pseudobulk_experiment_stransform

Sean Maden

2022-10-10

Overview

This vignette investigates the impact of performing transformations at the cell type level. Specifically, we apply either static or randomized transformations on individual cell types based on quantification of cells from independent analyses. We hope that by applying cell quantification factors to cell type signature data prior to deconvolution would improve the robustness of deconvolution across cells having large differences in cell abundances and available total expression (e.g. total reads).

Get s-transformed signature matrices

We wish to evaluate the impact of performing transformations on the signature matrix data.

To do this, we need the transformation values and an outline of the experiment design.

Load the z-table series

Load the preprocessed signature matrix table.

```
lz.fname <- "lz_s-rescale-k4_dlpfc-ro1.rda"
lz.fpath <- file.path(proj.dpath,
                      "outputs/03_test-stransform",
                      lz.fname)
lz <- get(load(file.path(here(), lz.fpath)))
```

Define the transformation distributions

The s transformations apply some factor to all cells of a given type, e.g. all rows in a column for some $Z^{G \times K}$ signature matrix.

When we define some vector of factors, e.g. `meanv`, we wish to applye to cell type 1 the value corresponding to `meanv[1]`, to cell type 2 the value corresponding to `meanv[2]`, etc.

Below, we use the following data as obtained from the paper Huuki-Myers et al. (2022). These indicate the mean and SD in sizes of four types of cells in brain tissues.

cell_type	mean	sd
Excit	6	2
Inhib	8	3
Oligo	2	1

cell_type	mean	sd
other	2	1

For static s, we only need the means (or medians), and this same value is applied to all cells of a given cell type.

```
meanv <- c(6, 2, 2, 8)
```

For randomized S, we additionally define some standard deviations. Now, random values from the same normal distribution are applied to all cells of a given cell type.

```
sdv <- c(2, 1, 1, 3)
```

Run transformations

We transform the signature matrix using `s_rescale()`.

First, use the static transformation method. We pass the vector of means to the argument `factorv`.

```
lz[["zs.stat"]] <- s_rescale(lz[["z.final"]], factorv = meanv)
```

Second, use the randomized transformation method. Note, we use arguments `meanv` and `sdv` to define a distribution for each cell type.

```
lz[["zs.rand"]] <- s_rescale(lz[["z.final"]], meanv = meanv, sdv = sdv)
```

Performing pseudobulking experiments

We can use pseudobulking to evaluate the impact of the S-transformation on deconvolution outcomes. Here, we specifically test the relative cell proportions as well as the relative scaling, or total counts, on deconvolution with `nnls`.

Experiment setup

We wish to evaluate deconvolution outcomes in a variety of conditions. For instance, can we recover neuronal cell proportions as well as oligodendrocyte cell proportions when equal numbers of cells are present? For each of these conditions, we also test outcomes for the 3 Z signature matrices defined above to see if either of the S-transformation strategies had any effect.

Get data objects

We need to load the `SummarizedExperiment` from which to sample cell-level expression counts.

```
sef.fname <- "sef-markers_ct-treg_stransform-expt_dlpfc-ro1.rda"
sef.fpath <- file.path(proj.dpath,
                       "outputs/03_test-stransform",
                       sef.fname)
sef <- get(load(file.path(here(), sef.fpath)))
```

We also need to subset the `lz` list to include just the Z signature matrices of interest.

```
# subset lz
lexpt <- lz[c("z.final", "zs.stat", "zs.rand")]
```

Experiment design

We wish to evaluate a series of scales and cell proportions.

```
# scalev <- seq(500, 5000, 500)
# cell.propv <- seq(0, 1000, 225)
datv <- sample(10, 4*4, replace = T) + 1
```

Run experiments

```
# run experiment
pb.expt <- get_pb_experiment(lz = lexpt, sef = sef, datv = datv,
                             scale.range = 100:10000, ctvarname = "celltype.treg",
                             plot.results = FALSE, method.str = "nnls",
                             seed.num = 2)
```

```
## Loading required package: SummarizedExperiment
```

```
## Loading required package: MatrixGenerics
```

```
## Loading required package: matrixStats
```

```
##
## Attaching package: 'MatrixGenerics'
```

```
## The following objects are masked from 'package:matrixStats':
##
##     colAlls, colAnyNAs, colAnys, colAvgsPerRowSet, colCollapse,
##     colCounts, colCummaxs, colCummins, colCumprods, colCumsums,
##     colDiffs, colIQRDiffs, colIQRs, colLogSumExps, colMadDiffs,
##     colMads, colMaxs, colMeans2, colMedians, colMins, colOrderStats,
##     colProds, colQuantiles, colRanges, colRanks, colSdDiffs, colSds,
##     colSums2, colTabulates, colVarDiffs, colVars, colWeightedMads,
##     colWeightedMeans, colWeightedMedians, colWeightedSds,
##     colWeightedVars, rowAlls, rowAnyNAs, rowAnys, rowAvgsPerColSet,
##     rowCollapse, rowCounts, rowCummaxs, rowCummins, rowCumprods,
##     rowCumsums, rowDiffs, rowIQRDiffs, rowIQRs, rowLogSumExps,
##     rowMadDiffs, rowMads, rowMaxs, rowMeans2, rowMedians, rowMins,
##     rowOrderStats, rowProds, rowQuantiles, rowRanges, rowRanks,
##     rowSdDiffs, rowSds, rowSums2, rowTabulates, rowVarDiffs, rowVars,
##     rowWeightedMads, rowWeightedMeans, rowWeightedMedians,
##     rowWeightedSds, rowWeightedVars
```

```
## Loading required package: GenomicRanges
```

```
## Loading required package: stats4
```

```
## Loading required package: BiocGenerics
```

```
##
## Attaching package: 'BiocGenerics'
```

```
## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs
```

```
## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, basename, cbind, colnames,
##     dirname, do.call, duplicated, eval, evalq, Filter, Find, get, grep,
##     grepl, intersect, is.unsorted, lapply, Map, mapply, match, mget,
##     order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,
##     rbind, Reduce, rownames, sapply, setdiff, sort, table, tapply,
##     union, unique, unsplit, which.max, which.min
```

```
## Loading required package: S4Vectors
```

```
##
## Attaching package: 'S4Vectors'
```

```
## The following objects are masked from 'package:base':
##
##     expand.grid, I, unname
```

```
## Loading required package: IRanges
```

```
##
## Attaching package: 'IRanges'
```

```
## The following object is masked from 'package:grDevices':
##
##     windows
```

```
## Loading required package: GenomeInfoDb
```

```
## Loading required package: Biobase
```

```
## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.
```

```
##
## Attaching package: 'Biobase'
```

```
## The following object is masked from 'package:MatrixGenerics':
##
##     rowMedians
```

```
## The following objects are masked from 'package:matrixStats':
##
##     anyMissing, rowMedians
```

```
## Loading required package: dplyr
```

```
##
## Attaching package: 'dplyr'
```

```
## The following object is masked from 'package:Biobase':
##
##     combine
```

```
## The following objects are masked from 'package:GenomicRanges':
##
##     intersect, setdiff, union
```

```
## The following object is masked from 'package:GenomeInfoDb':
##
##     intersect
```

```
## The following objects are masked from 'package:IRanges':
##
##     collapse, desc, intersect, setdiff, slice, union
```

```
## The following objects are masked from 'package:S4Vectors':
##
##     first, intersect, rename, setdiff, setequal, union
```

```
## The following objects are masked from 'package:BiocGenerics':
##
##     combine, intersect, setdiff, union
```

```
## The following object is masked from 'package:matrixStats':
##
##     count
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
## Loading required package: reshape2
```

```
## z.final
```

```
## running nnls...
```

```
## Loading required package: nnls
```

```
## zs.stat
```

```
## running nnls...
```

```
## zs.rand
```

```
## running nnls...
```

```
pb.expt$lpb$pb_report
```

```
##    cell_type sample_id    pi_est   pi_true      pi_diff  method scale
## 1      Inhib       j_1 0.0000000 0.1875000  0.187500000 z.final  4905
## 2      Oligo       j_1 0.0000000 0.3437500  0.343750000 z.final  4905
## 3      other       j_1 1.0000000 0.3437500 -0.656250000 z.final  4905
## 4      Excit       j_1 0.0000000 0.1250000  0.125000000 z.final  4905
## 5      Inhib       j_2 0.0000000 0.2000000  0.200000000 z.final  8564
## 6      Oligo       j_2 0.3902462 0.4000000  0.009753806 z.final  8564
## 7      other       j_2 0.0000000 0.2000000  0.200000000 z.final  8564
## 8      Excit       j_2 0.6097538 0.2000000 -0.409753806 z.final  8564
## 9      Inhib       j_3 1.0000000 0.4000000 -0.600000000 z.final  5568
## 10     Oligo       j_3 0.0000000 0.1000000  0.100000000 z.final  5568
## 11     other       j_3 0.0000000 0.4000000  0.400000000 z.final  5568
## 12     Excit       j_3 0.0000000 0.1000000  0.100000000 z.final  5568
## 13     Inhib       j_4 1.0000000 0.4347826 -0.565217391 z.final  3375
## 14     Oligo       j_4 0.0000000 0.1304348  0.130434783 z.final  3375
## 15     other       j_4 0.0000000 0.2173913  0.217391304 z.final  3375
## 16     Excit       j_4 0.0000000 0.2173913  0.217391304 z.final  3375
## 17     Inhib       j_1 0.0000000 0.1875000  0.187500000 zs.stat  4905
## 18     Oligo       j_1 0.0000000 0.3437500  0.343750000 zs.stat  4905
## 19     other       j_1 1.0000000 0.3437500 -0.656250000 zs.stat  4905
## 20     Excit       j_1 0.0000000 0.1250000  0.125000000 zs.stat  4905
## 21     Inhib       j_2 0.0000000 0.2000000  0.200000000 zs.stat  8564
## 22     Oligo       j_2 0.1379322 0.4000000  0.262067820 zs.stat  8564
## 23     other       j_2 0.0000000 0.2000000  0.200000000 zs.stat  8564
## 24     Excit       j_2 0.8620678 0.2000000 -0.662067820 zs.stat  8564
## 25     Inhib       j_3 1.0000000 0.4000000 -0.600000000 zs.stat  5568
## 26     Oligo       j_3 0.0000000 0.1000000  0.100000000 zs.stat  5568
## 27     other       j_3 0.0000000 0.4000000  0.400000000 zs.stat  5568
## 28     Excit       j_3 0.0000000 0.1000000  0.100000000 zs.stat  5568
## 29     Inhib       j_4 1.0000000 0.4347826 -0.565217391 zs.stat  3375
## 30     Oligo       j_4 0.0000000 0.1304348  0.130434783 zs.stat  3375
## 31     other       j_4 0.0000000 0.2173913  0.217391304 zs.stat  3375
## 32     Excit       j_4 0.0000000 0.2173913  0.217391304 zs.stat  3375
## 33     Inhib       j_1 0.0000000 0.1875000  0.187500000 zs.rand  4905
## 34     Oligo       j_1 0.0000000 0.3437500  0.343750000 zs.rand  4905
## 35     other       j_1 1.0000000 0.3437500 -0.656250000 zs.rand  4905
## 36     Excit       j_1 0.0000000 0.1250000  0.125000000 zs.rand  4905
## 37     Inhib       j_2 0.0000000 0.2000000  0.200000000 zs.rand  8564
## 38     Oligo       j_2 0.1059933 0.4000000  0.294006655 zs.rand  8564
## 39     other       j_2 0.0000000 0.2000000  0.200000000 zs.rand  8564
## 40     Excit       j_2 0.8940067 0.2000000 -0.694006655 zs.rand  8564
## 41     Inhib       j_3 1.0000000 0.4000000 -0.600000000 zs.rand  5568
## 42     Oligo       j_3 0.0000000 0.1000000  0.100000000 zs.rand  5568
## 43     other       j_3 0.0000000 0.4000000  0.400000000 zs.rand  5568
## 44     Excit       j_3 0.0000000 0.1000000  0.100000000 zs.rand  5568
## 45     Inhib       j_4 1.0000000 0.4347826 -0.565217391 zs.rand  3375
## 46     Oligo       j_4 0.0000000 0.1304348  0.130434783 zs.rand  3375
## 47     other       j_4 0.0000000 0.2173913  0.217391304 zs.rand  3375
## 48     Excit       j_4 0.0000000 0.2173913  0.217391304 zs.rand  3375
```

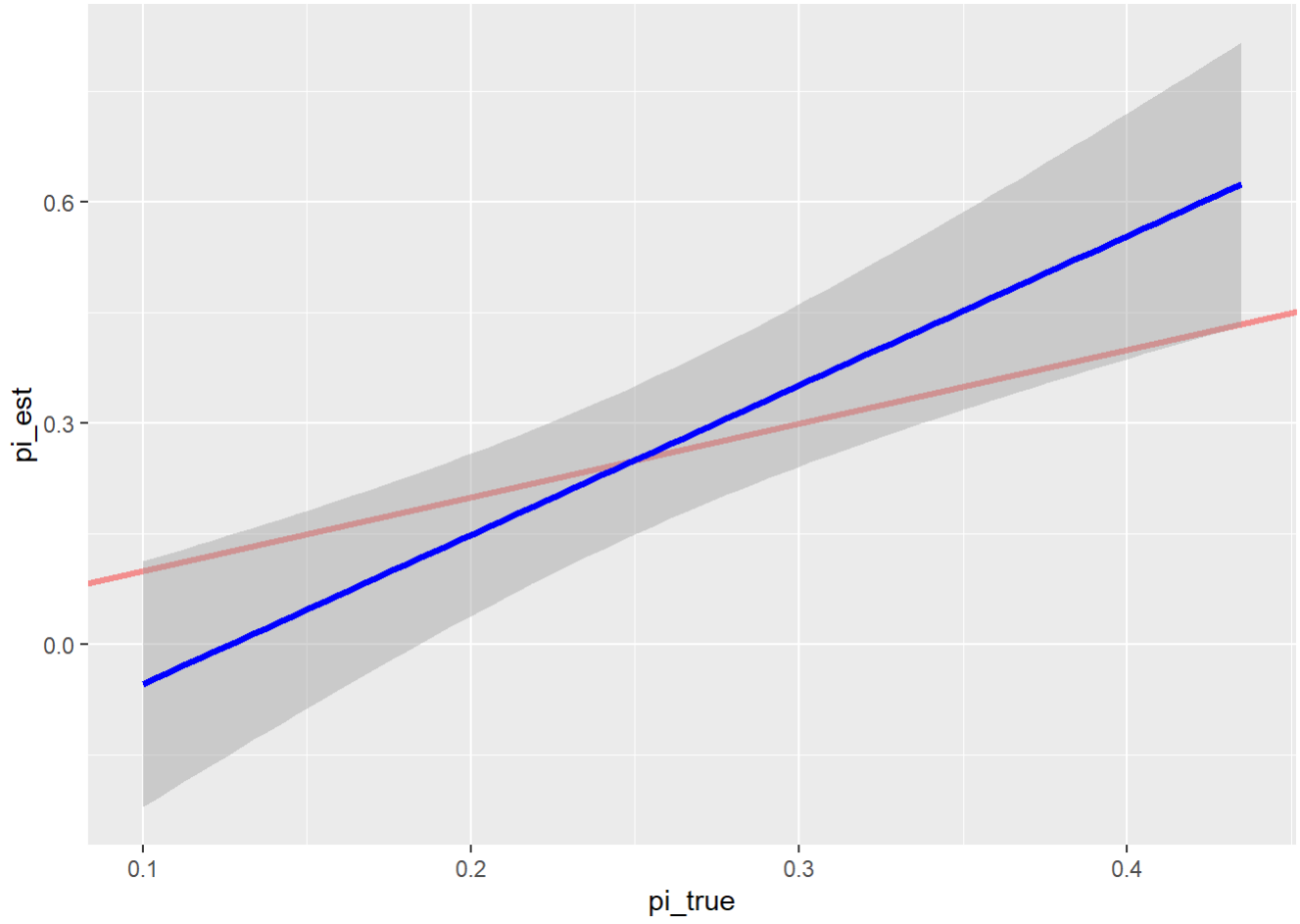
Results

Plotting pi_est versus pi_true

```
df.tall <- pb.expt$lpb$pb_report
alpha.value <- 0.4
```

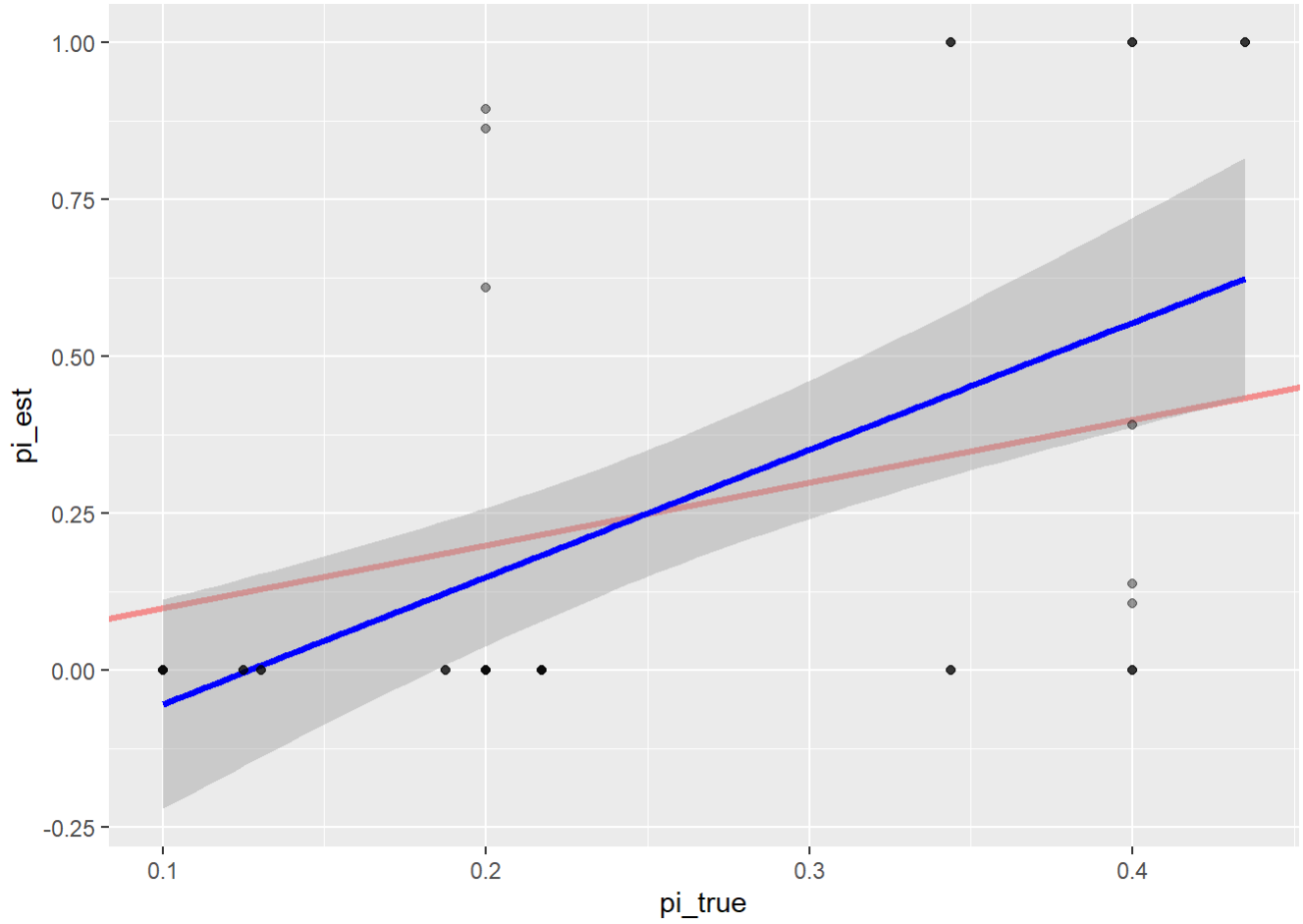
```
ggpt.main <- ggplot(df.tall, aes(x = pi_true, y = pi_est),
                      environment = environment()) +
    geom_abline(intercept = 0, slope = 1, color = "red", lwd = 1.2,
                alpha = alpha.value) +
    geom_smooth(method = "lm", color = "blue", lwd = 1.2,
                alpha = alpha.value)
ggpt.main
```

```
## `geom_smooth()` using formula 'y ~ x'
```



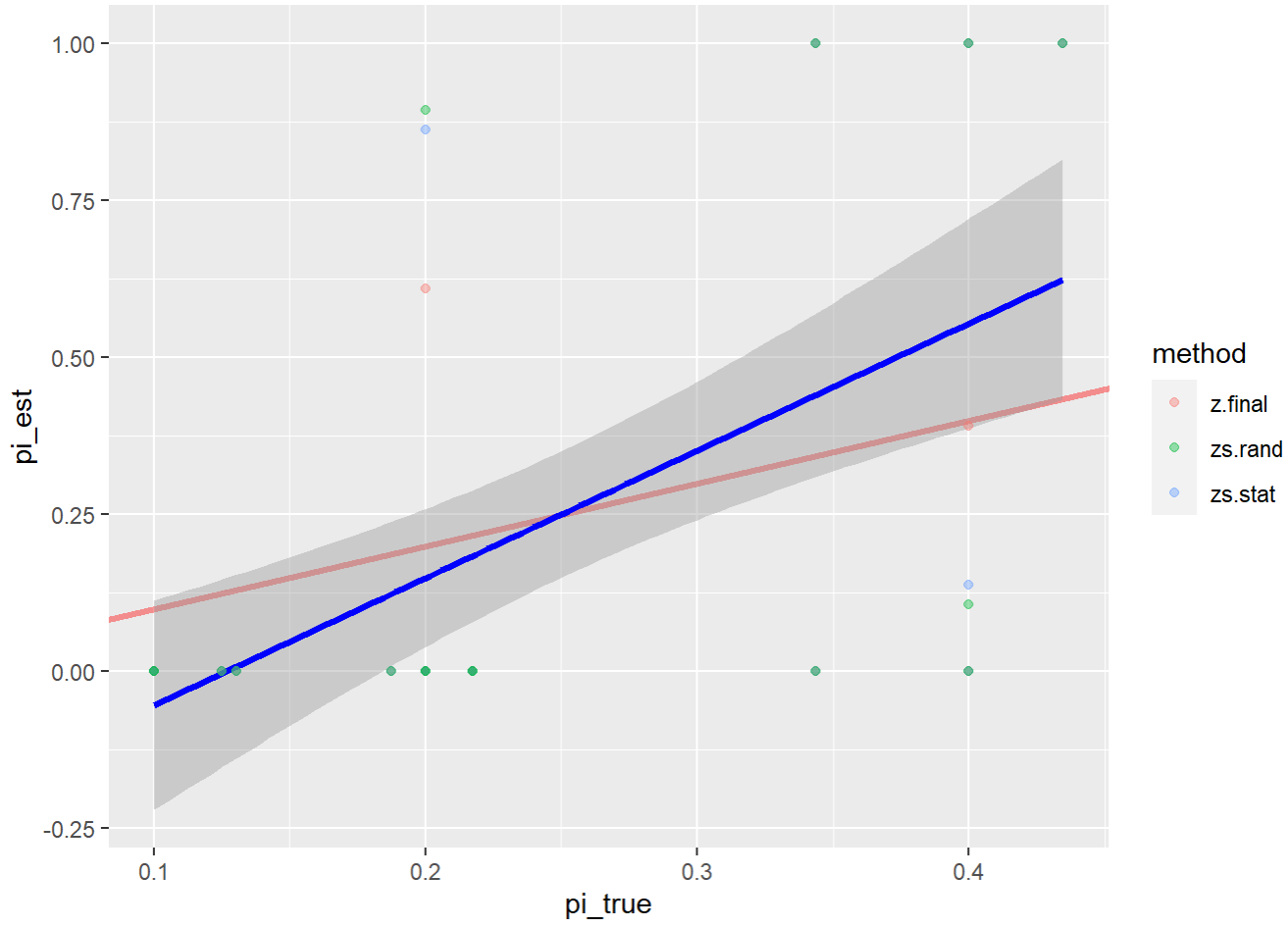

```
# label series
ggpt.all <- ggpt.main + geom_point(alpha = alpha.value)
ggpt.all
```

```
## `geom_smooth()` using formula 'y ~ x'
```



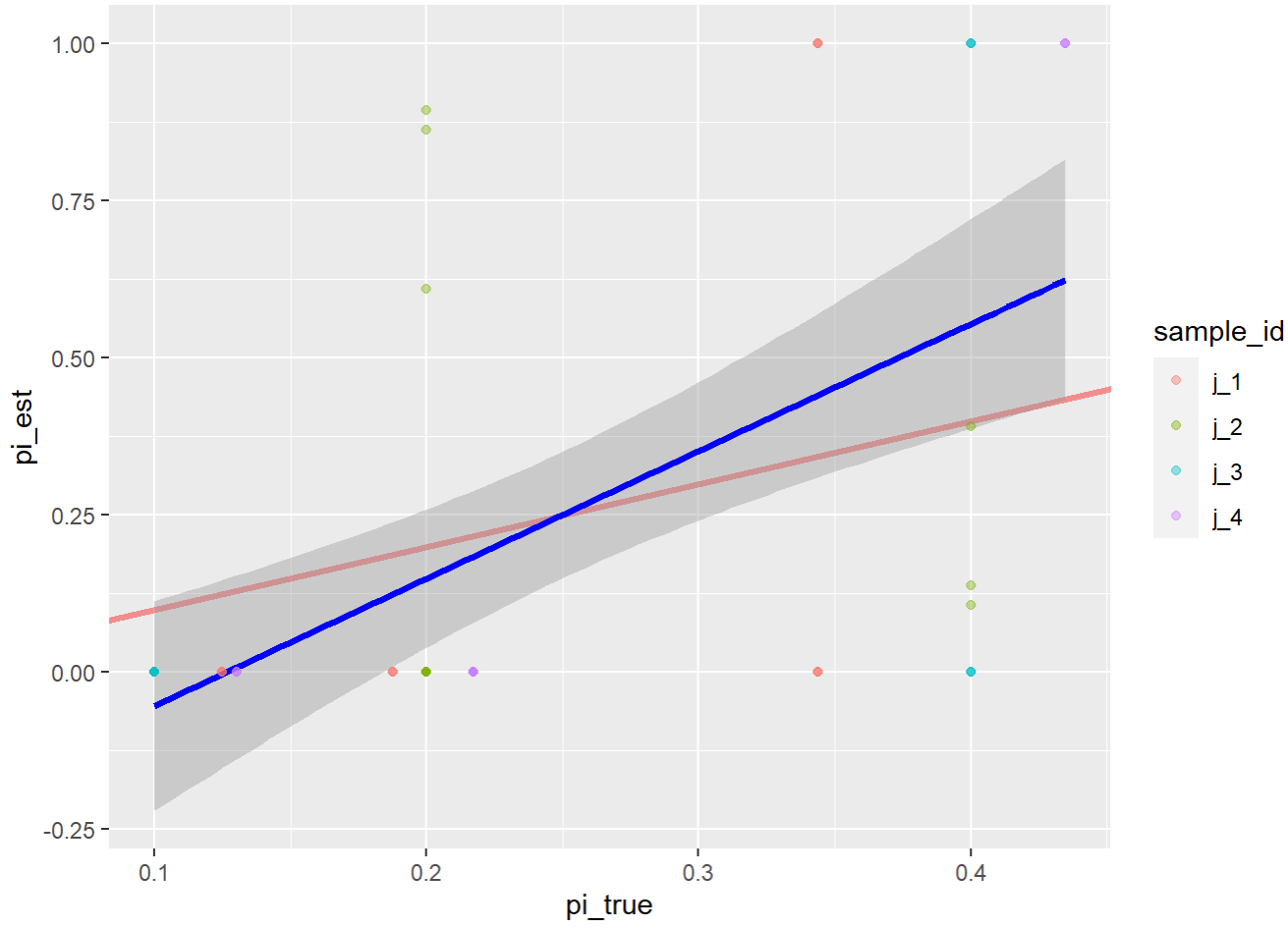

```
ggpt.all.method <- ggpt.main + geom_point(aes(color = method),
                                          alpha = alpha.value)
ggpt.all.method
```

```
## `geom_smooth()` using formula 'y ~ x'
```



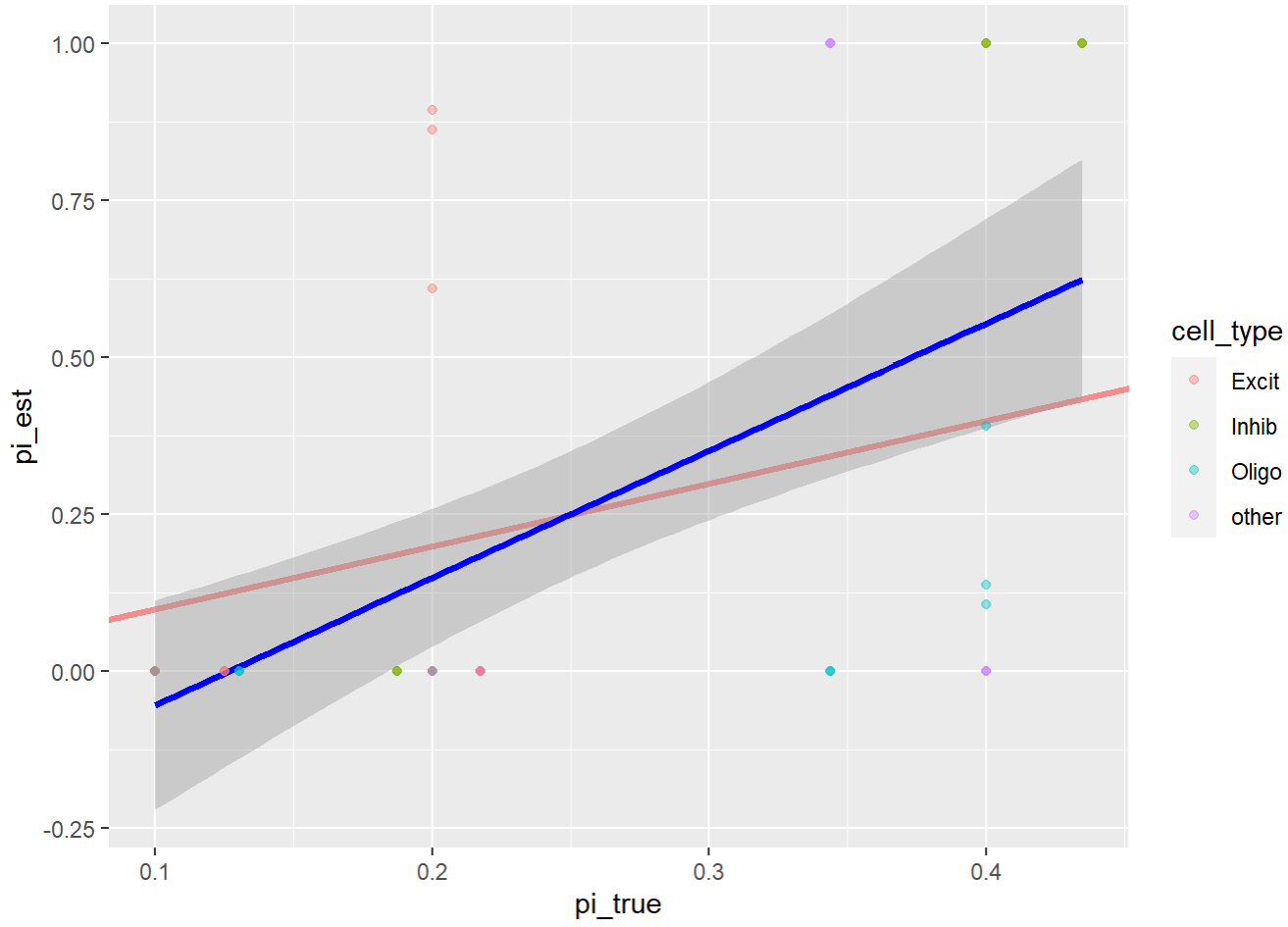

```
ggpt.all.sampleid <- ggpt.main +
  geom_point(aes(color = sample_id), alpha = alpha.value)
ggpt.all.sampleid
```

```
## `geom_smooth()` using formula 'y ~ x'
```



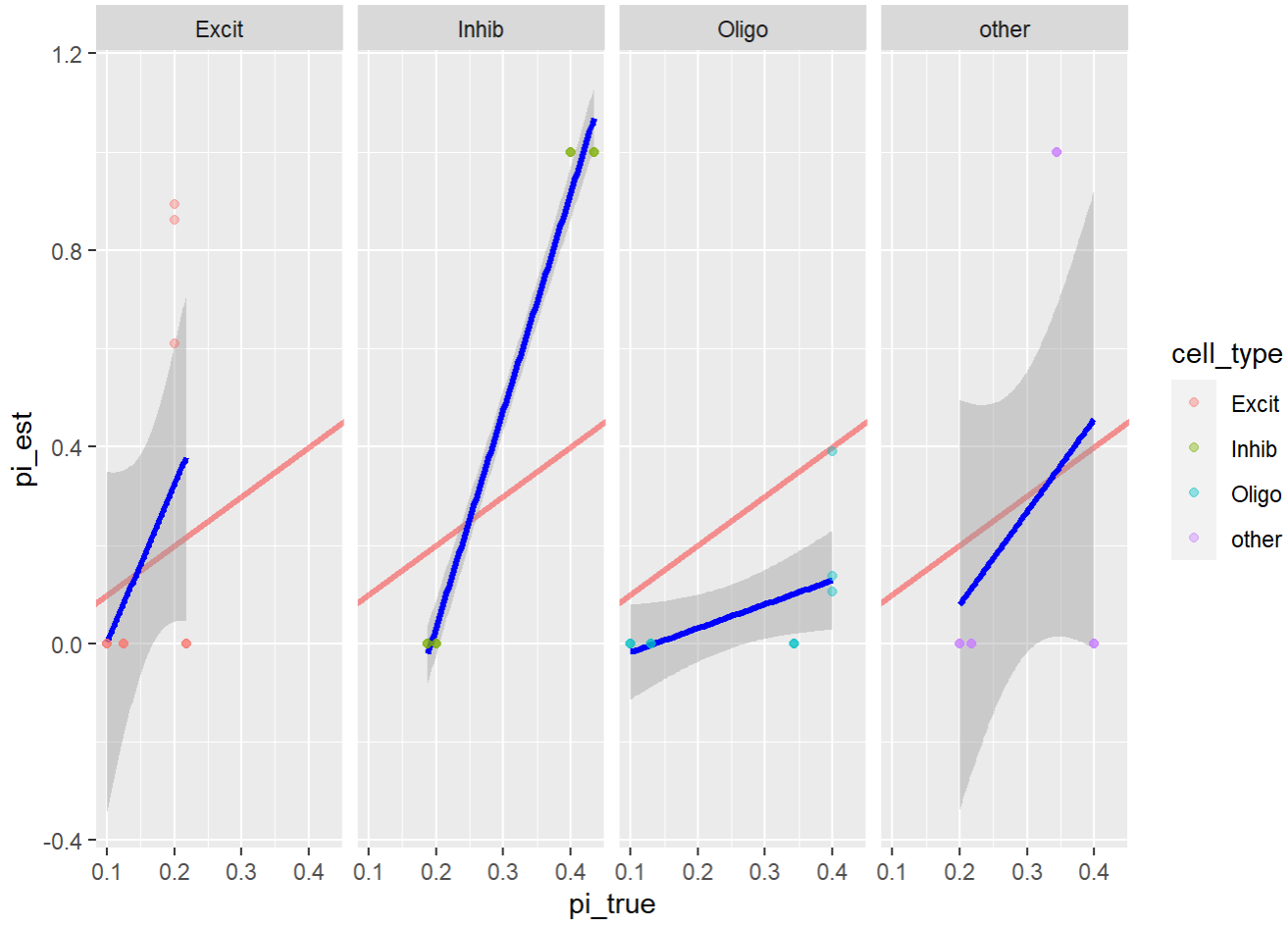

```
ggpt.all.col <- ggpt.main +
  geom_point(aes(color = cell_type), alpha = alpha.value)
ggpt.all.col
```

```
## `geom_smooth()` using formula 'y ~ x'
```



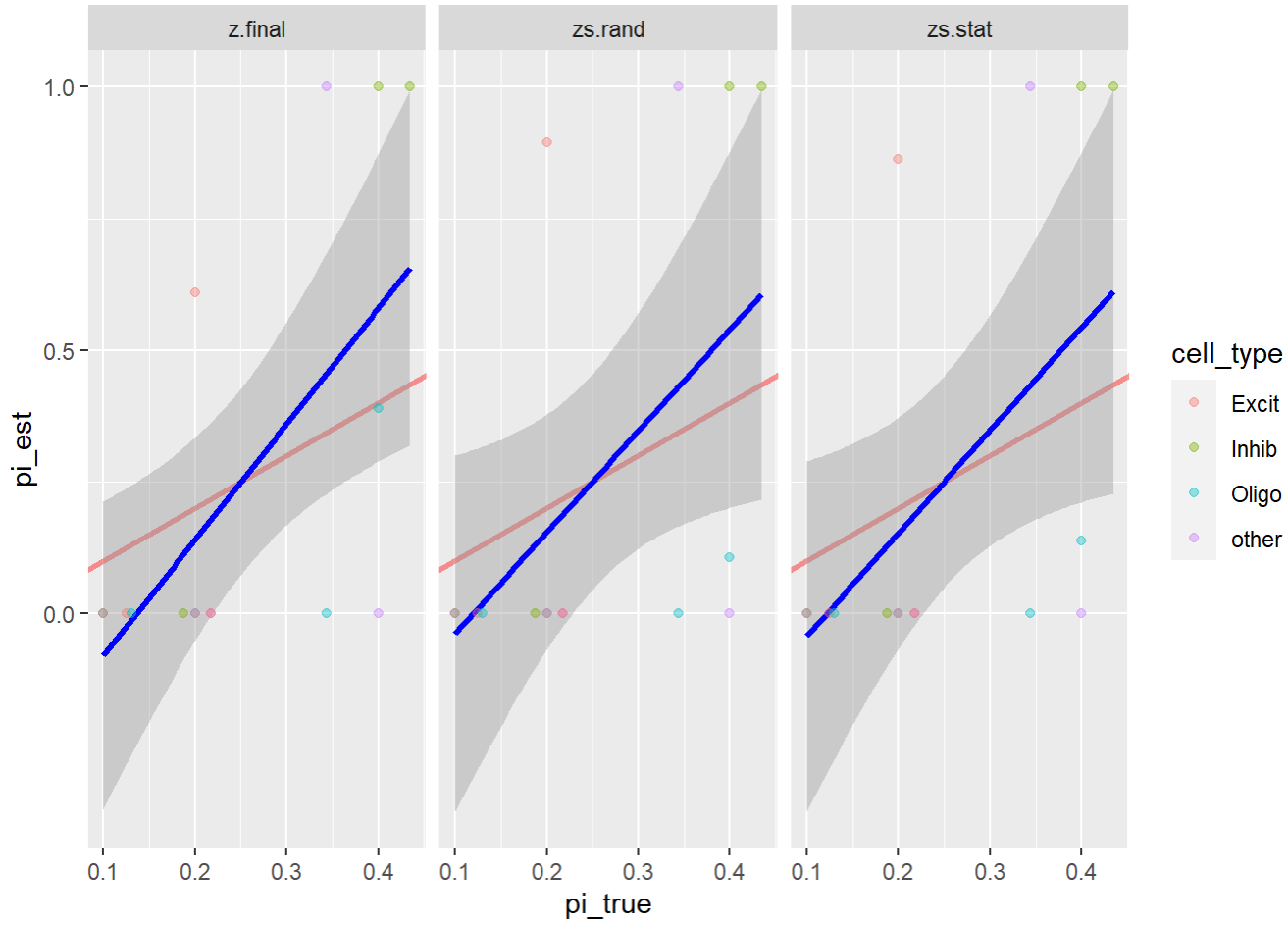

```
# get facet series
ggpt.all.celltype.facet <- ggpt.all.col + facet_wrap(vars(cell_type), nrow = 1)
ggpt.all.celltype.facet
```

```
## `geom_smooth()` using formula 'y ~ x'
```



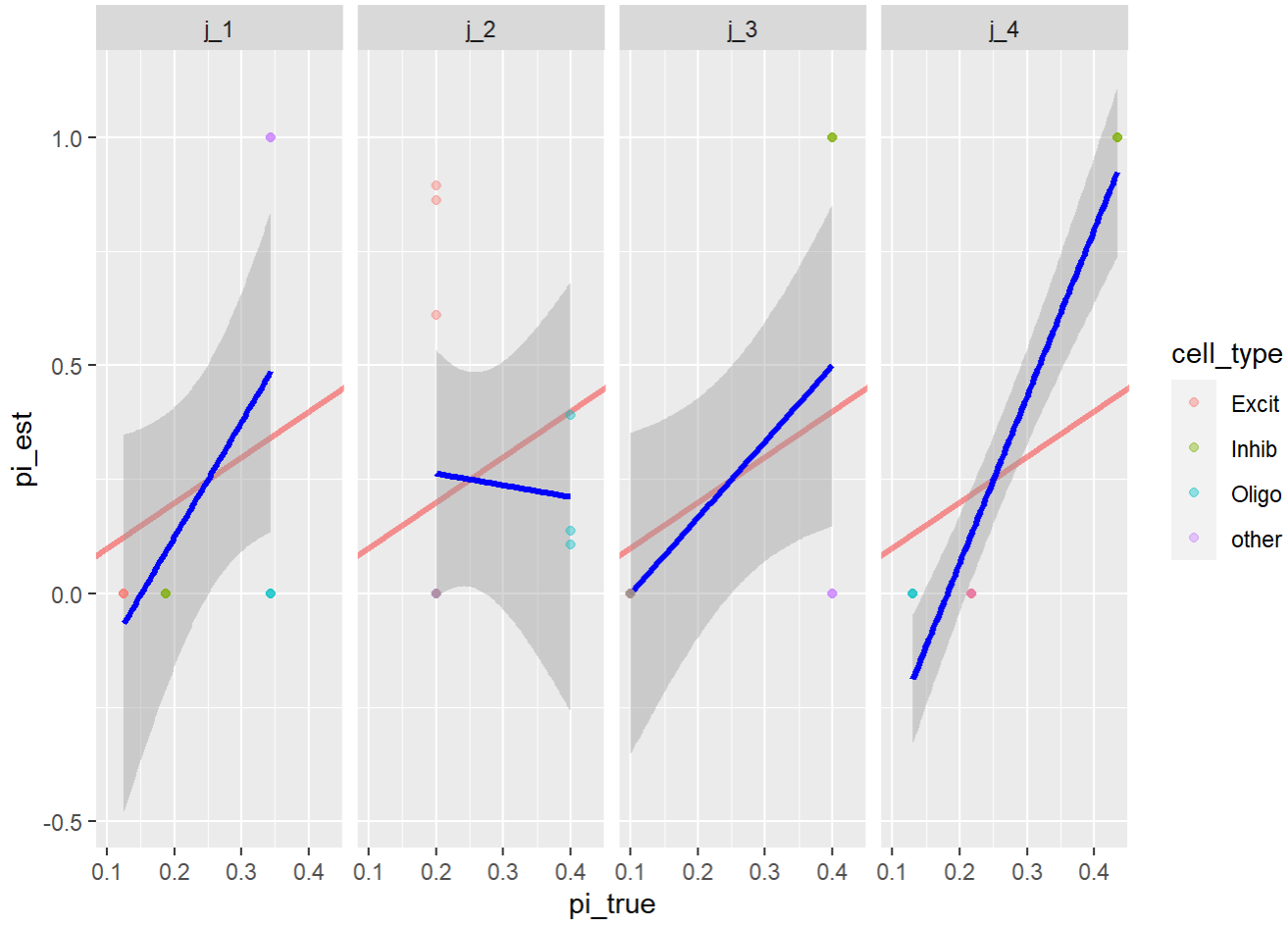

```
ggpt.all.method.facet <- ggpt.all.col +
    facet_wrap(~method, nrow = 1)
ggpt.all.method.facet
```

```
## `geom_smooth()` using formula 'y ~ x'
```



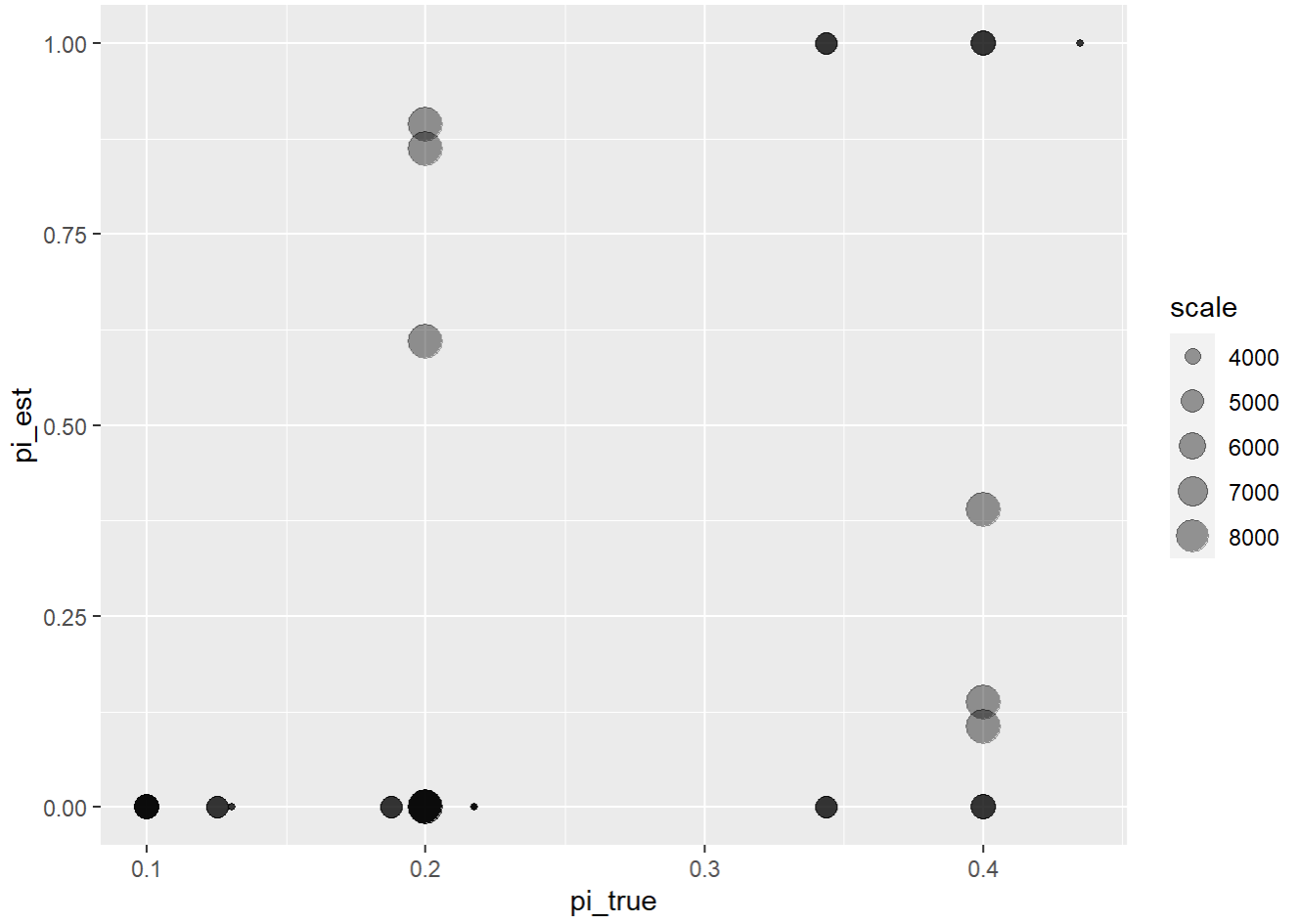

```
ggpt.all.sampleid.facet <- ggpt.all.col +
    facet_wrap(~sample_id, nrow = 1)
ggpt.all.sampleid.facet
```

```
## `geom_smooth()` using formula 'y ~ x'
```

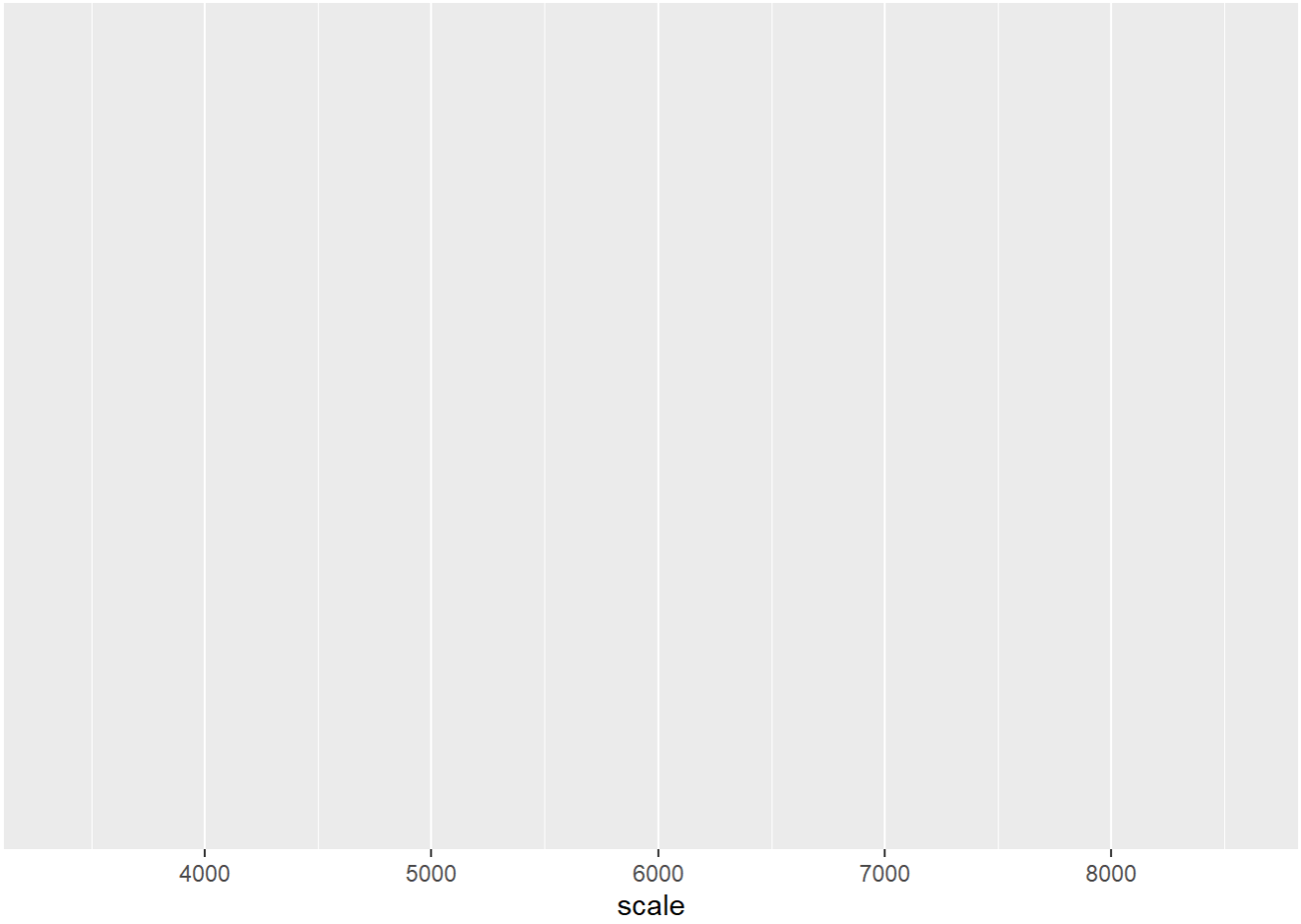


Plotting by scale/total counts

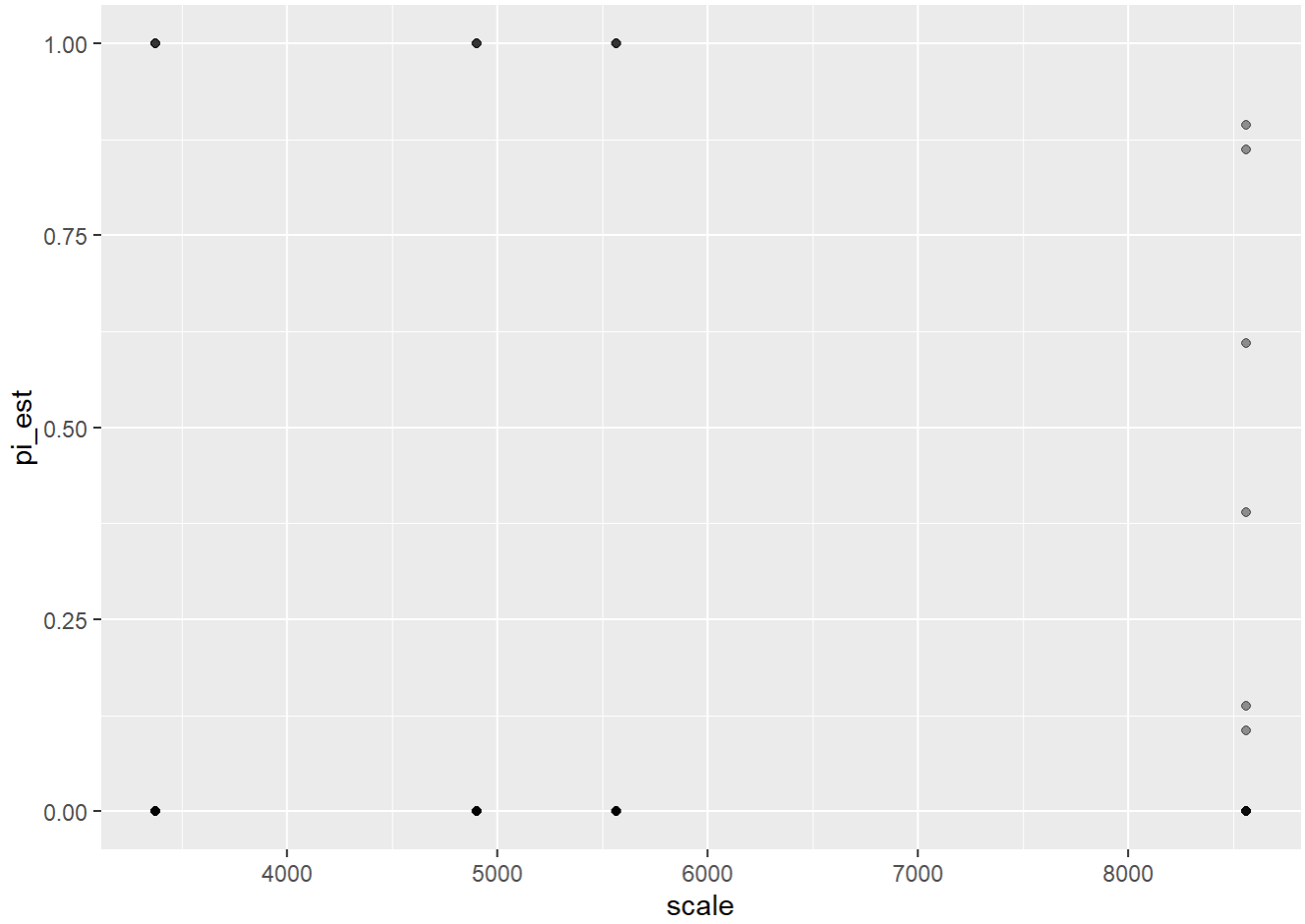
```
ggpt.scalesize <- ggplot(df.tall, aes(x = pi_true, y = pi_est, size = scale)) +
    geom_point(alpha = alpha.value)
ggpt.scalesize
```



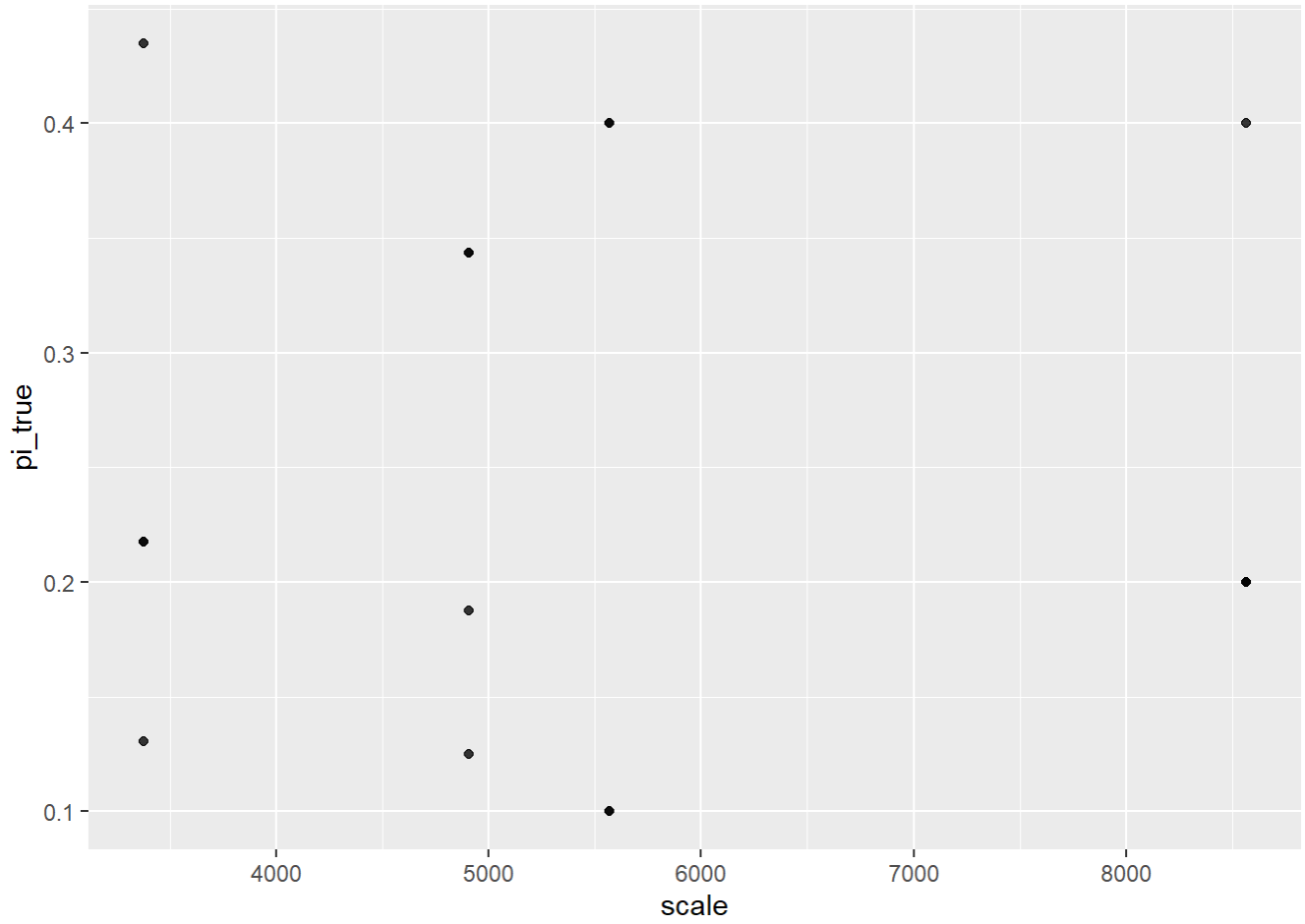

```
# get main template plot for series
ggpt.main <- ggplot(df.tall, aes(x = scale))
ggpt.main
```



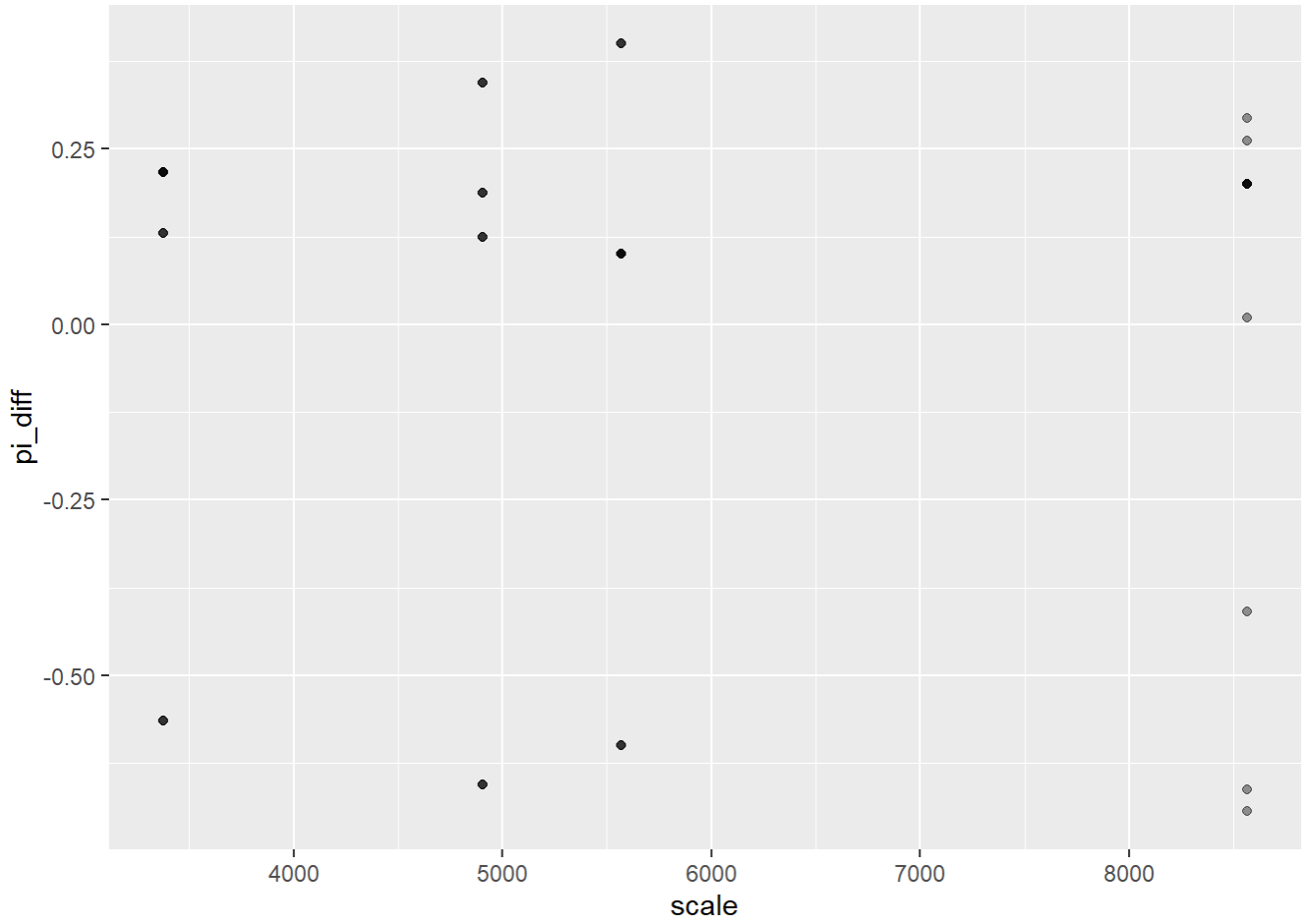

```
# get plot series
ggpt.main.piest <- ggpt.main + geom_point(aes(y = pi_est), alpha = alpha.value)
ggpt.main.piest
```



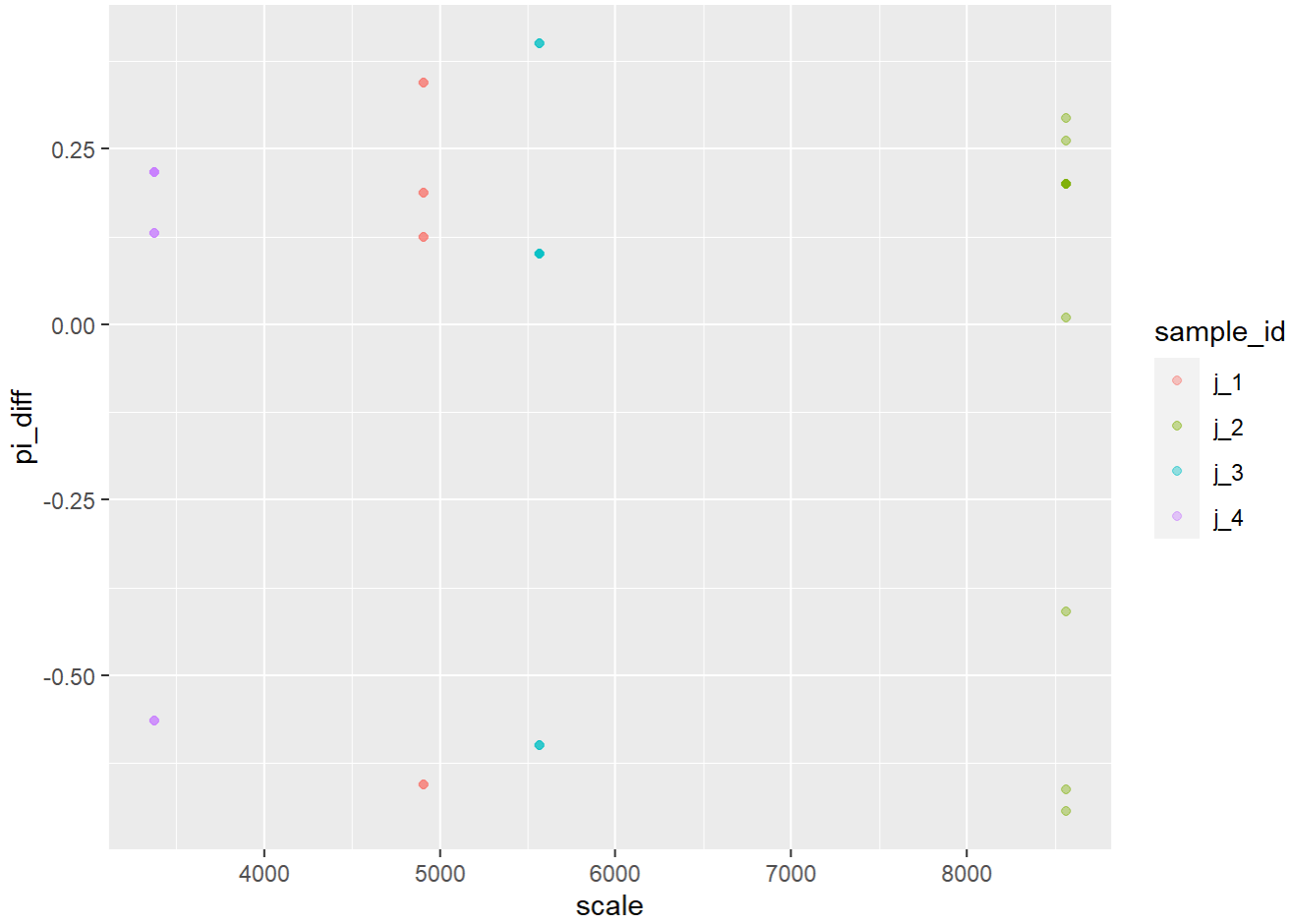

```
ggpt.main.pitrue <- ggpt.main + geom_point(aes(y = pi_true), alpha = alpha.value)
ggpt.main.pitrue
```



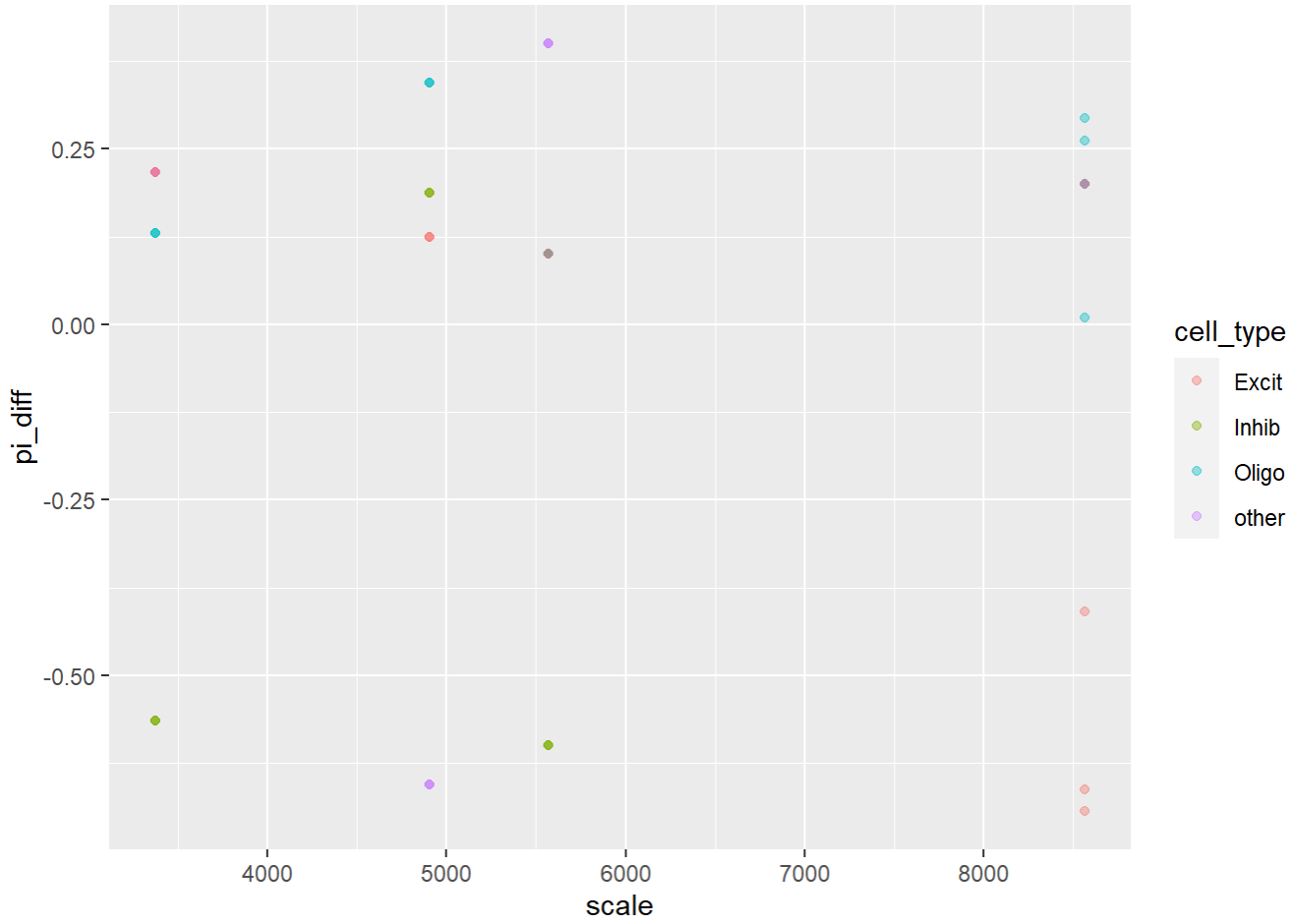

```
ggpt.main.pidiff <- ggpt.main + geom_point(aes(y = pi_diff), alpha = alpha.value)
ggpt.main.pidiff
```



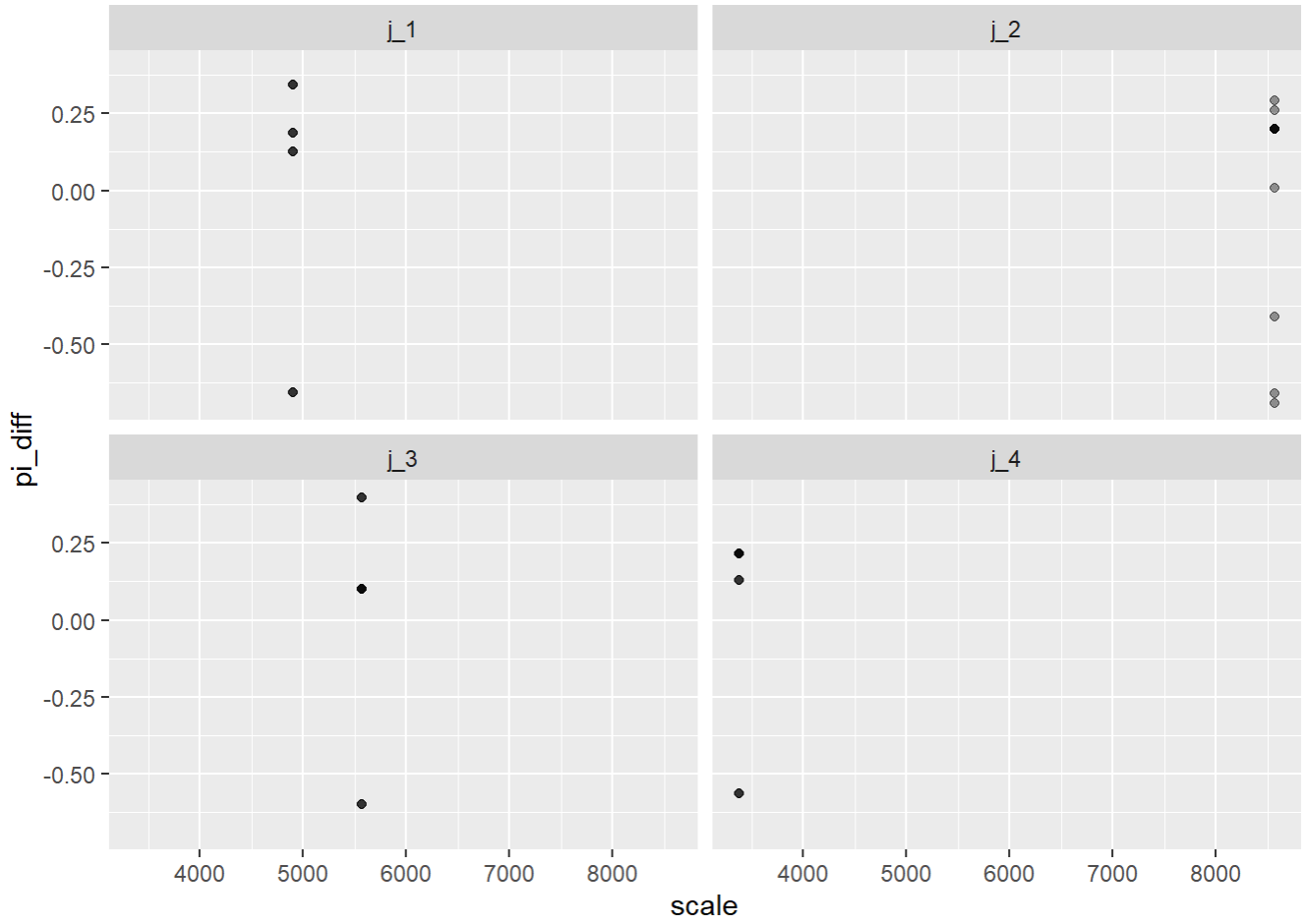

```
ggpt.main.pidiff.sampleid.col <- ggpt.main +
  geom_point(aes(y = pi_diff, color = sample_id), alpha = alpha.value)
ggpt.main.pidiff.sampleid.col
```



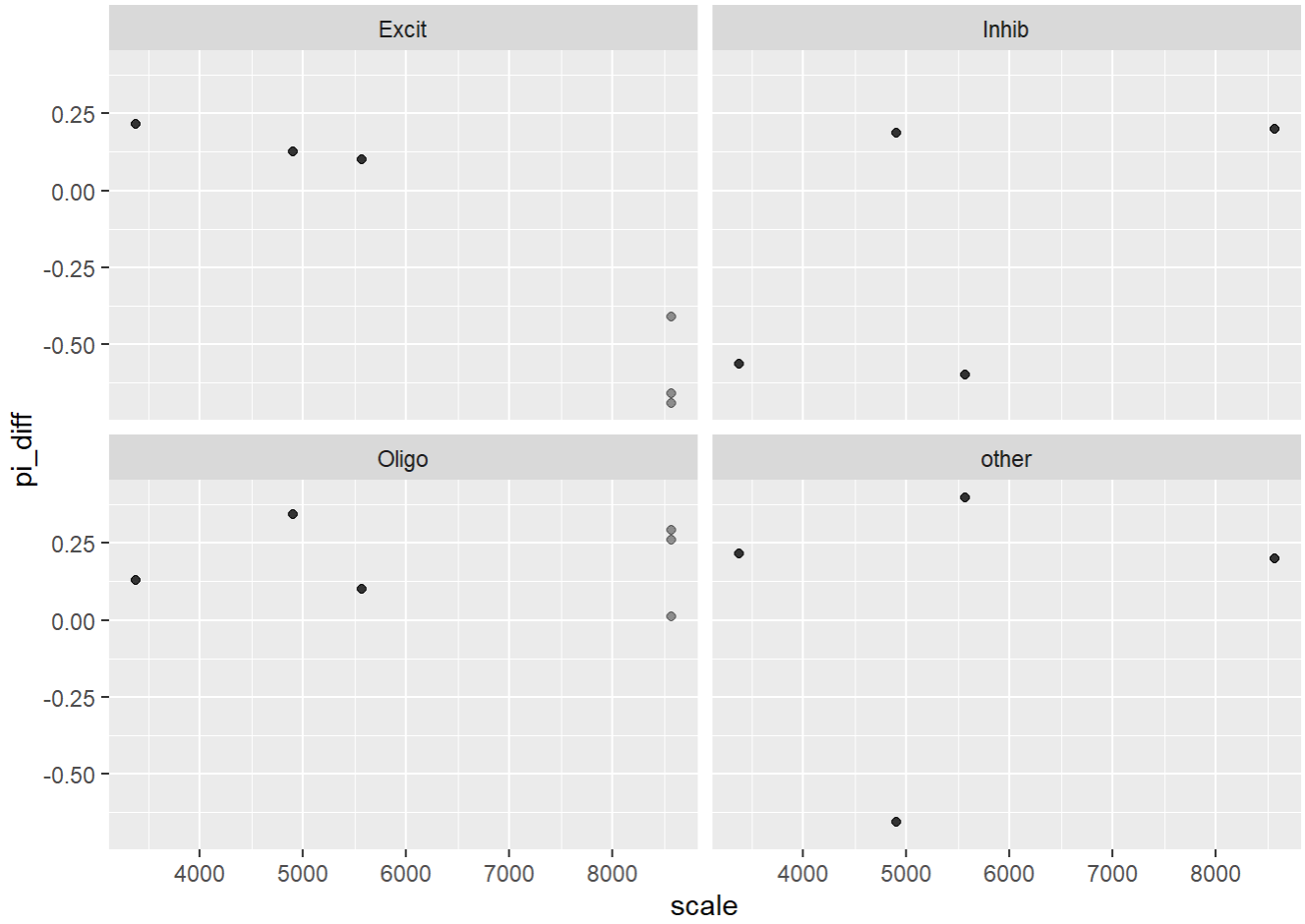

```
ggpt.main.pidiff.celltype.col <- ggpt.main +
  geom_point(aes(y = pi_diff, color = cell_type), alpha = alpha.value)
ggpt.main.pidiff.celltype.col
```



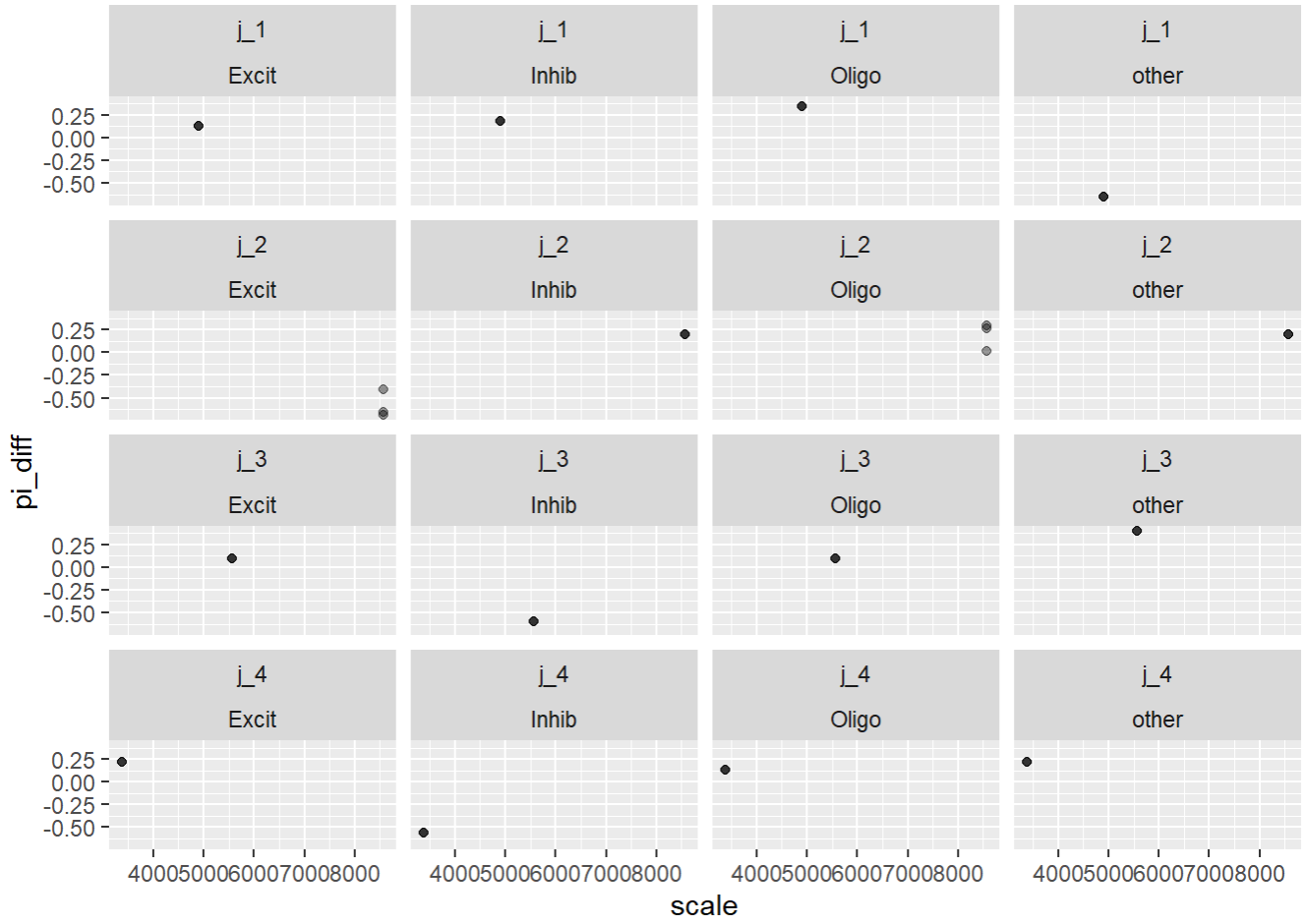

```
# get facets
ggpt.main.pidiff.sampleid.facet <- ggpt.main.pidiff + facet_wrap(~sample_id)
ggpt.main.pidiff.sampleid.facet
```




```
# get facets
ggpt.main.pidiff.celltype.facet <- ggpt.main.pidiff + facet_wrap(~cell_type)
ggpt.main.pidiff.celltype.facet
```




```
ggpt.main.pidiff.sampleid.celltype.facet <- ggpt.main.pidiff +
  facet_wrap(~sample_id+cell_type)
ggpt.main.pidiff.sampleid.celltype.facet
```



Huuki-Myers, Louise A., Kelsey D. Montgomery, Sang Ho Kwon, Stephanie C. Page, Stephanie C. Hicks, Kristen R. Maynard, and Leonardo Collado-Torres. 2022. “Data-Driven Identification of Total RNA Expression Genes (TREGs) for Estimation of RNA Abundance in Heterogeneous Cell Types.” bioRxiv. https://doi.org/10.1101/2022.04.28.489923 (https://doi.org/10.1101/2022.04.28.489923).