Run a minimal deconvolution simulation

Sean Maden

2022-12-08

This vignette shows how to run minimal deconvolution simulations on some example synthetic data objects.

Run a minimal simulation

This section walks through how to set up, run, and analyze a deconvolution simulation series. The entire code to run the simulation is as follows:

```r
num.sim <- 1e3
marker1 <- c(1, 0)
marker2 <- c(0, 1)
lgv <- lapply(seq(num.sim),function(ii){list(marker1, marker2)})
size1 <- 1
size2 <- 100
lsv <- lapply(seq(num.sim), function(ii){c(size1, size2)})
prop1 <- seq(1e-3, 1, 1e-3)
prop2 <- rev(prop1)
lpv <- lapply(seq(num.sim), function(ii){c(prop1[ii], prop2[ii])})
lres <- decon_analysis(lgv, lpv, lsv)
```

Simulation setup

```r
num.sim <- 1e3
```

We have set the number of simulations to 1000. We need to define the core deconvolution objects to run the simulation. These are the following 3 list variables:

- `lgv` : Marker signals. These are used to calculate Z. We can assign these as follows:

```r
marker1 <- c(1, 0)
marker2 <- c(0, 1)
lgv <- lapply(seq(num.sim),function(ii){list(marker1, marker2)})
```

- `lsv` : Size factors. These are used to transform Z and also to make the Y pseudo-bulked sample. To assign these values, use:

```
size1 <- 1
size2 <- 100
lsv <- lapply(seq(num.sim), function(ii){c(size1, size2)})
```

- `lpv` : True prediction values. These are used to make the Y pseudo-bulked sample and to compare predictions returned from deconvolution. We can vary complementary proportions for 2 types using:

```
prop1 <- seq(1e-3, 1, 1e-3)
prop2 <- rev(prop1)
lpv <- lapply(seq(num.sim), function(ii){c(prop1[ii], prop2[ii])})
```

Run simulations

Now run simulations and store the results as `lres`.

```
lres <- decon_analysis(lgv, lpv, lsv)
```

```
## Loading required package: ggplot2
```

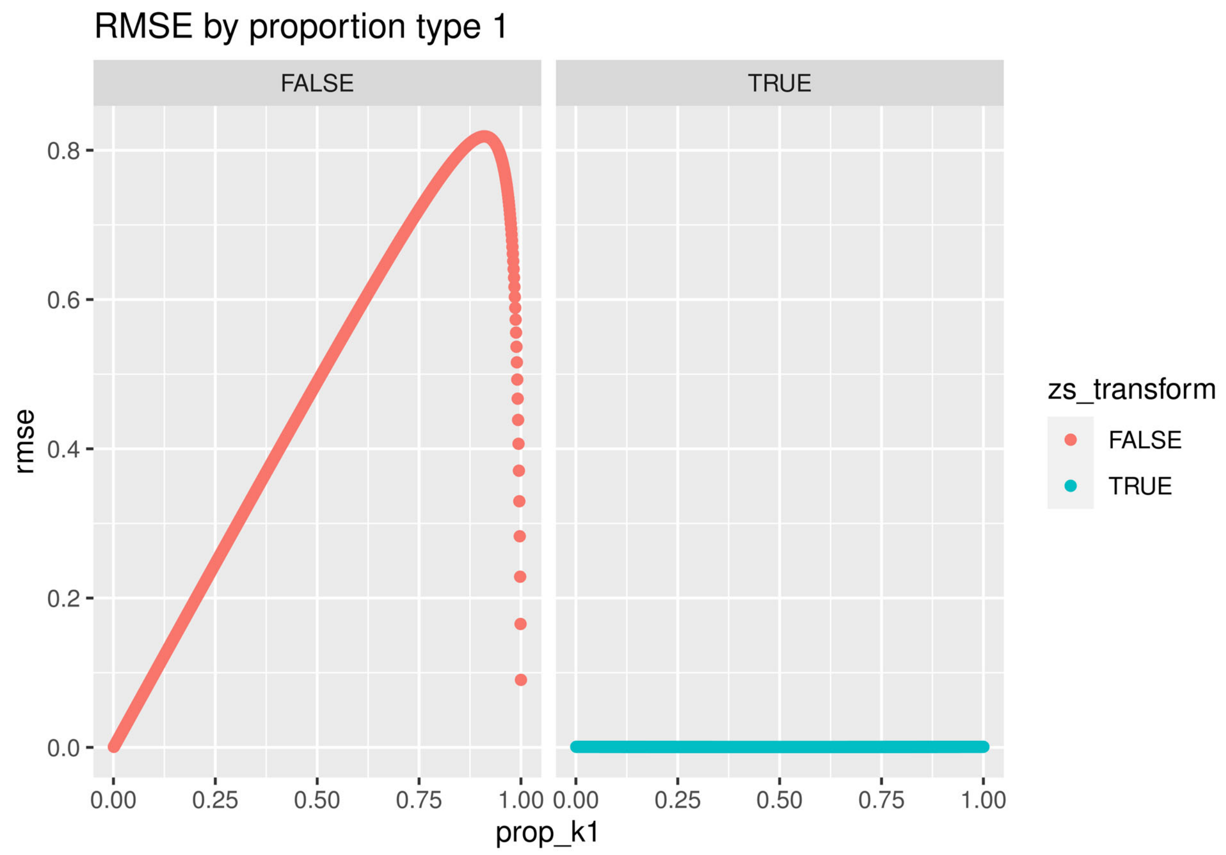
Analyze results

We access the plot objects from `lres`.

```
lgg <- lres$lgg
```

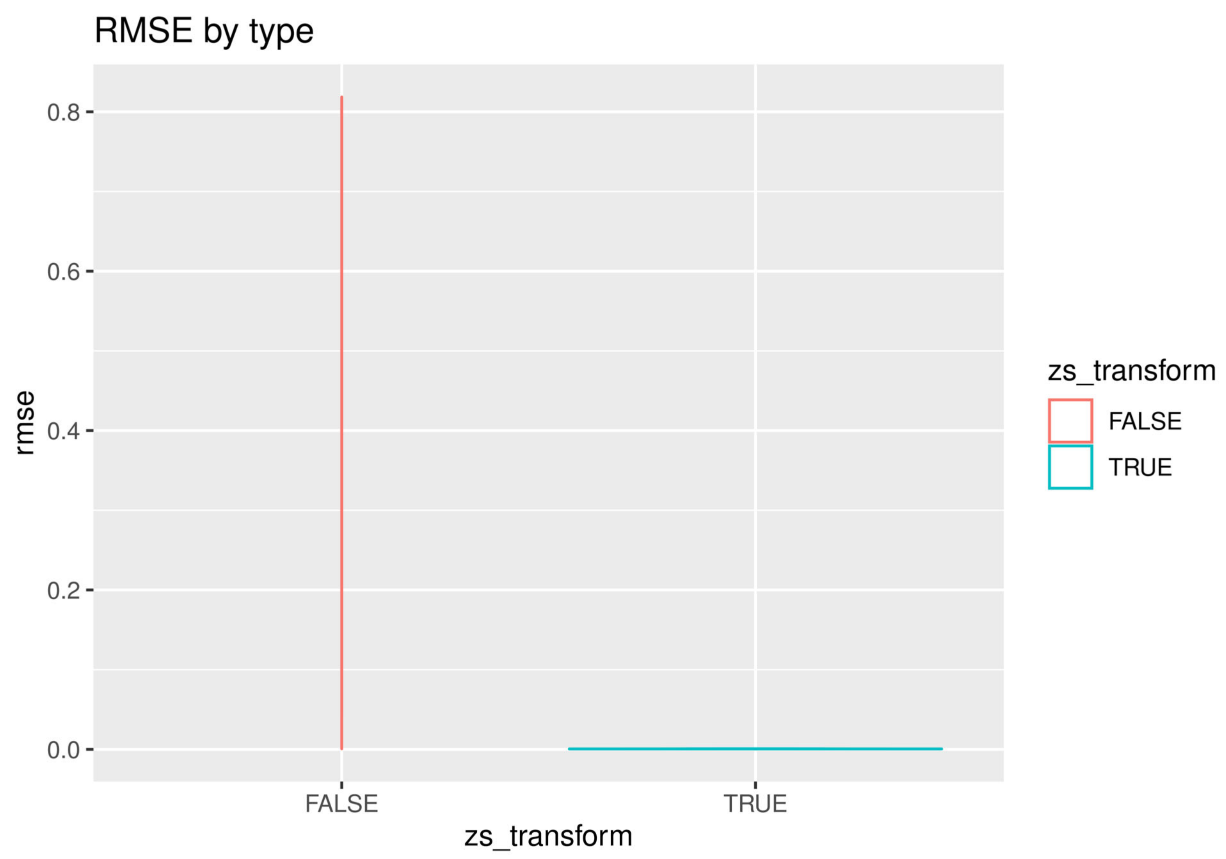
We can view the scatter plots of NNLS-predicted proportions for type 1 (x-axis) by the root mean squared error (RMSE), grouped on whether the S-transform was first applied to the Z reference, using:

```
lgg$ggpt1
```



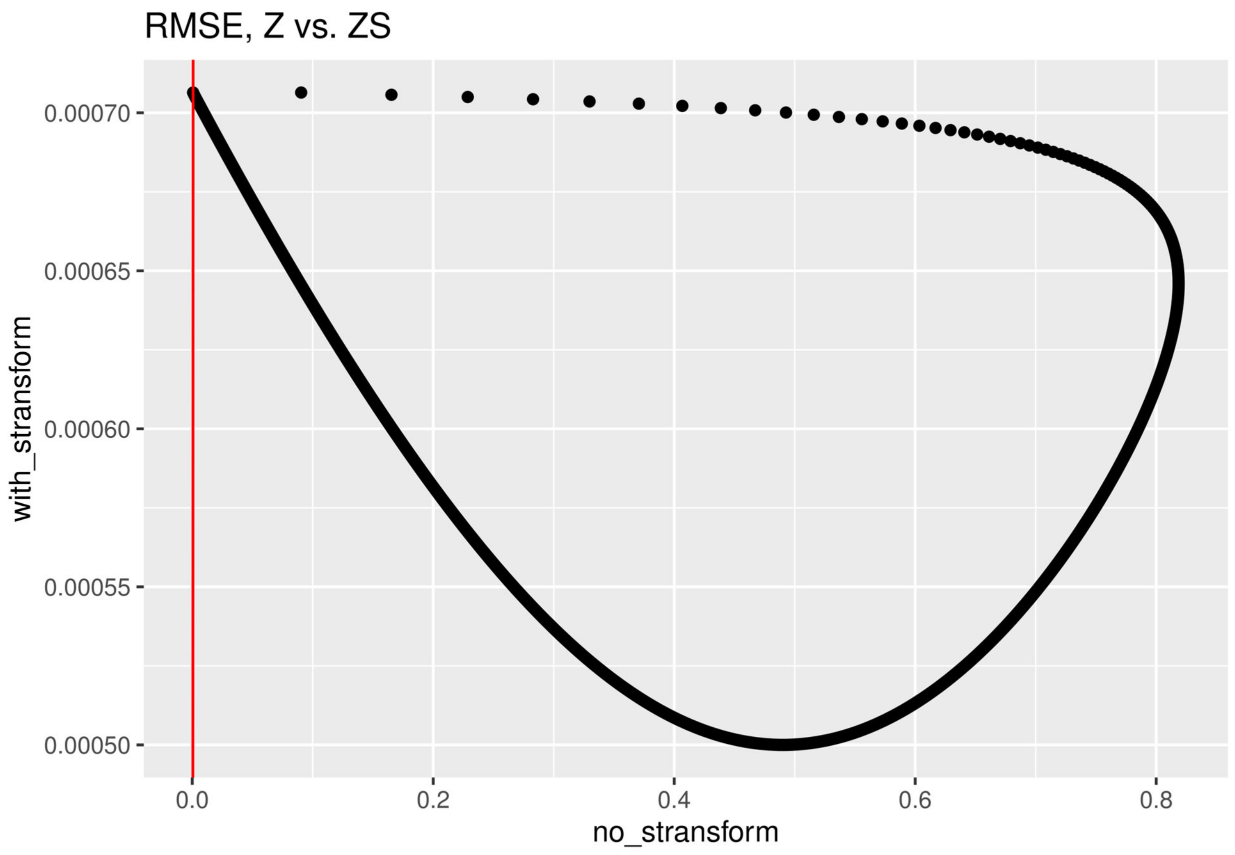
We can view the violin plots of RMSE grouped on S transformation status using:

```
lgg$ggvp
```



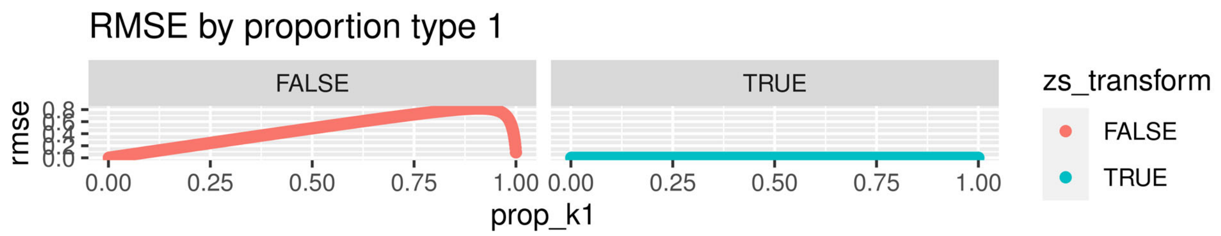
Finally, we can view the scatterplot of the RMSEs for the non-transformed data (x-axis) versus the S-tranformed data (y-axis), with a red reference line (y-intercept = 0, slope = 1), using:

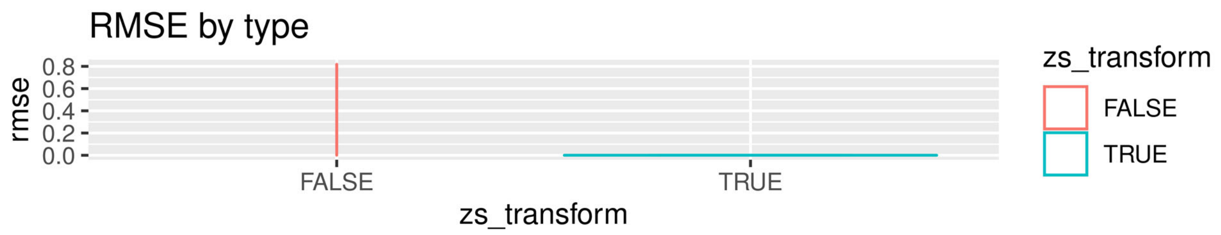
```
lgg$ggpt2
```

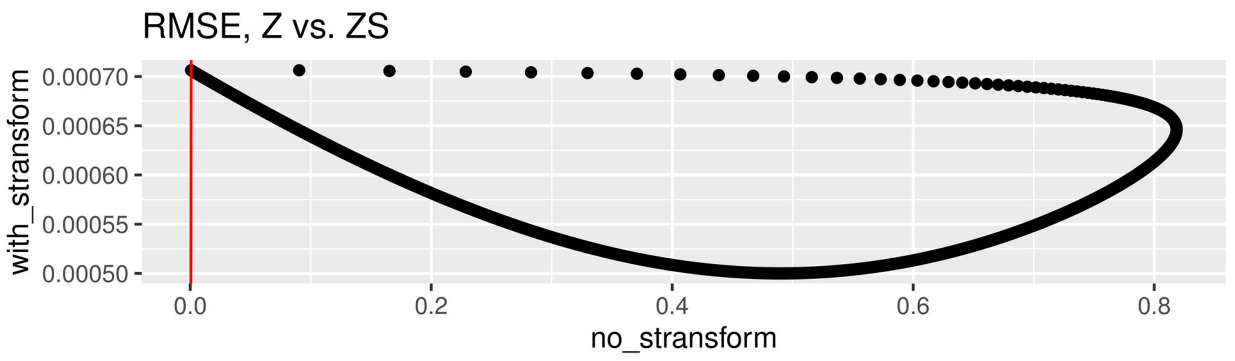


Finally, we can arrange these plots together using `gridExtra::grid.arrange()`.

```
grid.arrange(lgg$ggpt1, lgg$ggvp, lgg$ggpt2,
             layout_matrix = matrix(c(rep(1, 8), rep(2, 8), rep(3, 12)),
                                    ncol = 2, byrow = T))
```







Conclusions

This vignette showed how to perform a minimal simulation and analyze the results.

For general information about the `lute` R package, see the User's Guide.

For additional simulation examples, see `size_factor_experiments.Rmd` vignette.