the highest leverage points.

The first striking feature of this figure is how quickly and strongly the (Z) and (RR) transformations are able to recover the staining batch effect. The AUC for these curves is in excess of .95 which, looking at Figure 3.7, means it recovers the staining batch better than the majority of the features individually. This demonstrates the power of the ASVs for identifying common latent effects. When we average the Gram matrices to calculate the ASVs the feature-specific effects are "averaged-out" while the common effects like staining batch are amplified. This makes identifying these strong, common batch effects easy.

In addition to how strongly the staining batch is recovered, it is notable that the (Z) and (RR) transformations recover the batch significantly better than the other processing options (O), (G), and (NT). This happens because we find the ASVs by element-wise averaging Gram matrices across features. If these Gram matrices are on vastly different scales (as in (NT), (G), and (O)) then their average will be biased towards the feature(s) with the largest values. This bias will not consider the information from all features equally but will arbitrarily weight the information by the features' scales. To equitably integrate information across features, all feature matrices should have values in a similar range before averaging. This is precisely what is done by the (Z) step. This step converts the disparate scales all to z-scores. Thus the values in the Gram matrices will be commensurate and we can integrate them with a simple average.

Finally, it is notable in the left panel of Figure 3.8 that the (RR) and (Z) transformations do approximately as well as each other. This happens because the Gram matrix averaging conveys some of the same benefits as the (G) and (O) steps. While outliers or skewness might affect individual Gram matrices such effects will have less influence on the average of the Gram matrices. Thus the (G) and (O) steps are less critical. The problems they solve are naturally attenuated by the averaging. This is true so long as we do not have either (1) a small number of features or (2) features that share systematic skewness or outliers. To illustrate this, consider the plot in the right panel of Figure 3.8. Here, we mimic the left plot but calculate the ASVs using only five features with several extremely high-leverage points. In this plot we see a separation between the (Z) and (RR) transformations. This is the case because there is benefit to the (G) and (O) steps since the average is over a small number of features. In any case, including (G) and (O) steps does not seem to hurt the analysis and thus we still recommend the full three-step (RR) transformation for integrating features in this manner. Finally, we note that a similar, but attenuated, story can be told for integrating data to recover other effects like plate, well and ligand. These results are shown in Figures 6.15 - 6.17.

3.4.5 Discovering Biological and Spatial Effects within Wells

A primary goal in the analysis of MEMA data is the discovery of important biological effects. Much the same way that the left singular vectors (PCs) revealed latent effects across the wells, plates, and staining batches the right singular vectors (RSVs) reveal interesting variation across the spots in each well. In Figure 3.9 we display a scatter plot of the elements first two RSVs of total cytoplasmic DAPI intensity. This is done for each of the five transformations (NT), (G), (Z), (O) and (RR). The color and shape of the points indicate to which ECMp the RSV component corresponds.

The first thing to notice in Figure 3.9 is the striking separation between the ECMps ELN and

NID1, and the rest. This effect manifests because the cells in the ELN and NID1 spots have difficulty adhering to the MEMA substrate. Notice the cell count heat-map in Figure 6.7 shows that the cell count in the ELN and NID1 spots are significantly lower than other spots. While this ELN-NID1 effect is present in the un-transformed data, we can see from Figure 3.9 that the (G), (O) and (RR) transformations reveal the effect better. The first RSV from the un-transformed data does capture the effect however the second RSV seems to almost entirely be focused on explaining a single anomalous CDH8 spot. Conversely, the transformations of (G), (O) and (RR) are not distracted by this outlier spot. Indeed, they not only separate NID1 and ELN from the other ECMps but they also separate the effects of ELN and NID1 from each other. This is especially prominent in the three-step (RR) transformation.

Figure 3.9: Scatter plot of elements of top two right singular vectors against each other for the total cytoplasmic DAPI intensity feature. Shape and color indicate ECMp of the spot corresponding to the elements of the singular vector.

To explore this further, in Figure 3.10 we plot heat-maps of the first three RSVs for cytoplasmic DAPI intensity. This is done across all five transformations. We again see the effect of NID1 and ELN in these heat-maps. This effect manifests as the isolated points contrasting with their immediate surroundings. However what we can see in these plots that we cannot see in Figure 3.9

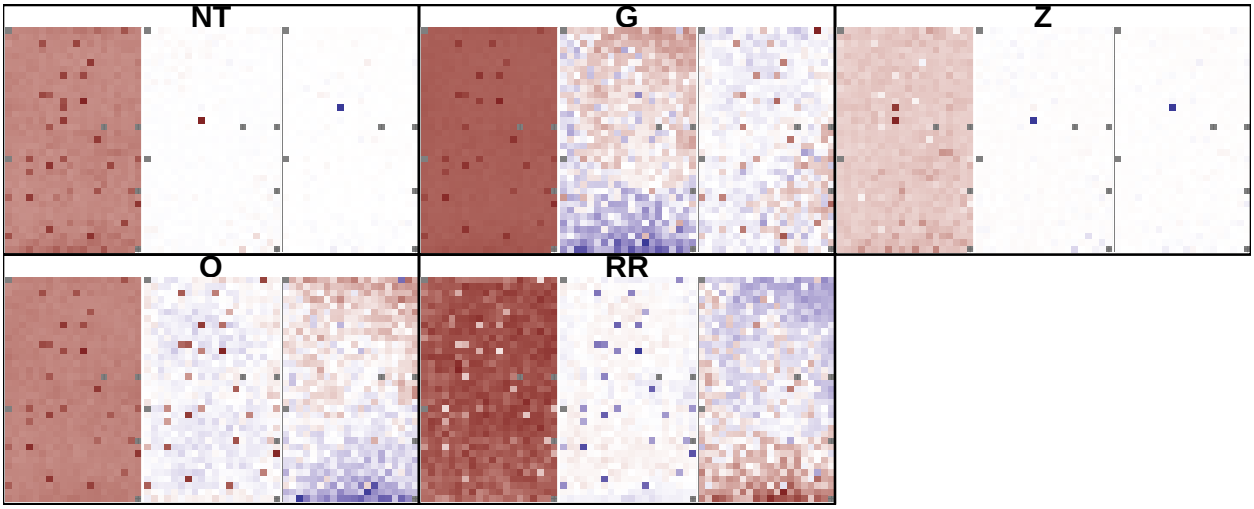


Figure 3.10: Heat map of elements of top three right singular vectors for the total cytoplasmic DAPI intensity feature.

is the spatial patterns captured by the RSVs. In Figure 3.10 we see a very strong spatial pattern differentiating the top of the well from the bottom. This is most visible for the (G), (O) and (RR) transformations. Notably, we do not identify this effect with the (NT) and (Z) transformations. Instead, their second and third RSVs are entirely capturing outlying points. Likely, this spatial effect is an unwanted technical artifact from the experiment. Nonetheless, it is important to identify such an effect so that we can be sure to properly account for it.

Looking at the RSVs of the other three example features largely tells a similar story. In Figures 6.19 - 6.23 we display similar scatter plots and heat-maps for the other example features. These plots show that the (G) and (O) steps, and their combination in the (RR) transformation, help reveal important latent spatial and biological effects.

3.4.6 Data Integration for Discovering Within-Well Effects

In section 3.4.4 we saw that data integration helped make salient important between-well effects. In a similar fashion, the average right singular vectors (ASVs) help bring out within-well effects. In Figure 3.11 we plot the first two right ASVs against each other. As previously, we use color and shape of the points to indicate ECMp. The results here mimic what we saw in our previous data integration effort. The (Z) and (RR) transformations equitably integrate information from all features and thus help bring out important latent effects. In Figure 3.11 we can see that these right ASVs are highlighting the ELN-NID1 effect. On the other hand the (NT), (G) and (O) transformations do not properly adjust the features' scales and so they do not capture this common latent effect.

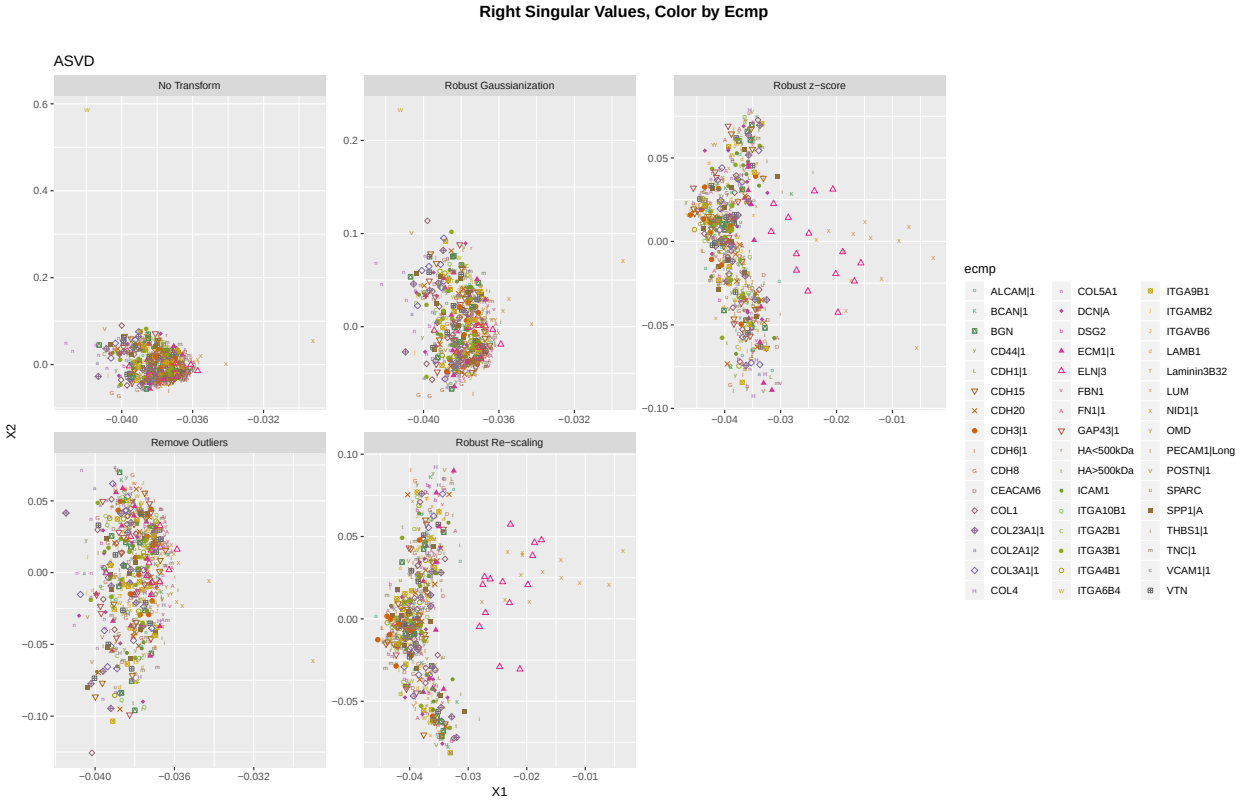


Figure 3.11: Scatter plot of elements of top two right ASVs calculated over 21 features measured on all MEMAs. Shape and color indicate ECMp of the spot corresponding to the elements of the singular vector.

Finally, we display heat-maps for the the first three right ASVs in Figure 3.12. While the (G) and (O) transformations did not pick up the ELN-NID1 effect, we can see from this figure that they do capture spatial effects within the wells. The only transformation that does not strongly pick up the spatial effects well is (NT). The right ASVs for this un-transformed data seem to be mostly picking up a single outlier. On the other hand, the (G), (Z), (O), and (RR) transformations seem to pick up two interesting spatial effects. The first effect is a top versus bottom effect, and the second effect is a middle versus top/bottom edge spatial effect. Nonetheless, while the (G) and (O) transformation do pick up the spatial effects we still recommend the (Z) or (RR) transformation for recovering latent effects from the ASVs. These transformations will properly allow integration of features so as to pick up common effects within the wells. The (G) and (O) transformations are just picking up the spatial effects because they happen to capture a feature with similar spatial effects. Conversely, the (Z) and (RR) transformations help recover both important biological effects and shared within-well spatial effects.

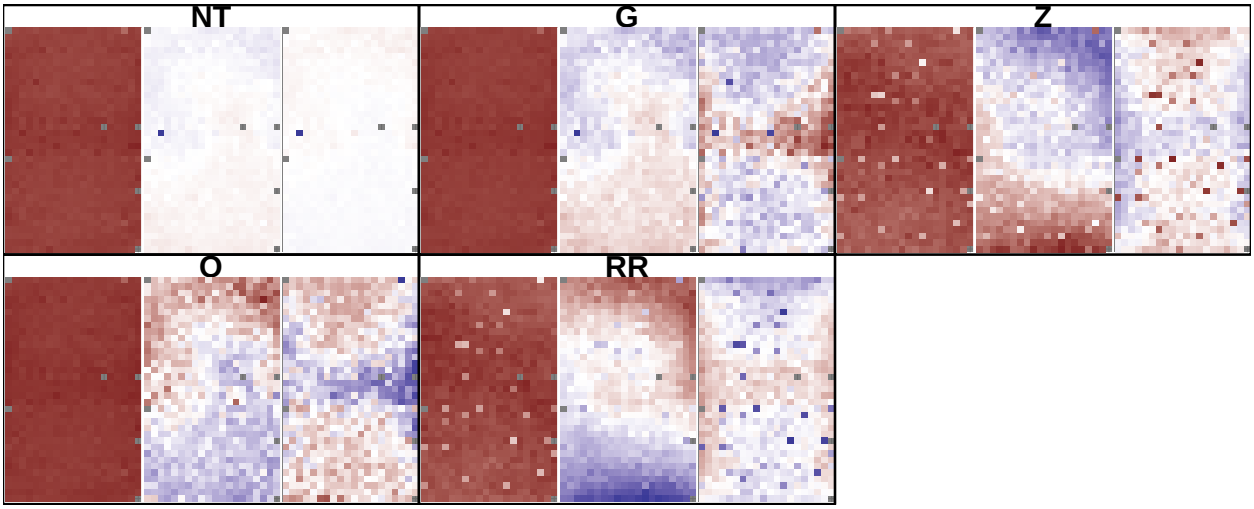


Figure 3.12: Heat-map of top three right ASVs calculated over 21 features measured on all MEMAs.

3.5 Discussion

The microenvironment of cells is an important component of many cell and tissue level processes. Studying cellular microenvironment not only furthers a fundamental understanding of these processes but also helps us understand the interaction of the microenvironment with disease-targeting therapies. In this paper we have explored the effects of several simple processing steps on the analysis of MEMA data. We have considered not transforming data (NT), Gaussianizing data (G), z-scoring data (Z), removing outliers (O), and a three-step sequence (G), (Z), then (O) that robustly re-scales (RR) the data. Broadly, our analysis of these transformations is as follows.

First we found that un-transformed feature data is often encumbered by skewed measurement scales, outliers, or both. These aspects can obscure important effects in the data and this hinders discovery of interesting latent effects through qualitative visualizations like heat-maps and quantitative analysis like PCA. In pursuit of attenuating such effects, we saw that both the (G) and (O) transformations were helpful. The (O) transformation simply discards data outside of a conservative threshold. Often this transformation enhances analysis by removing influential but uninformative measurements. Unfortunately the simple threshold sometimes fails to properly identify outliers. If the data is highly skewed, informative points might be spuriously removed. This can be detrimental to analysis. Conversely, if the data is too compact, the thresholding can fail to detect misleading data that probably should be removed. Another way to deal with such problems is Gaussianize the data. This is what our Gaussianizing transformation (G) does. Like the (O) transformation, we saw that reducing skewness by Gaussianizing the data can make interesting latent effects prominent. This happens because the (G) transformation attenuates the influence

of the data's tail. However to make the transformation robust, our (G) step largely ignores extreme outliers. Thus the effects from these outliers are still potentially problematic after the (G) transformation.

Many of the problems with applying the (G) and (O) transformations individually are resolved by combining the two steps. We see this in the (RR) transformation where (G) and (O) are the first and last steps, respectively. First the (G) step is applied which non-linearly transforms measurements to reduce data skewness in the body of the distribution. After this transformation, any points which appear to be anomalously outlying are removed using the (O) step. The order of these two steps is important. By first applying the (G) transformation and following with the (O) step we attempt to minimize the changes made to the data. The (G) step transforms the data only if the distribution is fundamentally skewed. The (G) step does not apply a strong non-linear transformation simply to ameliorate the effects of outliers. Instead, these outliers are dealt with simply by discarding points beyond a conservative threshold. In this way only the few outlying points are changed. While one might consider first removing the outliers before a Gaussianizing transformation, accurately identifying points that should be removed depends critically on the data scale. The preemptive Gaussianization means we are less likely to spuriously identify points in the tail of a skewed distribution as outliers. Thus the (O) step in tandem the (G) step helps identify and remove only true outliers that are not accounted for in the (G) step.

Finally, in this paper we looked at a z-score transformation (Z) alone and as part of the (RR) transformation. We saw that transforming feature matrices with a robust z-score allowed them to be straight-forwardly integrated. This integration allowed us to extract important latent effects like staining batch from a simple arithmetic average of Gram matrices. The (Z) transformation is crucial for this data integration because it puts the values in the Gram matrices on commensurate scales so that an arithmetic average equitably integrates them. Furthermore we saw that the averaging of Gram matrices conveys some of the same benefits as the (G) and (O) transformations. Specifically, the averaging dampens the effects of anomalous outliers and skewed distributions. Nonetheless, we also saw that the (G) and (O) transformations do not tend to harm this analysis. Moreover, these (G) and (O) transformations can still convey benefits when the number of feature integrated is small or there exist systematic skewness or outliers across features.

In conclusion, we saw that a combination of a Gaussianizing transformation (G), z-score transformation (Z), and removal of outliers (O) can improve visualization and discovery of biological and technical latent effects in both individual features and when integrating features. The application of each of these transformation alone and sequentially has the potential to make salient important effects in the data. Conversely, these transformation rarely have a negative effect on analysis. For these reasons we believe that such transformations are generally advisable for the analysis and integration of MEMA data. More broadly, the robust nature of these transformations

suggests that they may prove beneficial for processing data from other image-based cell profiling technologies.

Chapter 4

Summary and Conclusions

4.1 Major Conclusions of This Work

In this work we have carefully considered the role that scale transformations play in the analysis of high-throughput -omics data. In this section we will briefly summarize those results.

In Chapter 2 we looked at the dtangle estimator for cell type proportions. Here we showed that a combination of a plausible biological model on the linear scale with a fitting procedure using log-scale data produces a powerful cell type estimator. The dtangle estimator broadly outperformed existing cell type deconvolution methods across eleven bench-mark data sets. These data sets included a wide range of cell types, technologies, and many more experimental factors. Not only does dtangle perform as well or better than existing method, it is also very robust to outliers in the data and tuning parameters. In addition, we explored the role of scale in deconvolution methods through a large number of simulations. We also applied dtangle to real gene expression data from Lyme disease patients and showed that dtangle produces results consistent with existing knowledge.

In Chapter 3 we looked at the role of transformations in the analysis of microenvironment microarray data. Here we explored a Gaussianizing transformation, a z-score transformation, and a step to remove outliers. We observed that the Gaussianizing and outlier removal steps can improve visualizations and recovery of important latent effects by attenuating the effects of skewed measurement scales and outliers. We also saw that their sequential application can produce a powerful transformation towards this end. In addition to these two transformations, we looked at a z-score transformation. This transformation was crucial in the proper integration of MEMA features. The z-score transformation allowed features on very different scales to be integrated using a simple averaging of Gram matrices. This allowed the strong recovery of important latent effects shared across features. By averaging the Gram matrices the effects of outliers and skewed data could largely be ameliorated. However the Gaussianizing and outlier-removal steps were potentially beneficial if there were a small number of features being integrated or systematic skewness or outliers. In any case, these transformations were not harmful.

The unifying thread across these projects is the importance of data scale in analysis. In Chapter 2 we showed that the scales used to model and fit the data can impact the efficacy with which cell types can be estimated. Furthermore, in Chapter 3 we showed that simple scale transformations can enhance visualization, recovery of important biological and technical latent effects, and integration of MEMA image features. Both of these project underline the fact that simple data transformations combined with a careful consideration of data scale has the potential to greatly enhance analysis.

4.2 Future Work

We can see several ways forward from these projects. For dtangle we see a potential of research in the following directions:

1. application to DNA methylation data
2. relaxation of the marker gene assumption using optimization
3. Box-Cox-like transformations in lieu of a log-transformation
4. exploring the effects of data normalization techniques and deconvolution
5. removing unwanted variation as part of dtangle.

In the pursuit of the analysis of MEMA data we presently have the two following lines of inquiry we would like to explore:

1. delving into feature integration and imputation of unknown features using known features
2. looking at the interaction between transformations and adjustments for unwanted variation.

Chapter 5

Supplement: dtangle

5.1 Assessing The Relationship Between Actual and Measured Expression

One of the main components of dtangle's approach is a linear model relating actual gene expression to measured gene expression. To explore its plausibility, we consider this linear model's application to Affymetrix DNA microarray data and Illumina RNA-seq data.

5.1.1 Microarray Data

To explore the relationship between the amount of transcripts and the measured expression from microarray technology we consider the Latin Square data set from Affymetrix [Irizarry *et al.*, 2003]. This data set was created by hybridizing a solution of complex human background mRNA with 42 transcripts spiked in at concentrations ranging from 0.125pM to 512pM. The spike-ins were done with 3 technical replicates of 14 hybridization experiments in a Latin square design. This data set lets us explore the relationship between measured expression and abundance of the transcripts because for each of the spiked-in transcripts we know both the expression measured by the array and the amount in which the transcript was spiked in.

The expression as measured by the microarray is best explained by a logistic fit in the spike-in amount (Supplementary Figure 5.28a). However the linear fit that dtangle assumes does quite well. The logistic fit has a slightly smaller R^2 than the linear fit however there are several reasons we choose to model the relationship between spike-in amount and measured expression as linear. Firstly, the linear model is much simpler than the logistic model and has almost as good of a fit. For the linear model $R^2 = 0.957$ while for the logistic least squares fit we have $R^2 = 0.992$. Thus we gain relatively little for using the more complex model. Furthermore, the simplicity of the linear fit can also be thought of as a regularization of the logistic model. The non-linearity of the logistic curve means it is a very unstable model for measured expressions on both the high

and low ends. That is, its inverse is undefined at or beyond these points. If the logistic model is used to estimate the true gene transcriptional abundance from measured expression data then small changes in measured expression might correspond to large changes in predicted amount. Indeed, the logistic curve will fail completely for measured expressions above its maximum or below its minimum. The linear model can be thought of as a regularized model between these two quantities. It ensures that a linear change in measured expression will only ever effect a linear change in amount. While there is probably a true non-linear relationship between the expression measured by microarrays and the amounts of transcripts in the samples, a linear fit does quite well at approximating this relationship and is a regularized model for the truth.

5.1.2 RNA-seq

Another reason we favor linear modeling of the relationship between amount and measured expression is because it is not only reasonable for microarray technology but a reasonable model for RNA-seq. To explore how our model interacts with RNA-seq technology we consider data from the Sequencing Quality Control project [SEQC Consortium, 2015]. These data are available on GEO with accession GSE47774. Here we look at RNA-seq analysis run on Ambion ERCC Spike-In Control Mix 1 using Illumina HiSeq technology. The ERCC spike-in control mix contains 92 transcripts spiked-in at known concentrations. Hence this data set allows us to look at the relationship between measured expression and amount because both are known.

For this data the measured expression values ($\log_2$ of the read count plus one) are well approximated by a linear relationship to the spike-in concentration amount (Supplementary Figure 5.28b). Unlike the previously discussed microarray data the RNA-seq data does not seem well approximated by a logistic fit. For a simple linear regression we find $R^2 = 0.955$ and so a linear fit seems reasonable.

5.1.3 Estimating The Slope

We have thus seen that both microarray and RNA-seq measured gene expressions are well modeled as linear (on the log-scale) in the actual expressions. For the RNA-seq data we find that the slope of the linear relationship is approximately one. However for microarray data the relationship is better modeled as linear with a slope slightly smaller than one. Doing so will help account for the true logistic relationship that is affected by saturation and attenuation of the measured expressions at the low and high ends.

We denote the slope of this relationship as γ in our model and replace it with its estimate $\widehat{\gamma}$ when estimating the cell type proportions using dtangle. The value of $\widehat{\gamma}$ in dtangle's algorithm may be set by the user if desired. However a pre-set value of $\widehat{\gamma}$ will be used by default if none

is supplied. If $\widehat{\gamma}$ is not specified by the user, one need only specify the type of technology as either probe-level microarray, gene-level microarray, or RNA-seq. From here a default value of $\widehat{\gamma}$ is chosen. These default values are estimated from spike-in experiments like those just discussed. For both RNA-seq and microarray spike-in data we fit regression models of measured expression on spike-in amount. These are the linear models seen in in the previous sections. We then take the median value of all the estimates of the slopes from each gene's regression model. These form the estimate of $\widehat{\gamma}$. This is done for the RNA-seq data (on the $\log_2$ of the counts plus one) and microarray data (at the RMA-summarized gene level and raw $\log_2$ probe-level). These estimates set the default values for $\widehat{\gamma}$ at $.452$ for probe-level microarrays, $.699$ for gene-level microarrays, and $.943$ for RNA-seq data. For other applications or situations lacking intuition for γ we recommend setting γ to one.

5.1.3.1 Slope Sensitivity

In order to evaluate the sensitivity of dtangle to changes in $\widehat{\gamma}$ we conduct a meta-analysis of dtangle over many values for $\widehat{\gamma}$ (Supplementary Figure 5.10, 5.11, 5.12). dtangle seems to perform poorly if $\widehat{\gamma}$ is less than 0.5. However for $\widehat{\gamma}$ above about 0.5 dtangle is not particularly sensitive to the parameter. In any case dtangle seems robust to changes in $\widehat{\gamma}$ with best performance when $\widehat{\gamma}$ is between 0.5 and 1.

5.2 Investigations Using Simulated Mixtures

To further investigate the role of robust scales, marker genes, cell type co-linearity, and the accuracy of dtangle we investigated the performance of deconvolution methods on a wide range of simulated data.

5.2.1 Methods and Data

Broadly, the data simulation approach we take is to generate a matrix $U \in \mathbb{R}^{K \times N}$ of K reference cell type profiles across N genes, and a matrix $M \in \mathbb{R}^{S \times K}$ of K cell type mixing proportions across S samples and take their product (with some noise) to form a mixture gene expression matrix $X \in \mathbb{R}^{S \times N}$. We simulate data using both Gaussian and Poisson error at the log and linear scales, respectively, so that

1. in the Gaussian case $X \stackrel{def}{=} \exp(\log(MU) + E)$ where $E_{mn} \overset{iid}{\sim} N(0, f\sigma)$, $\sigma = sd(\log(1 + \text{vec}(U))$, and f is a multiplicative error factor controlling the level of noise
2. and in the Poisson case we let $Y_{mn} \overset{iid}{\sim} Pois((MU)_{mn})$.

The Gaussian error model will simulate data with character similar microarray data while the Poisson model will simulate data that more closely resembles RNA-seq.

We generate M with the following structure

$$M = (I_K \mid I_K \mid R)'$$

where the $S - 2K$ columns of R are uniformly drawn from the $(K-1)$-dimensional probability simplex. This structure of M means that the first $2K$ rows of X are just the references U with some error. These first rows of X are thus used as the reference data for deconvolution.

To generate U we follow two broad schemes. We will call the first the "artificial cell type" scheme and the second the "real cell type" scheme.

5.2.1.1 Artificial Cell Type Mixtures

The artificial cell type references were simulated as follows. First we generated a baseline profile $B \in \mathbb{R}^N$ by taking the un-normalized read counts from the Parsons data set and gene-wise taking the median across the 39 samples for each of the $N = 23459$ genes. This baseline profile was then perturbed to create reference profiles for $K = 3$ artificial cell types as follows.

Let $\rho \in (0, 1)$ be the percentage of N genes that are markers of some cell type. Then let $G_k \subset \{1, \ldots, N\}$ be a set of $\lfloor N\rho/K \rfloor$ randomly selected type k marker genes. These genes are randomly selected among those genes in the top quartile of expression in B so that the G_k are mutually disjoint. We then form U through the following two steps:

1. make each reference profile a copy of B,

$$U \leftarrow \mathbb{1}_K \otimes B$$

2. for each cell type k set the expression level of marker genes G_k to some small value $\mu \in \mathbb{R}$ for all reference profiles other than the type k reference,

$$U_{tn} = \mu \text{ for all } n \in G_k, t \neq k \text{ and } k = 1, \ldots, K.$$

This scheme ensures that each cell type k has some set of marker genes G_k that are highly expressed in the type k reference (they are among the top 25% of overall expression) but lowly expressed in all other cell type references (at a low expression level μ).

5.2.1.2 Real Cell Type Mixtures

Our scheme to generate "real" cell type references is much simpler. We let U be the reference of an existing data set. We use references from the Parsons or Linsley data in our simulations. Given the data set (Parsons or Linsley), we let the k^{th} row of U be the median of the reference profiles for the k^{th} cell type in the data set. For both the Parsons and Linsley data $K = 3$, while $N = 23459$ for the Parsons data and $N = 21421$ for the Linsley.

The artificial cell type simulations are useful because they allow us control over many simulation parameters. In addition to controlling the noise level f in the Gaussian case, we can control the percentage of actual marker genes ρ, and the expression level of marker genes in other cell types μ. On the other hand the real cell type simulations are interesting because they are more realistic than the artificially created cell types but also because they allow us to investigate realistic situations where the cell types are very different (Parsons) and very similar (Linsley).

In all cases after X has been generated the data is TPM normalized and analyzed using precisely the same procedure as described in the main body of this paper treating the first $2K$ rows of X as reference samples of the K cell types. Notably we do not reveal the true marker genes to the deconvolution methods in the case of the artificial cell type simulations.

We evaluated the performance of dtangle, the four other partial deconvolution algorithms (CIBERSORT, EPIC, LS Fit, and Q Prog) and a simple linear regression approach, on this simulated data. For all methods other than dtangle we evaluated the algorithms using both the linear scale data as generated, and logarithmically transforming the data using a base-2 logarithm of one plus the expression. Importantly, the other partial deconvolution methods do not fit using log scale data. For example, CIBERSORT's code explicitly forces data to be on a linear scale and EPIC uses linear-scale TPM-transformed read counts. Nonetheless, it will be instructive to look at these methods using both linear and log-scale expressions. We do not do this similar comparison for dtangle because its approach does not fall nicely into either category, it combines both scales. Hence such a comparison does not make sense for dtangle. Instead we put dtangle in its own hybrid category.

The simple regression approach, mentioned above, simply estimates M by regressing the mixture samples' expressions onto the reference expressions. This is done using both linear and log-scale expressions. We included this regression approach because it serves as an easily understandable baseline against which we may compare other methods.

5.2.2 Scale and Robustness

In Supplementary Figure 5.29 we plot boxplots of error and correlation along with scatter plots of estimates against true mixing proportions showing the performance of dtangle, the four partial deconvolution methods, and the linear regression approach, on artificial cell type simulated data

with a low level of Gaussian error. We display these plots of the methods using both linear and log-scale expressions. The data was simulated so that 15% of the genes were markers ($\rho = .15$), the marker genes were only expressed by the cell type they mark ($\mu = 0$), and the added Gaussian error is 2.5% of the typical variance among expressions ($f = .025$). We plot similar figures using the Poisson error structure and the same values of ρ and μ in Supplementary Figure 5.34. Similarly we plots these figures for the real cell type mixtures using the Parsons data (Gaussian error: Supplementary Figure 5.38, Poisson error: Supplementary Figure 5.41) and the Linsley data (Gaussian error: Supplementary Figure 5.43, Poisson error: Supplementary Figure 5.46). The same value of f is used for the real cell type simulations with Gaussian error.

We can see from all of these figures that broadly dtangle out-performs other methods but also that the other partial deconvolution methods tend to perform better deconvolving linear scale expressions than the log-scale expressions. This makes sense because our data has been simulated as a linear mixture of linear scale expressions and since the simulation error in these figures is small ($f = .025$ in for the Gaussian simulations). The simulated data follows exactly the model presumed by these methods.

We argue that while a linear mixing of linear expressions is a plausible model, it is not robust. To show that this is true we adjust our simulations in two ways. First, we look at the same Gaussian simulations but change the error factor f from 2.5% to 75% (i.e. $f = .75$). The same plots with a high level of Gaussian error are Supplementary Figure 5.30, 5.39, 5.44. While dtangle still out-performs other methods, we see now that the other partial deconvolution algorithms perform better using log-scale expressions than linear scale expressions. Notably the data has still been generated using a linear mixing model of linear expressions, we have only increased the Gaussian error. Yet the log-scale expressions give a better fit even though the model is mis-specified fitting with log-scale expressions. The reason the log-scale expressions give a better fit in the high-error situation is because the logarithmic transformation attenuates the effects of the highly skewed data and the undue influence of points in the tail of the data.

A similar situation occurs if we leave the error low ($f = .025$ for Gaussian simulations) but add outliers to the data. For both the Gaussian and Poisson error structure we simulate data as previously described but then add five random outliers to each of the reference profiles in X. The value of these outlying points is set (on the log-scale) as 1.25 times the largest observation before adding any outliers. We plot similar figures for each of the simulations after adding outliers (see Supplementary Figure 5.31, 5.35, 5.40, 5.42, 5.45, 5.47). Largely the results are the same as the high-error Gaussian case. We see that the other partial deconvolution algorithms perform better after a logarithmic transformation since this ameliorates the effects of the outliers. Notably dtangle is relatively unperturbed by the outliers. This is because dtangle robustly combines expressions on a log-scale before averaging. Furthermore dtangle's averaging approach is not highly influenced by

a single outlying point. In contrast, the outlying point becomes a high-leverage point for the other regression-like partial deconvolution approaches and thus is highly-influential on the estimates. This effect of outliers is also seen very prominently in the Shen-Orr dataset. In this dataset, as in the simulations, dtangle's robust approach of working with log-scale expressions but using a linear mixing model of linear expressions does exceptionally well. The other deconvolution algorithms are completely misled by an outlier.

5.2.3 Marker Genes

5.2.3.1 Marker Gene Expression

A central feature of dtangle's approach its use of marker genes. Broadly, we define a marker gene as one which is expressed predominantly in only one cell type. All deconvolution methods seem to benefit from marker genes. Typically, they are used to sub-set the data on which the model is fit. dtangle has a unique use of marker genes and rigorously defines marker genes as only being expressed in one cell type. Nonetheless, we realize that this assumption is a mathematical approximation to the truth and so it is worth investigating what happens to dtangle when it is violated.

To investigate this we simulate data according to our artificial cell type simulation scheme with $\rho = .15$ so that 15% of the genes are marker of some cell type and using both a Gaussian error structure ($f = .025$) and a Poisson error structure. We then vary the expression of marker genes in other cell types μ. In Supplementary Figure 5.33 and Supplementary Figure 5.37 we plot (A) the absolute error and (B) correlation of dtangle's estimates from the truth varying the value of μ from the minimum of the data to letting μ be the maximum of the data. We plot the error or correlation on the y-axis and set μ to be the q^{th} quantile of the data varrying q along the x-axis.

In either case we see that as we increase the value of μ, and as it gets further from our mathematical assumption that $\mu = 0$, the error of dtangle increases. However this increase is very slow. Indeed, the marker genes do not need to have a true expression of $\mu = 0$ in all other cell types. So long as the expression of marker genes in other cell types is in, say, the bottom 25% of all gene expression dtangle does very well. Thus dtangle seems quite robust to this marker gene assumption.

5.2.3.2 Number of Marker Genes and Co-linearity

For all deconvolution algorithms, including dtangle, it is important to find a good set of marker genes. In real data, the primary reason it can be difficult to find marker genes is some combination of (1) that there are many cell types we wish to deconvolve and (2) the cell types we wish to deconvolve are closely related. This follows because our definition of a marker gene is a gene that

is highly expressed in only one cell type. Thus the more cell types we have, the harder it is to find a gene is that expressed highly in only one of the cell types. Similarly, if the cell types we wish to deconvolve a closely related and their expression profiles are highly co-linear then finding genes highly expressed in one cell type but not the others is difficult.

To explore the performance of dtangle in situations where marker genes are hard to identify we simulate according to our artificial cell type scheme and vary the percentage of genes that are markers of some cell type (ρ). For a Gaussian error structure we plot in Supplementary Figure 5.32 the (A) error or (B) correlation of dtangle's estimates against the true mixing proportions, on the y-axis, against the percentage of marker genes in the data (ρ), on the x-axis. We vary ρ from 0.01 to 0.2. We plot a similar plot in Supplementary Figure 5.36 using the Poisson error structure. What we can see from these two figures is that dtangle's performance only suffers drastically when less than about 2-3% of the genes are good markers of a cell type. So long as at least 3-5% of the genes in the data are markers of some type dtangle does quite well.

To explore this issue further we also revisit the results from the real cell type mixture simulations, Supplementary Figure 5.38-5.47. In these simulations we simulated mixtures of using the references from the Parsons and Linsley data sets. The Parsons data set is a mixture of three very distinct cell types: Brain, Liver and Muscle. On the other hand the Linsley data set is a mixture of three closely related white blood cell classes: Lymphocytes, Monocytes, and Neutrophils. It should be relatively easy to find marker genes for the Parsons data set, because the cell types are very distinct, and relatively more difficult to find marker genes for the simulated mixture of closely-related white blood cells using the Linsley reference data. While we do see that dtangle has relatively more trouble deconvolving the Linsley-derived simulations than the Parsons-derived simulations, e.g. compare Supplementary Figure 5.38 to Supplementary Figure 5.43, dtangle still does well over-all. Indeed dtangle still out-performs the other partial deconvolution methods.

Over all we see that dtangle, like other deconvolution algorithms, will suffer if there are almost no marker genes of the cell types. However dtangle is quite robust and works well with as few as a couple of percent of the genes being marker of some cell type.

5.2.4 Other Remarks

5.2.4.1 Accuracy as a Function of the Truth

We see from all of these simulations that the accuracy of dtangle's estimates do not seem to depend strongly on the true mixing proportion. That is, dtangle estimates accurately when the true mixing proportion is close to zero and when the true mixing proportion is close to one.

5.2.4.2 Gamma

The simulations we have explored in this section follow a linear mixing model of linear expressions. This is a simplification of the model that dtangle posits. It is simplified because dtangle's model also includes an adjustment term γ. Hence simulations strictly according to dtangle's model would posit a linear mixing of slightly transformed linear expressions. We chose to simulate linear mixing on a linear scale because this is the model assumed by other deconvolution algorithms. Thus our simulations should be a fair analysis of these other deconvolution algorithms because it follows their model, not dtangle's. Effectively, we have simulated data assuming $\gamma = 1$. This shows that dtangle works quite well even when γ is not required in the model.

5.3 The Mathematics of dtangle

Assume we have a mixture sample of K cell types. Let $Y \in \mathbb{R}^N$ be the (base-2) log-scale expression measurements of this mixture sample and $p_1, \ldots, p_K$ be the mixing proportions of the cell types. For $k = 1, \ldots, K$ assume that there are ν_k reference samples of cell type k and let $Z_{kr} \in \mathbb{R}^N$ be the log-scale expressions of the r^{th} type k reference. Furthermore, let $G_k \subset \{1, \ldots, N\}$ be the set of type k marker genes. We require that these marker gene sets are mutually disjoint.

Let $g_k = |G_k|$ and define $\overline{Y_{G_k}} = \frac{1}{g_k} \sum_{n \in G_k} Y_n$ and $\overline{Z_{G_k}} = \frac{1}{g_k \nu_k} \sum_{n \in G_k} \sum_{r=1}^{\nu_k} Z_{krn}$ to be the average of all type k marker genes across the mixture and reference samples, respectively. Finally, denote our "adjustment term" as $\gamma \approx 1$.

Let η_{kn} be the actual linear-scale expression of the n^{th} gene in a sample of type k cells and η_n be the actual linear-scale expression in the mixture, then dtangle assumes these actual expressions mix linearly,

$$\eta_n = \sum_{k=1}^{K} p_k \eta_{kn}. \tag{5.1}$$

Furthermore dtangle assumes that the measured log-scale expressions are linear in the actual log-scale expressions,

$$\begin{aligned} Y_n &= \mu + \theta_n + \gamma \log_2(\eta_n) + \varepsilon_n \\ Z_{krn} &= \alpha + \theta_n + \gamma \log_2(\eta_{kn}) + \varepsilon_{krn}. \end{aligned} \tag{5.2}$$

and that marker genes are (approximately) expressed by only one cell type so that if n is a marker gene for cell type k ($n \in G_k$) then

$$\eta_{\ell n} = 0 \text{ for all } \ell \neq k. \tag{5.3}$$

Combining Equation 5.1 , Equation 5.2 and Equation 5.3 we then have that for $n \in G_k$,

$$\begin{aligned} Y_n &= \mu + \theta_n + \gamma \log_2 (p_k \eta_{kn}) + \varepsilon_n \\ &= \mu + \theta_n + \gamma \log_2 (p_k) + \gamma \log_2 (\eta_{kn}) + \varepsilon_n \\ Z_{krn} &= \alpha + \theta_n + \gamma \log_2 (\eta_{kn}) + \varepsilon_{krn}. \end{aligned} \tag{5.4}$$

So for $n \in G_k$ we have

$$\begin{aligned} \overline{Y_{G_k}} &= \mu + \overline{\theta_{G_k}} + \gamma \log_2 (p_k) + \gamma \overline{\log_2 (\eta_{G_k})} + \overline{\varepsilon_{G_k}} \\ \overline{Z_{G_k}} &= \alpha + \overline{\theta_{G_k}} + \gamma \overline{\log_2 (\eta_{G_k})} + \overline{\overline{\varepsilon_{G_k}}}. \end{aligned} \tag{5.5}$$

where

$$\begin{aligned} \overline{\theta_{G_k}} &= \frac{1}{g_k} \sum_{n \in G_k} \theta_n \\ \overline{\log_2 (\eta_{G_k})} &= \frac{1}{g_k} \sum_{n \in G_k} \log_2 (\eta_{kn}) \\ \overline{\varepsilon_{G_k}} &= \frac{1}{g_k} \sum_{n \in G_k} \varepsilon_n \\ \overline{\overline{\varepsilon_{G_k}}} &= \frac{1}{g_k \nu_k} \sum_{n \in G_k} \sum_{r=1}^{\nu_k} \varepsilon_{krn}. \end{aligned}$$

This means

$$\begin{aligned} \overline{Y_{G_k}} - \overline{Y_{G_t}} &= \gamma \log_2 (p_k / p_t) \\ &\quad + \overline{\theta_{G_k}} - \overline{\theta_{G_t}} + \gamma \overline{\log_2 (\eta_{G_k})} - \gamma \overline{\log_2 (\eta_{G_t})} \\ &\quad + \overline{\varepsilon_{G_k}} - \overline{\varepsilon_{G_t}} \\ \overline{Z_{G_k}} - \overline{Z_{G_t}} &= \overline{\theta_{G_k}} - \overline{\theta_{G_t}} + \gamma \overline{\log_2 (\eta_{G_k})} - \gamma \overline{\log_2 (\eta_{G_t})} \\ &\quad + \overline{\overline{\varepsilon_{G_k}}} - \overline{\overline{\varepsilon_{G_t}}} \end{aligned} \tag{5.6}$$

and so

$$\begin{aligned} D_{kt} &= \frac{1}{\gamma} \left(\left(\overline{Y_{G_k}} - \overline{Y_{G_t}} \right) - \left(\overline{Z_{G_k}} - \overline{Z_{G_t}} \right) \right) \\ &= \log_2 (p_k / p_t) + \delta \end{aligned} \tag{5.7}$$

where $\delta = \frac{1}{\gamma} \left(\overline{\varepsilon_{G_k}} - \overline{\overline{\varepsilon_{G_k}}} - \overline{\varepsilon_{G_t}} + \overline{\overline{\varepsilon_{G_t}}} \right)$.

Now as $g_k \to \infty$ for all k then $\delta \to 0$ and so for a reasonably large number of marker genes

$$D_{kt} \approx \log_2 (p_k / p_t)$$

and so since $D_k = (D_{k1}, \ldots, D_{kK})$ then

$$D_k \approx \left(\log_2\left({}^{p_k}/_{p_1}\right), \ldots, \log_2\left({}^{p_k}/_{p_K}\right)\right)$$

and so if $L_k(x) = {}^{1}/_{\left(1+\sum_{t\neq k} 2^{-x_t}\right)}$ then

$$L_k(D_k) \approx p_k$$

and so the dtangle estimator $L_k(D_k)$ approximates p_k.

Scale	Actual Expr. (Unobserved)
Linear	mRNA amount
Log	$\log_2$(mRNA amount)

GEP tech. ⇨

Scale	Measured Expr. (Observed)
Linear	gene expr. measurement
Log	$\log_2$(gene expr. measurement)

Figure 5.1: Measured expressions ($\log$ or linear) arise from a measurement process on the actual expressions ($\log$ or linear).

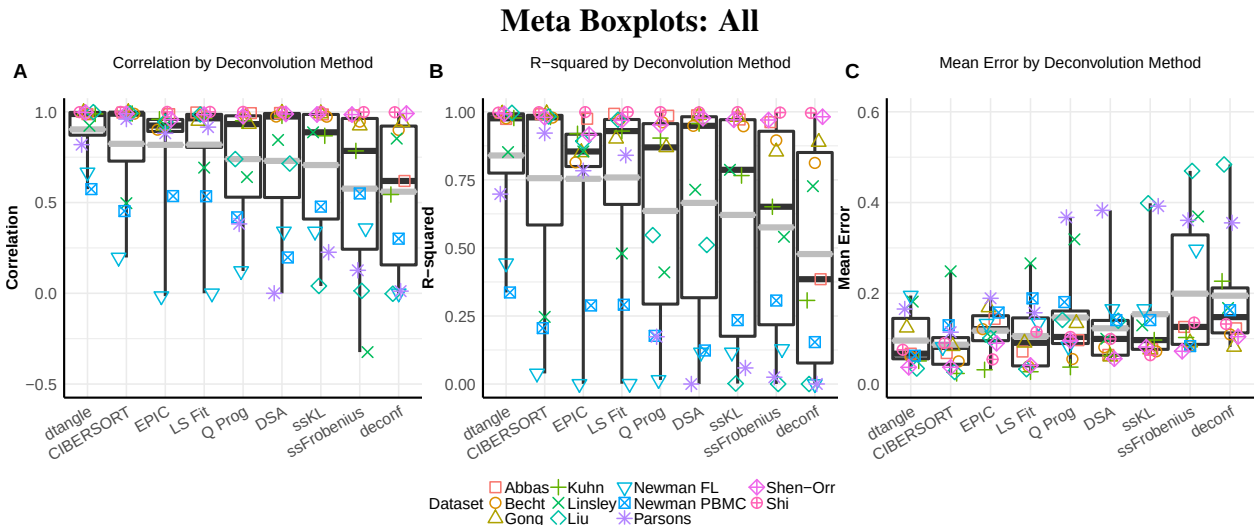
Deconvolution Methods

Name	Citation	
dtangle	(This publication)	Partial
CIBERSORT	[Newman *et al.*, 2015]	
EPIC	[Valencia *et al.*, 2017]	
LS Fit	[Abbas *et al.*, 2009]	
Q. Prog	[Gong *et al.*, 2011]	
deconf	[Repsilber *et al.*, 2010]	Full
DSA	[Zhong *et al.*, 2013]	
ssFrobenius	[Gaujoux and Seoighe, 2012]	
ssKL	[Gaujoux and Seoighe, 2012]	

Table 5.1: Nine deconvolution algorithms we compare.

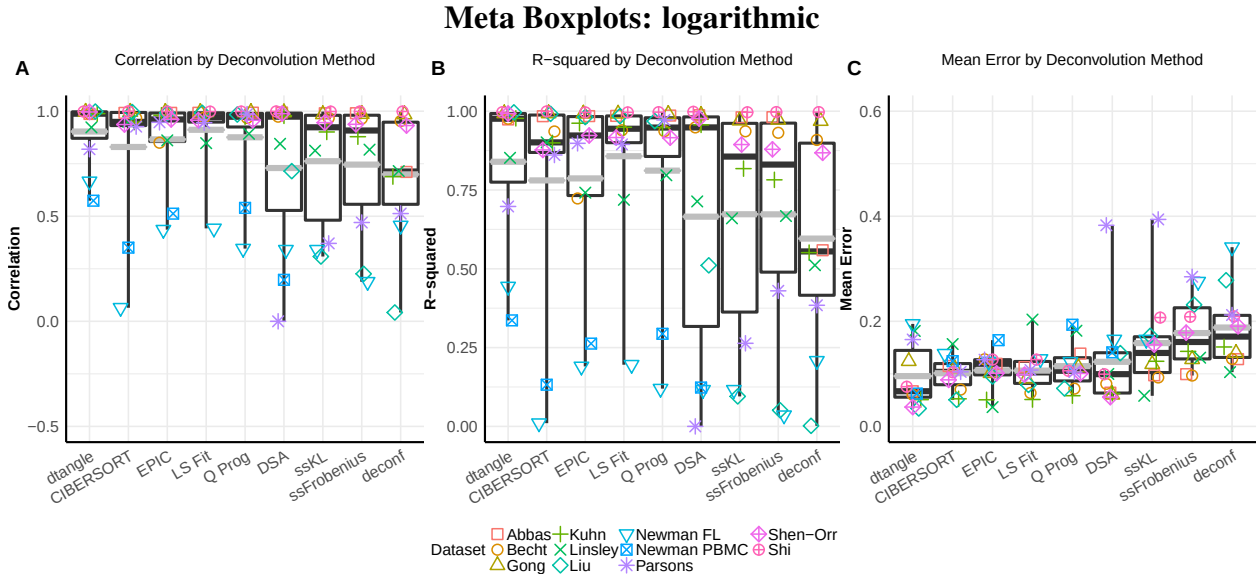
Name	Citation	Accession	Tech.	Truth	Genes	Samples	Reference (num., source)	Cell Types (num., source)	Species
Shi	MAQC [2006]	GSE5350	ma	mix	54,676	60	60, internal	2, universal, brain	human
Gong	Gong *et al.* [2011]	GSE29832	ma	mix	54,676	9	6, internal	2, blood, breast	human
Shen-Orr	Shen-Orr *et al.* [2010]	GSE19830	ma	mix	31,100	33	9, internal	3, liver, brain, lung	rat
Abbas	Abbas *et al.* [2009]	GSE11058	ma	mix	54,676	12	12, internal	4, leukocytes	human
Becht	Becht *et al.* [2016]	GSE64385	ma	mix	54,676	10	766, external	6, colorectal carcinoma, leukocytes	human
Kuhn	Kuhn *et al.* [2011]	GSE19380	ma	mix	31,100	10	16, internal	4, brain	rat
Newman FL	Newman *et al.* [2015]	GSE65136	ma	cyto.	11,189	14	113, external	12, leukocytes	human
Newman PBMC	Newman *et al.* [2015]	GSE65133	ma	cyto.	11,049	20	113, external	22, leukocytes	human
Parsons	Parsons *et al.* [2015]	PRJEB8231	seq	mix	23,459	30	9, internal	3, brain, liver, muscle	human
Liu	Liu *et al.* [2015]	GSE64098	seq	mix	23,056	24	16, internal	2, adenocarcinoma	human
Linsley	Linsley *et al.* [2014]	GSE60424	seq	cyto.	21421	5	3, external	3, lymphocytes, monocytes, neutrophils	human

Table 5.2: Benchmark data sets on which we compare deconvolution algorithms. The accession key is for GEO (or in the case of Parsons, ENA). The technology producing the data is either "ma" for microarray or "seq" for RNA-seq. The column "Truth" distinguishes between mixture experiments "mix" or data where the truth is known from flow cytometry "ctyo." The number of gene expression measurements made by the technology is the column "Genes" and the number of unknown heterogeneous samples deconvolved is the column "Samples." The column "Reference" lists the number of samples in the reference data along with the designation of "internal" if the pure reference samples were created part and parcel with the mixture experiment or "external" if the reference samples were collected from external data sources (typically GEO). The column "Cell Types" lists the number of cell types in the mixture samples and provides a description of the cell types along with the species from which the cell types come (in the column "Species").



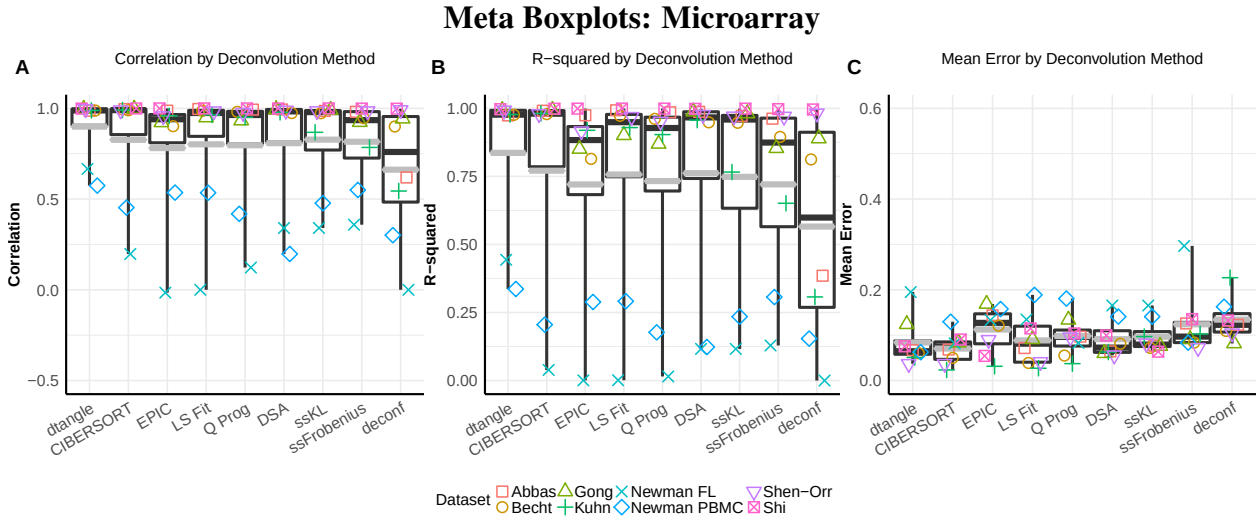
label=paperplots:metaboxplotsall

Figure 5.2: Boxplots of all deconvolution methods across all data-sets. Top 10% of the 25% of most variable genes are used as marker genes used for deconvolution. Marker genes determined by median differences across reference samples. Slope (γ) for dtangle determined automatically by data-type. (A) For each cell type the correlation of the true mixing proportions against the estimated mixing proportions is calculated. If the s.d. of the estimates is zero, we say the correlation is zero. If the s.d. of the true proportions is zero we do not calculate the correlation. Each point is the median of the correlations across cell types. We calculate this median correlation for each data-set and each deconvolution method. (B) Similar to (A) except using R^2 instead of correlation. (C) is similar to (A) but using grand means instead of correlation. For each cell type the absolute value of the error of the estimated mixing proportions from the true mixing proportions is calculated. Each point is the mean of the errors across cell types. We calculate this mean for each data-set and each deconvolution method.



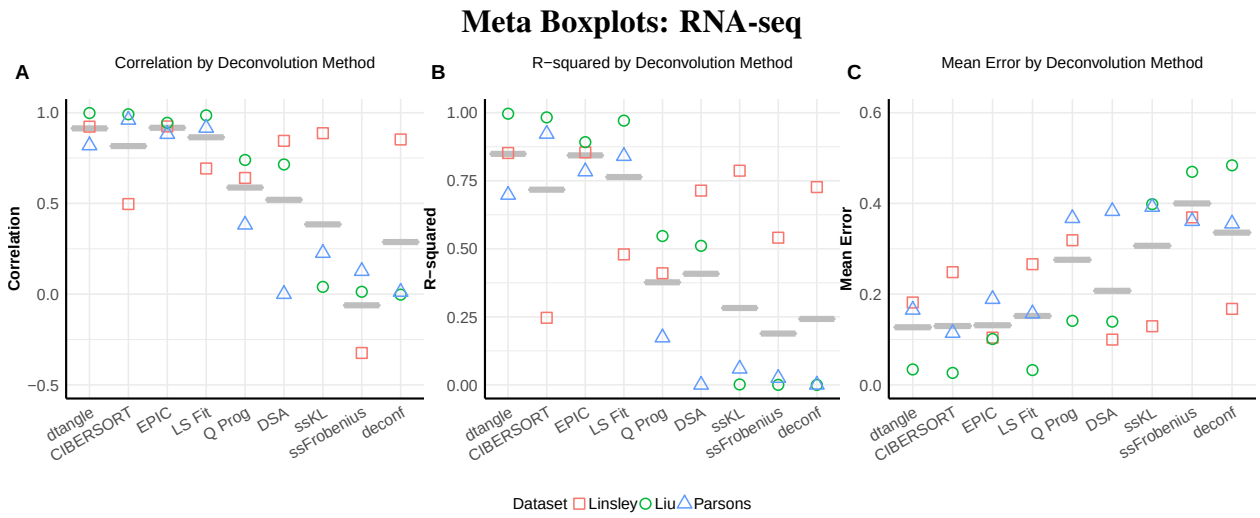
label=logpaperplots:metaboxplotsall

Figure 5.3: Similar to Figure 5.2 but applying methods to log transformed data.



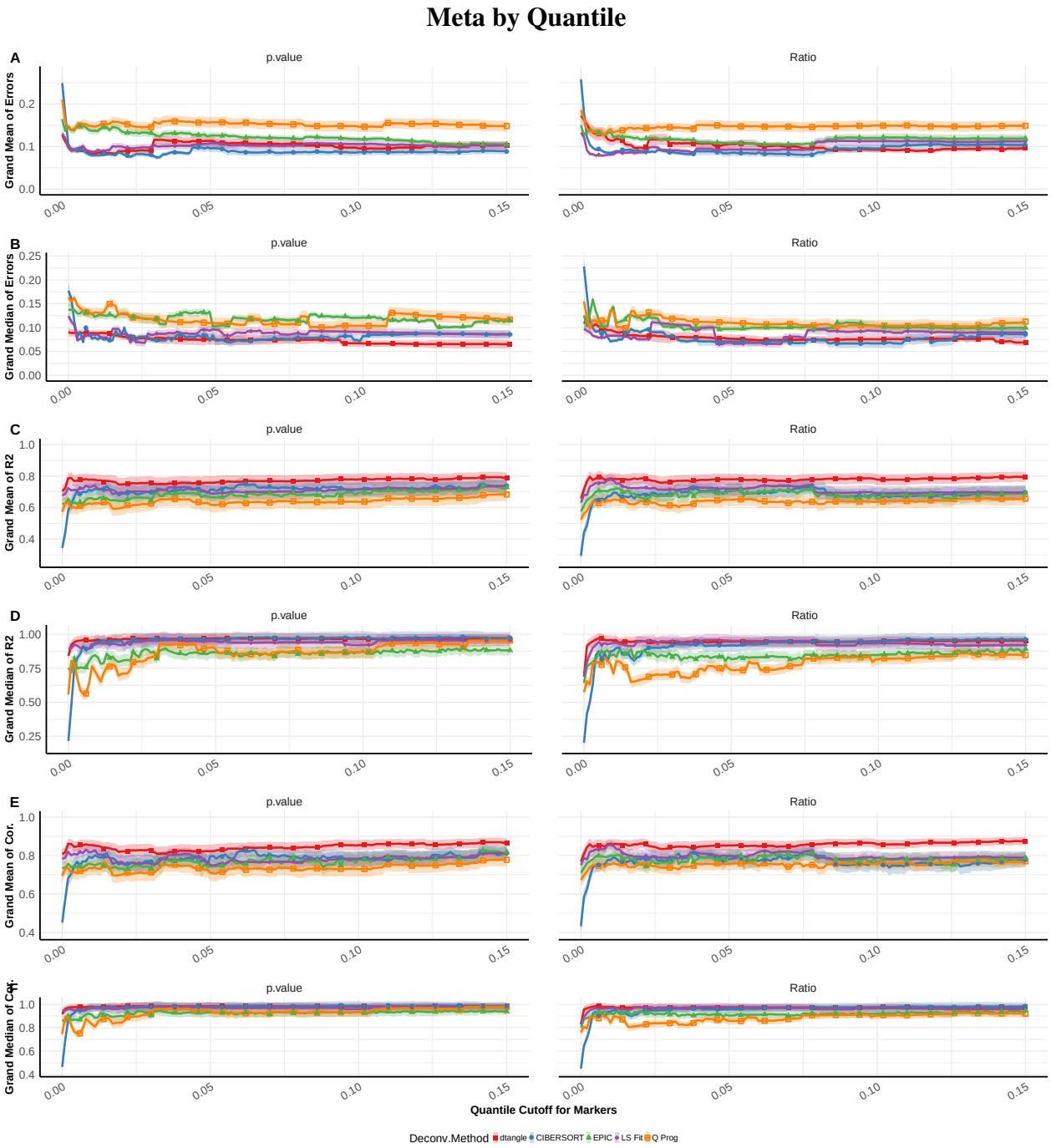
label=paperplots:metaboxplotsma

Figure 5.4: Similar to Figure 5.2 but only comparing microarray data-sets.



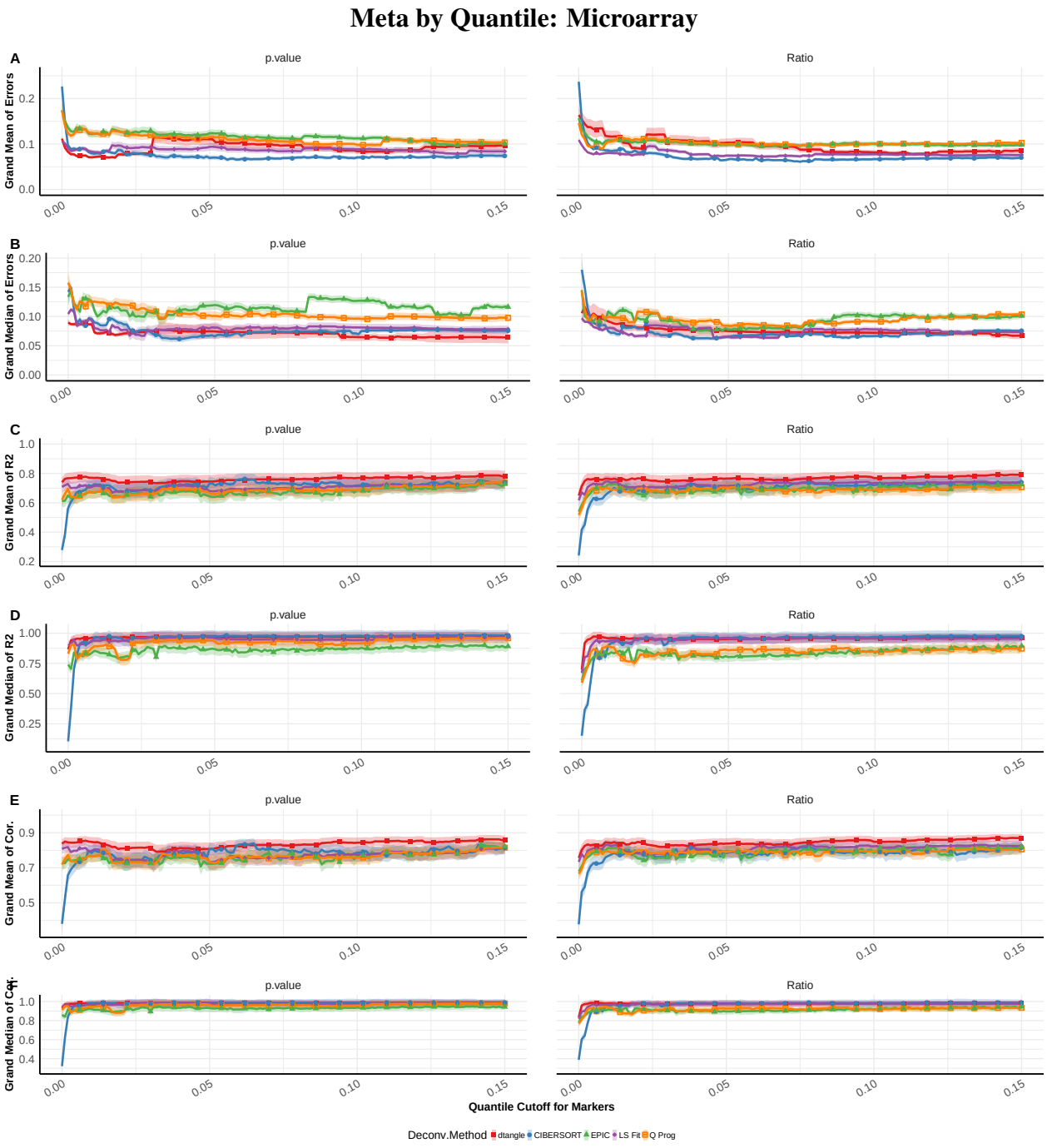
label=paperplots:metaboxplotsseq

Figure 5.5: Similar to Figure 5.2 but only comparing RNA-seq data-sets.



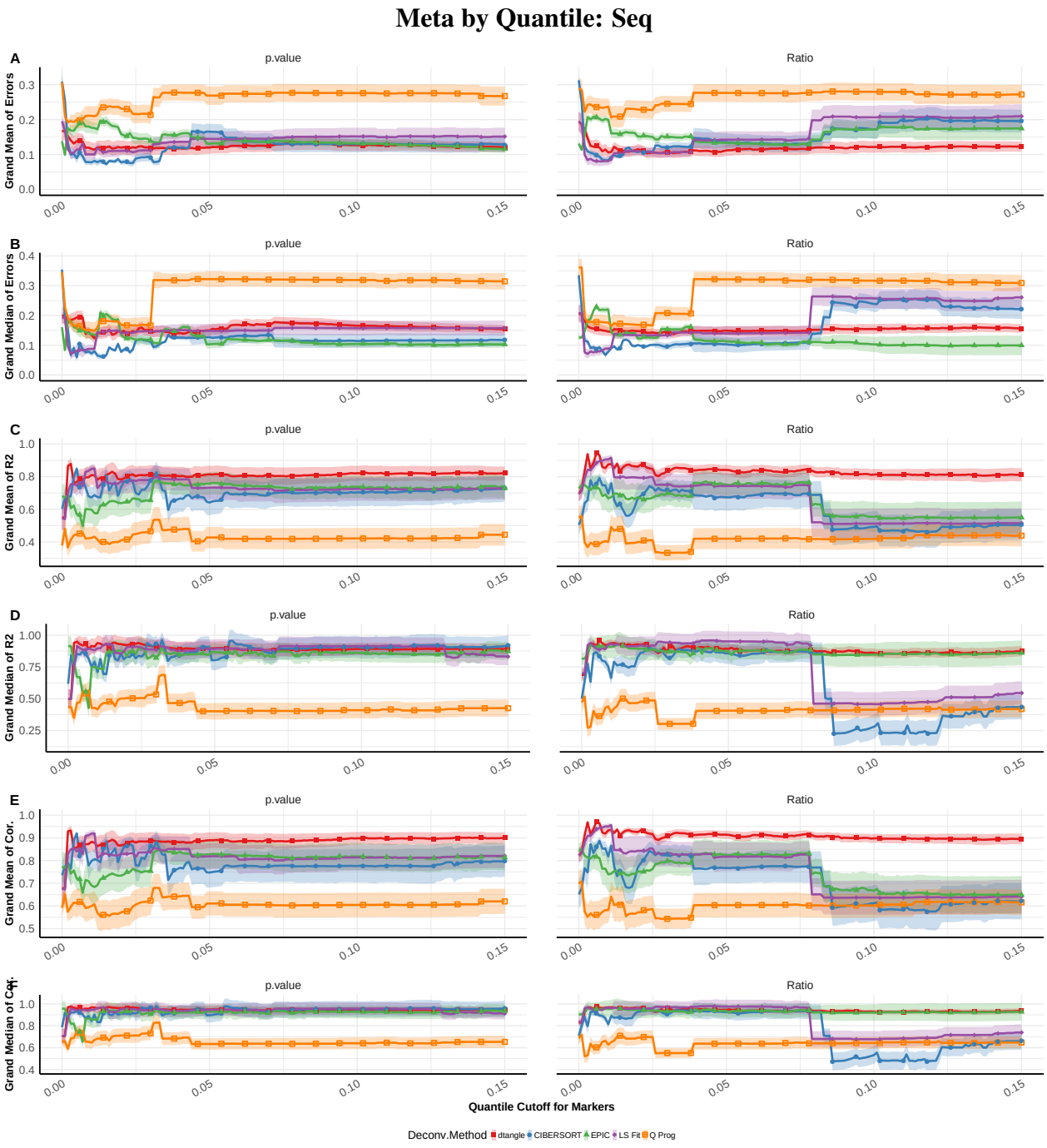
label=paperplots:quantile:all

Figure 5.6: Partial deconvolution methods performance (y-axis) by number of marker genes (quantile, x-axis). Slope (γ) for dtangle determined automatically by data-type. Top $q\%$ of top 25% of most variable genes used for deconvolution where q varies over the x-axis from 1% to 15% (in increments of 1%). Marker genes determined by p-value (Left) and ratio of the linear expression of each type to the expression in all other types (Right). The y-axis is the grand (A) mean or (B) median (over data-sets and cell types) of the absolute error of the true proportions from the estimated proportions, or the grand (C) mean or (D) median of the R^2 or correlations (E, F) of the estimated proportions against the true proportions. The correlation is zero if the s.d. of the estimates is zero and the correlation is not computed if the s.d. of the true proportions is zero. One line is plotted for each partial deconvolution method. Error ribbons displaying 95% confidence intervals.



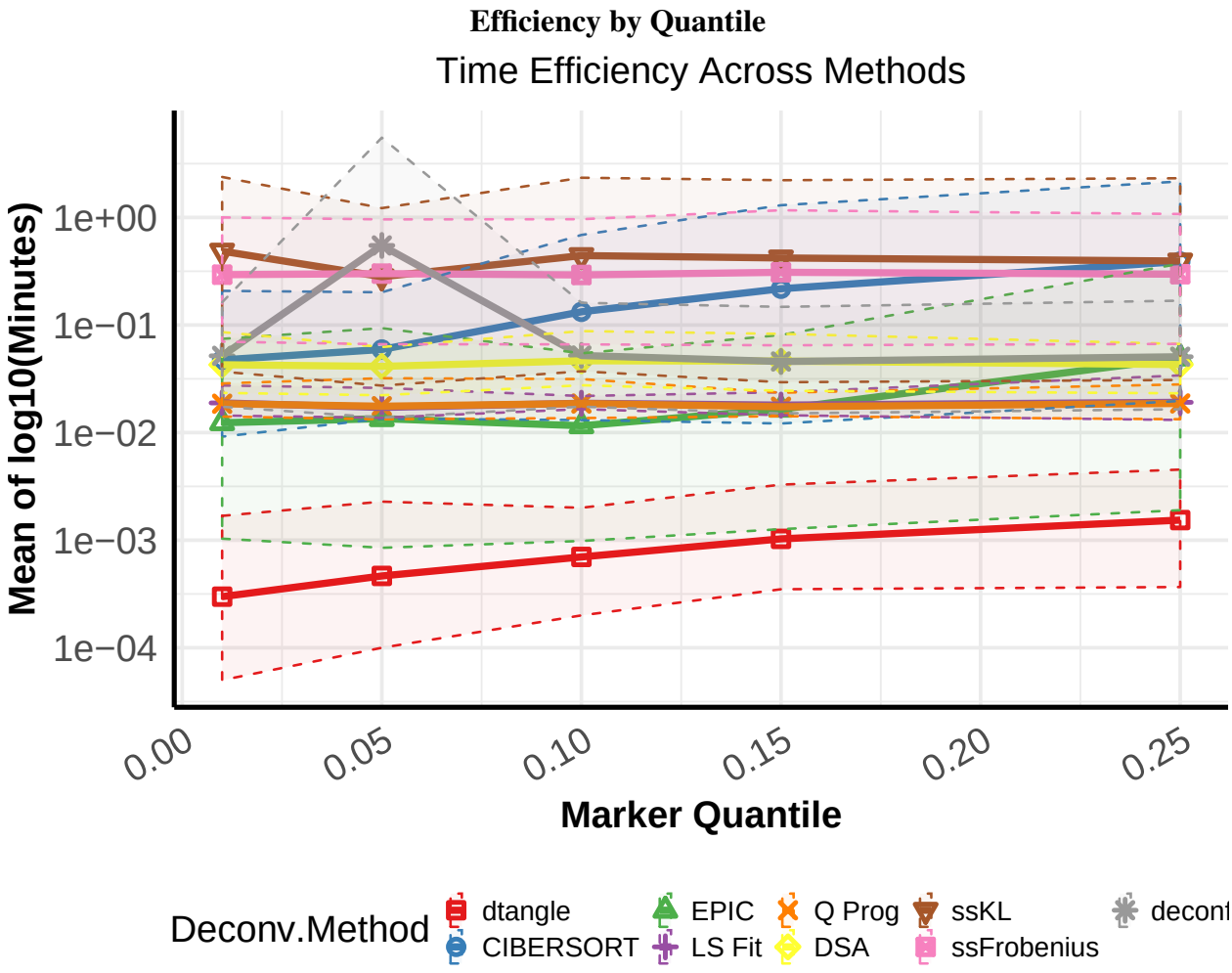
label=paperplots:quantile:ma

Figure 5.7: Similar to Figure 5.6 except only comparing microarray datasets.



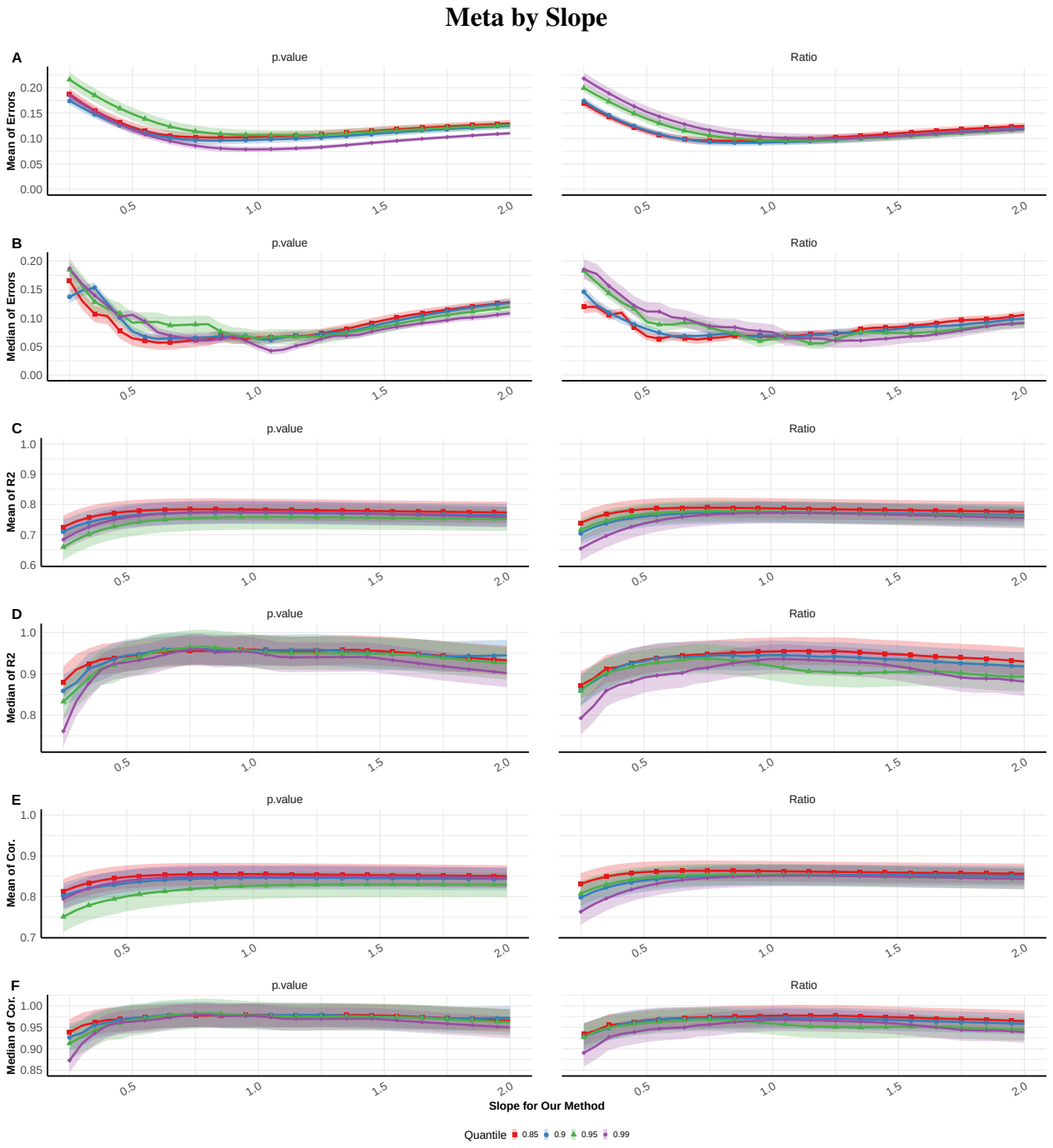
label=paperplots:quantile:seq

Figure 5.8: Similar to Figure 5.6 except only comparing RNA-seq datasets.



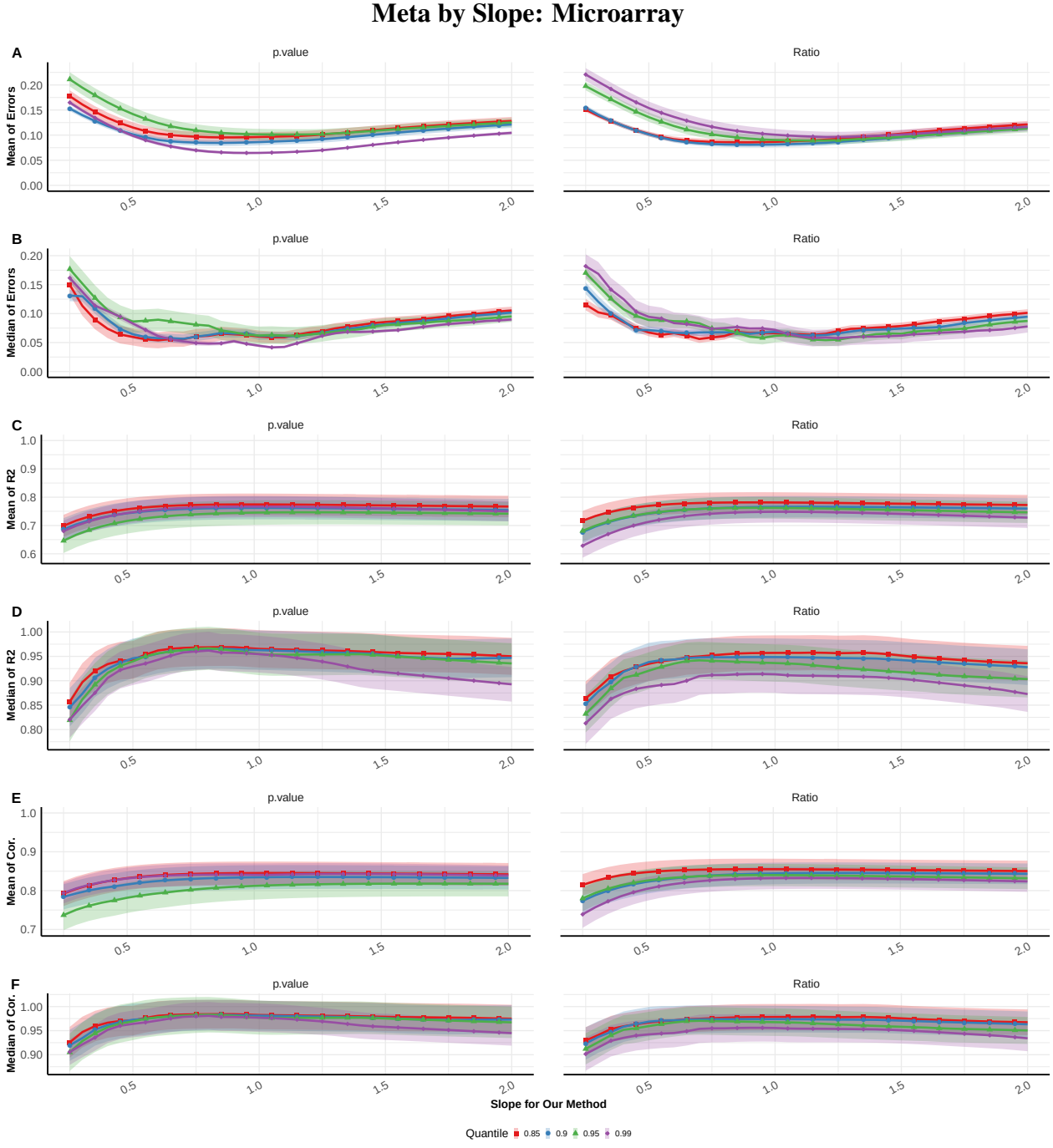
label=time:timeplot

Figure 5.9: Mean of $\log_{10}$ of time (in minutes) each algorithm took to deconvolve all data sets. Maximum and minimum value envelope is included.



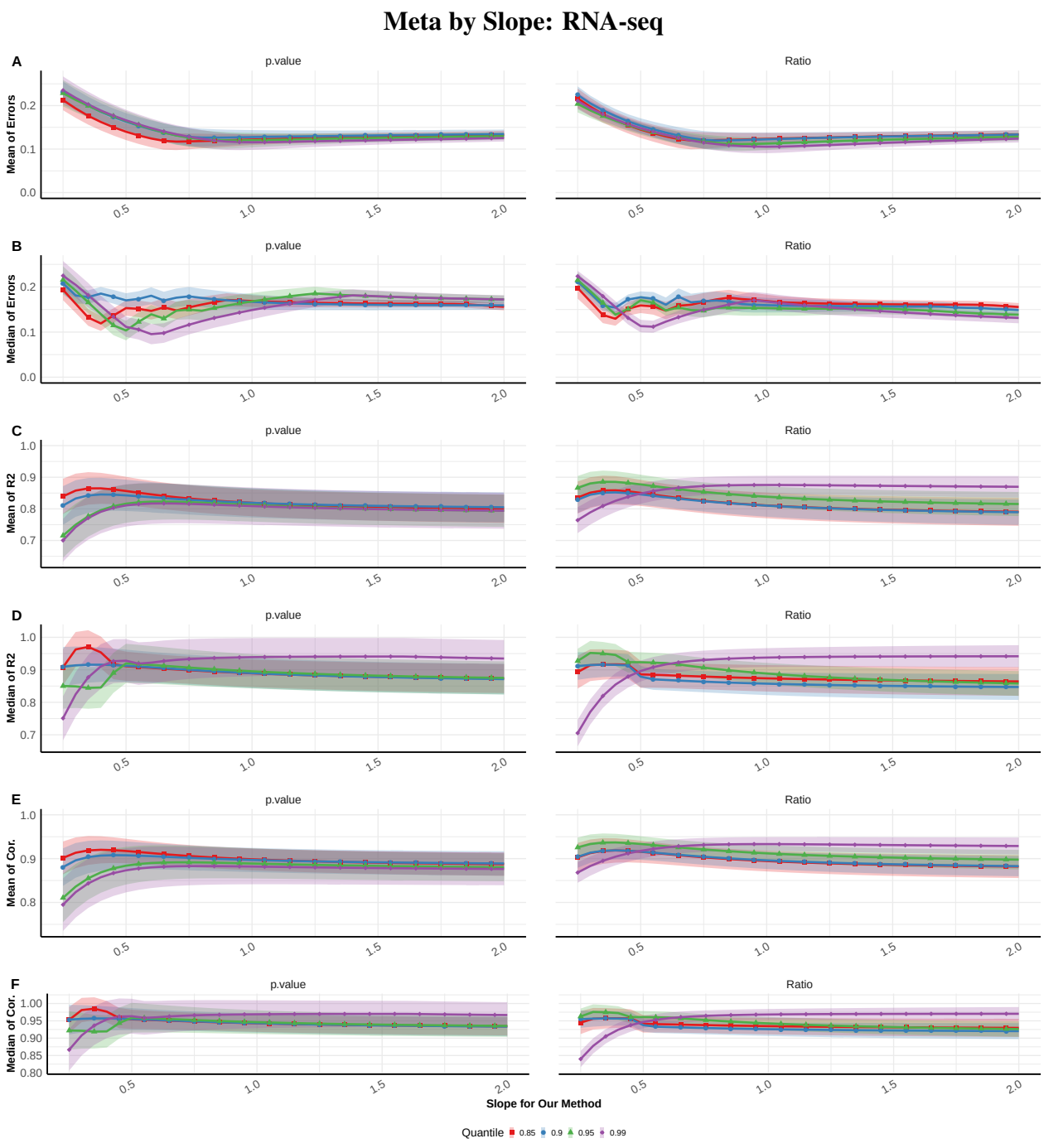
label=ss:all

Figure 5.10: dtangle performance (y-axis) by slope (γ) varrying over x-axis from 0.25 to 2 (in increments of 0.05). Marker genes determined by p-value (Left) and ratio of the linear expression of each type to the expression in all other types (Right). The y-axis is the grand (A) mean or (B) median (over data-sets and cell types) of the absolute error of the true proportions from the estimated proportions, or the grand (C) mean or (D) median of the correlations of the estimated proportions against the true proportions. The correlation is zero if the s.d. of the estimates is zero and the correlation is not computed if the s.d. of the true proportions is zero. One line is plotted for four choices of number of markers using only the top 1%, 5%, 10% or 15% of top 25% most variables genes as markers. Error ribbons displaying 95% confidence intervals.



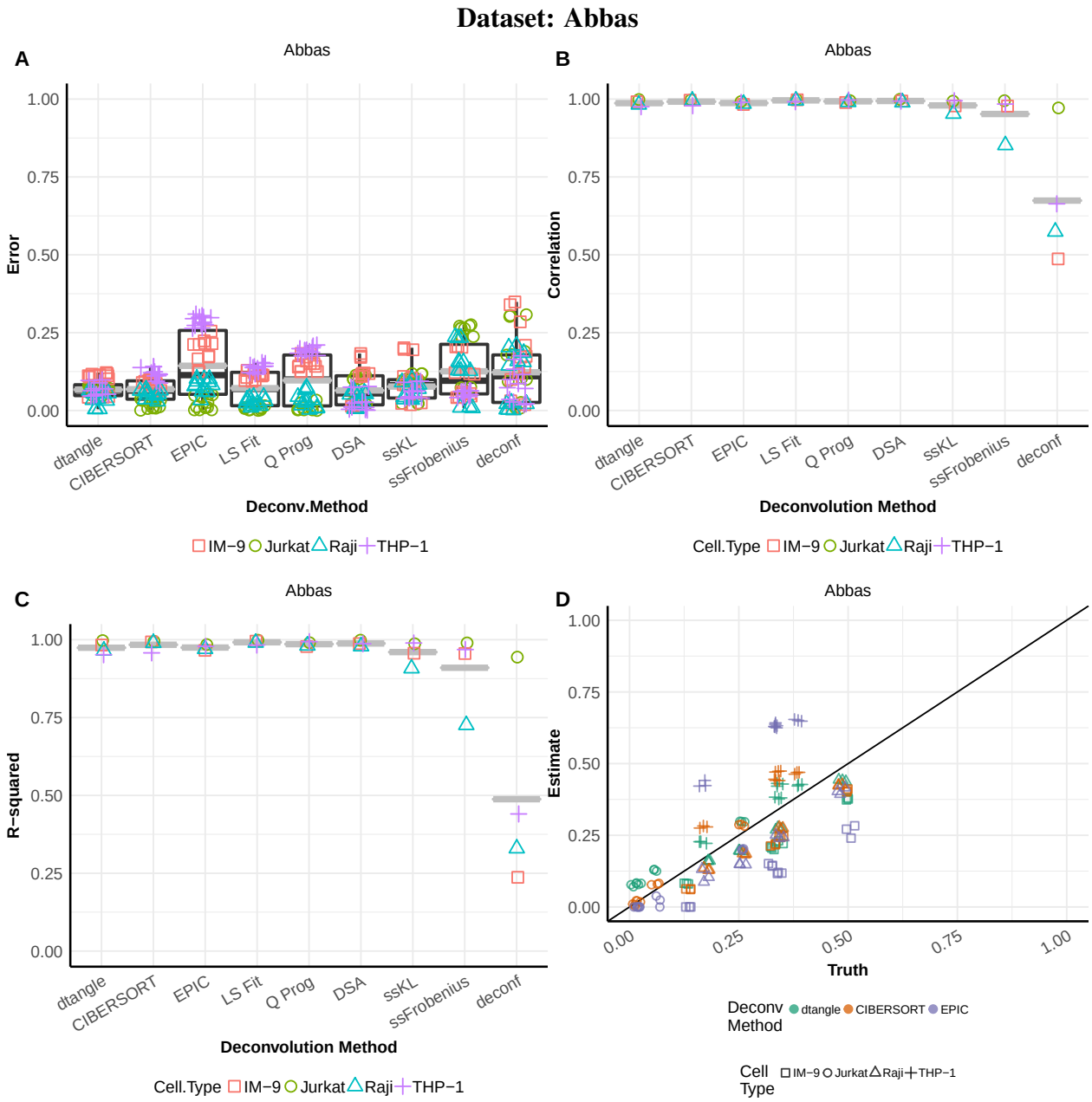
label=ss:ma

Figure 5.11: Similar to Figure 5.10 but only comparing microarray data-sets.



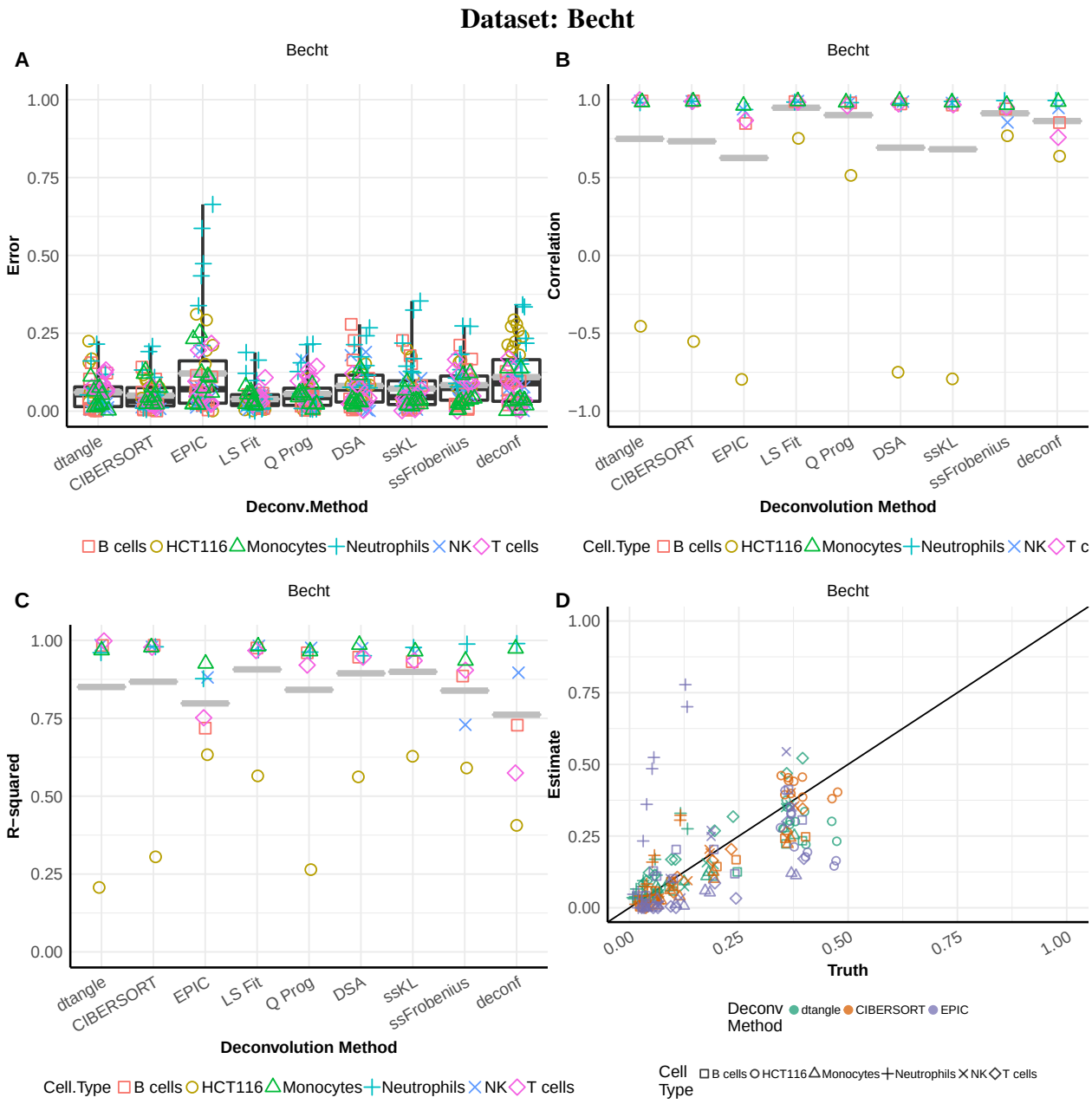
label=ss:seq

Figure 5.12: Similar to Figure 5.10 but only comparing RNA-seq data-sets.



label=paperplots:datasets:Abbasall

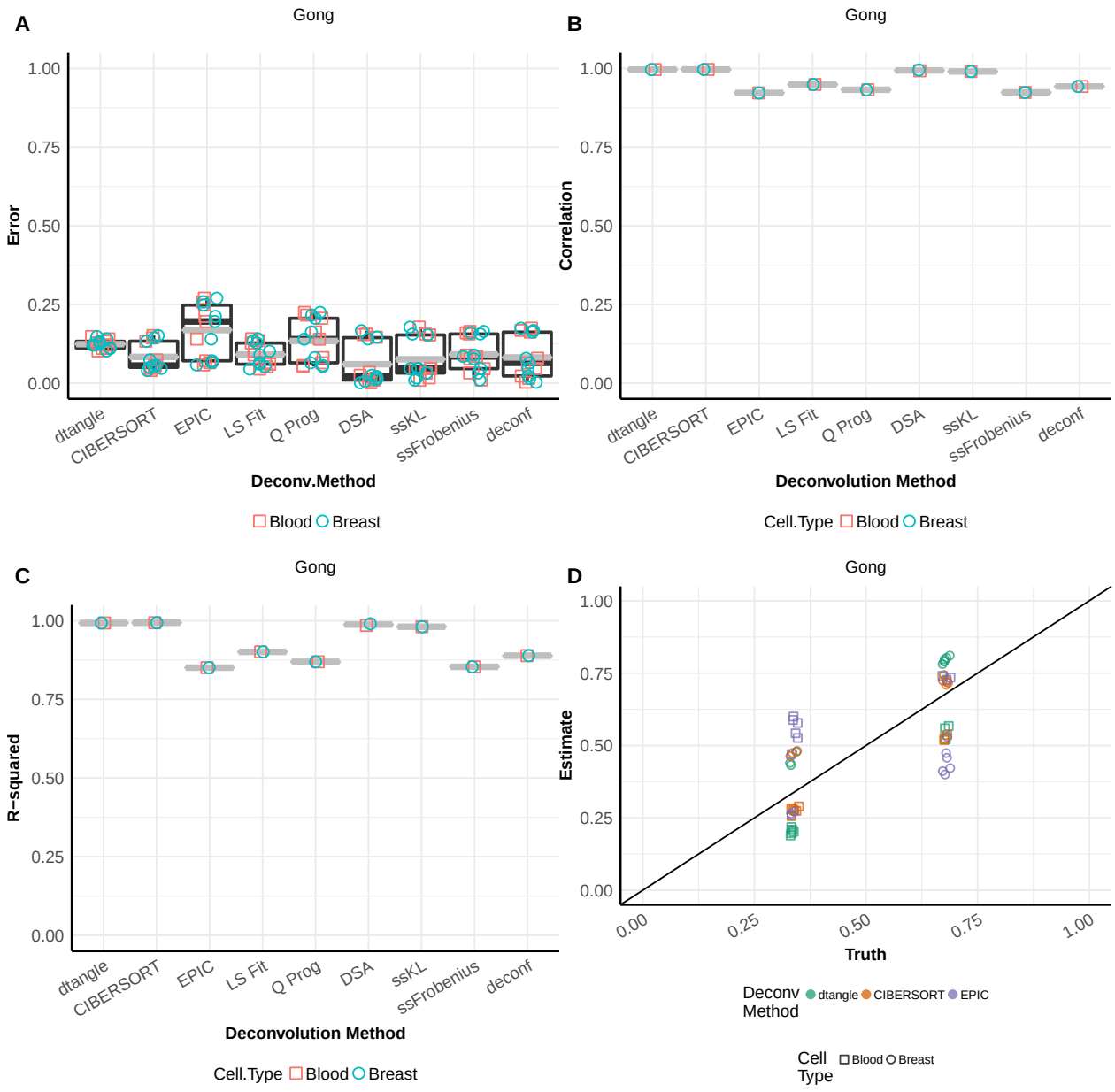
Figure 5.13: Deconvolution methods performance on Abbas data-set. Slope (γ) for dtangle determined automatically by data-type. Top 10% of marker genes among the 25% most variable genes are used for deconvolution. Marker genes determined by median differences across reference samples. (A) Boxplots of error for each algorithm. y-axis is the absolute value of the error of the estimates from the true mixing proportions. Black line is the median absolute error, grey line is the mean absolute error. (B) Boxplots of correlation. For each cell type the correlation of the true mixing proportions against the estimated mixing proportions is calculated. If the s.d. of the estimates is zero, we say the correlation is zero. If the s.d. of the true proportions is zero we do not calculate the correlation. (C) Simlar to (B) but using R^2 instead of correlation. (D) Scatter plots of estimated mixing proportions again true mixing proportions for dtangle, CIBERSORT and EPIC. Orange line is a 45° line through zero.



label=paperplots:datasets:Bechtall

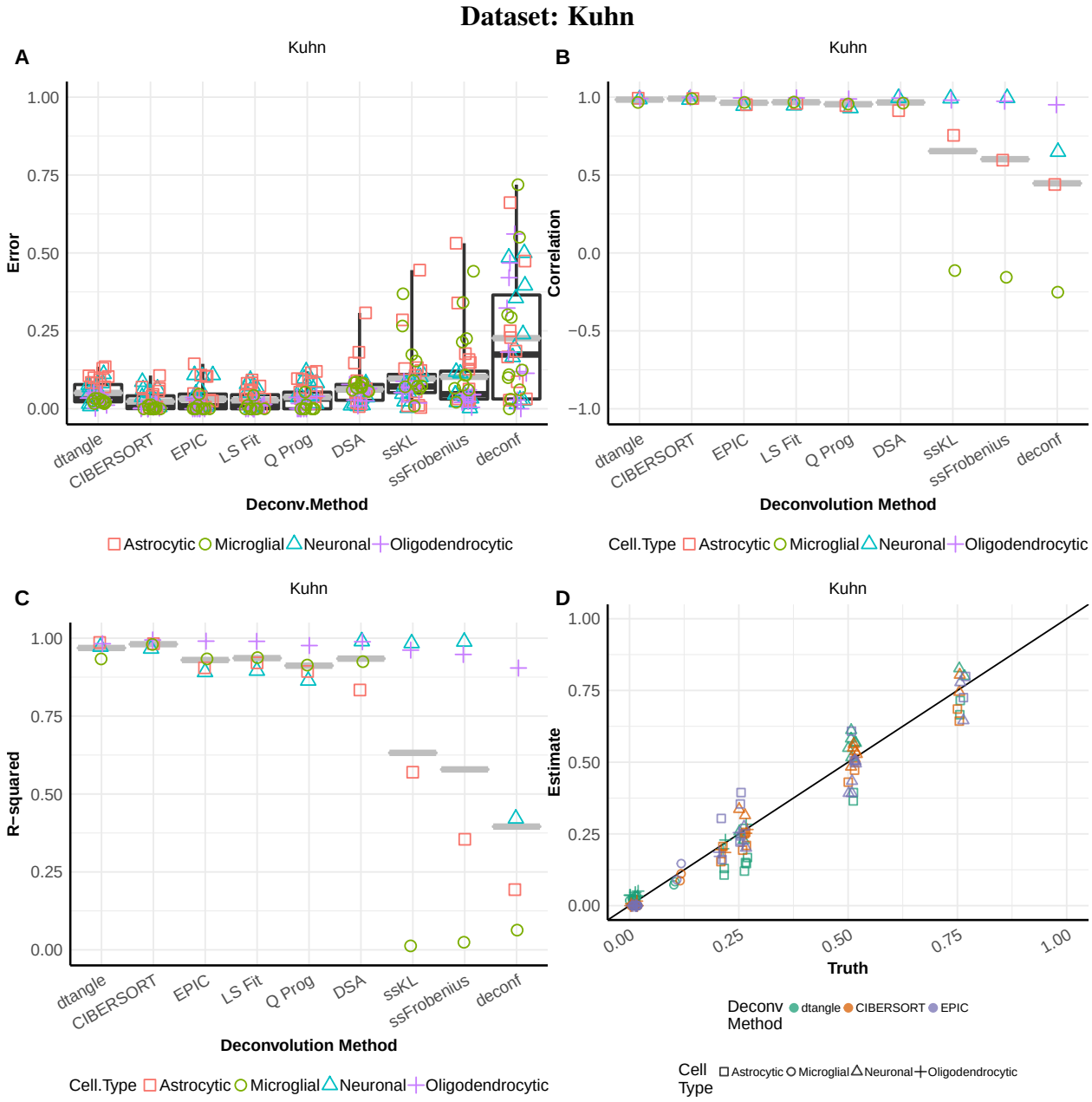
Figure 5.14: Similar to Figure 5.13 but for the Becht data-set.

Dataset: Gong



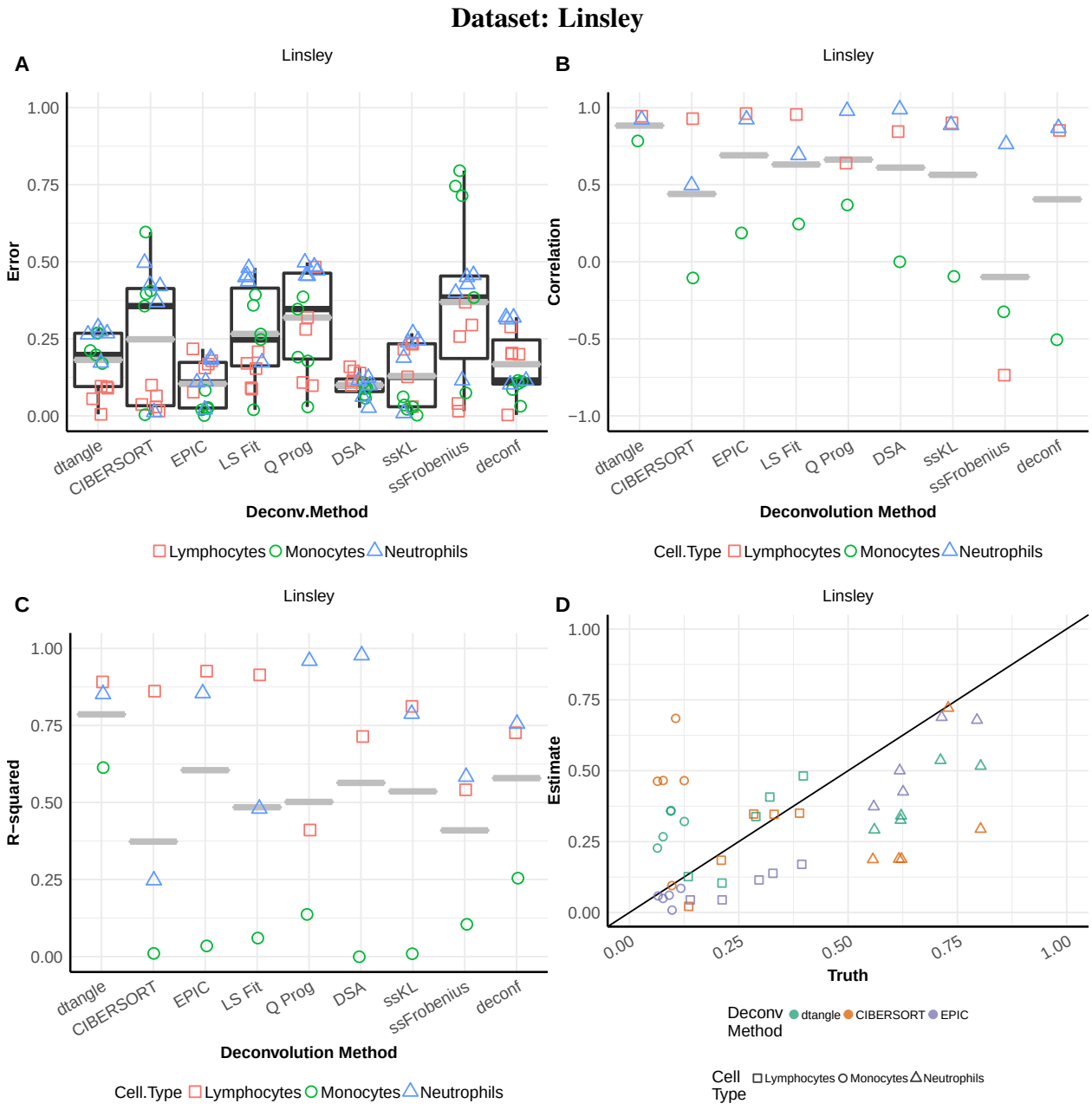
label=paperplots:datasets:Gongall

Figure 5.15: Similar to Figure 5.13 but for the Gong data-set.



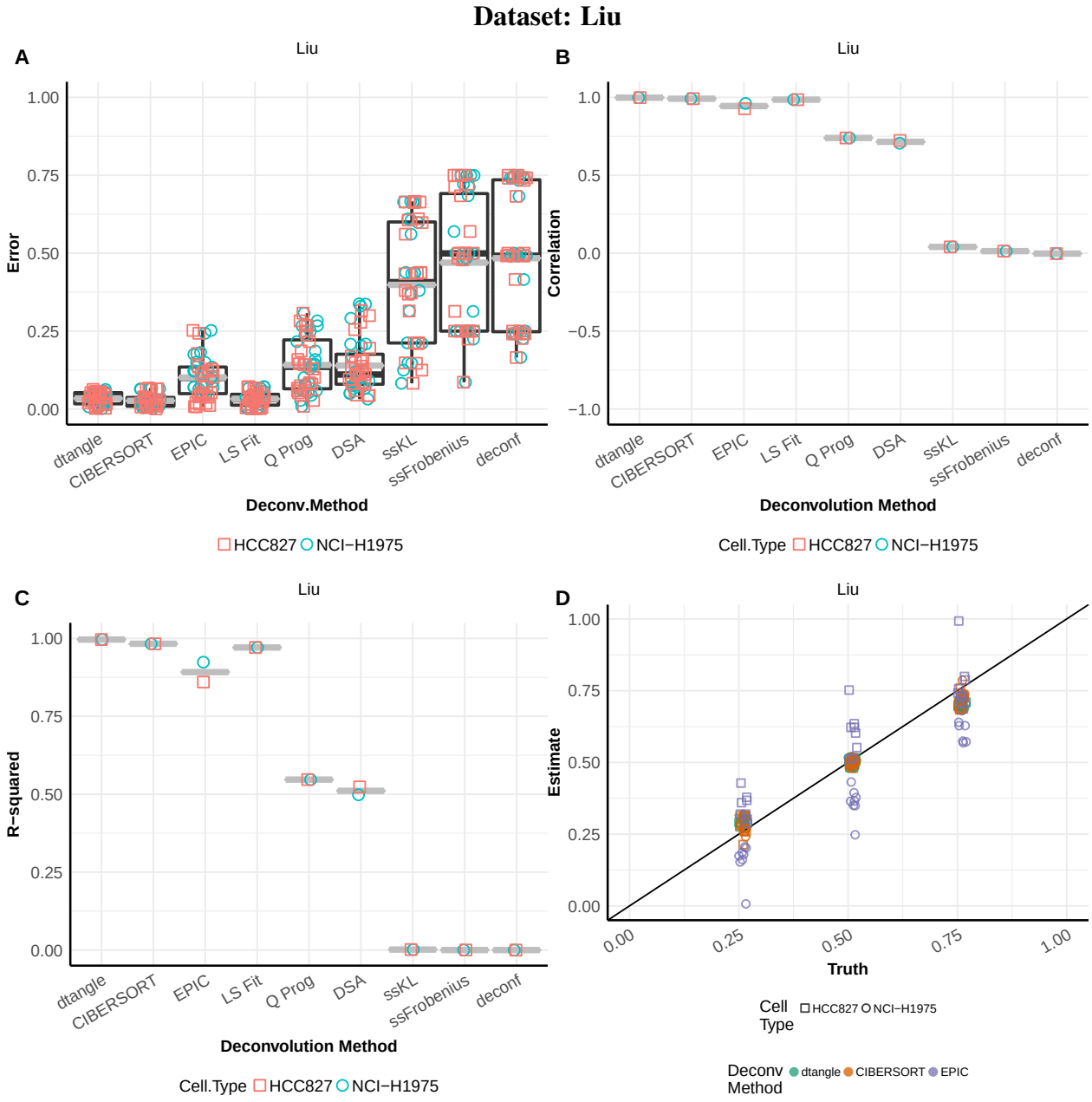
label=paperplots:datasets:Kuhnall

Figure 5.16: Similar to Figure 5.13 but for the Kuhn data-set.



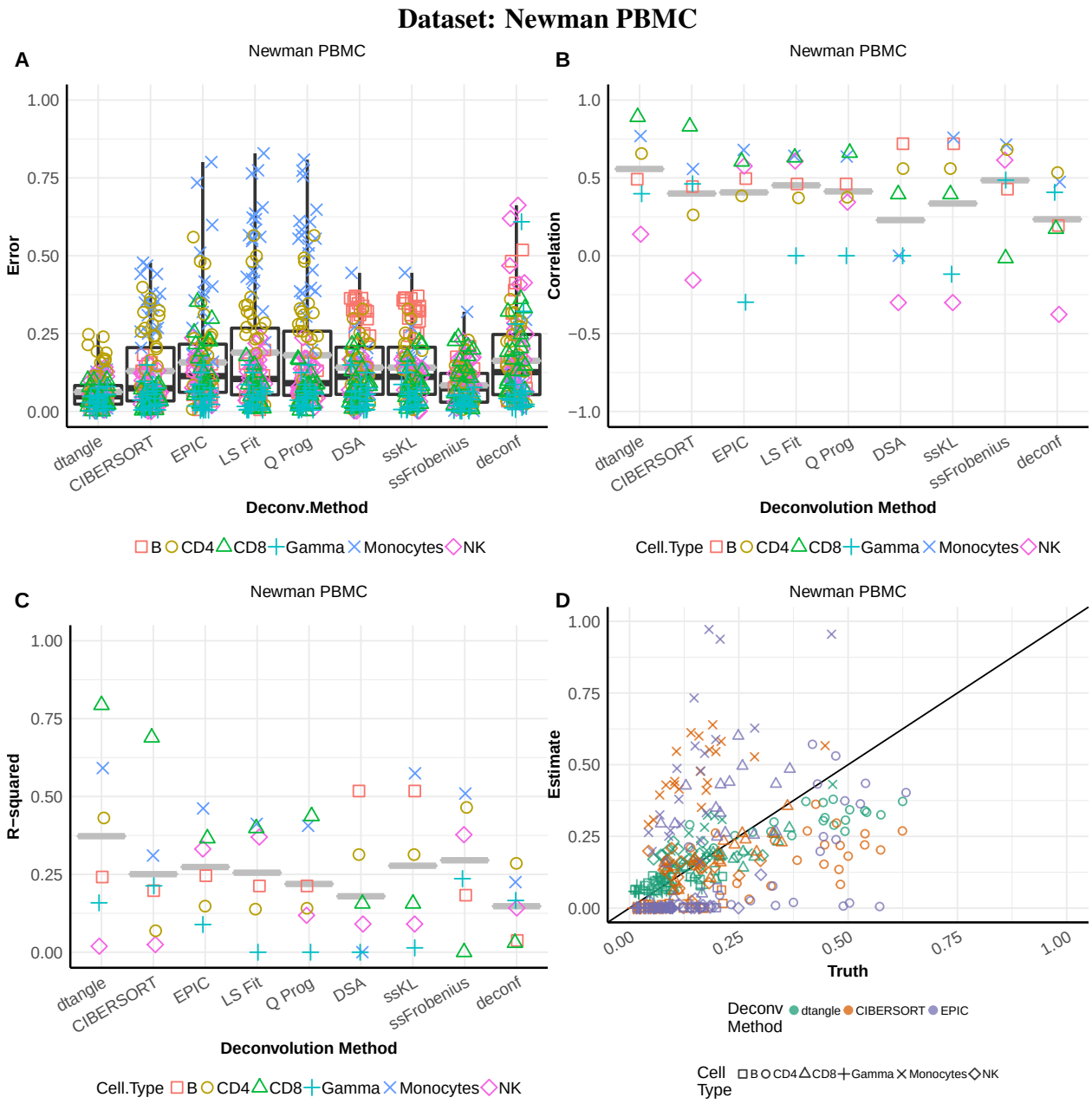
label=paperplots:datasets:Linsleyall

Figure 5.17: Similar to Figure 5.13 but for the Linsley data-set.



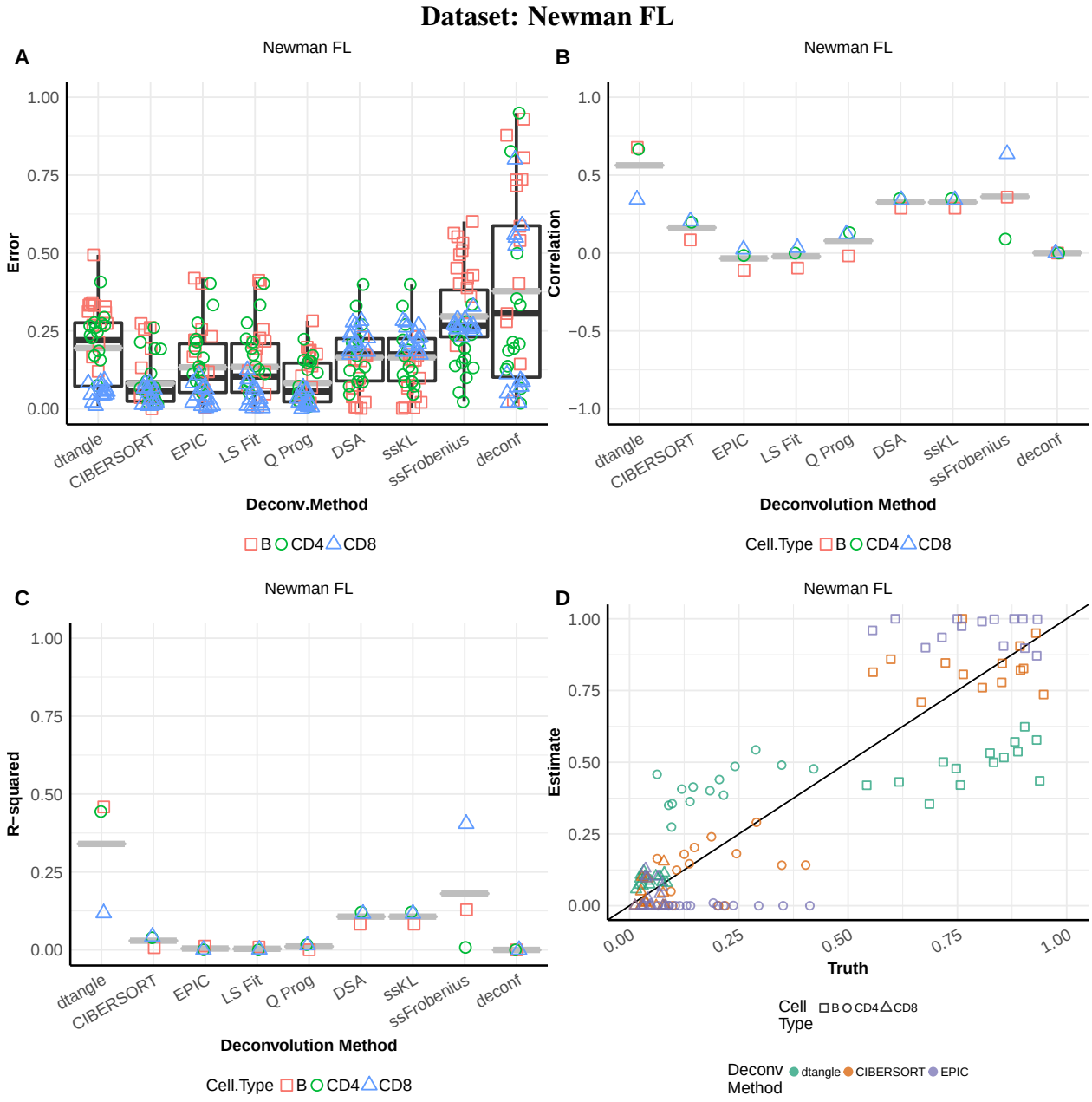
label=paperplots:datasets:Liuall

Figure 5.18: Similar to Figure 5.13 but for the Liu data-set.



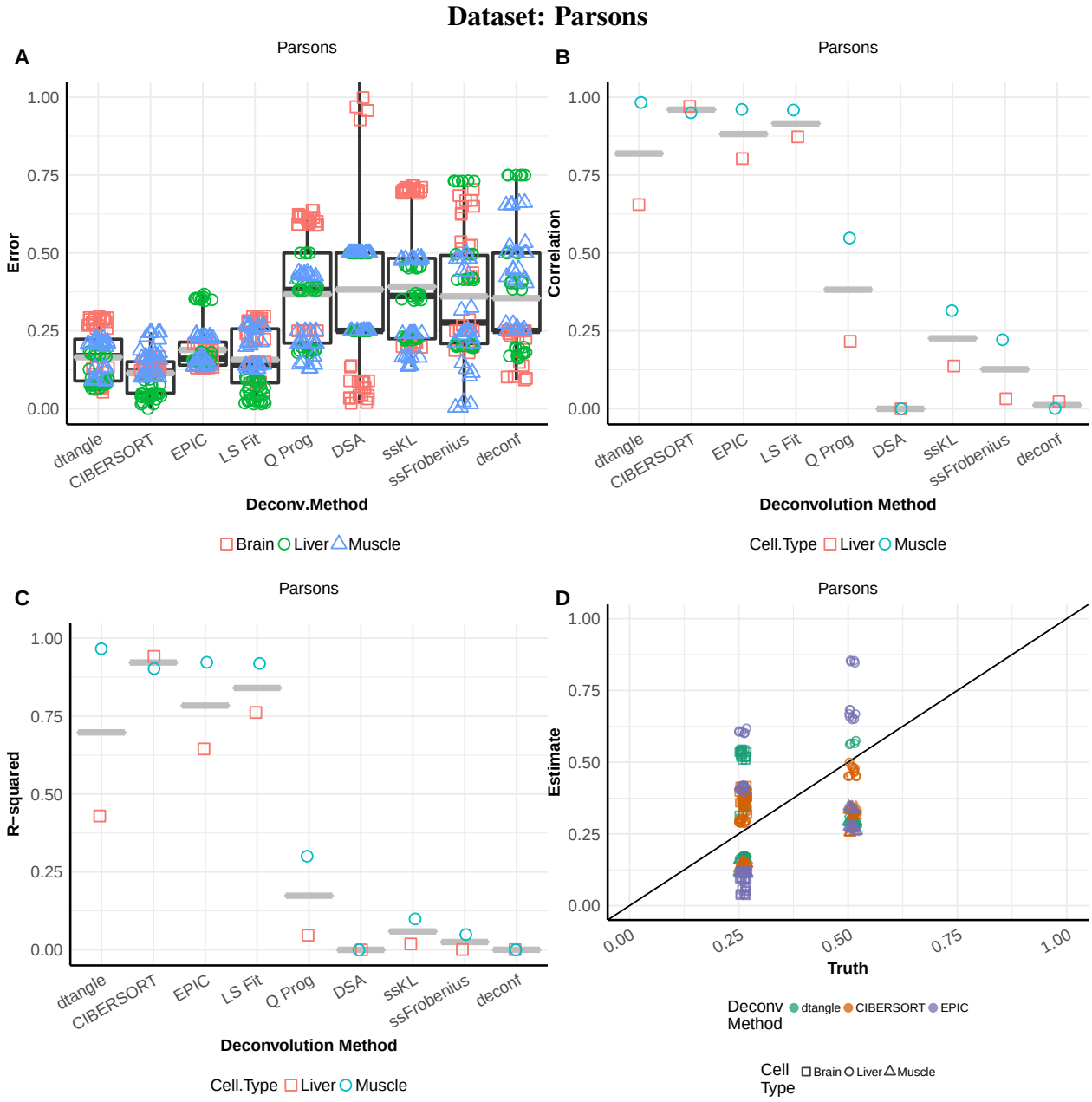
label=paperplots:datasets:NewmanPBMCall

Figure 5.19: Similar to Figure 5.13 but for the Newman PBMC data-set.



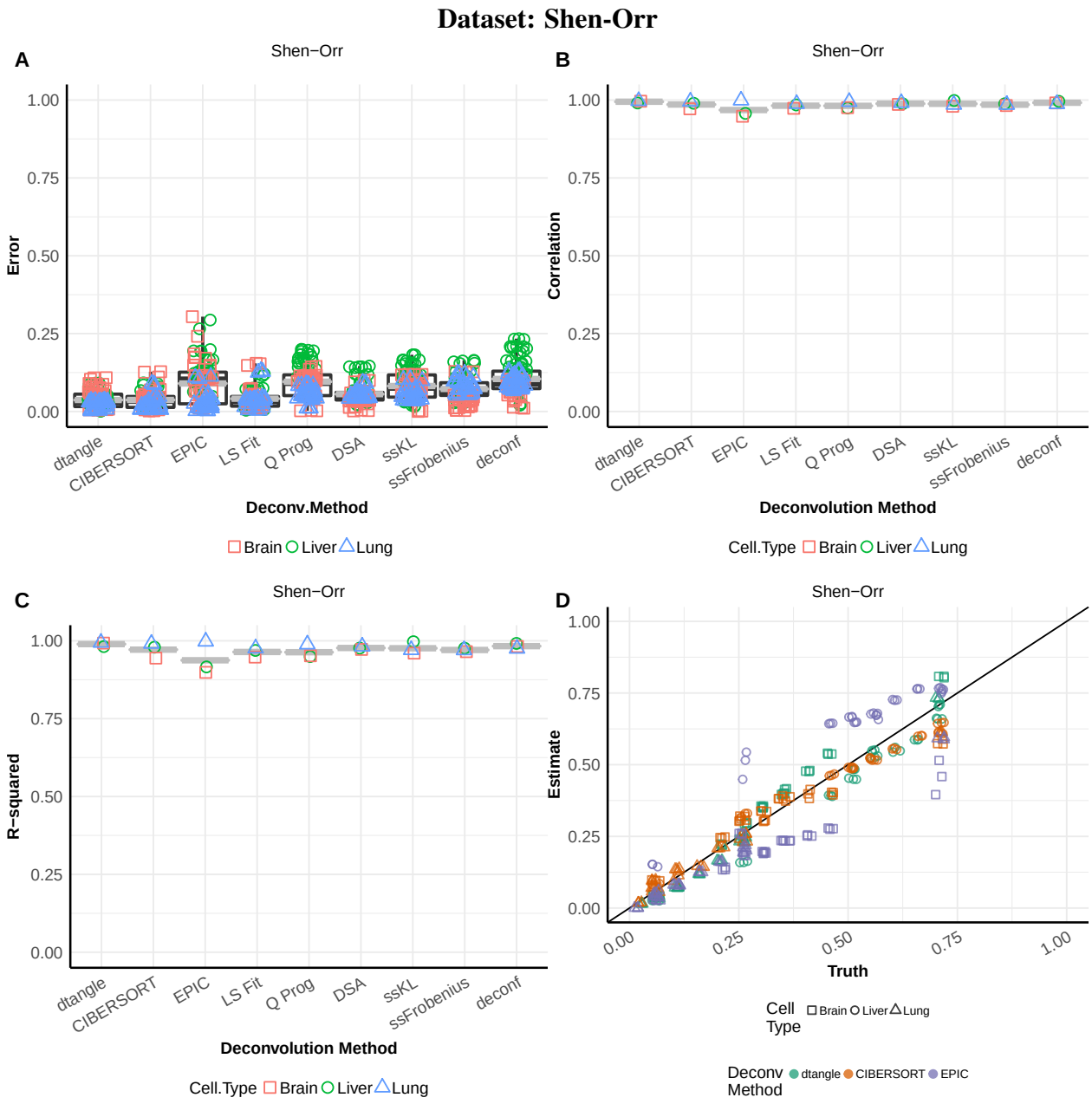
label=paperplots:datasets:NewmanFLall

Figure 5.20: Similar to Figure 5.13 but for the Newman FL data-set.



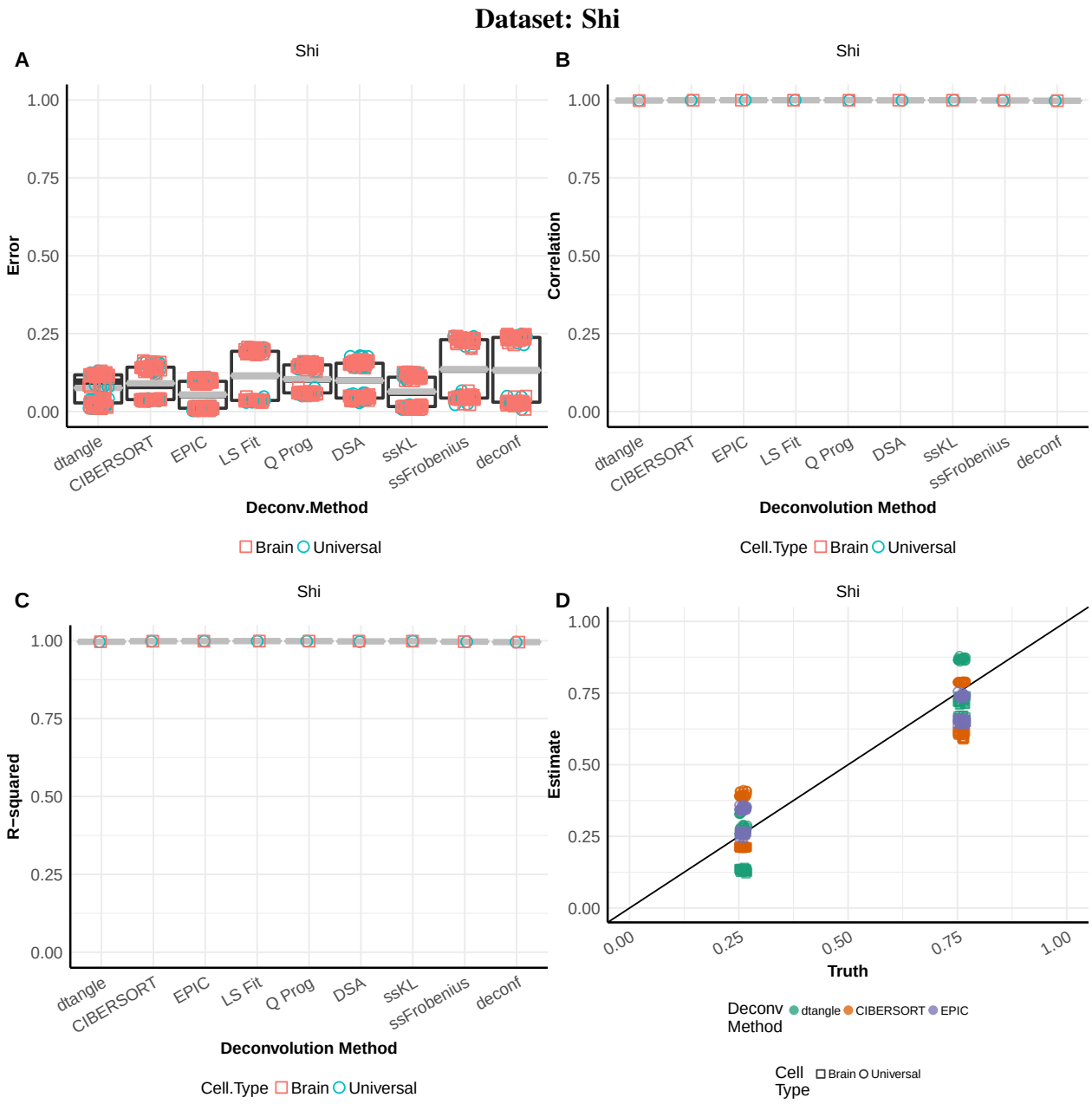
label=paperplots:datasets:Parsonsall

Figure 5.21: Similar to Figure 5.13 but for the Parsons data-set.



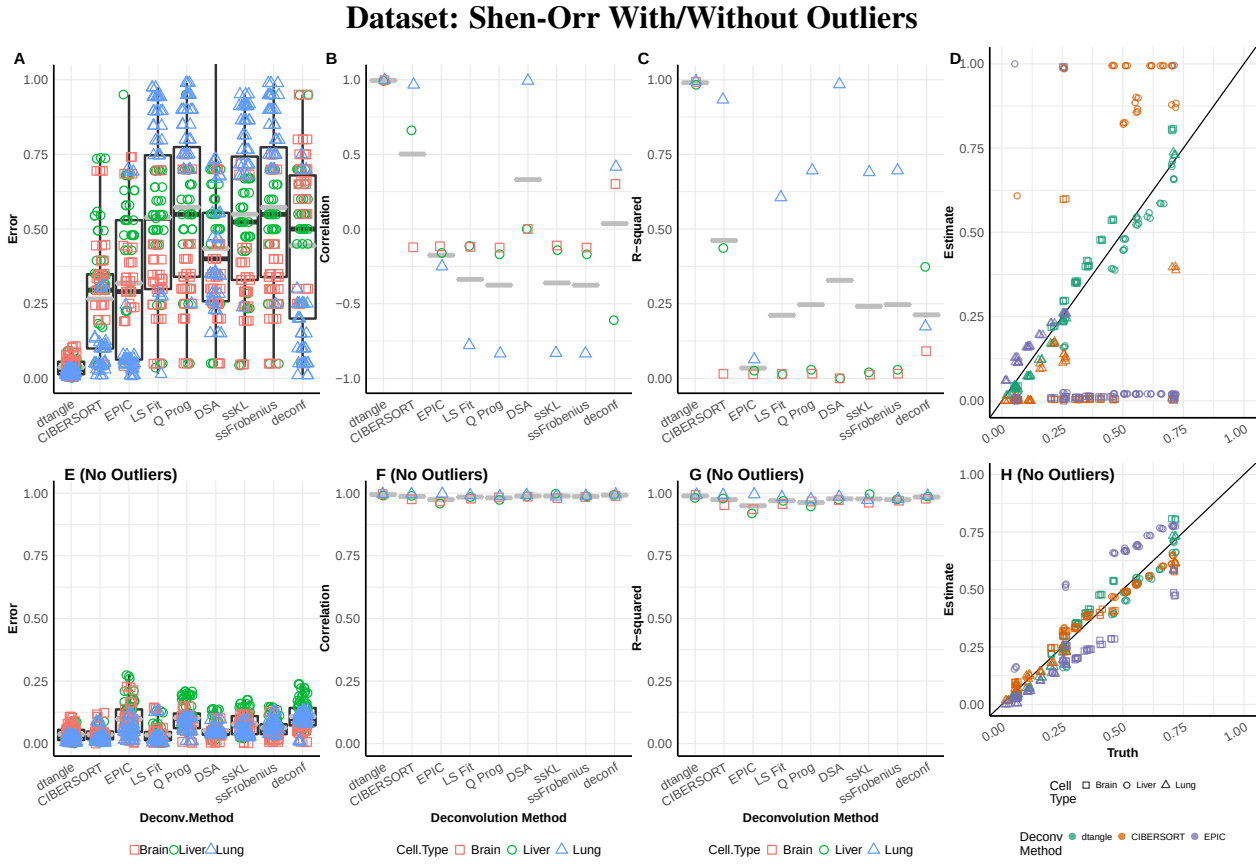
label=paperplots:datasets:ShenOrrall

Figure 5.22: Similar to Figure 5.13 but for the Shen-Orr data-set.



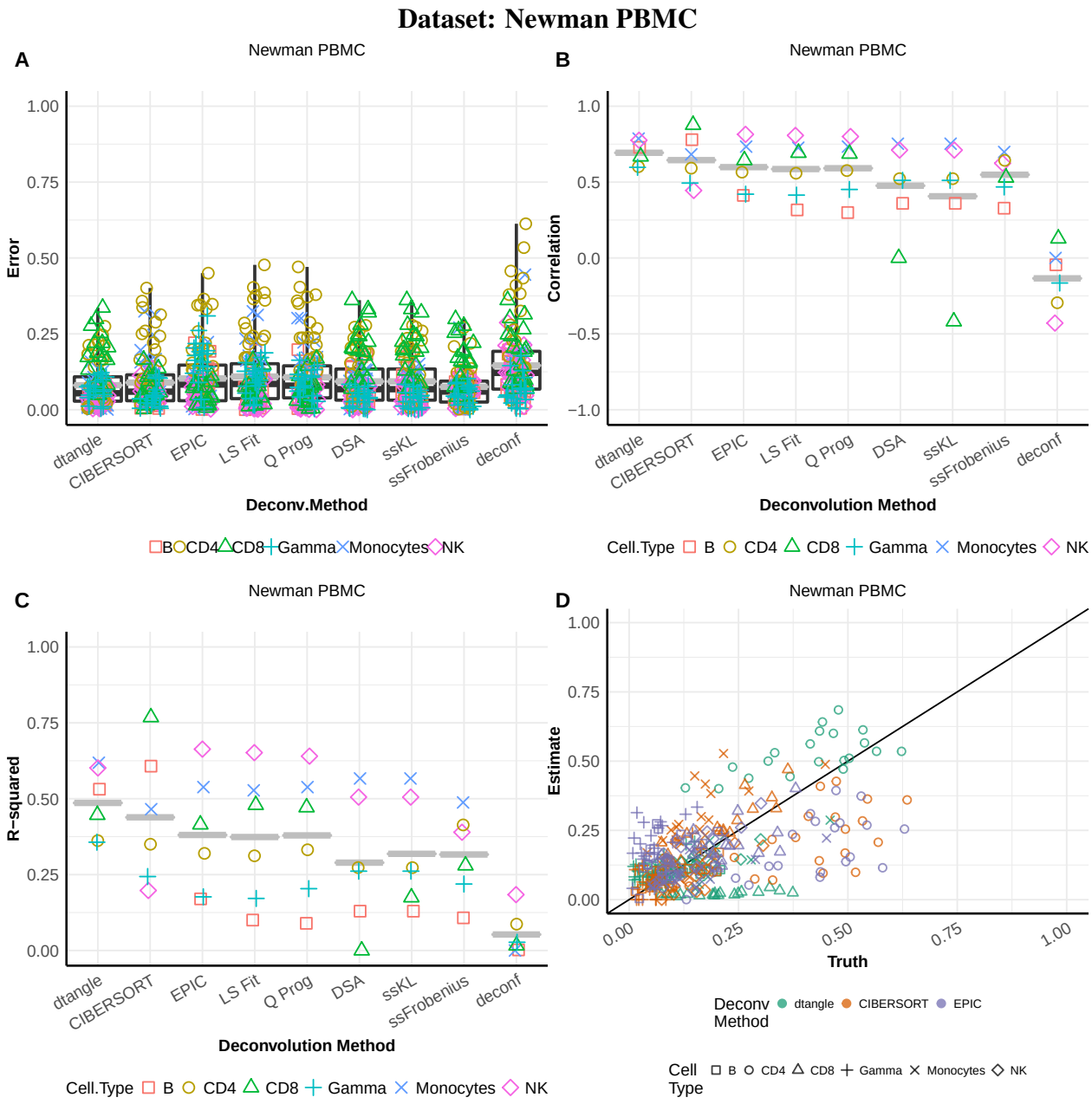
label=paperplots:datasets:Shiall

Figure 5.23: Similar to Figure 5.13 but for the Shi data-set.



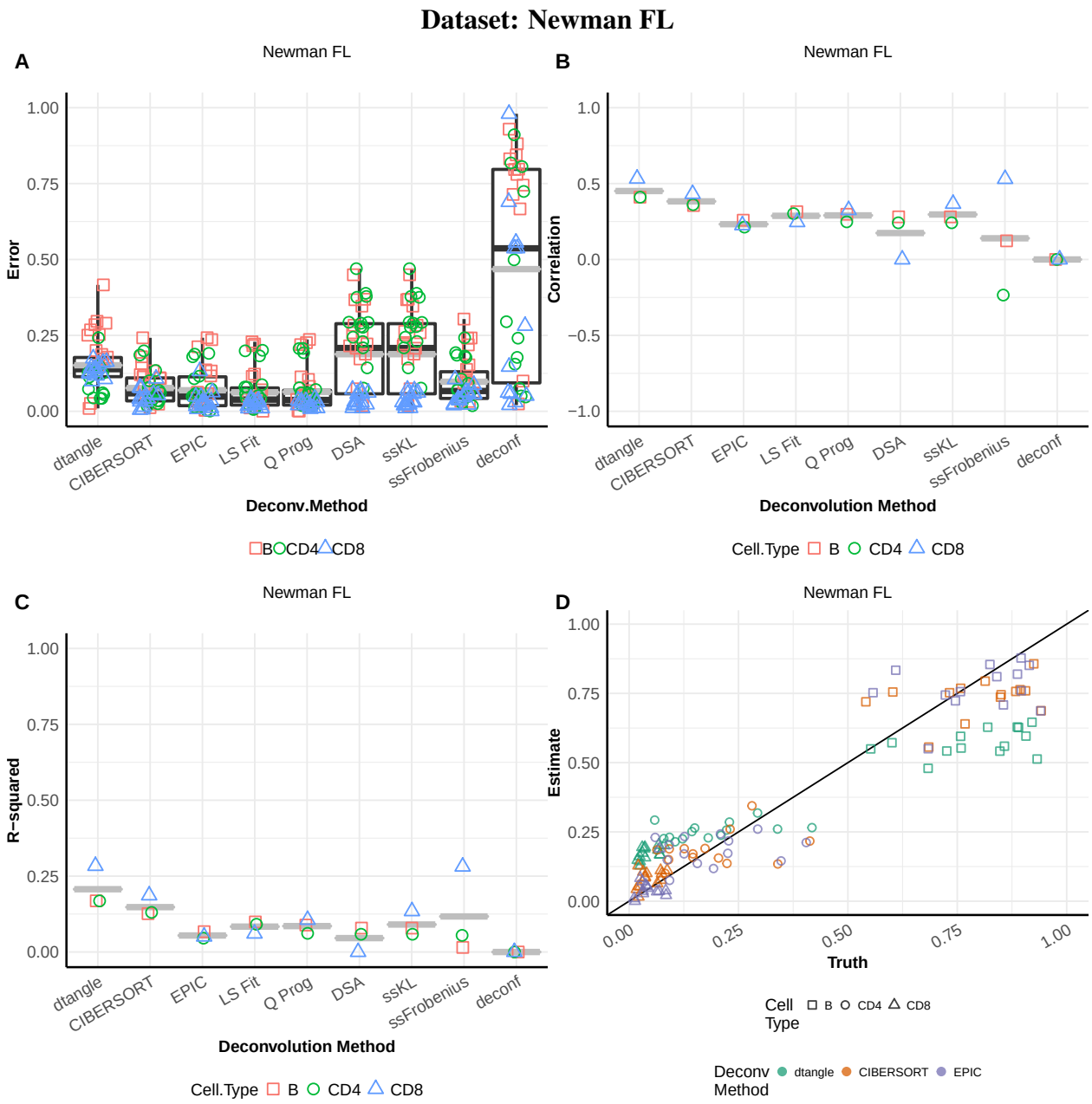
label=shenorrplots:all

Figure 5.24: (A-D) same as Figure 5.22. (E-H) same as (A-D) but with outliers removed.



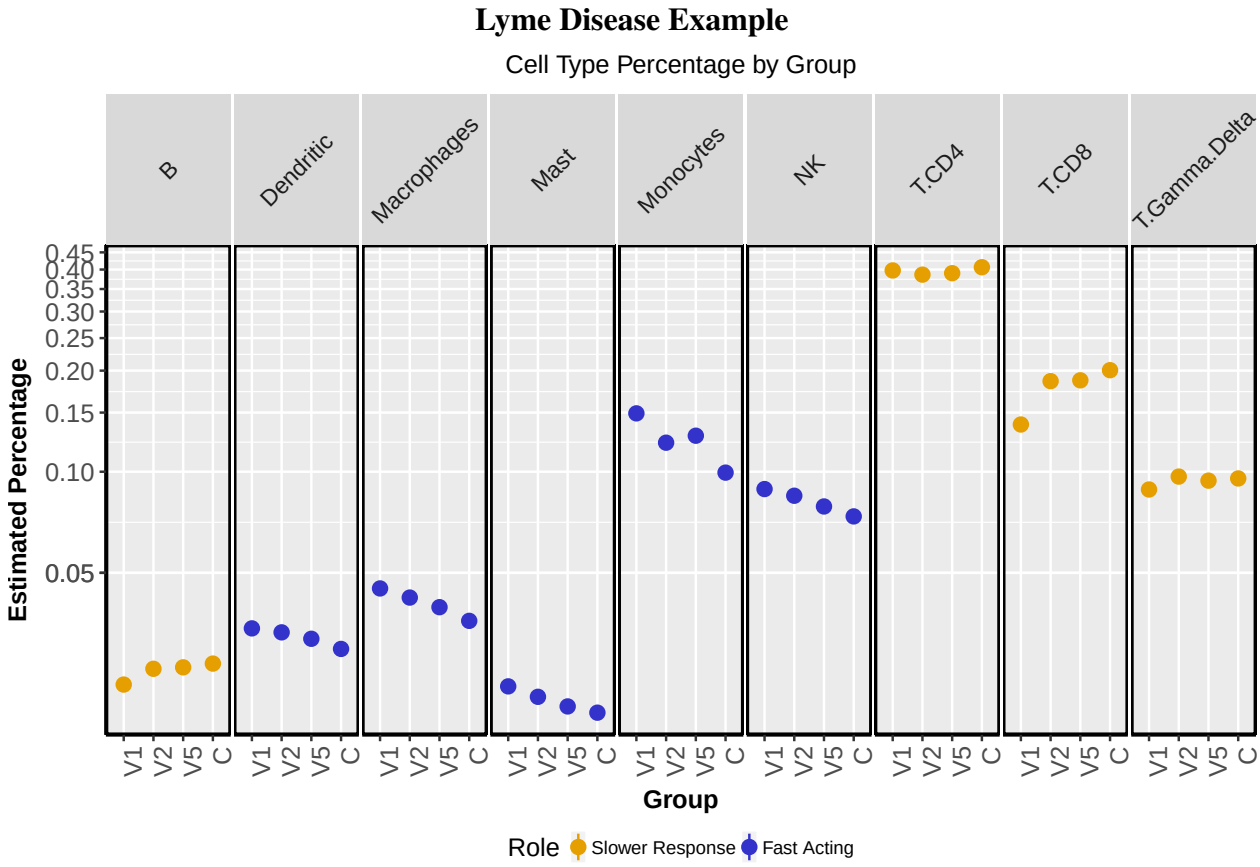
label=newmanmarkersplots:datasets:NewmanPBMCall

Figure 5.25: Same as Figure 5.19 but using references, mixtures samples, and marker genes directly from Newman paper supplement.



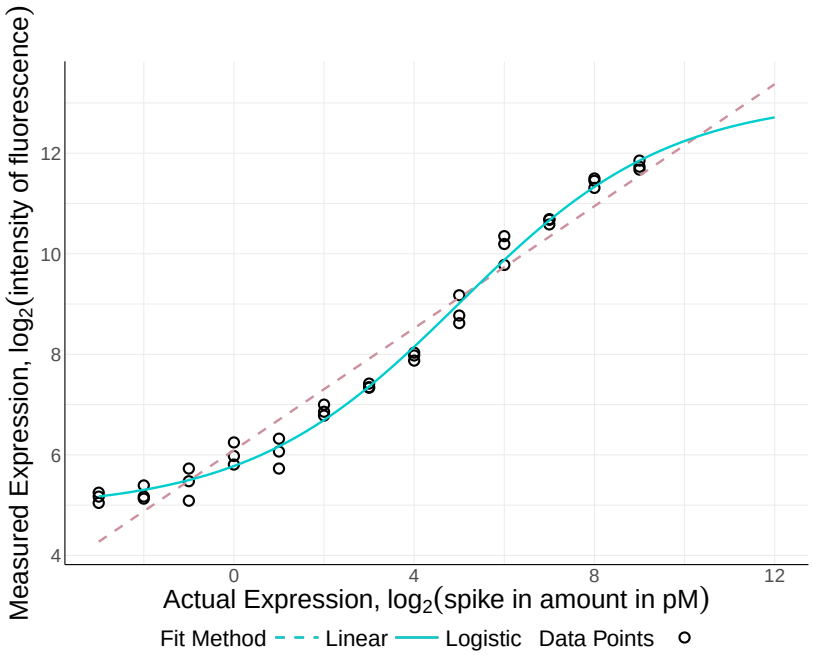
label=newmanmarkersplots:datasets:NewmanFLall

Figure 5.26: Same as Figure 5.20 but using references and marker genes directly from Newman paper supplement.

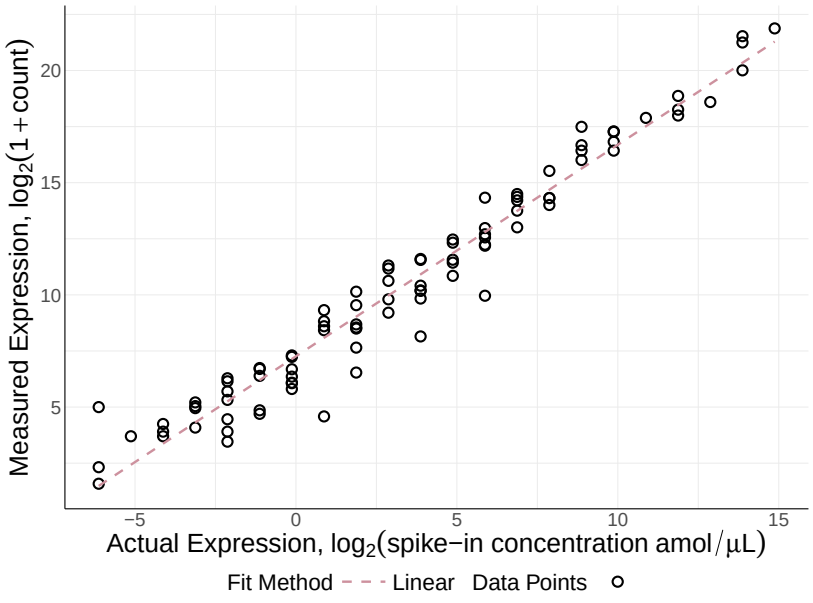


label=lyme:ctypes

Figure 5.27: Estimated cell type proportions over time.

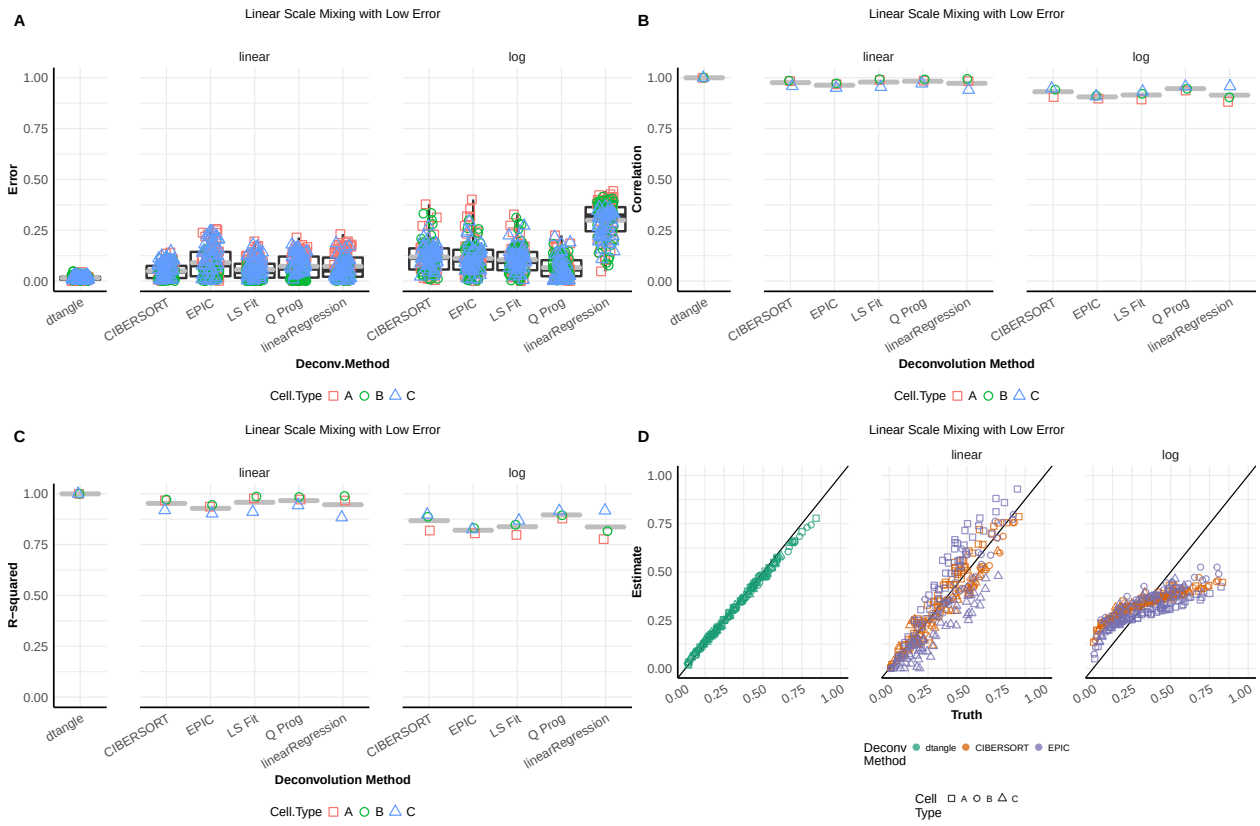


(a) $\log_2$ measured expression versus $\log_2$ concentration of a probe for gene TNFRSF1B. The relationship is approximated well by a linear model. While we have plotted amount against measured expression for one particular gene the results are generalizable to all genes. Points are plotted for the 13 experiments where the gene is spiked-in at a amount above zero and for each of the three technical replicates of each experiment. Along with the data points are plotted a linear and logistic least squares fit. The linear fit is a simple linear regression of measured expression on amount and the logistic fit is the least squares fit of a generalized logistic function of the form $\beta_0 + \beta_1 / \left(1 + \exp\left(\beta_2 x + \beta_3\right)\right)$.



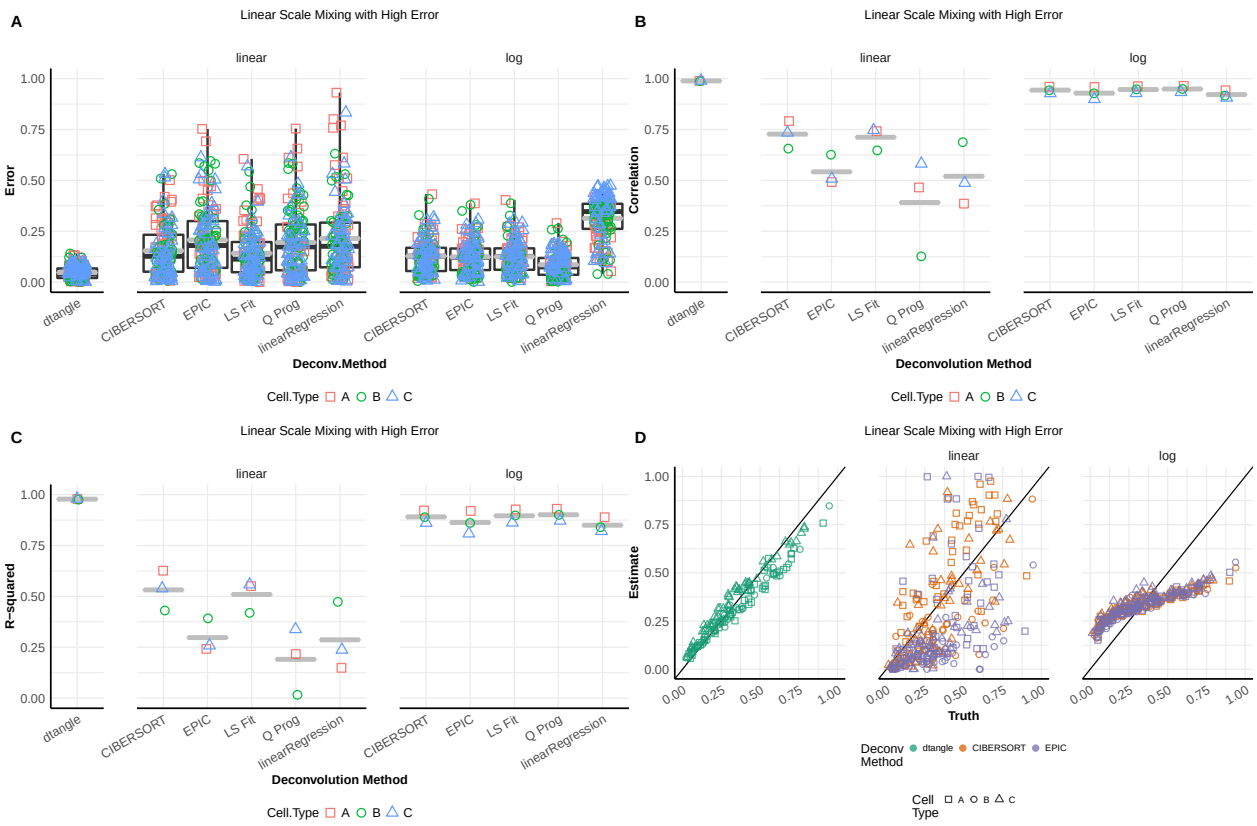
(b) $\log_2$ measured expression versus $\log_2$ concentration of ERCC spike-in controls in RNA-seq data. This relationship is highly linear. A linear least squares regression fit is plotted as a line.

Figure 5.28: Plots of actual v. measured expression.



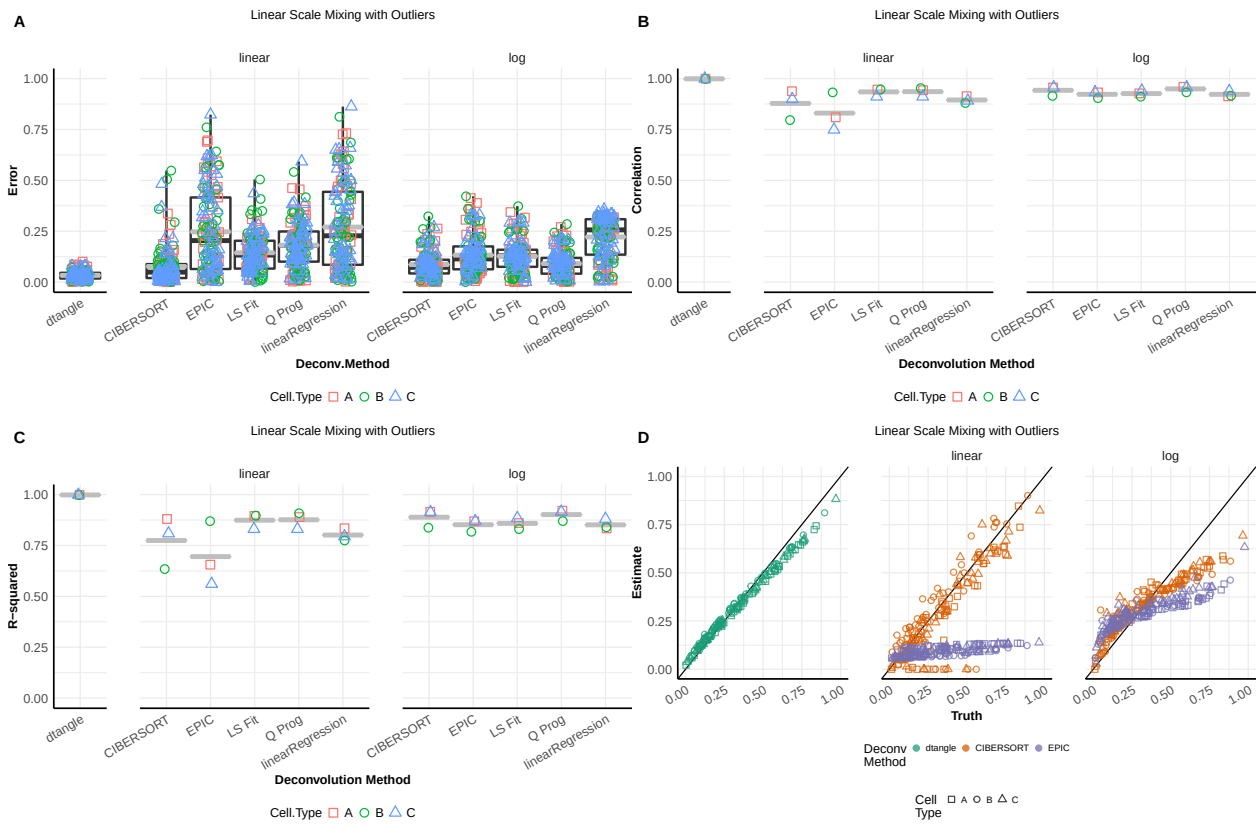
label=sim:gaussian:lowerror:lowerrorgaussianall

Figure 5.29: Partial deconvolution methods performance on simulated gaussian data with low error. Computation for methods other than dtangle was done for data both on the $\log_2$ scale and the linear un-transformed scale. Slope (γ) for dtangle is set to one. Top 10% of 25% most variable genes used for deconvolution. Marker genes determined by median differences across reference samples. (A) Boxplots of error for each algorithm. y-axis is the absolute value of the error of the estimates from the true mixing proportions. Black line is the median absolute error, grey line is the mean absolute error. (B) Boxplots of correlation. For each cell type the correlation of the true mixing proportions against the estimated mixing proportions is calculated. If the s.d. of the estimates is zero, we say the correlation is zero. If the s.d. of the true proportions is zero we do not calculate the correlation. (C) Scatter plots of estimated mixing proportions again true mixing proportions for dtangle, CIBERSORT and EPIC. Orange line is a 45° line through zero.



label=sim:gaussian:higherror:higherrorgaussianall

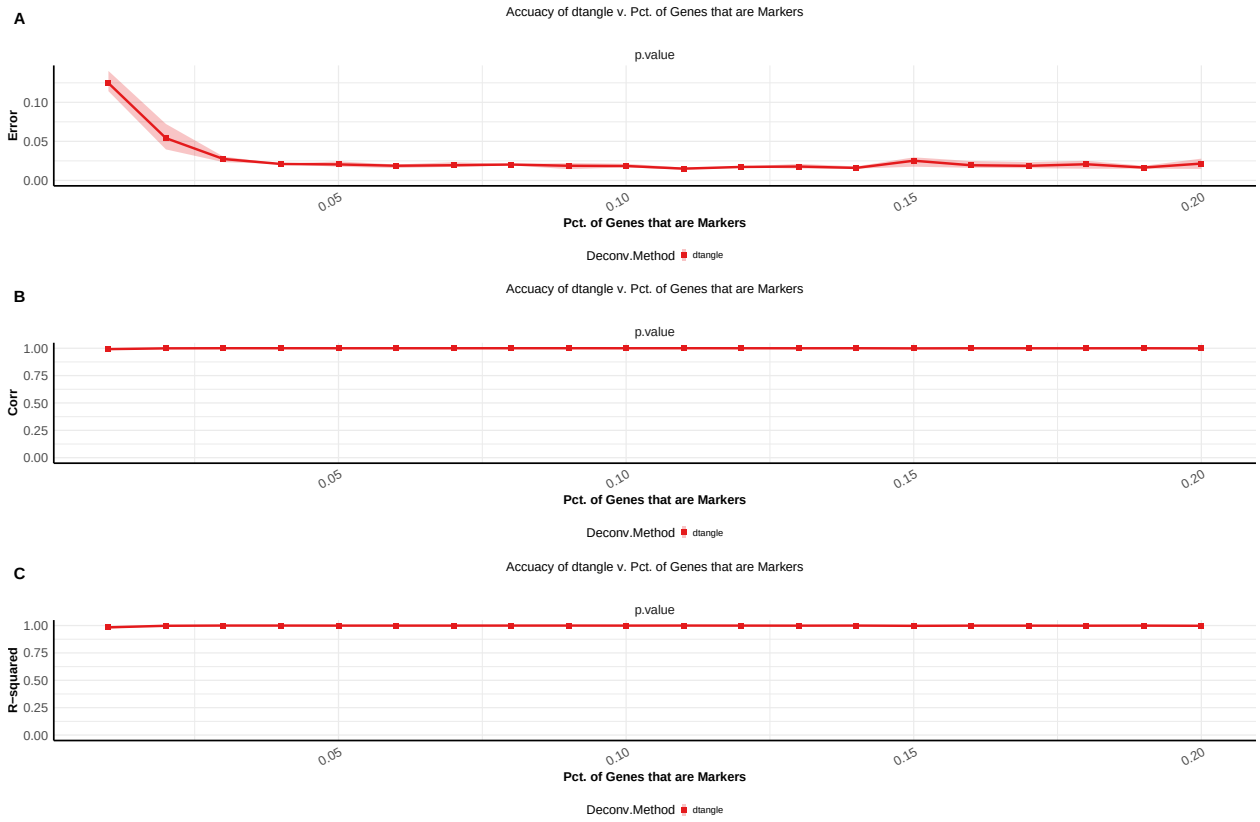
Figure 5.30: Similar to Figure 5.29 but with a high error variance used in simulation.



label=sim:gaussian:outliers:outliersgaussianall

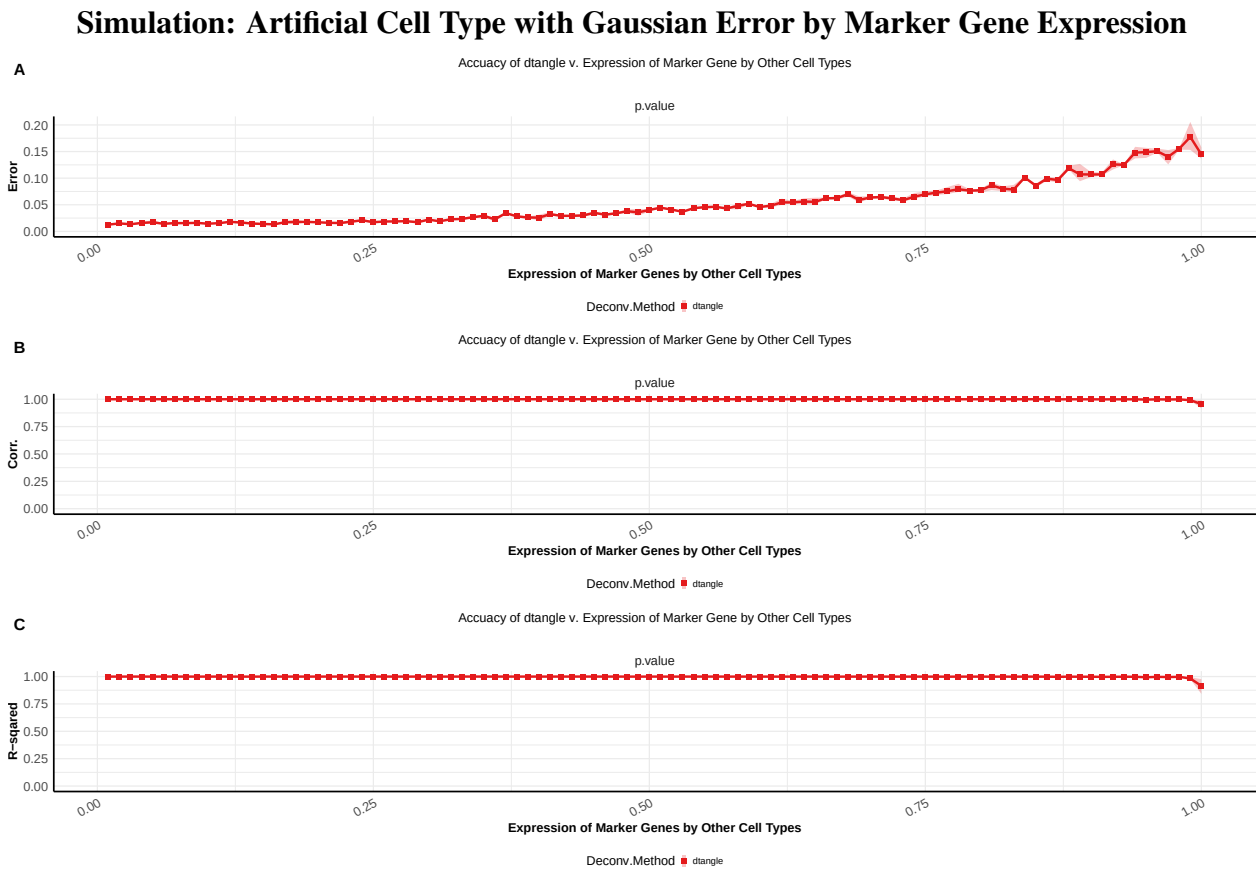
Figure 5.31: Similar to Figure 5.29 but with outliers added to the simulated data.

Simulation: Artificial Cell Type with Gaussian Error by Number of Marker Genes



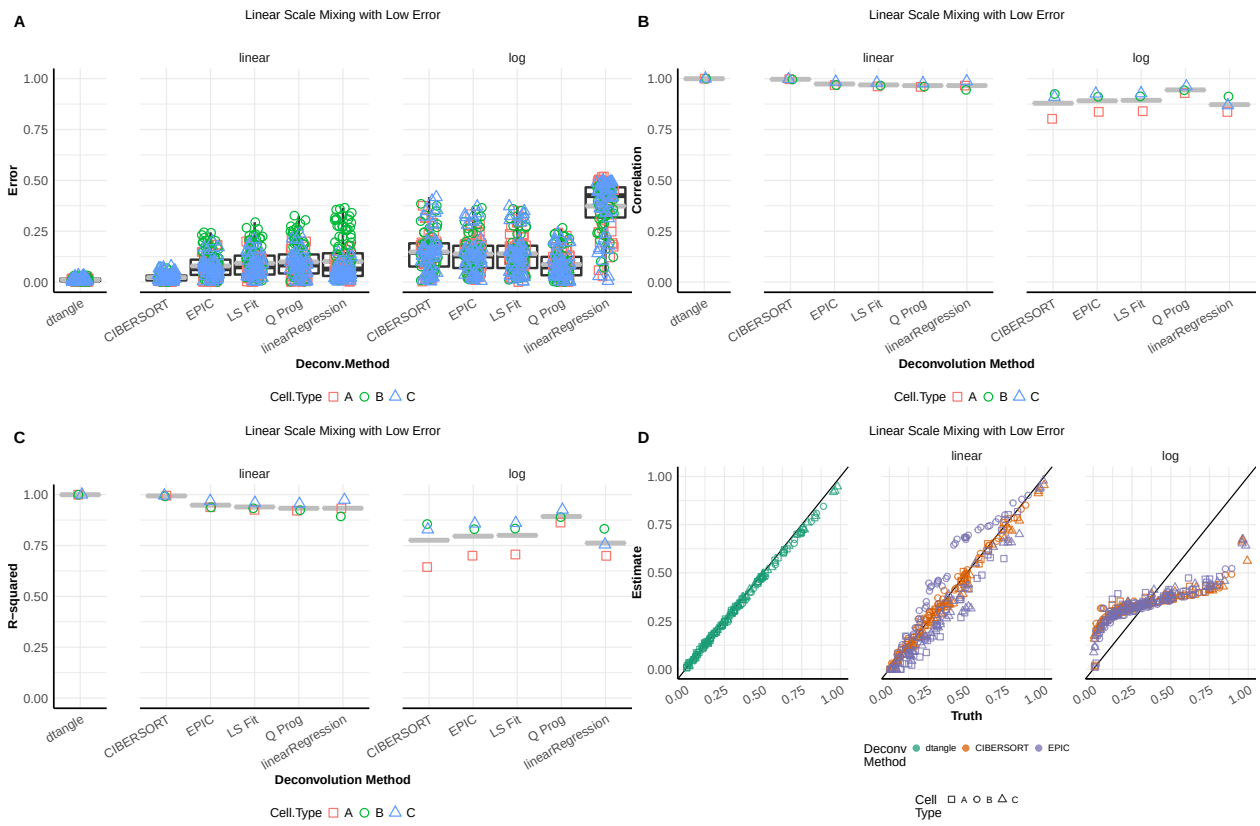
label=sim:gaussian:nmk

Figure 5.32: Accuracy of dtangle by the number of marker genes present in gaussian simulated data with low error variance. y-axis is accuracy measured by (A) grand mean of the absolute value of the error of the true proportions from the estiamted proportions and (B) mean correlation within each cell type. The x-axis is the percentage of the data set that is comprised of marker genes as defined by dtangle.



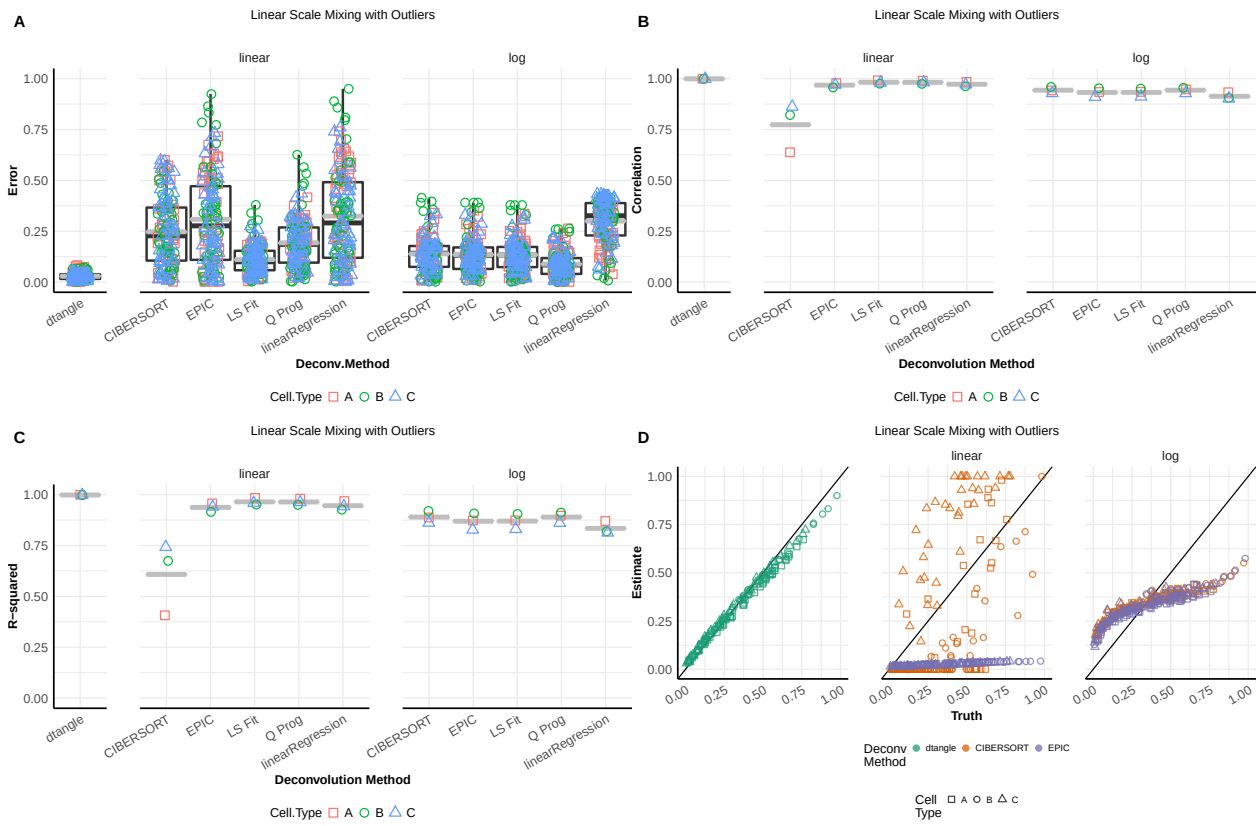
label=sim:gaussian:nmz

Figure 5.33: Accuracy of dtangle by expression level of marker genes in gaussian simulated data with low error variance. y-axis is accuracy measured by (A) grand mean of the absolute value of the error of the true proportions from the estiamted proportions and (B) mean correlation within each cell type. The x-axis is the quantile of the over-all data at which marker genes are expressed in all other cell types.



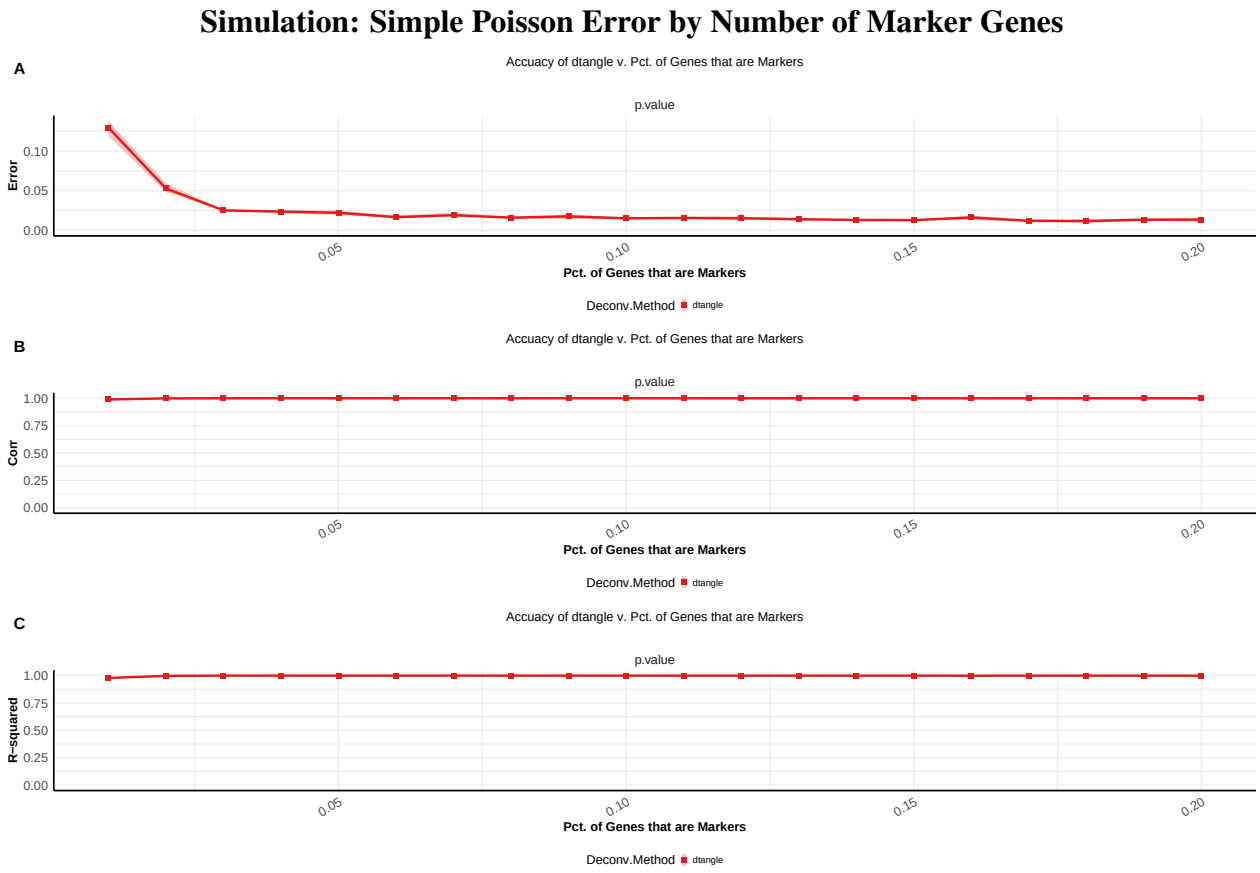
label=sim:poisson:lowerror:lowerrorpoissonall

Figure 5.34: Similar to Figure 5.29 but using a poisson error.



label=sim:poisson:outliers:outlierspoissonall

Figure 5.35: Similar to Figure 5.34 but with outliers added to the simulated data.

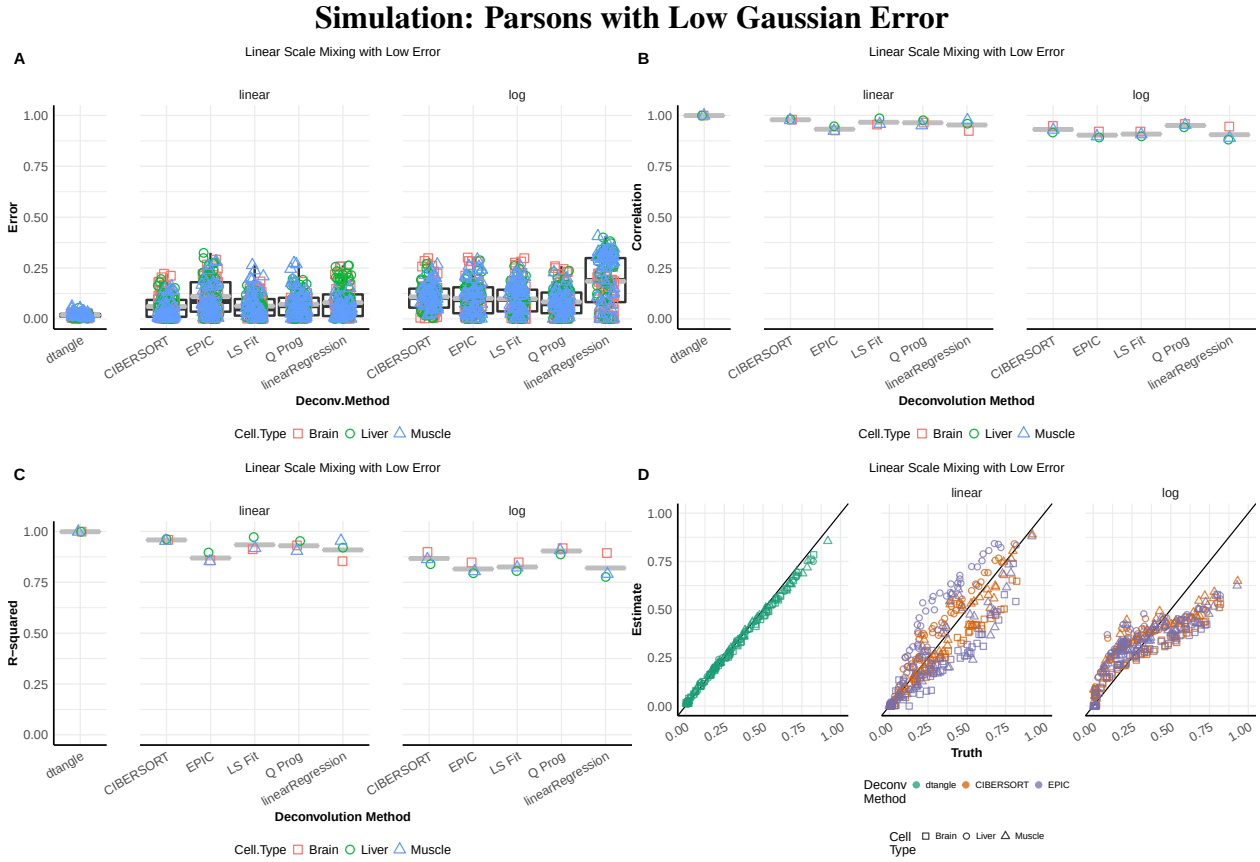


label=sim:poisson:nmk

Figure 5.36: Similar to Figure 5.32 but using a poisson error.

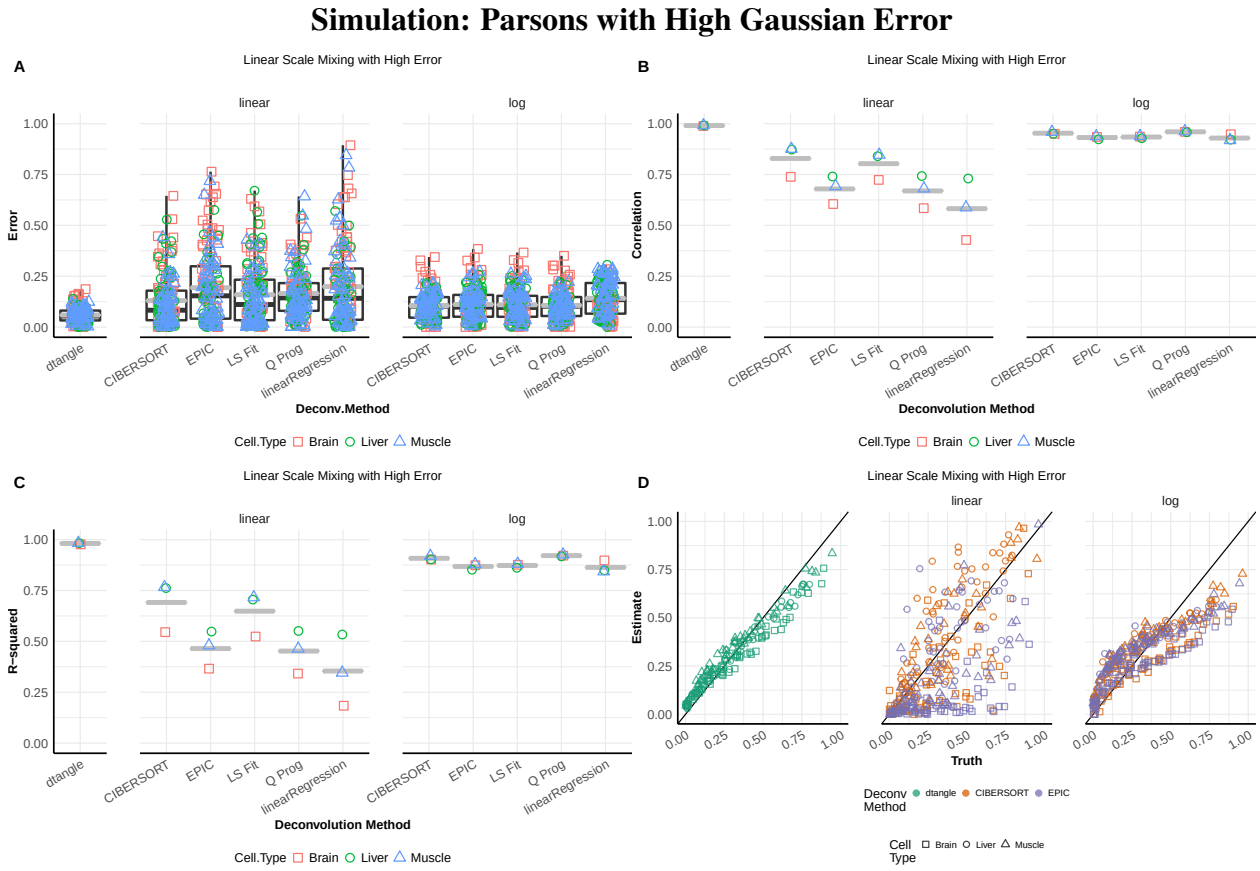
label=sim:poisson:nmz

Figure 5.37: Similar to Figure 5.33 but using a poisson error.



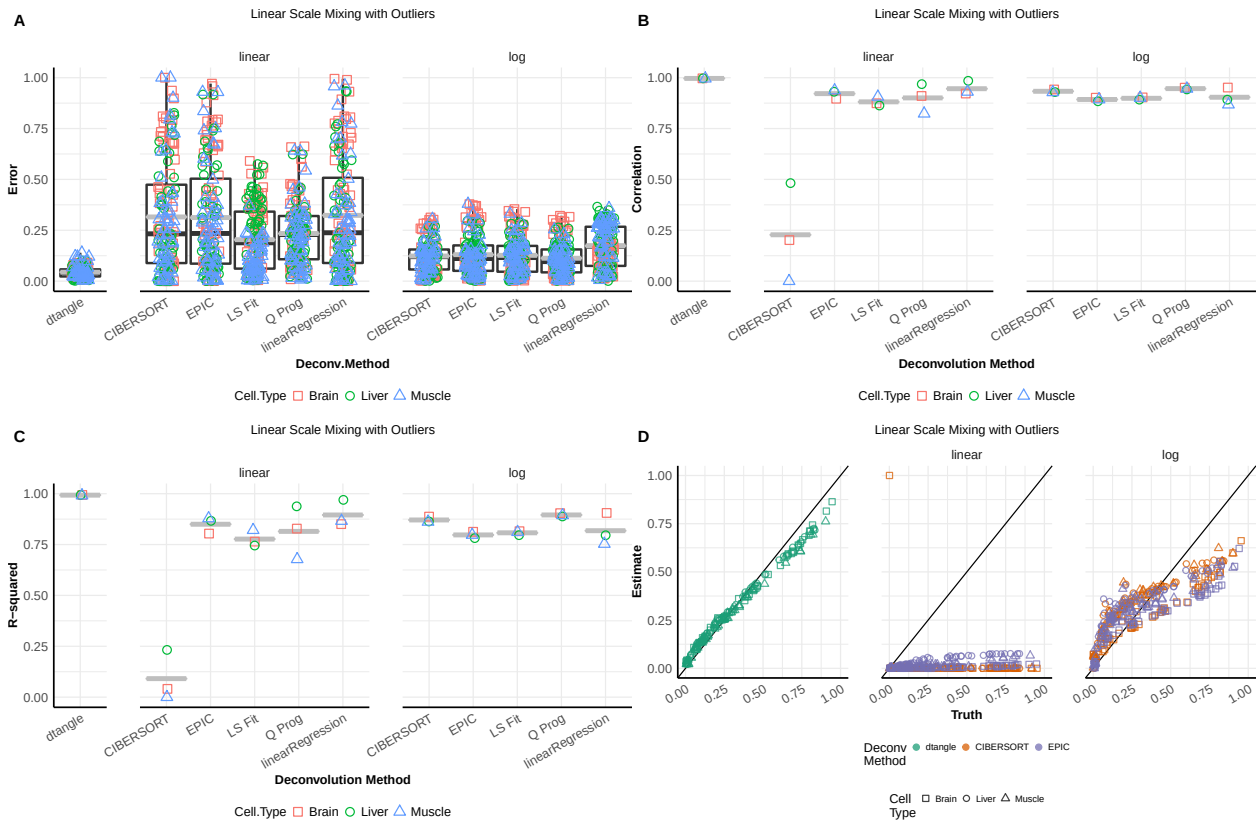
label=sim:mixgaussian:parsons:lowerror:lowerrormixgaussianall

Figure 5.38: Similar to Figure 5.29 but simulation was done by in-silico mixtures of reference cell type profiles from the Parsons data set.



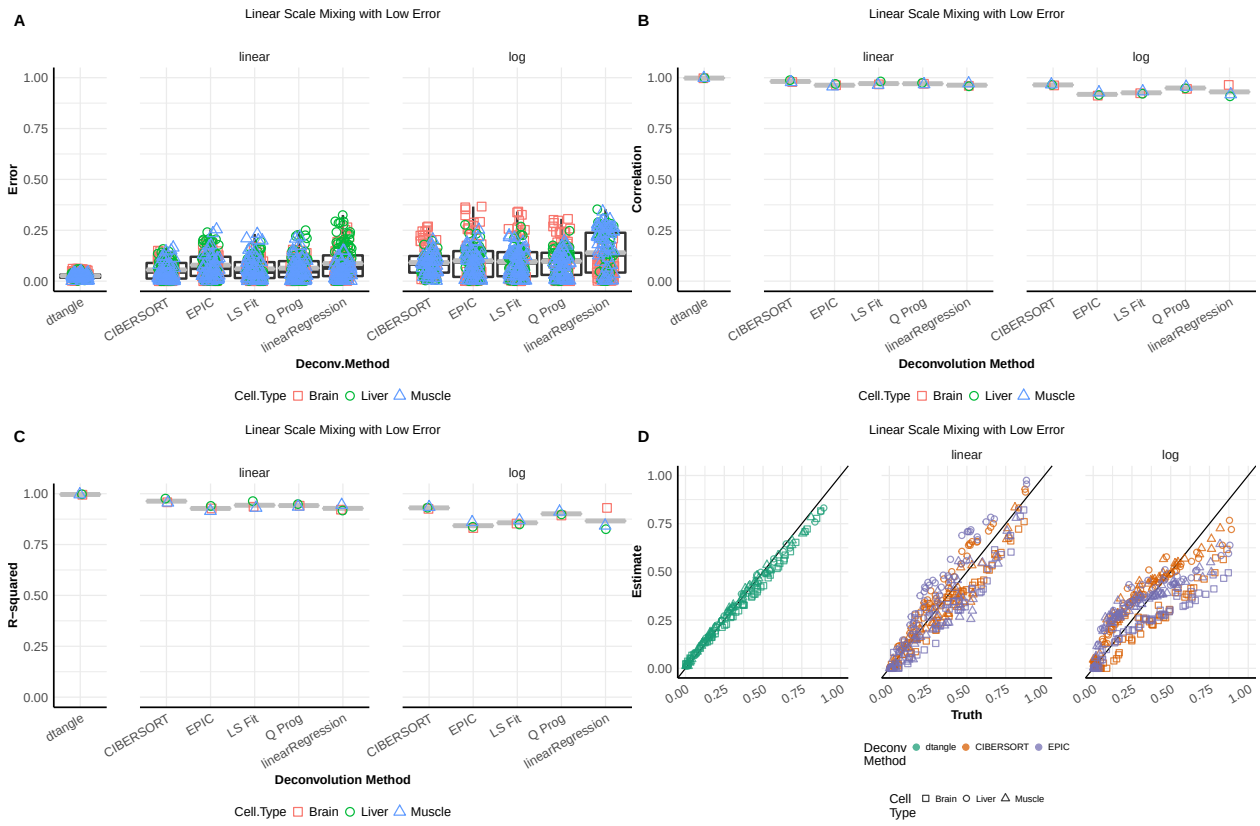
label=sim:mixgaussian:parsons:higherror:higherrormixgaussianall

Figure 5.39: Similar to Figure 5.38 but with a high error variance used in simulation.



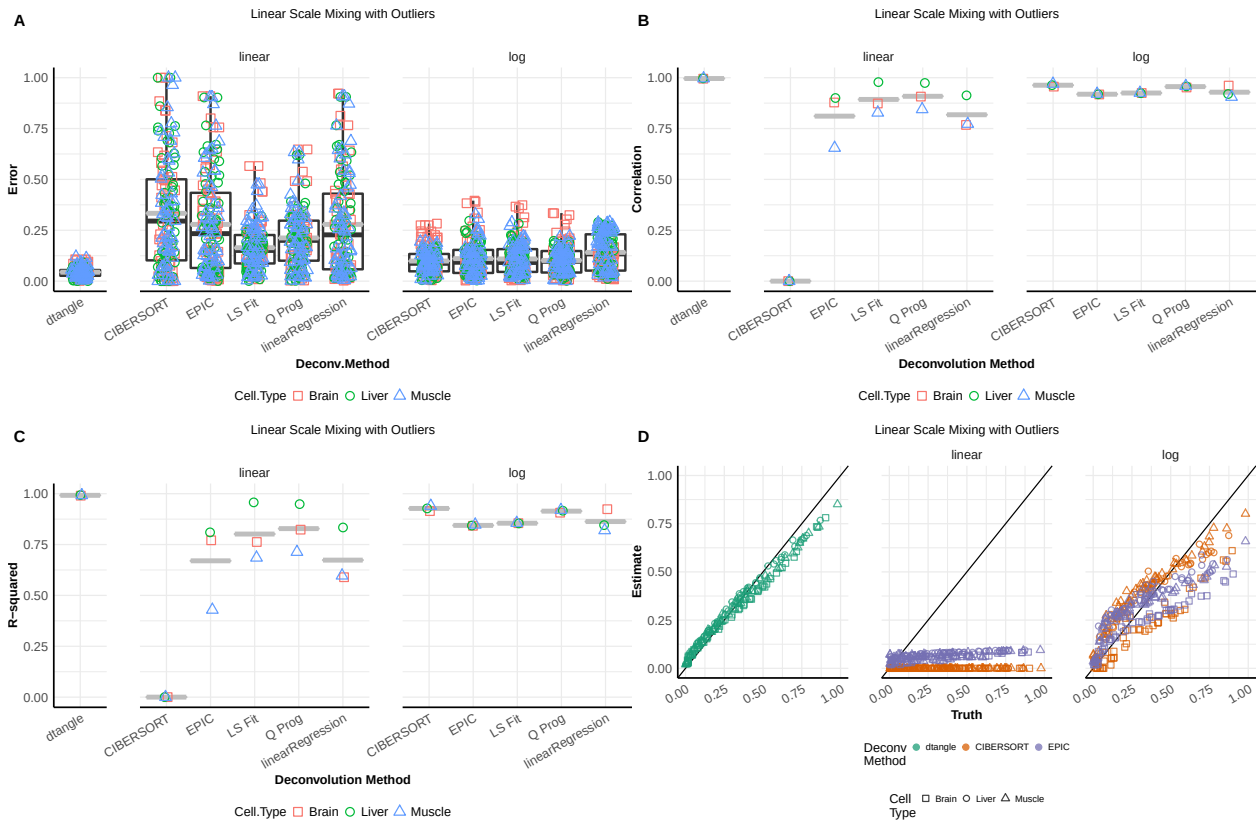
label=sim:mixgaussian:parsons:outliers:outliersmixgaussianall

Figure 5.40: Similar to Figure 5.38 but with outliers added to the simulated data.



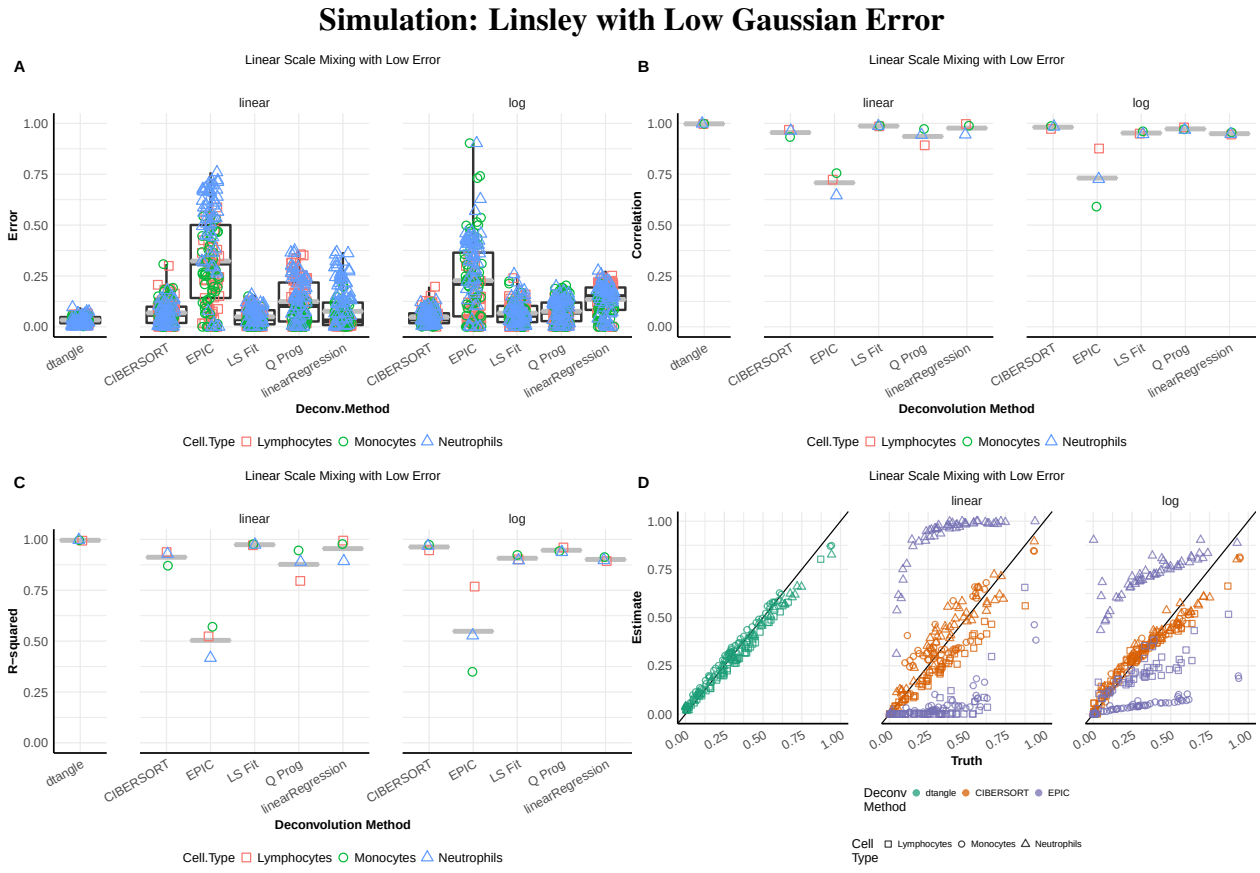
label=sim:mixpoisson:parsons:lowerror:lowerrormixpoissonall

Figure 5.41: Similar to Figure 5.38 but using poisson error.



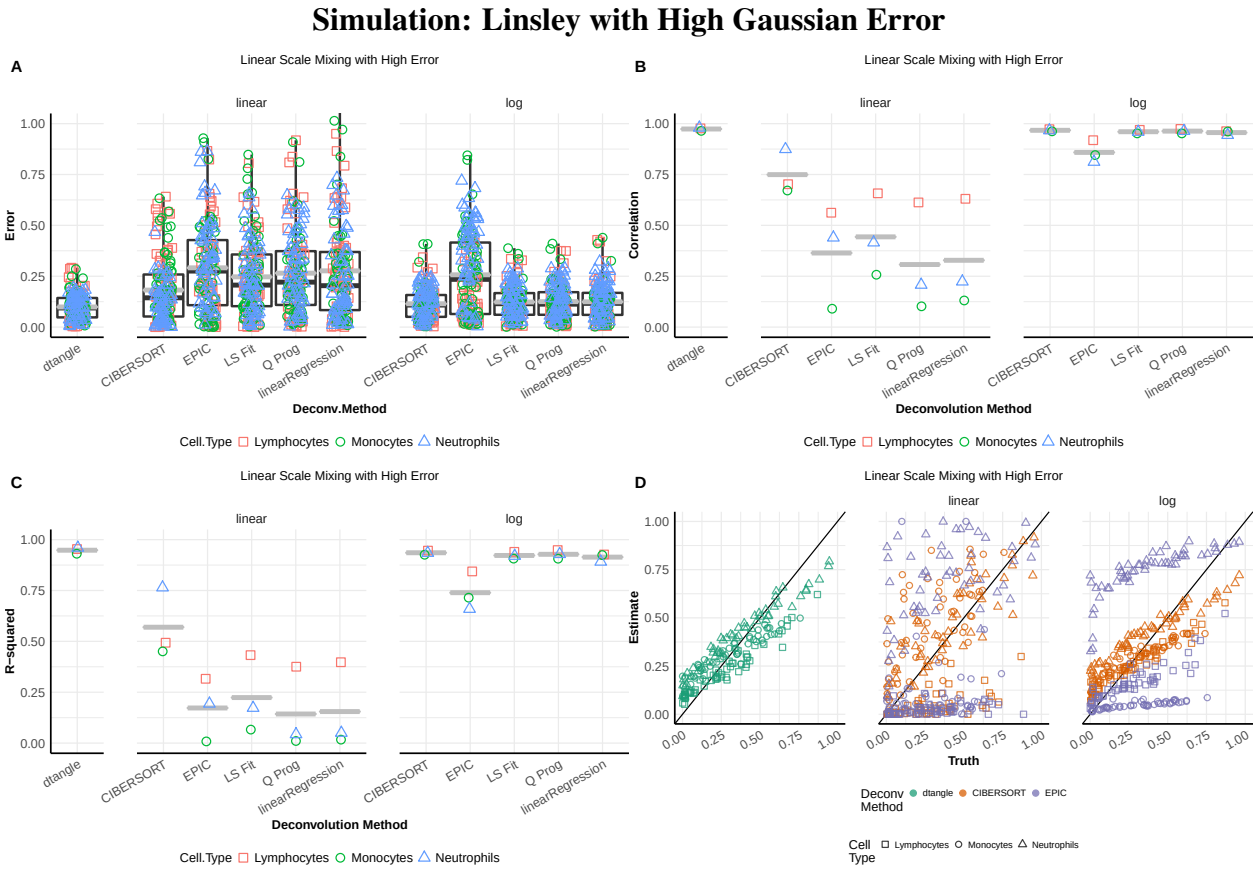
label=sim:mixpoisson:parsons:outliers:outliersmixpoissonall

Figure 5.42: Similar to Figure 5.41 but with outliers added to the simulated data.



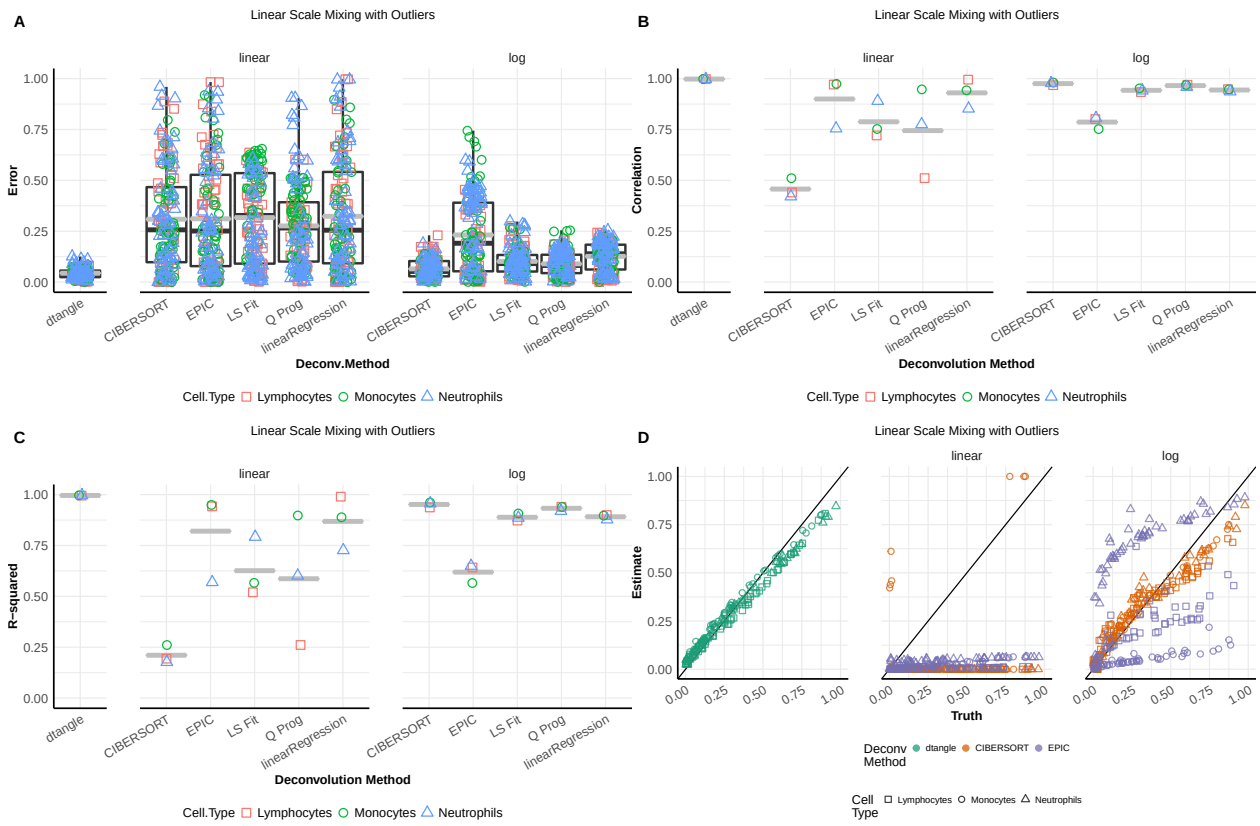
label=sim:mixgaussian:linsley:lowerror:lowerrormixgaussianall

Figure 5.43: Similar to Figure 5.29 but simulation was done by in-silico mixtures of reference cell type profiles from the Linsley data set.



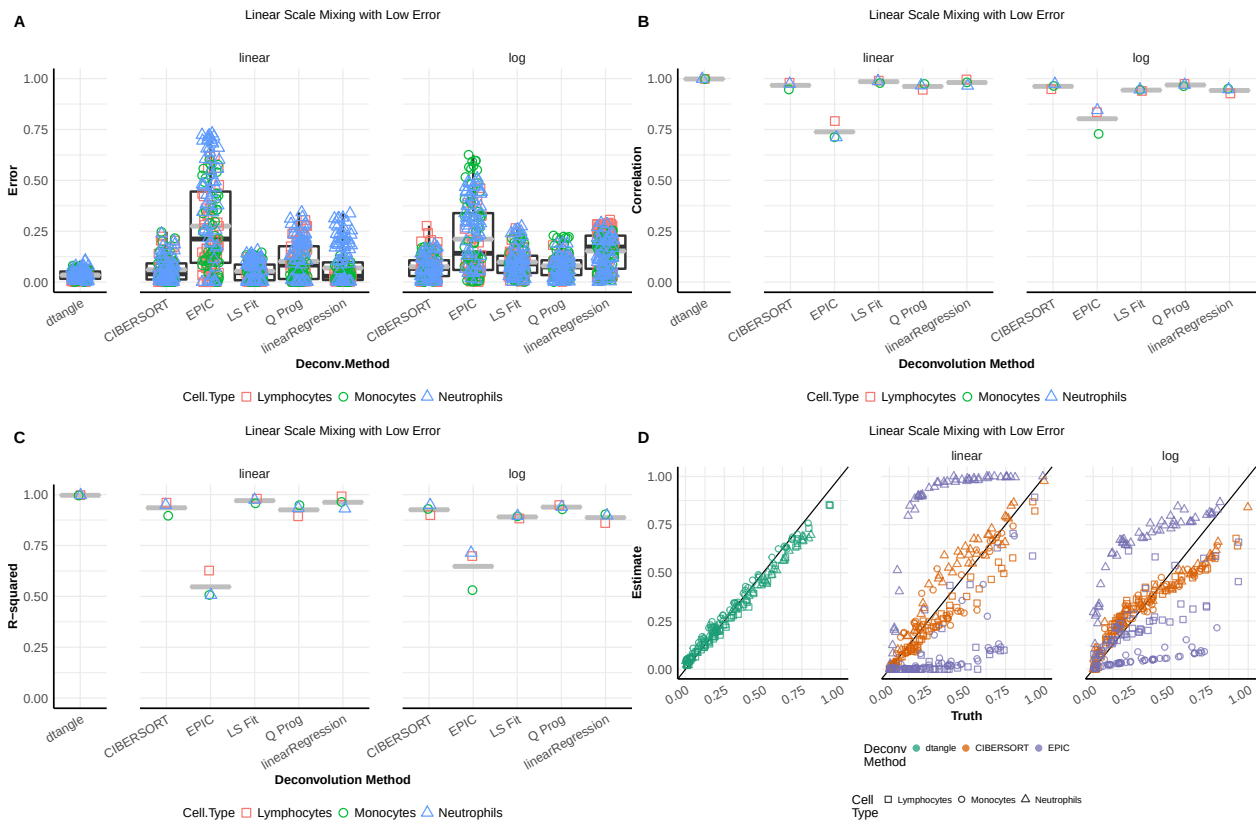
label=sim:mixgaussian:linsley:higherror:higherrormixgaussianall

Figure 5.44: Similar to Figure 5.43 but with a high error variance used in simulation.



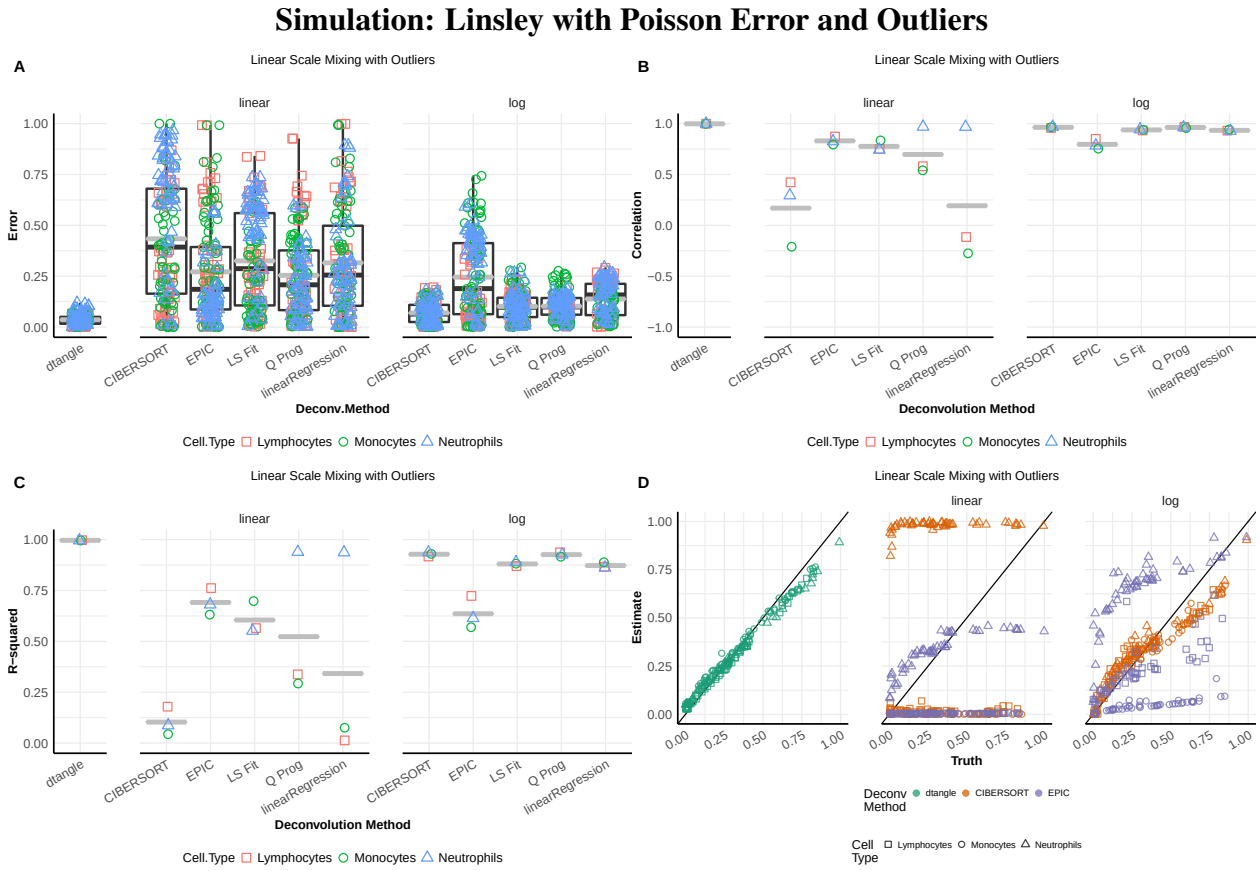
label=sim:mixgaussian:linsley:outliers:outliersmixgaussianall

Figure 5.45: Similar to Figure 5.43 but with outliers added to the simulated data.



label=sim:mixpoisson:linsley:lowerror:lowerrormixpoissonall

Figure 5.46: Similar to Figure 5.43 but using poisson error.



label=sim:mixpoisson:linsley:outliers:outliersmixpoissonall

Figure 5.47: Similar to Figure 5.46 but with outliers added to the simulated data.

Chapter 6

Supplement: MEMA Transformations

6.1 Robust Re-scaling Mathematics

In this section we expound upon the mathematical details of our three transformation steps (G), (Z), and (O). In brief, the three steps are:

1. (G) a robust Gaussianizing non-linear scale change,
2. (Z) a robust z-score transformation
3. (O) an outlier removal step.

In the following sections we will discuss each step in detail.

6.1.1 (G) Gaussianizing non-linear scale change.

Let $Y \in \mathbb{R}^{M\times N}$ be a data matrix. Let $\mathcal{T}$ be an indexed family $\mathcal{T} = \{T_\lambda \mid \lambda \in \Lambda\}$ of differentiable, monotonic, functions $T_\lambda : S_\lambda \to \mathbb{R}$ on some $S_\lambda \subseteq \mathbb{R}$. In this manuscript we choose $\mathcal{T} = \{(\text{sign}(y)|y|^\lambda - 1)/\lambda, \lambda \in \mathbb{R}\} \cup \{\text{asinh}(\lambda y)/\lambda, \lambda \geq 0\}$ however many reasonable choices of parameterized families are available. Let $\widehat{\lambda}$ be an estimate of the value of $\lambda \in \Lambda$ that makes of $T_\lambda(Y)$ as normally distributed as possible among all $T_\lambda \in \mathcal{T}$. We estimate this value through a robust Box-Cox-like procedure [Box and Cox, 1964]. Essentially this procedure is to estiamte $\widehat{\lambda} = \text{median}_j \widehat{\lambda}_j$ where $\widehat{\lambda}_j$ is the Box-Cox transformation parameter estimate using the j^{th} column of Y. The specifics for estimating λ are as follows.

Define Y_{*j} as the j^{th} column of Y, and

$$\nu_{*j} = T_\lambda^{-1}(Y_{*j}), \text{ equiv. } Y_{*j} = T_\lambda(\nu_{*j}). \tag{6.1}$$

If we assume that λ makes ν_{*j} approximately distributed $N(\mu_j, \sigma_j^2)$ then we can estimate it using the traditional Box-Cox approach through maximizing the likelihood. Define $L_j(\lambda)$ to be the

profile likelihood (over μ and σ) for λ based upon Y_{*j} and let

$$\widehat{\lambda_j} = \arg\max_{\lambda \in \Lambda} L_j(\lambda)$$

be the MLE. Typically, for some S the space of possible parameters Λ is a non-null subset of $\mathbb{R}^S$. In this case, we define $\widehat{\lambda}$ as

$$\widehat{\lambda} = \text{median}_j \widehat{\lambda_j}.$$

Unfortunately, the estimate of $\widehat{\lambda}$ is more complicated if Λ is a null set. This typically arises when the parameterized family $\mathcal{T}$ is the union of two families. For example, consider the family

$$\mathcal{T} = \{x^\alpha \mid \alpha \in \mathbb{R}\} \cup \{\exp(\beta x) \mid \beta \in \mathbb{R}\}.$$

If we define $\Lambda = \{0, 1\} \times \mathbb{R}$ this family may be equivalently parameterized as

$$\mathcal{T} = \{\lambda_1 x^{\lambda_2} + (1 - \lambda_1)\exp(\lambda_2 x) \mid (\lambda_1, \lambda_2) \in \Lambda\}.$$

To deal with this case, we split Λ into two spaces: its discrete coordinates and its continuous coordinates. For the discrete coordinates we define $\widehat{\lambda}$ as the mode over the $\widehat{\lambda_j}$. For the continuous coordinates we use the median as before. Specifically, assume that Λ may be written as the Cartesian product of a discrete space Λ_D and a continuous space Λ_C so that $\Lambda = \Lambda_D \times \Lambda_C$. In our example above, $\Lambda_D = \{0, 1\}$ and $\Lambda_C = \mathbb{R}$. In a similar fashion we can decompose each $\widehat{\lambda_j}$ as $\widehat{\lambda_j} = (\widehat{\lambda_j^D}, \widehat{\lambda_j^C})$. Given these decompositions, define

$$\widehat{\lambda^D} = \text{mode}_j \widehat{\lambda_j^D}$$

and then let $\Lambda|_D = \left\{\widehat{\lambda^D}\right\} \times \Lambda_C$. The space $\Lambda|_D$ is simply the space Λ with the discrete coordinates fixed at the mode of their column-wise estimates. Finally, if we define

$$\widetilde{\lambda_j} = \arg\max_{\lambda \in \Lambda|_D} L_j(\lambda)$$

then, similar to the continuous case, we let

$$\widehat{\lambda} = \text{median}_j \widetilde{\lambda_j}.$$

Finally, the Gaussianized version of Y is

$$T_{\widehat{\lambda}}^{-1}(Y).$$

6.1.2 (Z) Standardizing z-score.

If $Y \in \mathbb{R}^{M \times N}$ is a data matrix we define the robust z-score version of Y as follows. Let $\widetilde{Y}$ be a $(q = 0.001)$ winsorized version of Y. Then let

$$\widehat{\mu} = \frac{\mathbb{1}_M' \widetilde{Y} \mathbb{1}_N}{MN} \text{ and } \widehat{\sigma} = \sqrt{\frac{\mathbb{1}_M' \widetilde{Y}^2 \mathbb{1}_N}{MN}}$$

where $\widetilde{Y}^2$ is an element-wise exponential. Finally, then the robust z-score version of Y is

$$\frac{Y - \widehat{\mu}}{\widehat{\sigma}}$$

where the subtraction and division are element-wise.

6.1.3 (O) Outlier removal.

Again let $Y \in \mathbb{R}^{M \times N}$ be our data matrix. Our outlier removal step is a simple thresholding procedure. First let Z be the robust z-scored version of Y according to the previous section. We then define $[Y]$ as version of Y with outliers removed so that

$$\begin{cases} [Y]_{ij} = Y_{ij}, & |Z_{ij}| \leq 4 \\ [Y]_{ij} = \text{NA}, & |Z_{ij}| > 4 \end{cases}$$

where "NA" denotes a missing value.

6.2 Average and Missing Singular Vectors

First let us define a product for matrices with missing values. Let $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{n \times k}$ be two matrices with (potentially) missing values. Define A_0 and B_0 as the matrices A and B with missing values replaced by zeros, respectively. Furthermore define $A_{\mathbb{1}}$ and $B_{\mathbb{1}}$ so that $(A_{\mathbb{1}})_{ij} = \mathbb{1}\{A_{ij} \text{ is not missing}\}$ and similarly for $B_{\mathbb{1}}$. Define the "missing product" $A \cdot B$ for two matrices as $A \cdot B = n(A_0 B_0 \oslash A_{\mathbb{1}} B_{\mathbb{1}})$ where $\oslash$ is element-wise "Hadamard" division of matrices. This is well-defined so long as none of the entries of $A_{\mathbb{1}} B_{\mathbb{1}}$ are zero. If there are no missing values this product is exactly the normal matrix product.

We will now define left and right singular vectors for a matrix with missing values. Let $Y \in \mathbb{R}^{M \times N}$ be a matrix with missing values. We define the missing left Gram matrix of Y as $G^L(Y) = Y \cdot Y'$ and the missing right Gram matrix of Y as $G^R(Y) = Y' \cdot Y$. While symmetric, these missing Gram matrices may not be positive semi-definite (PSD). Let Π_+ be the function that projects a

real symmetric matrix onto the space of PSD matrices. If $A \in \mathbb{R}^{m \times m}$ is a symmetric matrix with eigen-decomposition $A = QDQ'$ then $A_+ \stackrel{def}{=} \Pi_+(A) = QD_+Q'$ where $(D_+)_{ij} = max(D_{ij}, 0)$. Define $G^L_+(Y) = \Pi_+(G^L(Y))$ and $G^R_+(Y) = \Pi_+(G^R(Y))$ as the PSD missing left and right Gram matrices. Let $G^L_+(Y)$ and $G^R_+(Y)$ have the eigen-decompositions $G^L_+(Y) = UD_LU'$ and $G^R_+(Y) = VD_RV'$. We call U the missing left singular vectors of Y and V the missing right singular vectors of Y. If there are no missing values U and V are precisely the normal left and right singular vectors of Y.

Let $\{Y^{(p)}\}_{p=1}^P$ be a collection of P different feature matrices each with (potentially different) missing values. Define the left and right average missing Gram matrices as

$$\overline{G^L} = \frac{1}{P}\sum_{i=1}^{P} G^L\left(Y^{(p)}\right) \quad \text{and} \quad \overline{G^R} = \frac{1}{P}\sum_{i=1}^{P} G^R\left(Y^{(p)}\right).$$

Similarly define the left and right PSD average missing Gram matrices as

$$\overline{G^L}_+ = \Pi_+\overline{G^L} \quad \text{and} \quad \overline{G^R}_+ = \Pi_+\overline{G^R}.$$

Let the eigen-decompositions of $\overline{G^L}_+$ and $\overline{G^R}_+$ be $\overline{G^L}_+ = UD_LU'$ and $\overline{G^R}_+ = VD_RV'$. We call U the left average singular vectors (ASVs) and V the right average singular values (ASVs). If $P = 1$ these reduce to the right and left missing singular vectors.

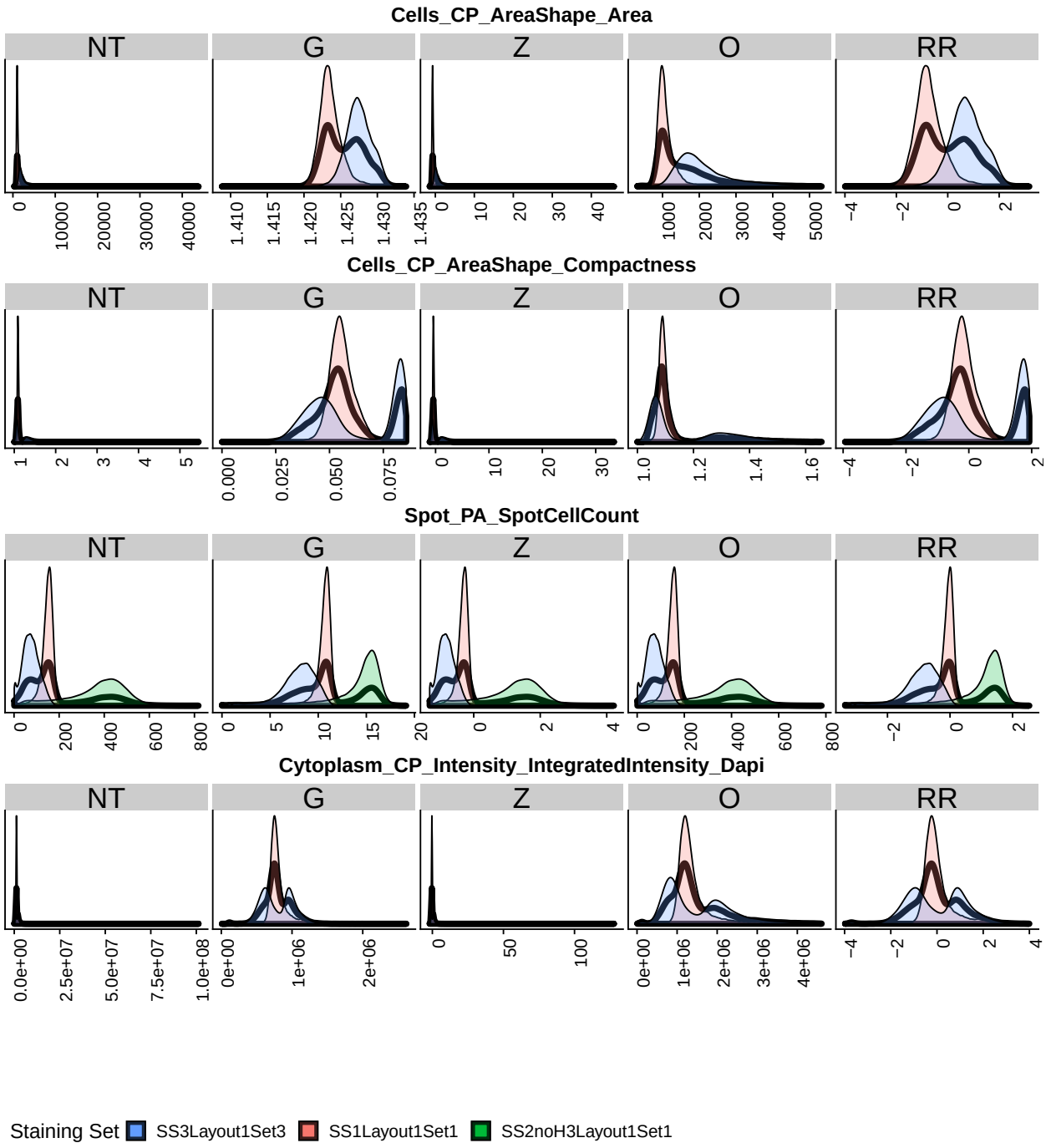


Figure 6.1: Density of elements of feature matrices. Black density is all elements combined. Colored densities are the densities denote staining batch. Subplots are for five processing transformations of this matrix: (NT) no transformation, (G) Gaussianization, (Z) z-score, (O) outlier removal, (RR) the three-step (G), (Z), and (O), robust re-scaling.

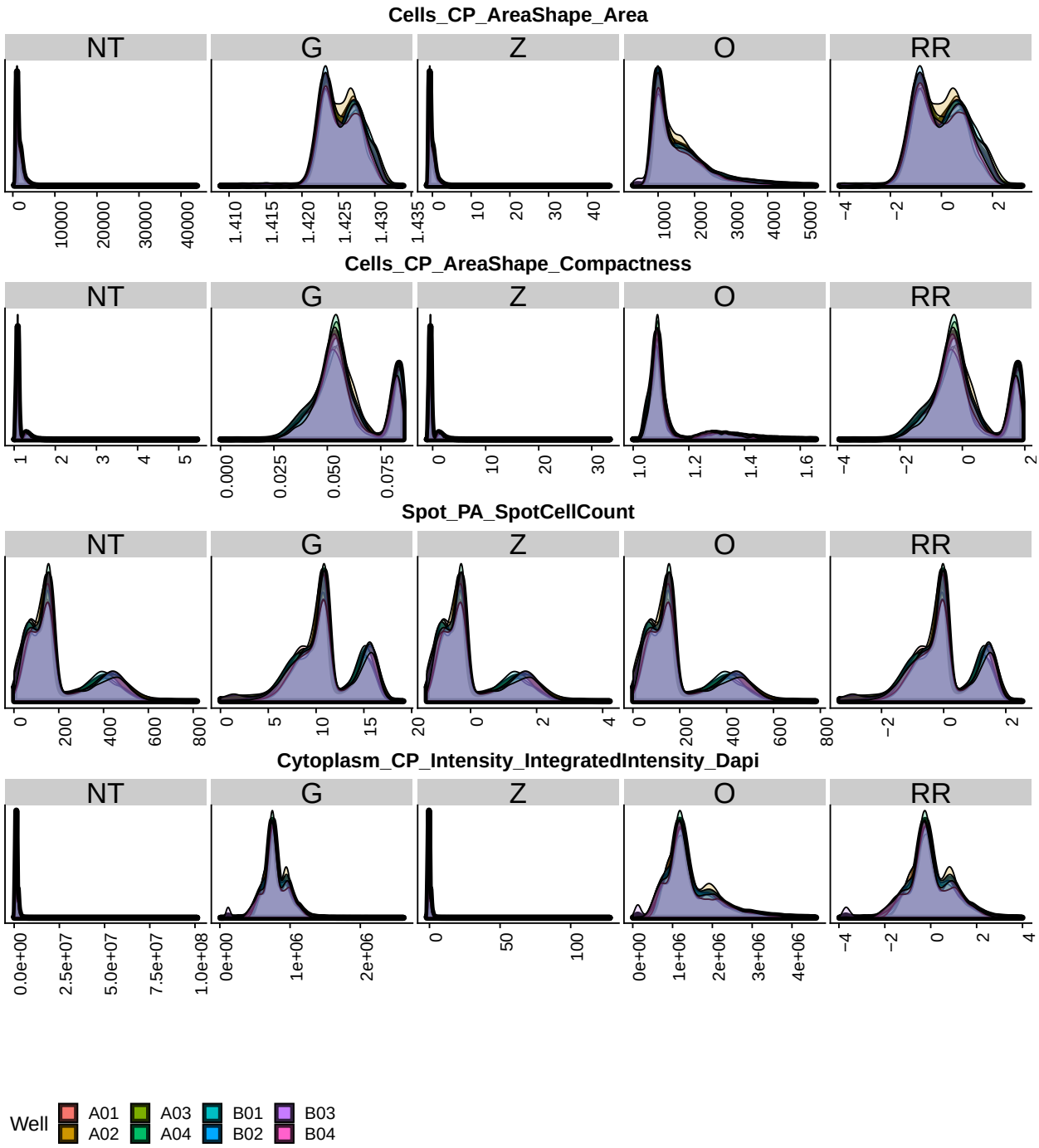


Figure 6.2: Similar to Figure 6.1 except colors indicate well.

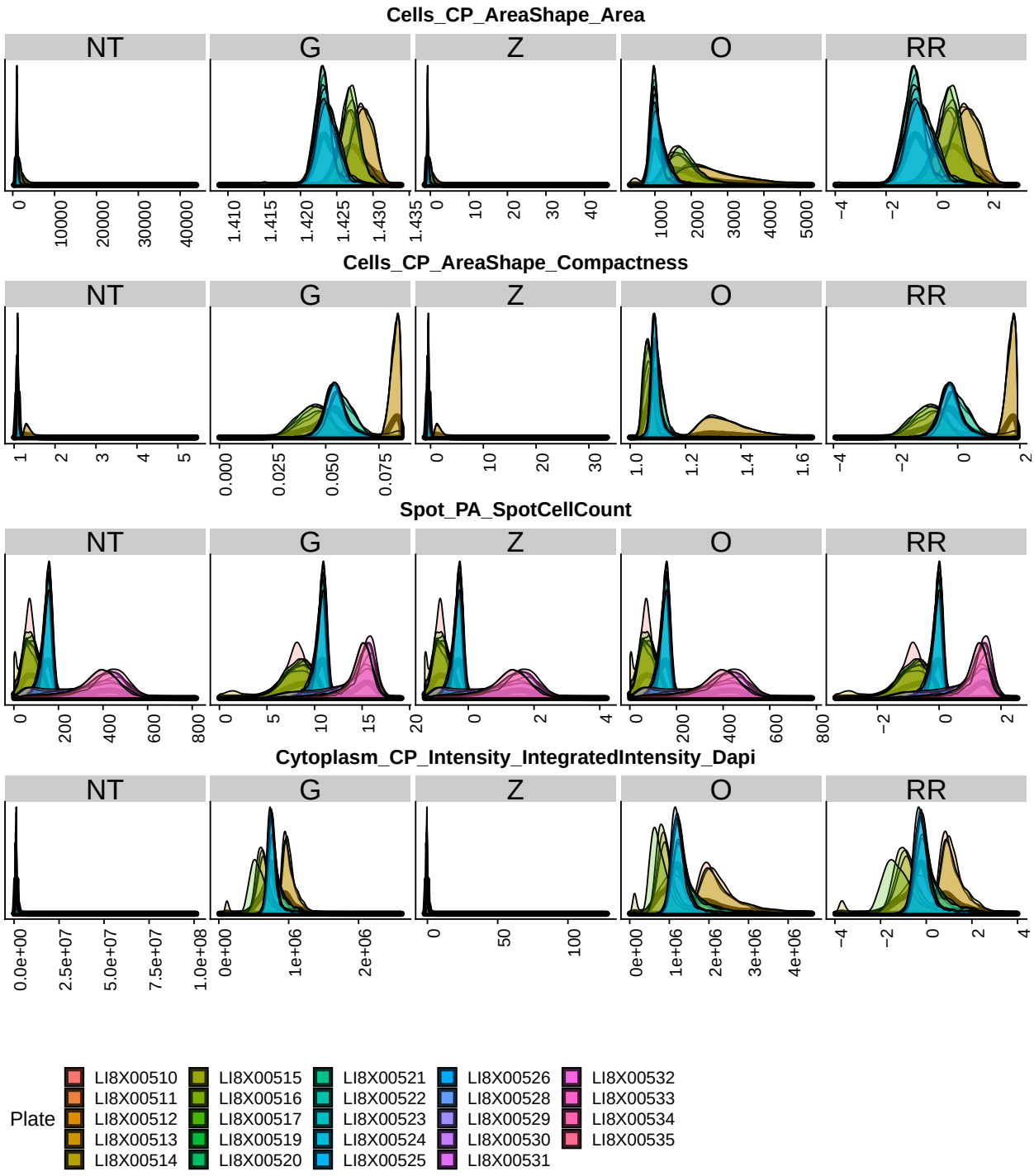


Figure 6.3: Similar to Figure 6.1 except colors indicate plate.

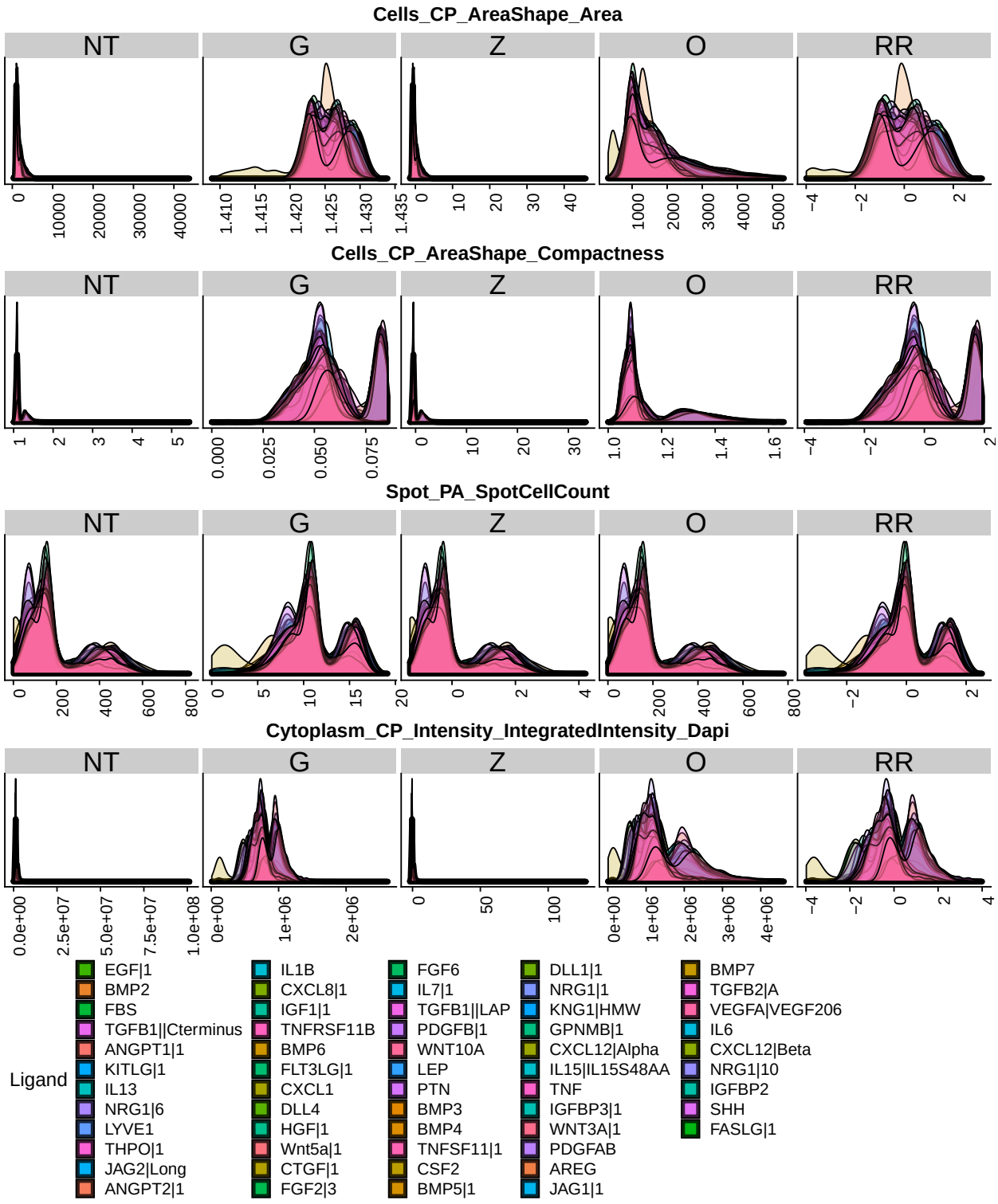


Figure 6.4: Similar to Figure 6.1 except colors indicate ligand.

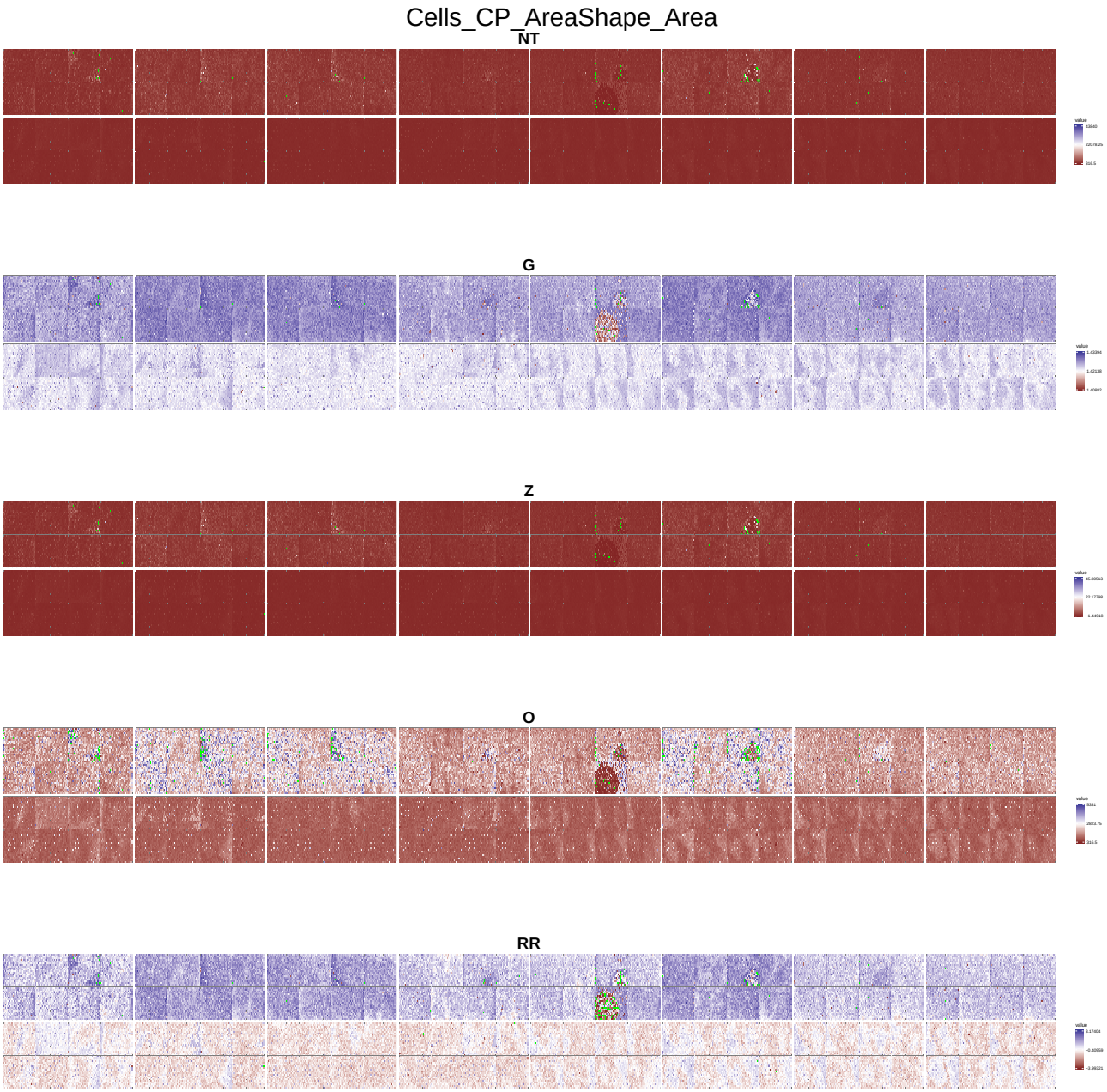


Figure 6.5: The next series of plots are heat-maps of MEMA plates across the five transformations (NT), (G), (Z), (O), (RR). Rows of each plot are the staining three batches. Colors are more blue if they are close to the minimum, red if they are close to the maximum, and white if they are close to half-way between. Green spots are missing. Dark grey spots are omitted according to the MEMA design.

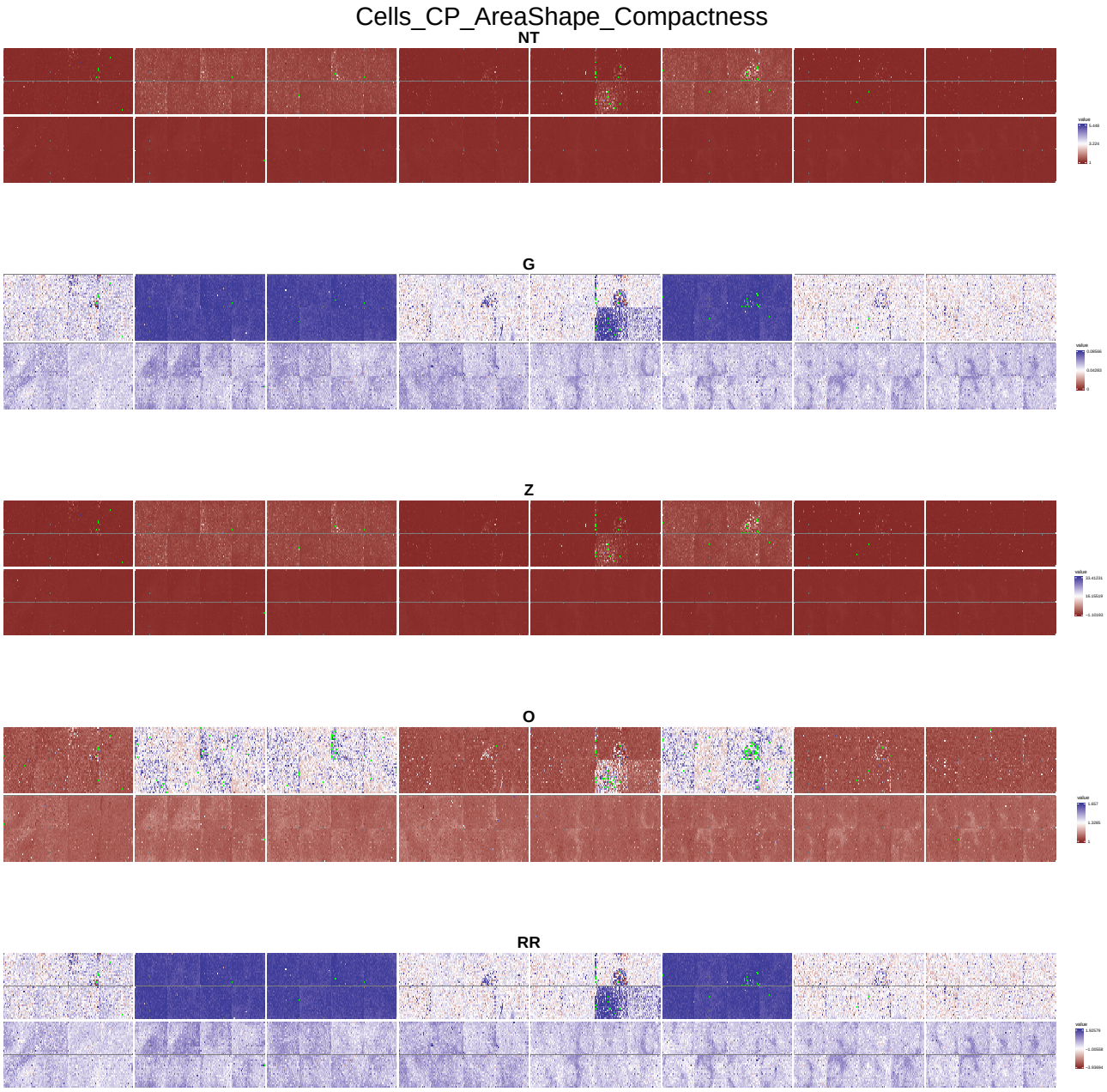


Figure 6.6: Similar to Figure 6.5 but for compactness.

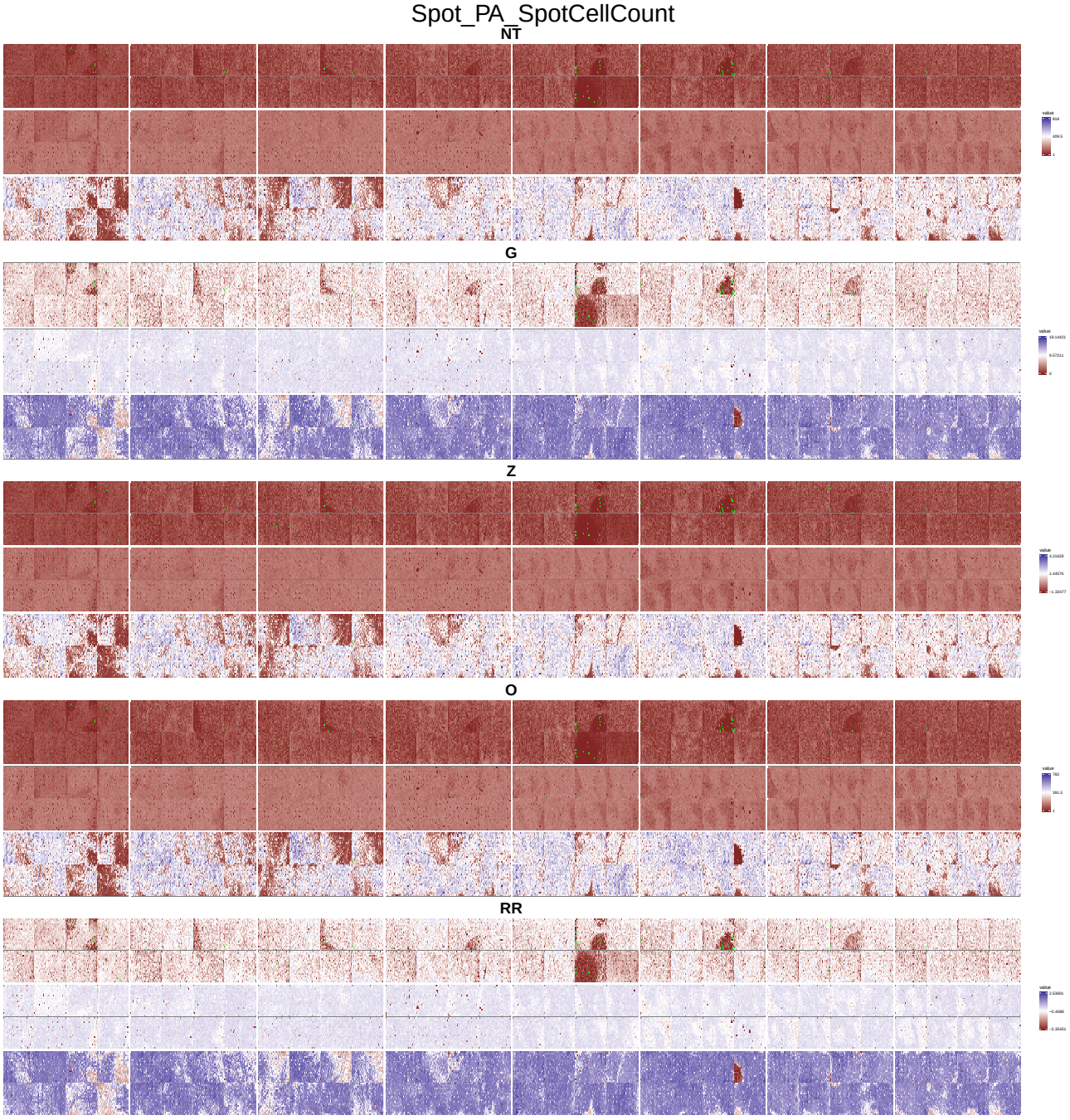


Figure 6.7: Similar to Figure 6.5 but for cell count.

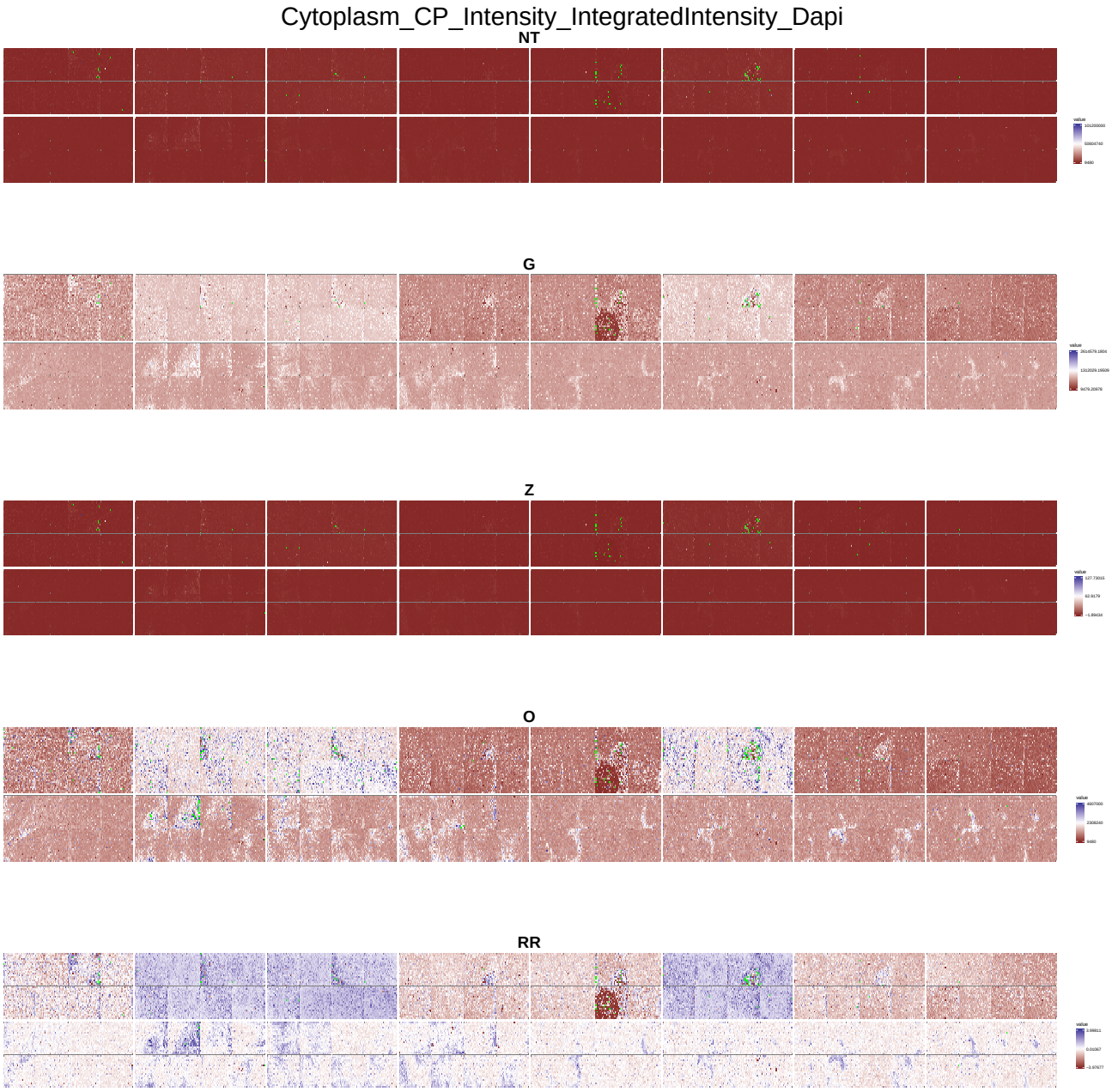


Figure 6.8: Similar to Figure 6.5 for for DAPI intensity.

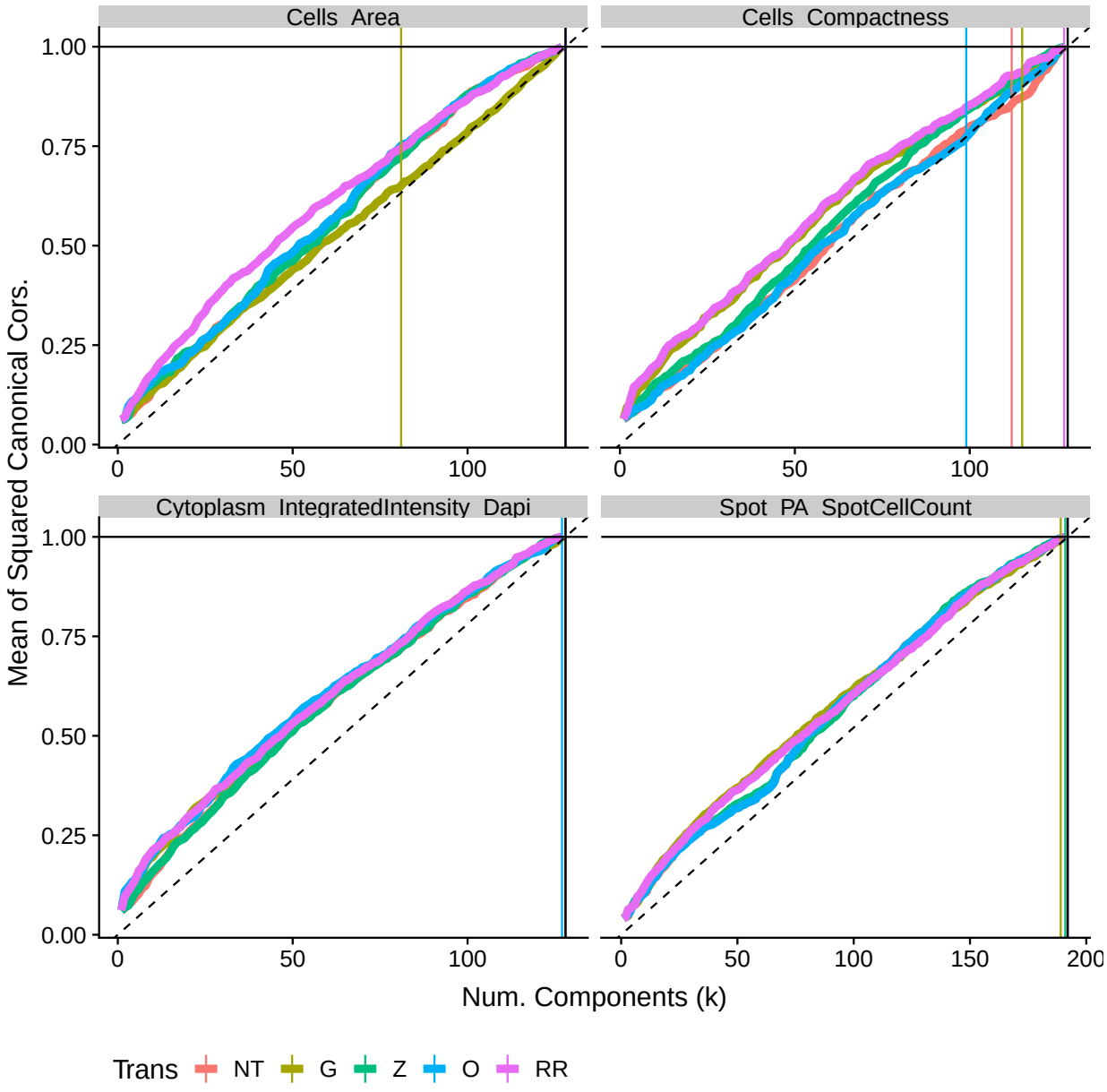


Figure 6.9: Mean of the squared canonical correlations between the first k principal components and the plate batch indicator variables.

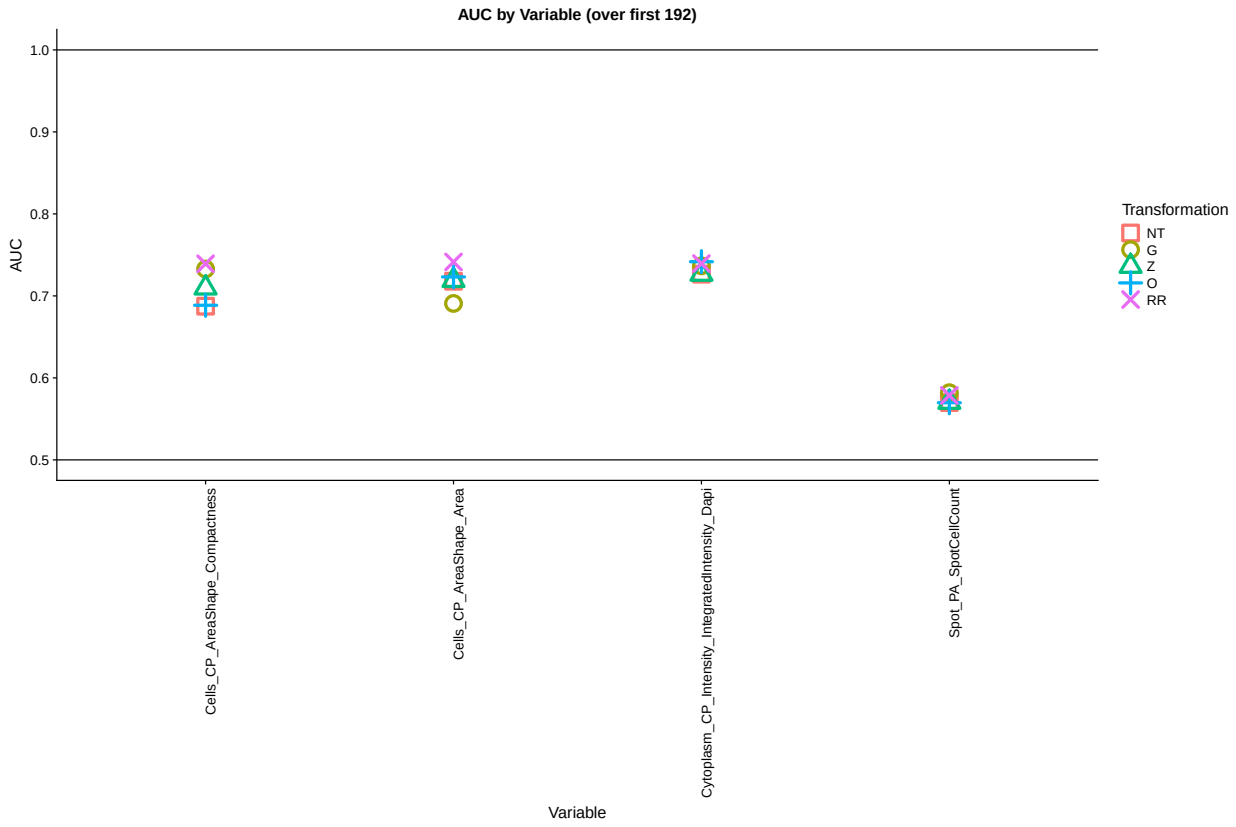


Figure 6.10: Grand mean of the squared canonical correlations across number of components (k). Canonical correlation is calculated between the first k principal components and the plate indicator variables.

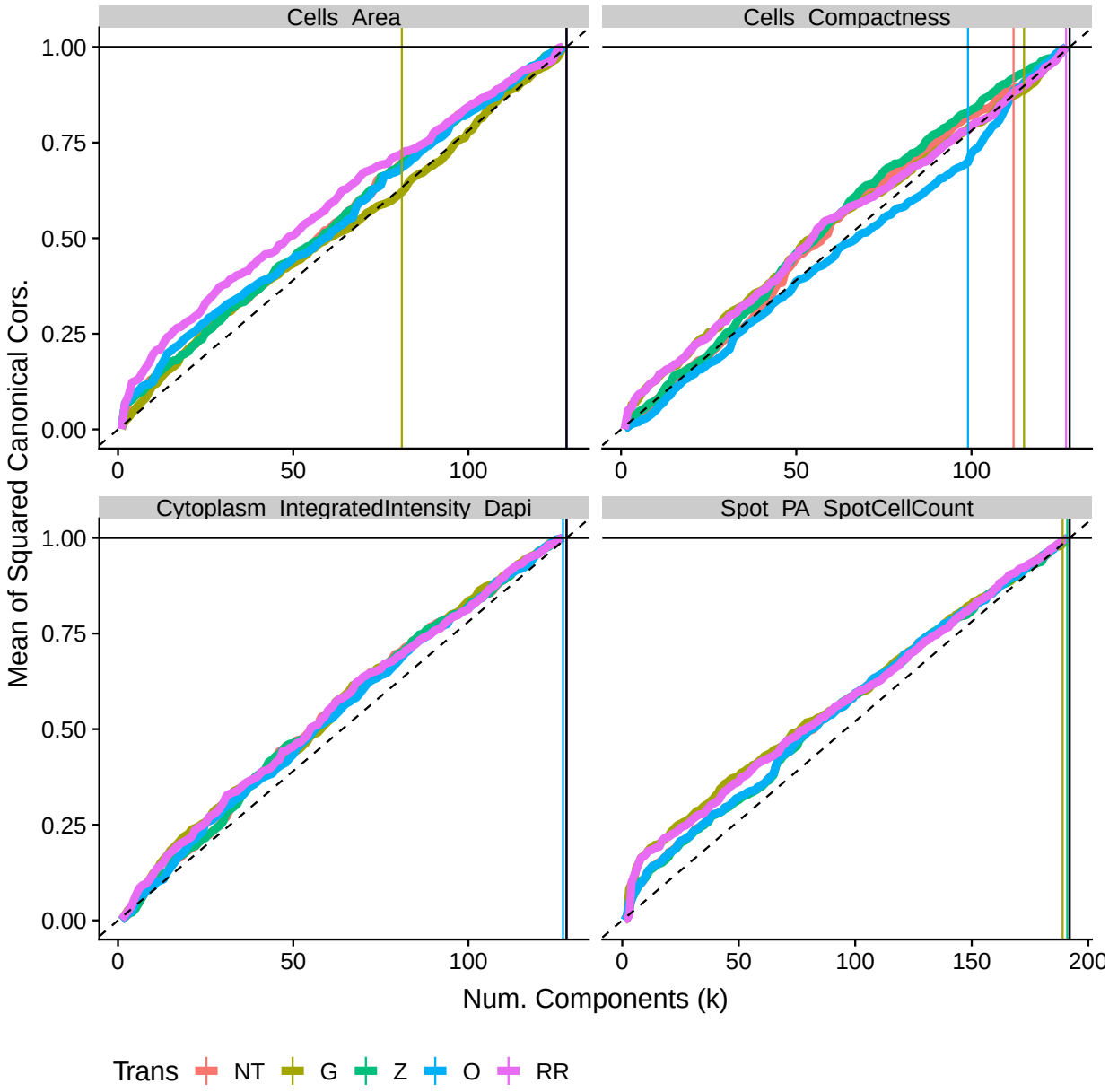


Figure 6.11: Similar to Figure 6.9 except correlation with well batch indicators.

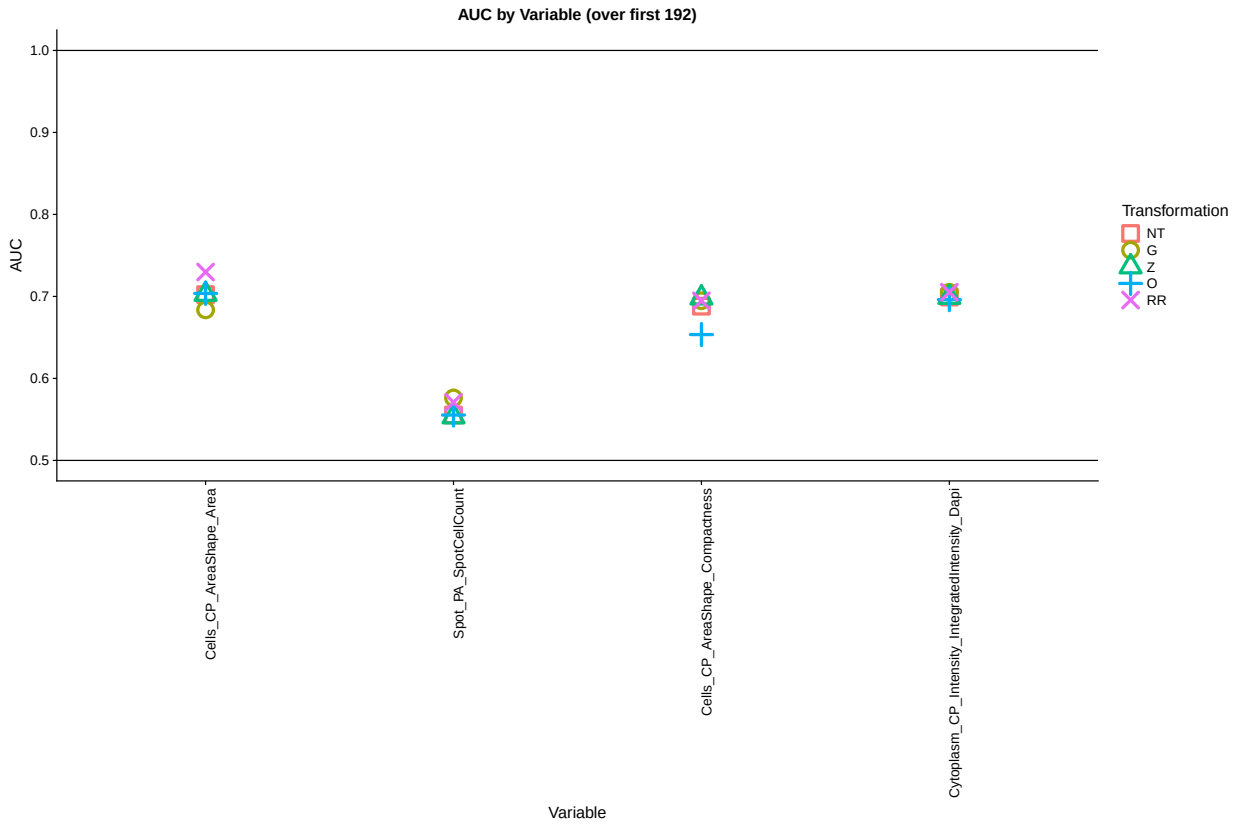


Figure 6.12: Similar to Figure 6.10 except correlation with well batch indicators.

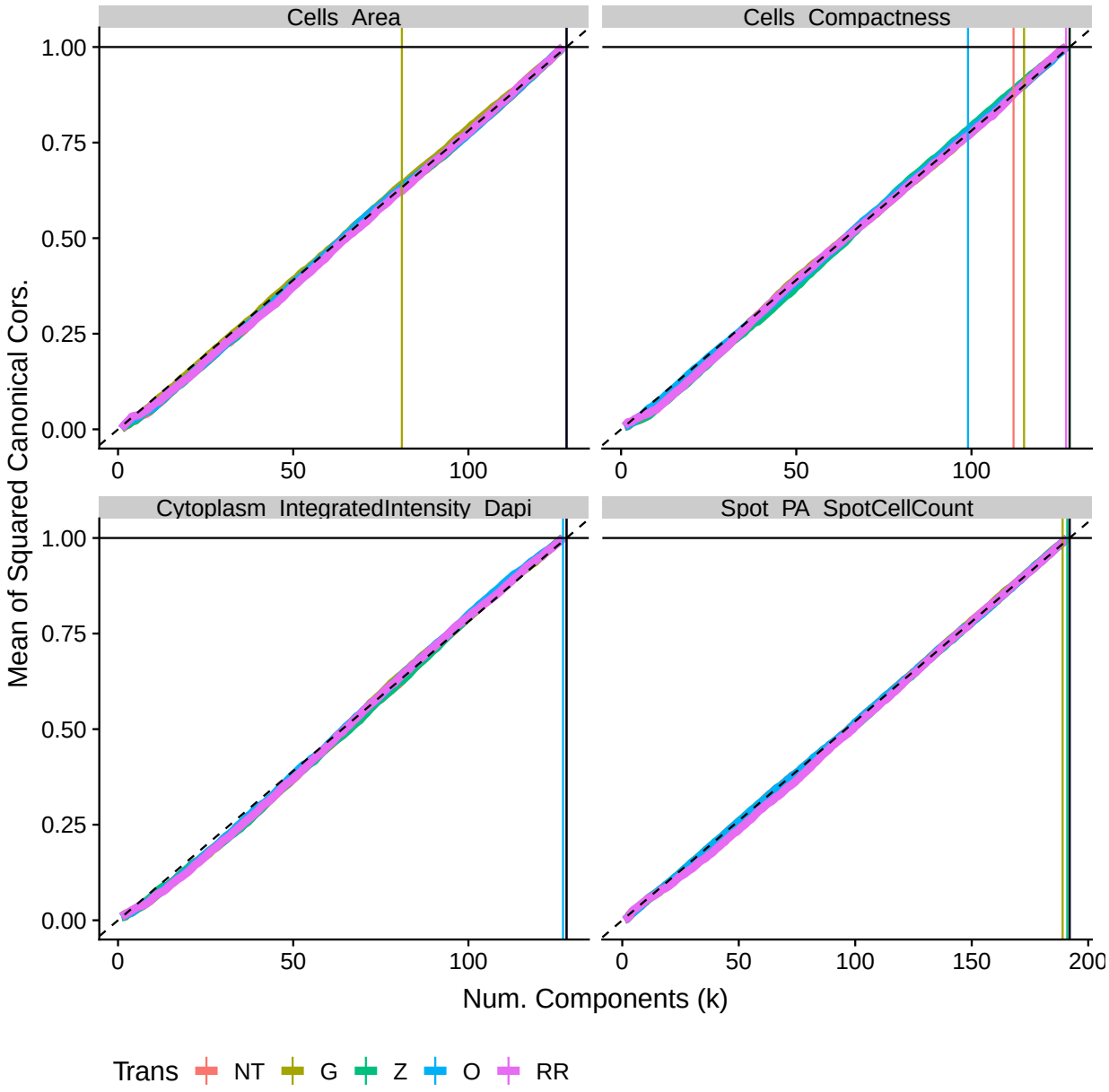


Figure 6.13: Similar to Figure 6.9 except correlation with ligand batch indicators.

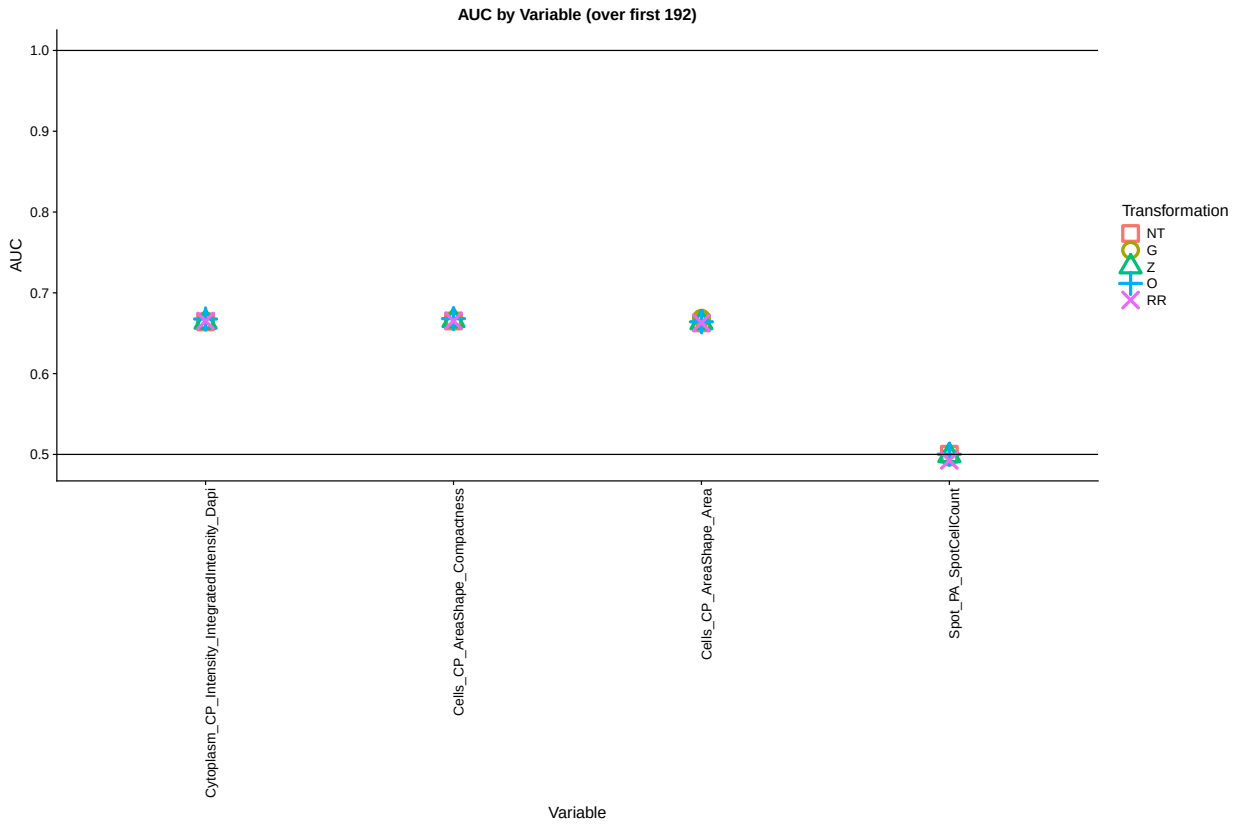


Figure 6.14: Similar to Figure 6.10 except correlation with well batch indicators.

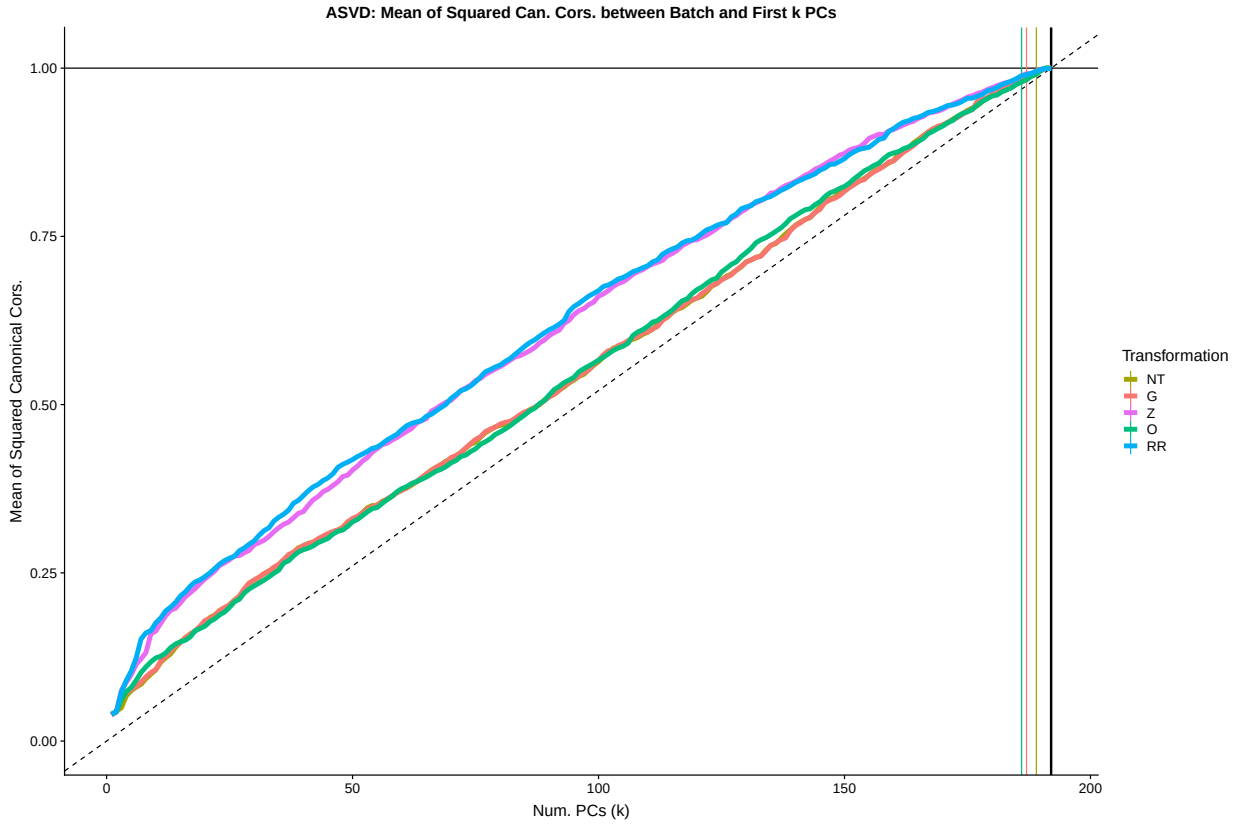


Figure 6.15: Mean of the squared canonical correlations between the first k principal components and the plate indicator variables. Principal components come from integration of the 21 features that are measured across all MEMAs.

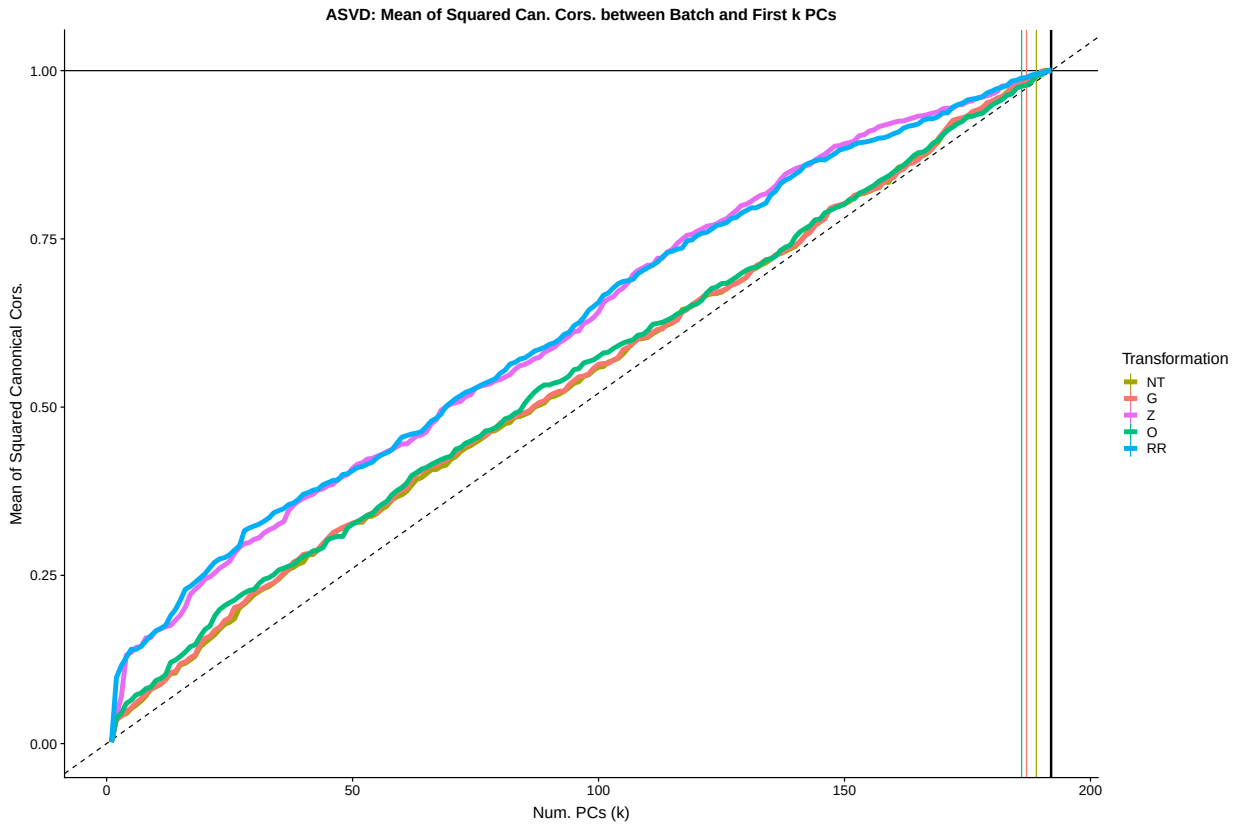


Figure 6.16: Similar to Figure 6.15 but calculating correlation with well indicators.

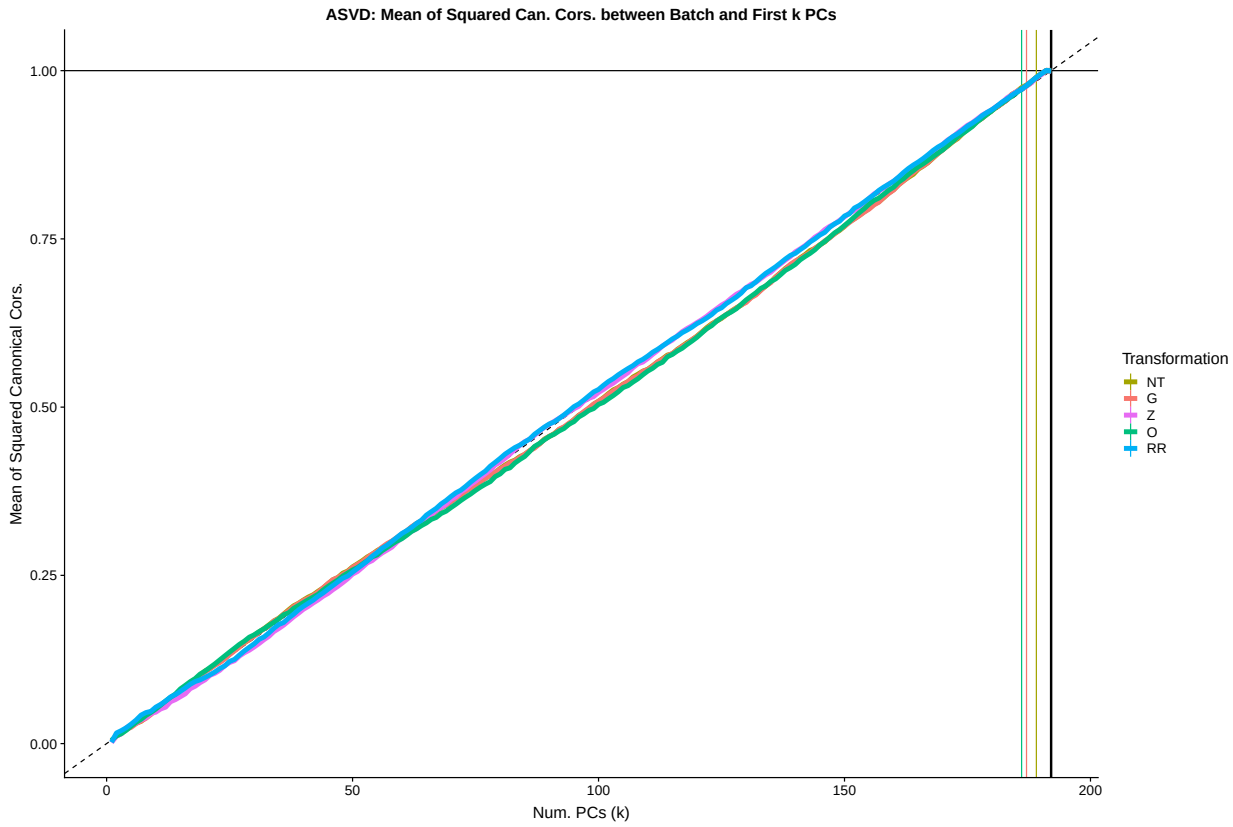


Figure 6.17: Similar to Figure 6.15 but calculating correlation with ligand indicators.

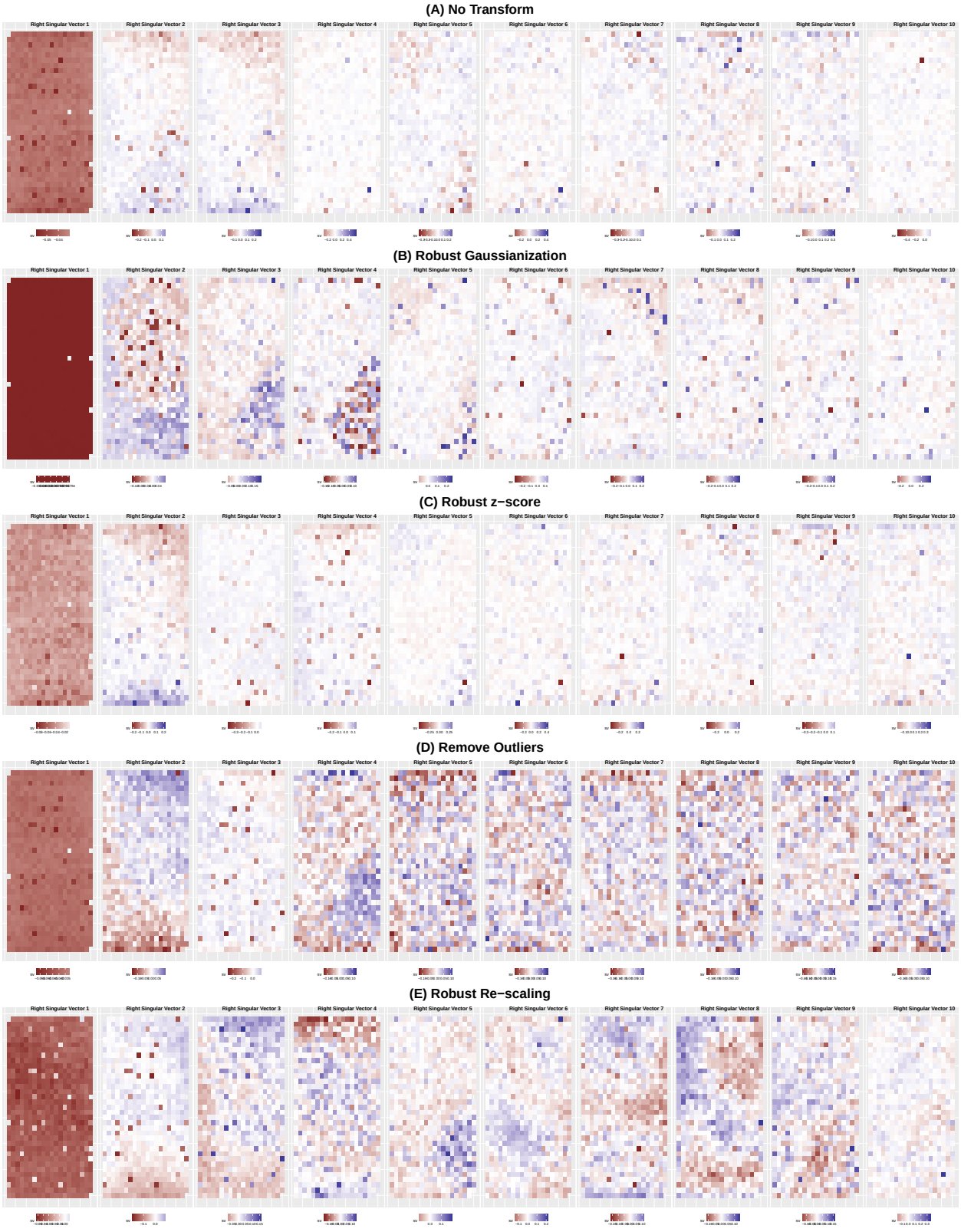


Figure 6.18: Heat map of elements of top ten right singular vectors for the cell area feature.

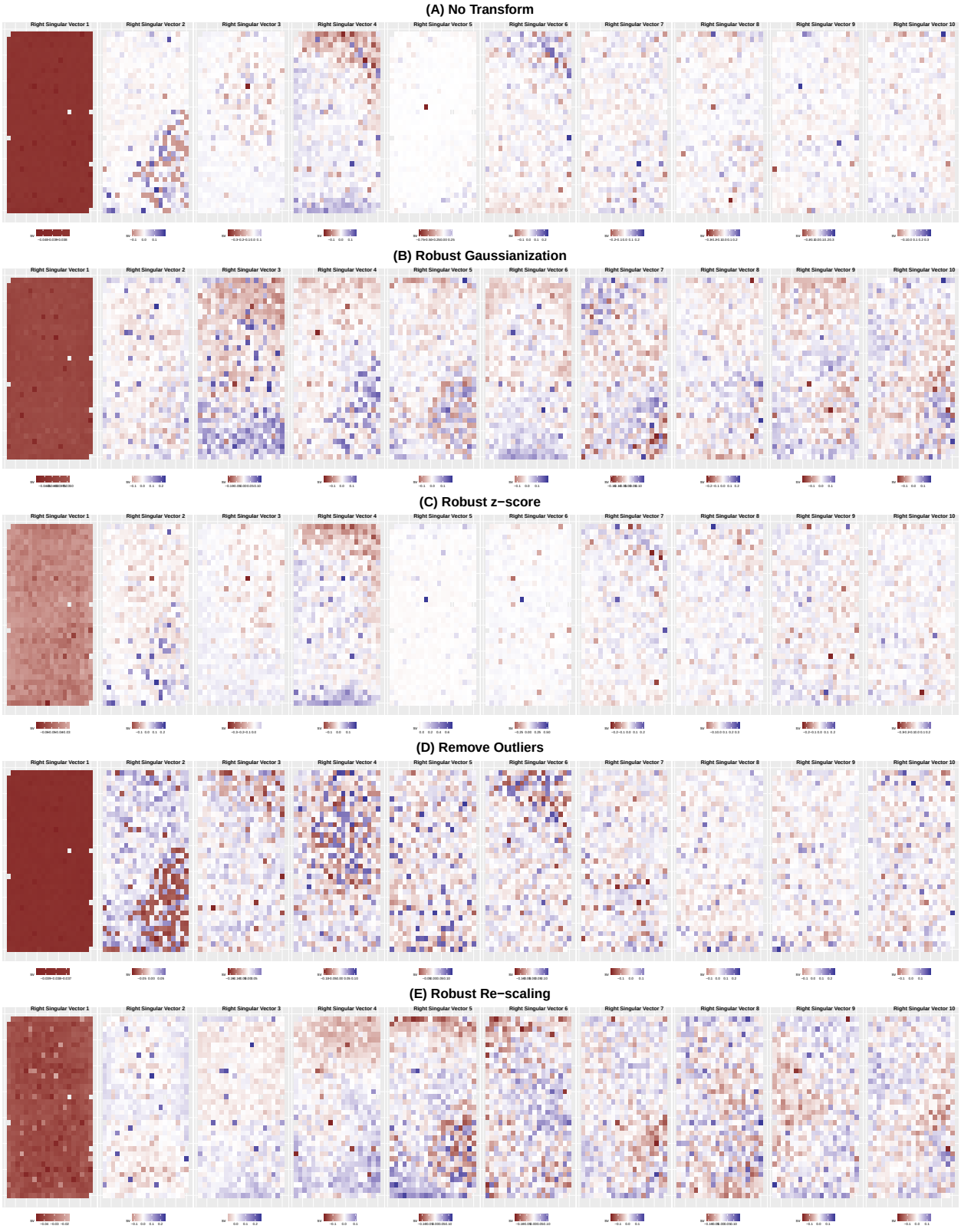


Figure 6.19: Similar to Figure 6.18 but for cell count feature.

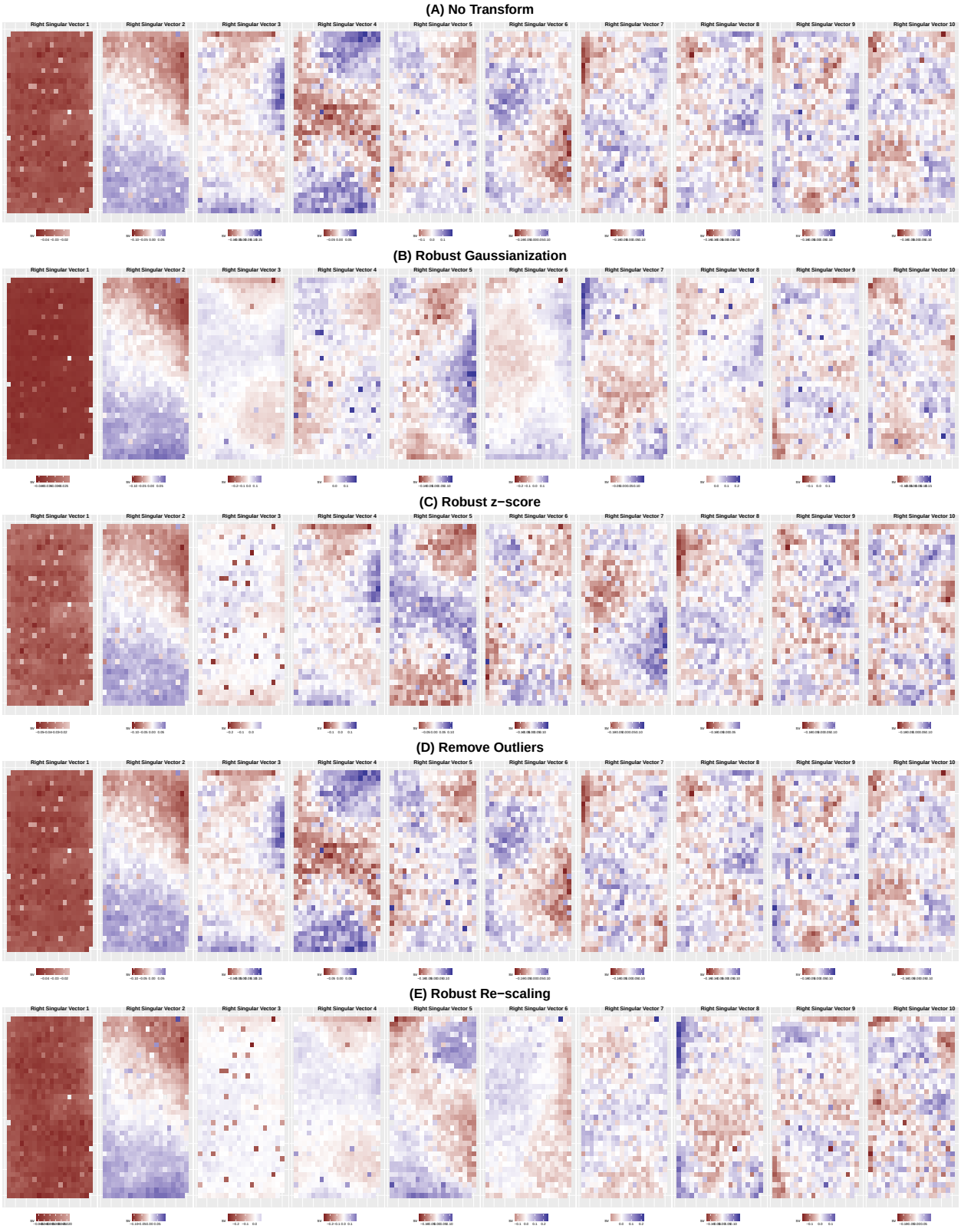


Figure 6.20: Similar to Figure 6.18 but for cell count feature.

Figure 6.21: Scatter plot of elements of top two right singular vectors against each other for the cell area feature. Shape and color indicate ECMp of the spot corresponding to the elements of the singular vector.

Figure 6.22: Similar to Figure 6.21 but for cell compactness feature.

Figure 6.23: Similar to Figure 6.21 but for cell count feature.

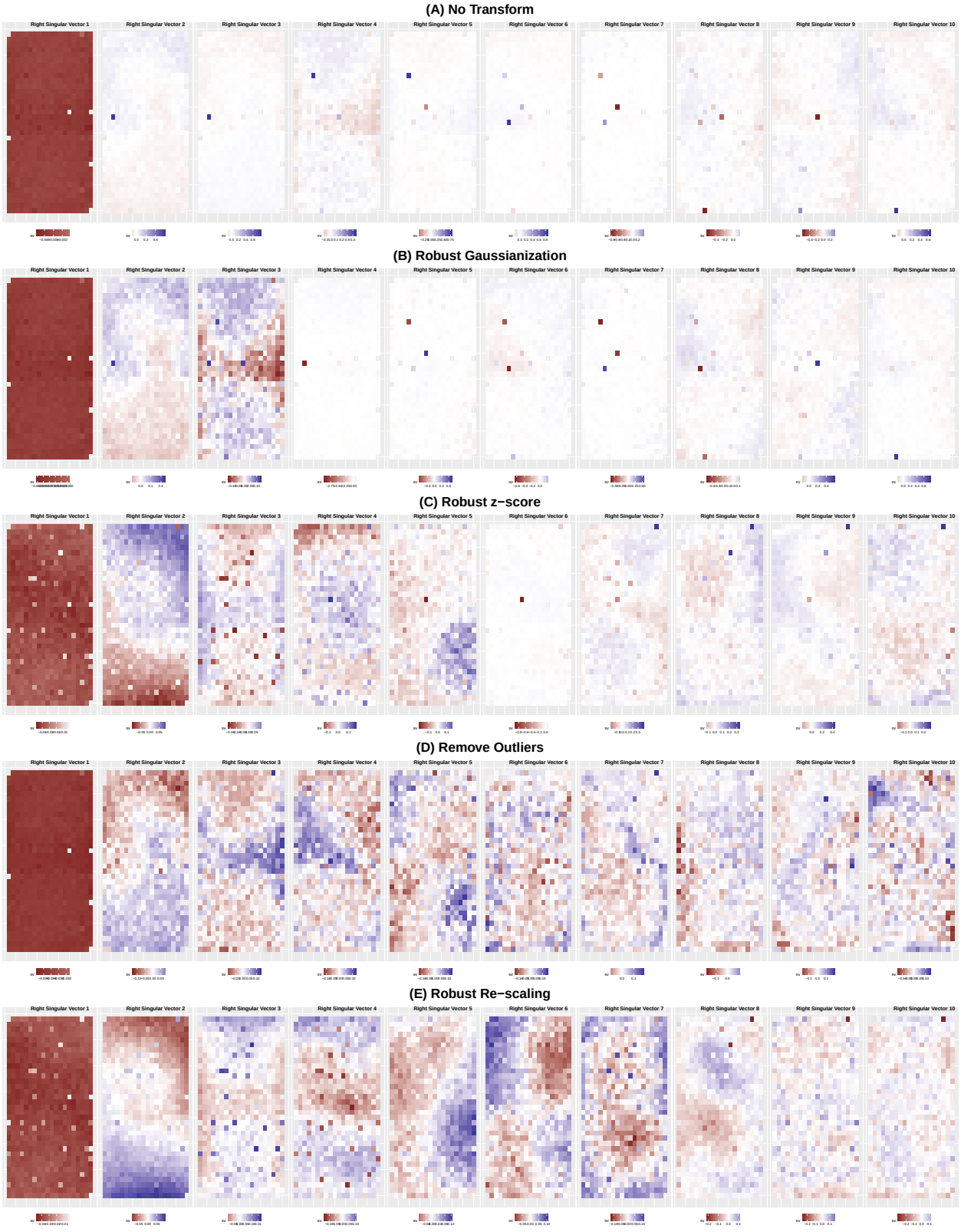


Figure 6.24: Heat-map of top ten right ASVs calculated over 21 features measured on all MEMAs.

Bibliography

(2018). Microenvironment Perturbagen (MEP) LINCS.

Abbas, A. R., Wolslegel, K., Seshasayee, D., Modrusan, Z., and Clark, H. F. (2009). Deconvolution of blood microarray data identifies cellular activation patterns in systemic lupus erythematosus. *PLoS ONE*, **4**(7).

Altboum, Z., Steuerman, Y., David, E., Barnett-Itzhaki, Z., Valadarsky, L., Keren-Shaul, H., Meningher, T., Mendelson, E., Mandelboim, M., Gat-Viks, I., and Amit, I. (2014). Digital cell quantification identifies global immune cell dynamics during influenza infection. *Molecular Systems Biology*, **10**(2), 1–14.

Aran, D., Hu, Z., and Butte, A. J. (2017). xCell: digitally portraying the tissue cellular heterogeneity landscape. *Genome Biology*, **18**(1), 220.

Becht, E., Giraldo, N. A., Lacroix, L., Buttard, B., Elarouci, N., Petitprez, F., Selves, J., Laurent-Puig, P., Sautès-Fridman, C., Fridman, W. H., and de Reyniès, A. (2016). Estimating the population abundance of tissue-infiltrating immune and stromal cell populations using gene expression. *Genome biology*, **17**(1), 218.

Bouquet, J., Soloski, M. J., Swei, A., Cheadle, C., Federman, S., Billaud, J.-n., and Rebman, A. W. (2016). Longitudinal Transcriptome Analysis Reveals a Sustained Differential Gene Expression Signature in Patients Treated for Acute Lyme Disease. **7**(1), 1–11.

Bowling, K. M., Thompson, M. L., Amaral, M. D., Finnila, C. R., Hiatt, S. M., Engel, K. L., Cochran, J. N., Brothers, K. B., East, K. M., Gray, D. E., Kelley, W. V., Lamb, N. E., Lose, E. J., Rich, C. A., Simmons, S., Whittle, J. S., Weaver, B. T., Nesmith, A. S., Myers, R. M., Barsh, G. S., Bebin, E. M., and Cooper, G. M. (2017). Genomic diagnosis for children with intellectual disability and/or developmental delay. *bioRxiv*, page 084251.

Box, G. E. P. and Cox, D. R. (1964). An Analysis of Transformations. *Journal of the Royal Statistical Society. Series B (Methodological)*, **26**(2), 211–252.

Burger, J. A., Ghia, P., Rosenwald, A., and Caligaris-Cappio, F. (2009). The microenvironment in mature B-cell malignancies: a target for new treatment strategies. *Blood*, **114**(16), 3367–75.

Capurro, A., Bodea, L. G., Schaefer, P., Luthi-Carter, R., and Perreau, V. M. (2015). Computational deconvolution of genome wide expression data from Parkinson's and Huntington's disease brain tissues using population-specific expression analysis. *Frontiers in Neuroscience*, **9**(JAN), 1–12.

Dame, T. M., Orenzoff, B. L., Palmer, L. E., and Furie, M. B. (2007). Endothelium to Favor Chronic Inflammation 1. *The Journal of Immunology*.

Edgar, R. (2002). Gene Expression Omnibus: NCBI gene expression and hybridization array data repository. *Nucleic Acids Research*, **30**(1), 207–210.

Gaujoux, R. (2013). An introduction to gene expression deconvolution and the CellMix package. pages 1–45.

Gaujoux, R. and Seoighe, C. (2012). Semi-supervised Nonnegative Matrix Factorization for gene expression deconvolution: A case study. *Infection, Genetics and Evolution*, **12**(5), 913–921.

Gong, T., Hartmann, N., Kohane, I. S., Brinkmann, V., Staedtler, F., Letzkus, M., Bongiovanni, S., and Szustakowski, J. D. (2011). Optimal deconvolution of transcriptional profiling data using quadratic programming with application to complex clinical blood samples. *PLoS ONE*, **6**(11).

Hagenauer, M. H., Li, J. Z., Walsh, D. M., Vawter, M. P., Thompson, R. C., Turner, C. A., Bunney, W. E., Myers, R. M., Barchas, J. D., Schatzberg, A. F., Watson, S. J., and Akil, H. (2016). INFERENCE OF CELL TYPE COMPOSITION FROM HUMAN BRAIN TRANSCRIPTOMIC DATASETS ILLUMINATES THE EFFECTS OF AGE, MANNER OF DEATH, DISSECTION, AND PSYCHIATRIC DIAGNOSIS. *bioRxiv*.

Horowitz, A., Stegmann, K. A., and Riley, E. M. (2012). Activation of natural killer cells during microbial infections. *Frontiers in Immunology*, **2**(JAN), 1–13.

Irizarry, R. A., Hobbs, B., Collin, F., BeazerBarclay, Y. D., Antonellis, K. J., Scherf, U., and Speed, T. P. (2003). Exploration, normalization, and summaries of high density oligonucleotide array probe level data. *Biostatistics*, **4**(2), 249–264.

Kuhn, A., Thu, D., Waldvogel, H. J., Faull, R. L. M., and Luthi-Carter, R. (2011). Population-specific expression analysis (PSEA) reveals molecular changepcts restrict to markers, s in diseased brain. *Nature methods*, **8**(11), 945–7.

Li, B., Severson, E., Pignon, J.-C., Zhao, H., Li, T., Novak, J., Jiang, P., Shen, H., Aster, J. C., Rodig, S., Signoretti, S., Liu, J. S., and Liu, X. S. (2016). Comprehensive analyses of tumor immunity: implications for cancer immunotherapy. *Genome Biology*, **17**(1), 174.

Lin, C.-H., Lee, J. K., and LaBarge, M. A. (2012). Fabrication and Use of MicroEnvironment microArrays (MEArrays). *Journal of Visualized Experiments*, (68), 1–7.

Linsley, P. S., Speake, C., Whalen, E., and Chaussabel, D. (2014). Copy Number Loss of the Interferon Gene Cluster in Melanomas Is Linked to Reduced T Cell Infiltrate and Poor Patient Prognosis. *PLoS ONE*, **9**(10), e109760.

Liu, R., Holik, A. Z., Su, S., Jansz, N., Chen, K., Leong, S., Blewitt, M. E., Smyth, G. K., and Ritchie, M. E. (2015). Why weight ? Modelling sample and observational level variability improves power in RNA-seq analyses. **43**(15).

Lu, P., Nakorchevskiy, A., and Marcotte, E. M. (2003). Expression deconvolution: a reinterpretation of DNA microarray data reveals dynamic changes in cell populations. *Proceedings of the National Academy of Sciences of the United States of America*, **100**(18), 10370–5.

MAQC (2006). The MicroArray Quality Control (MAQC) project shows inter- and intraplatform reproducibility of gene expression measurements. **24**(9), 1151–1161.

Marusyk, A. and Polyak, K. (2011). Tumor heterogeneity: causes and consequences. *Biochim Biophys Acta*, **1805**(1), 1–28.

Mohammadi, S., Zuckerman, N., Goldsmith, A., and Grama, A. (2015). A Critical Survey of Deconvolution Methods for Separating cell-types in Complex Tissues. *arXiv*, **X**(X), 1–20.

Newman, A. M., Chih Long Liu, Michael R. Green, Andrew J. Gentles, W. F., Yue Xu, C. D. H., Diehn, M., and Alizadeh, A. A. (2015). Robust enumeration of cell subsets from tissue expression profiles. *Nat Methods.*, **12**(5), 193–201.

Parsons, J., Munro, S., Pine, P. S., Mcdaniel, J., Mehaffey, M., and Salit, M. (2015). Using mixtures of biological samples as process controls for RNA-sequencing experiments. *BMC Genomics*, pages 1–13.

Qiao, W., Quon, G., Csaszar, E., Yu, M., Morris, Q., and Zandstra, P. W. (2012). PERT: A Method for Expression Deconvolution of Human Blood Samples from Varied Microenvironmental and Developmental Conditions. *PLoS Computational Biology*, **8**(12).

Repsilber, D., Kern, S., Telaar, A., Walzl, G., Black, G. F., Selbig, J., Parida, S. K., Kaufmann, S. H. E., and Jacobsen, M. (2010). Biomarker discovery in heterogeneous tissue samples -taking the in-silico deconfounding approach. *BMC bioinformatics*, **11**, 27.

SEQC Consortium (2015). HHS Public Access. **32**(9), 903–914.

Shen-Orr, S. S., Tibshirani, R., Khatri, P., Bodian, D. L., Staedtler, F., Perry, N. M., Hastie, T., Sarwal, M. M., Davis, M. M., and Butte, A. J. (2010). Cell type-specific gene expression differences in complex tissues. *Nat Methods*, **7**(4), 287–289.

Teti, A. (1992). Regulation of Cellular Functions by Extracellular Matrix. Technical report.

Valencia, A., Racle, J., de Jonge, K., Baumgaertner, P., Speiser, D. E., and Gfeller, D. (2017). Simultaneous enumeration of cancer and immune cell types from bulk tumor gene expression data.

Venet, D., Pecasse, F., Maenhaut, C., and Bersini, H. (2001). Separation of samples into their constituents using gene expression data. *Bioinformatics*, **17**(Suppl 1), S279–S287.

Wang, M., Master, S. R., and Chodosh, L. a. (2006). Computational expression deconvolution in a complex mammalian organ. *BMC bioinformatics*, **7**, 328.

Wang, N., Hoffman, E. P., Chen, L., Chen, L., Zhang, Z., Liu, C., Yu, G., Herrington, D. M., Clarke, R., and Wang, Y. (2016). Mathematical modelling of transcriptional heterogeneity identifies novel markers and subpopulations in complex tissues. *Scientific Reports*, **6**(1), 18909.

Watson, S. S., Dane, M., Chin, K., Jonas, O., Gray, J. W., and Korkola, J. E. (2018). Microenvironment-Mediated Mechanisms of Resistance to HER2 Inhibitors Differ between HER2+ Breast Cancer Subtypes. *Cell Systems*, **6**, 329–342.e6.

Zhong, Y., Wan, Y.-W., Pang, K., Chow, L. M. L., and Liu, Z. (2013). Digital sorting of complex tissues for cell type-specific gene expression profiles. *BMC bioinformatics*, **14**, 89.