Gridsemble on Platinum Spike Dataset - Models on Subsets

Jenna Landy

2023-12-21

```r
source('PAPER_metrics_helpers.R')
load("PAPER_platinum_data.RData")
```

```r
# remove.packages('gridsemblefdr')
# library(devtools)
# devtools::install_github('jennalandy/gridsemblefdr')
library(gridsemblefdr)
```

```r
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v forcats   1.0.0     v readr     2.1.4
v ggplot2   3.4.4     v stringr   1.5.1
v lubridate 1.9.2     v tibble    3.2.1
v purrr     1.0.1     v tidyr     1.3.0
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
```

```r
library(ggplot2)
library(ggthemes)
library(patchwork)
library(locfdr)
```

```
color_list = list(
  "gridsemble" = "#E69F00",
  "locfdr" = "#D55E00",
  "fdrtool" = "#009E73",
  "qvalue" = "#0072B2",
  "grid" = "#999999",
  "ensemble" = "#56B4E9",
  "ensemble_all" = "#CC79A7"
)
```

Subset Analyses

Goal: look at various subsets of the data with π_0 between 0.6 and 1. Plot metrics as a function of true π_0.

```
table(platinum_data$fold_change$DE)
```

```

FALSE  TRUE
 3426  1944
```

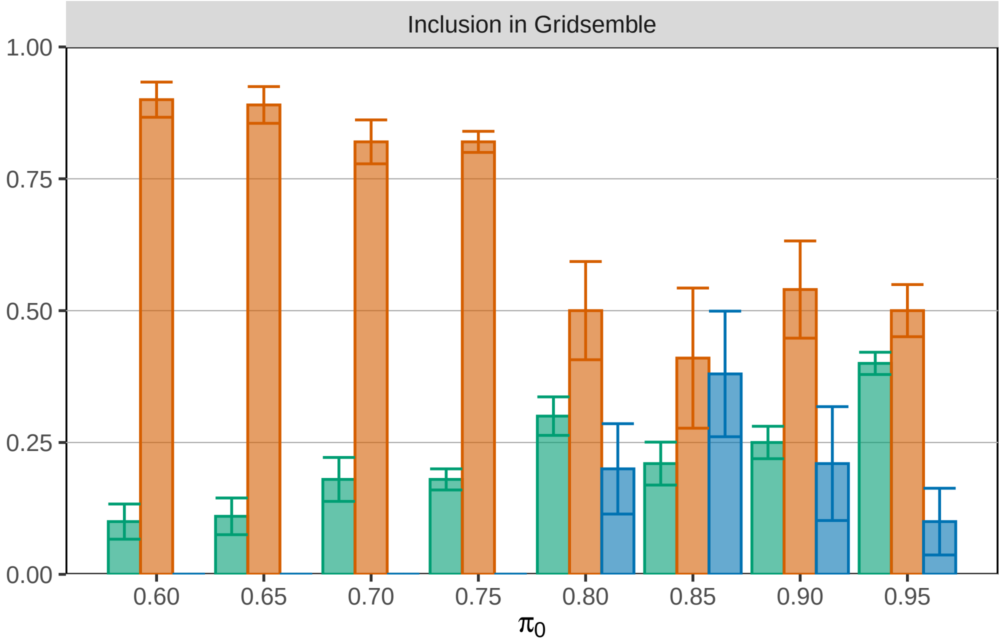
There are 5370 genes/hypotheses total, 3426 hypotheses are null and 1944 not-null. We let each subset be of size 1000. Sample 10 subsets of each size, and record metrics on fdr estimates as well as the 3 classification options on each.

```
set.seed(321)
subset_n = 1000

# consider pi0 0.6, 0.65, 0.7, ... 0.95
# sample 10 subsets for each pi0
subset_pi0_vec = rev(rep(seq(0.6, 0.95, by = 0.05), each = 10))
all_subset_mets <- list()
for (i in 1:length(subset_pi0_vec)) {
  pi0 = subset_pi0_vec[i]
  print(pi0)
  subset = get_subset(pi0 = pi0)

  subset_gridsemble_res <- gridsemble(
    subset$statistics,
    verbose = FALSE,
    df = 4,
    locfdr_grid = build_locfdr_grid(
      subset$statistics, grid_depth = 5
    ),
    fdrtool_grid = build_fdrtool_grid(
      subset$statistics, grid_depth = 20
    ),
    qvalue_grid = build_qvalue_grid(
      subset$statistics, grid_depth = 20
    )
  )
  subset_qvalue_res <- subset_gridsemble_res$default_qvalue
  subset_fdrtool_res <- subset_gridsemble_res$default_fdrtool
  subset_locfdr_res <- subset_gridsemble_res$default_locfdr

  gridsemble_cutoffs = c(
    '0.2' = 0.2,
    'pi0hat' = unname(quantile(
      subset_gridsemble_res$fdr,
      1-unname(subset_gridsemble_res$pi0)
    )),
    'pi0true' = unname(quantile(
      subset_gridsemble_res$fdr,
      1-pi0
    ))
  )
  gridsemble_metrics <- get_all_metrics(
    'gridsemble',
    subset_gridsemble_res$fdr,
    unname(subset_gridsemble_res$pi0),
    subset,
    cutoffs = gridsemble_cutoffs
  )


  fdrtool_cutoffs = c(
    '0.2' = 0.2,
    'pi0hat' = unname(quantile(
      subset_fdrtool_res$lfdr,
      1-subset_fdrtool_res$param[1,'eta0']
    )),
    'pi0true' = unname(quantile(
      subset_fdrtool_res$lfdr,
      1-pi0
    ))
  )
  fdrtool_metrics <- get_all_metrics(
    'fdrtool',
    subset_fdrtool_res$lfdr,
    subset_fdrtool_res$param[1,'eta0'],
    subset,
    cutoffs = fdrtool_cutoffs
  )

  qvalue_cutoffs = c(
    '0.2' = 0.2,
    'pi0hat' = unname(quantile(
      subset_qvalue_res$lfdr,
      1-subset_qvalue_res$pi0
    )),
    'pi0true' = unname(quantile(
      subset_qvalue_res$lfdr,
      1-pi0
    ))
  )
  qvalue_metrics <- get_all_metrics(
    'qvalue',
    subset_qvalue_res$lfdr,
    subset_qvalue_res$pi0,
    subset,
    cutoffs = qvalue_cutoffs
  )

  if (is.null(subset_locfdr_res)) {
    print(paste('uh oh', pi0))
    locfdr_metrics <- rep(NA, 47)
    names(locfdr_metrics) <- names(fdrtool_metrics)
    locfdr_metrics$method = 'locfdr'
  } else {
    locfdr_cutoffs = c(
      '0.2' = 0.2,
      'pi0hat' = unname(quantile(
        subset_locfdr_res$fdr,
        1-subset_locfdr_res$fp0['mlest','p0']
      )),
      'pi0true' = unname(quantile(
        subset_locfdr_res$fdr,
        1-pi0
      ))
    )
    locfdr_metrics <- get_all_metrics(
      'locfdr',
      subset_locfdr_res$fdr,
      subset_locfdr_res$fp0['mlest','p0'],
      subset,
      cutoffs = locfdr_cutoffs
    )
  }

  subset_package_counts = subset_gridsemble_res$top_grid %>%
    mutate(method = factor(method, levels = c('qvalue','locfdr','fdrtool'))) %>%
    pull(method) %>%
    table()
  subset_package_counts['pi0'] = pi0

  subset_mets = rbind(
    unlist(gridsemble_metrics),
    unlist(locfdr_metrics),
    unlist(fdrtool_metrics),
    unlist(qvalue_metrics)
  )

  subset_mets = data.frame(subset_mets)
  subset_mets$pi0 = pi0
  all_subset_mets[[i]] = subset_mets
  save(all_subset_mets, file = "PAPER_platinum_run_subsets.RData")

  if (i == 1) {
      all_package_counts = subset_package_counts
  } else {
      all_package_counts = rbind(
          all_package_counts,
          subset_package_counts
      )
  }
  save(all_package_counts, file = "PAPER_platinum_run_subsets_counts.RData")
}
```

```
load("PAPER_platinum_run_subsets.RData")
load("PAPER_platinum_run_subsets_counts.RData")
```

```
merge(
    all_package_counts %>%
        data.frame() %>%
        group_by(pi0) %>%
        summarize_all(function(x) {mean(x/10)}) %>%
        pivot_longer(2:4, values_to = 'mean'),
    all_package_counts %>%
        data.frame() %>%
        group_by(pi0) %>%
        summarize_all(function(x) {x = x/10; sd(x)/sqrt(length(x))}) %>%
        pivot_longer(2:4, values_to = 'se')
) %>%
    mutate(overall = "Inclusion in Gridsemble") %>%
    ggplot(aes(x = pi0, y = mean, fill = name, color = name)) +
    facet_grid(cols = vars(overall)) +
    geom_bar(stat = 'identity', position = 'dodge', alpha = 0.6) +
    geom_errorbar(aes(ymin = mean - se, ymax = mean + se), position = 'dodge') +
    scale_fill_manual(
        breaks = names(color_list),
        values = unlist(color_list)
    ) +
    scale_color_manual(
        breaks = names(color_list),
        values = unlist(color_list)
    ) +
    labs(
        x = expression(pi[0]),
        y = ''
    ) +
    theme(legend.position = "none") +
    scale_y_continuous(expand = c(0,0), limits = c(0, 1)) +
    theme(
        panel.grid.major.y = element_line(size = 0.2, color = 'darkgrey'),
        panel.grid.minor.y = element_blank(),
        panel.grid.major.x = element_blank(),
        panel.grid.minor.x = element_blank(),
        panel.background = element_rect(fill = 'white', color = 'black')
    ) +
    scale_x_continuous(
        breaks = seq(0.6, 0.95, by = 0.05)
    )
```

```
Warning: The `size` argument of `element_line()` is deprecated as of ggplot2 3.4.0.
i Please use the `linewidth` argument instead.
```



```
ggsave("PAPER_subsets_counts.png", width = 8, height = 3)
```

Plot

```r
g_legend <- function(a.gplot){
  tmp <- ggplot_gtable(ggplot_build(a.gplot))
  leg <- which(sapply(tmp$grobs, function(x) x$name) == "guide-box")
  legend <- tmp$grobs[[leg]]
  return(legend)
}
```

```r
plot_dat = do.call(rbind, all_subset_mets) %>%
  mutate(
    pi0_SE = (as.numeric(pi0) - as.numeric(pi0hat))**2
  ) %>%
  select(-starts_with("method_")) %>%
  group_by(method, pi0) %>%
  summarize_all(
    # .funs = c(
    #   center = function(vec) {median(as.numeric(vec))},
    #   lower = function(vec) {quantile(as.numeric(vec), 0.25, na.rm = TRUE)},
    #   upper = function(vec) {quantile(as.numeric(vec), 0.75, na.rm = TRUE)}
    # )
    .funs = c(
      center = function(vec) {
        mean(as.numeric(vec), na.rm = TRUE)
      },
      lower = function(vec) {
        if (length(vec) <= 1) {return(NA)}
        vec = as.numeric(vec)
        mean(vec, na.rm = TRUE) - sd(vec, na.rm = TRUE)/sqrt(sum(!is.na(vec)))
      },
      upper = function(vec) {
        if (length(vec) <= 1) {return(NA)}
        vec = as.numeric(vec)
        mean(vec, na.rm = TRUE) + sd(vec, na.rm = TRUE)/sqrt(sum(!is.na(vec)))
      }
    )
  )
```

```r
do.call(rbind, all_subset_mets) %>%
  filter(method == 'locfdr') %>%
  filter(!is.na(roc)) %>%
  group_by(pi0) %>%
  summarize(n = n())
```

```
# A tibble: 7 x 2
    pi0     n
  <dbl> <int>
1  0.6      4
2  0.65     7
3  0.7      4
4  0.8      2
5  0.85     1
6  0.9      2
7  0.95     2
```

```
  plot_dat %>%
    filter(method == 'locfdr') %>%
    select(pi0, roc_center, roc_lower, roc_upper)
```

```
Adding missing grouping variables: `method`
```

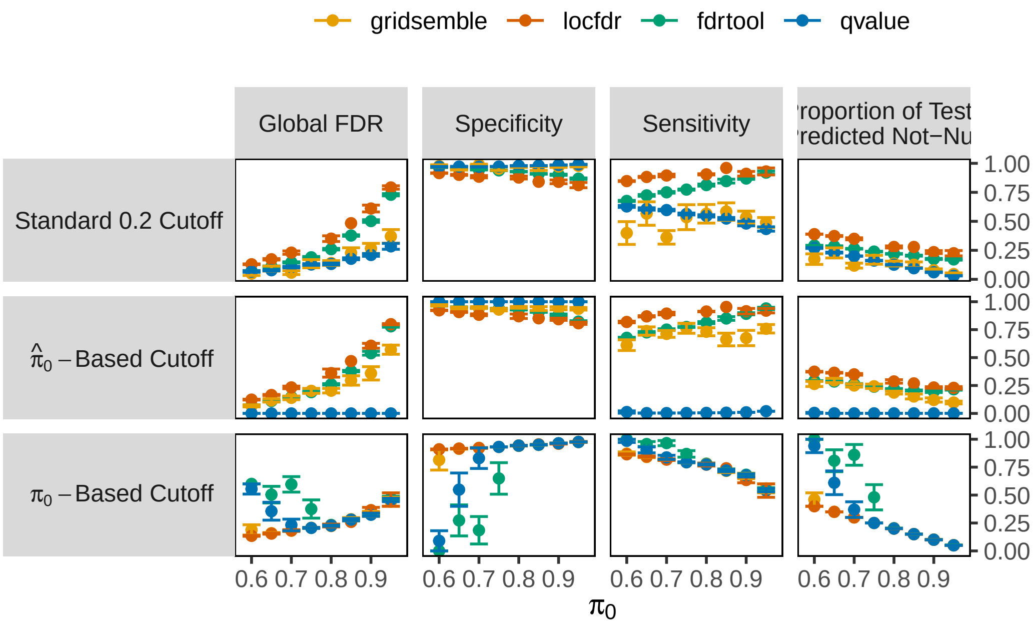
```
# A tibble: 8 x 5
# Groups:   method [1]
  method   pi0 roc_center roc_lower roc_upper
  <chr>  <dbl>      <dbl>     <dbl>     <dbl>
1 locfdr  0.6       0.934     0.931     0.938
2 locfdr  0.65      0.939     0.936     0.942
3 locfdr  0.7       0.938     0.933     0.942
4 locfdr  0.75    NaN       NaN       NaN
5 locfdr  0.8       0.944     0.938     0.949
6 locfdr  0.85      0.958    NA        NA
7 locfdr  0.9       0.922     0.904     0.940
8 locfdr  0.95      0.937     0.933     0.940
```

```
  plot_dat2 <- merge(
    merge(
      plot_dat %>%
        pivot_longer(ends_with("center"), names_to = 'metric', values_to = "center") %>%
        mutate(
          metric = str_replace(metric, "_center", "")
        ) %>%
        select(
          method, pi0, metric, center
        ),
      plot_dat %>%
        pivot_longer(ends_with("lower"), names_to = 'metric', values_to = "lower") %>%
        mutate(
          metric = str_replace(metric, "_lower", "")
        ) %>%
        select(
          method, pi0, metric, lower
        ),
      by = c('method','pi0','metric')
    ),
    plot_dat %>%
      pivot_longer(ends_with("upper"), names_to = 'metric', values_to = "upper") %>%
      mutate(
        metric = str_replace(metric, "_upper", "")
      ) %>%
      select(
        method, pi0, metric, upper
      ),
    by = c('method','pi0','metric')
  )
```

```
plot_dat3 <- plot_dat2

plot_dat3$cutoff_method = sapply(plot_dat3$metric, function(m) {
  m_parts = str_split(m, '_')[[1]]
  m_parts[length(m_parts)]
})
plot_dat3$metric = sapply(plot_dat3$metric, function(m) {
  m_parts = str_split(m, '_')[[1]]
  paste(m_parts[1:(length(m_parts)-1)], collapse = '_')
})

plot_dat3 <- plot_dat3 %>%
  filter(
    cutoff_method %in% c('0.2', 'pi0hat', 'pi0true')
  )
```

```r
plot_dat3 <- plot_dat3 %>%
  filter(metric %in% c(
    'global_FDR', 'specificity', 'sensitivity', 'prop_pred_T'
  )) %>%
  mutate(
    cutoff_method = factor(
      cutoff_method,
      levels = c("0.2",  "pi0hat", "pi0true"),
      labels = c('"Standard 0.2 Cutoff"',
                 '~ hat(pi)[0] ~ -"Based Cutoff"',
                 '~ pi[0] ~ -"Based Cutoff"')
    ),
    metric = factor(
      metric,
      levels = c('global_FDR', 'specificity', 'sensitivity', 'prop_pred_T'),
      labels = c(
        '"Global FDR"','"Specificity"','"Sensitivity"',
        '"Proportion of Tests\nPredicted Not-Null"'
      )
    )
  )

plot_dat3 %>%
  ggplot(aes(x = pi0, y = center, color = method)) +
  facet_grid(
    rows = vars(cutoff_method),
    cols = vars(metric),
    scales = 'free',
    switch = "y",
    labeller = label_parsed
  ) +
  geom_point() +
  geom_errorbar(aes(ymin = lower, ymax = upper)) +
  scale_color_manual(
    breaks = names(color_list),
    values = unlist(color_list)
  ) +
  labs(
    x = expression(pi[0])
  ) +
  scale_y_continuous(position = 'right') +
  theme(
    strip.text.y.left = element_text(angle = 0),
    axis.title.y.right = element_blank(),
    legend.position = 'top',
    legend.title = element_blank(),
    panel.background = element_blank(),
    panel.border = element_rect(fill = NA),
    legend.key = element_blank()
  )
```

```
Warning: Removed 12 rows containing missing values (`geom_point()`).
```



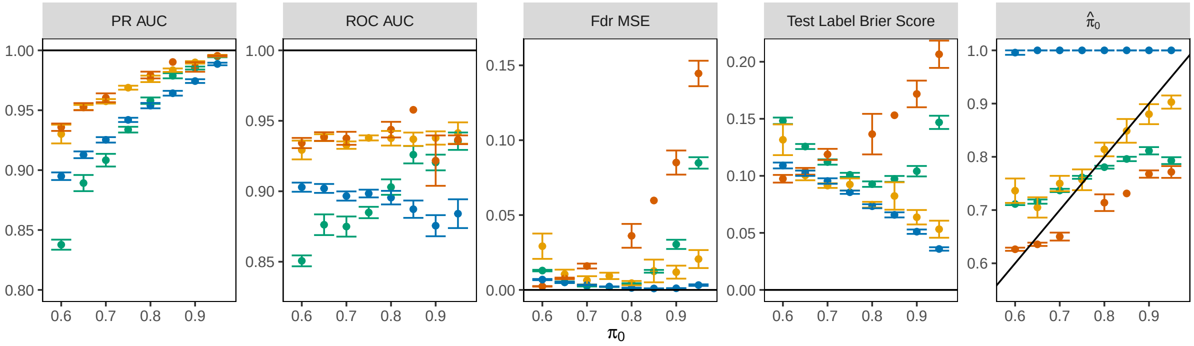
```
ggsave("PAPER_platinum_run_subsets_classification_metrics.png", width = 6, height = 4)
```

```
Warning: Removed 12 rows containing missing values (`geom_point()`).
```

```
library(ggh4x)
lines = data.frame(
  metric = factor(
    c('pr','roc','Fdr.MSE','brier','pi0hat.X1'),
```

```r
    levels = c('pr','roc','Fdr.MSE','brier','pi0hat.X1'),
    labels = c('"PR AUC"','"ROC AUC"','"Fdr MSE"','"Test Label Brier Score"','hat(pi)[0]')
  ),
  intercept = c(1, 1, 0, 0, 0),
  slope = c(0,0,0,0,1)
)

scales_y <- list(
  `brier` = scale_y_continuous(limits = c(0.8, 1)),
  `Fdr.MSE` = scale_y_continuous(limits = c(0.83, 1)),
  `pr` = scale_y_continuous(limits = c(0, 0.16)),
  `roc` = scale_y_continuous(limits = c(0, 0.21)),
  `pi0` = scale_y_continuous(limits = c(0.55, 1))
)

plot_dat2 %>%
  filter(metric %in% c('pr','roc','Fdr.MSE','brier','pi0hat')) %>%
  mutate(
    metric = factor(
      metric,
      levels = c('pr','roc','Fdr.MSE','brier','pi0hat'),
      labels = c('"PR AUC"','"ROC AUC"','"Fdr MSE"','"Test Label Brier Score"','hat(pi)[0]
    )
  ) %>%
  ggplot(aes(x = pi0, y = center, color = method)) +
  facet_wrap(
    ~ metric,
    nrow = 1,
    scales = 'free',
    labeller = label_parsed
  ) +
  geom_point() +
  geom_errorbar(aes(ymin = lower, ymax = upper)) +
  scale_color_manual(
    breaks = names(color_list),
    values = unlist(color_list)
  ) +
  labs(
    x = expression(pi[0])
  ) +
  theme(
    axis.title.y = element_blank(),
    legend.position = 'bottom',
    legend.title = element_blank(),
    panel.background = element_blank(),
    panel.border = element_rect(fill = NA),
    legend.key = element_blank()
  ) +
  geom_abline(data = lines, aes(intercept = intercept, slope = slope)) +
  ggh4x::facetted_pos_scales(
    y = scales_y
  )  +
  theme(
    legend.position = 'none'
  )
```

```
Warning: Removed 5 rows containing missing values (`geom_point()`).
```

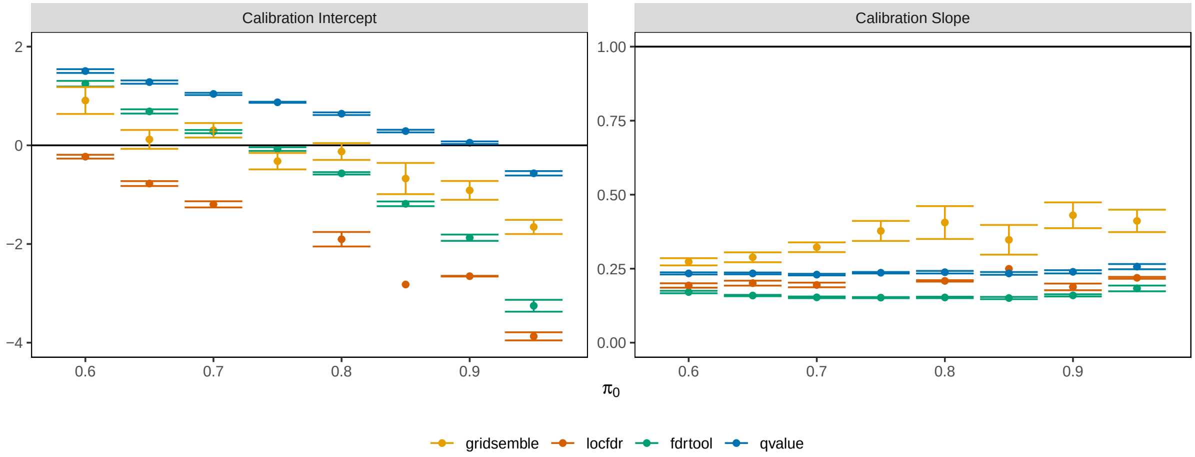


```
ggsave("PAPER_platinum_run_subsets_metrics.png", width = 8, height = 2)
```

```
Warning: Removed 5 rows containing missing values (`geom_point()`).
```

```
scales_y <- list(
  `intercept` = scale_y_continuous(limits = c(-4, 2)),
  `slope` = scale_y_continuous(limits = c(0, 1))
)

lines = data.frame(
  metric = factor(
    c('intercept','slope'),
    levels = c('intercept','slope'),
    labels = c('Calibration Intercept','Calibration Slope')
  ),
  intercept = c(0,1),
  slope = c(0,0)
)

plot_dat2 %>%
  filter(
    startsWith(metric, 'calib')
  ) %>%
  mutate(
    metric = str_replace(metric, 'calib_', ''),
    metric = factor(
      metric,
      levels = c('intercept','slope'),
      labels = c('Calibration Intercept','Calibration Slope')
    )
  ) %>%
  filter(
    metric != 'nEmpty'
  ) %>%
  ggplot(aes(x = pi0, y = center, color = as.factor(method))) +
  facet_wrap(~metric, scales = 'free_y') +
  geom_point() +
  geom_abline(data = lines, aes(intercept = intercept, slope = slope)) +
  theme(
    axis.title.y = element_blank(),
    legend.position = 'bottom',
    legend.title = element_blank(),
    panel.background = element_blank(),
    panel.border = element_rect(fill = NA),
    legend.key = element_blank()
  ) +
  ggh4x::facetted_pos_scales(
    y = scales_y
  ) +
  geom_errorbar(aes(ymin = lower, ymax = upper)) +
  scale_color_manual(
    breaks = names(color_list),
    values = unlist(color_list)
  ) +
  labs(
    x = expression(pi[0])
  )
```

```
Warning: Removed 2 rows containing missing values (`geom_point()`).
```



```
ggsave("PAPER_platinum_run_calibration_metrics.png", width = 8, height = 3)
```

```
Warning: Removed 2 rows containing missing values (`geom_point()`).
```