Pycno Run 5

Zack Gold

2025-12-30

Set Up

Load Libraries

Values Set

```
IPC_name = "Moa"
Target_name = "Pycno"
```

LoD + LoQ Analysis - Pycno Plate Run 5

Load Data

```
sample_set_up <- read_excel(here("2025-12-23_161523_Pycno_5_v3.xls"), sheet="Sample Setup", skip = 46)

amplification_data<- read_excel(here("2025-12-23_161523_Pycno_5_v3.xls"), sheet="Amplification Data",  s
results_data<- read_excel(here("2025-12-23_161523_Pycno_5_v3.xls"), sheet="Results",  skip = 46)
```

Merge Tables

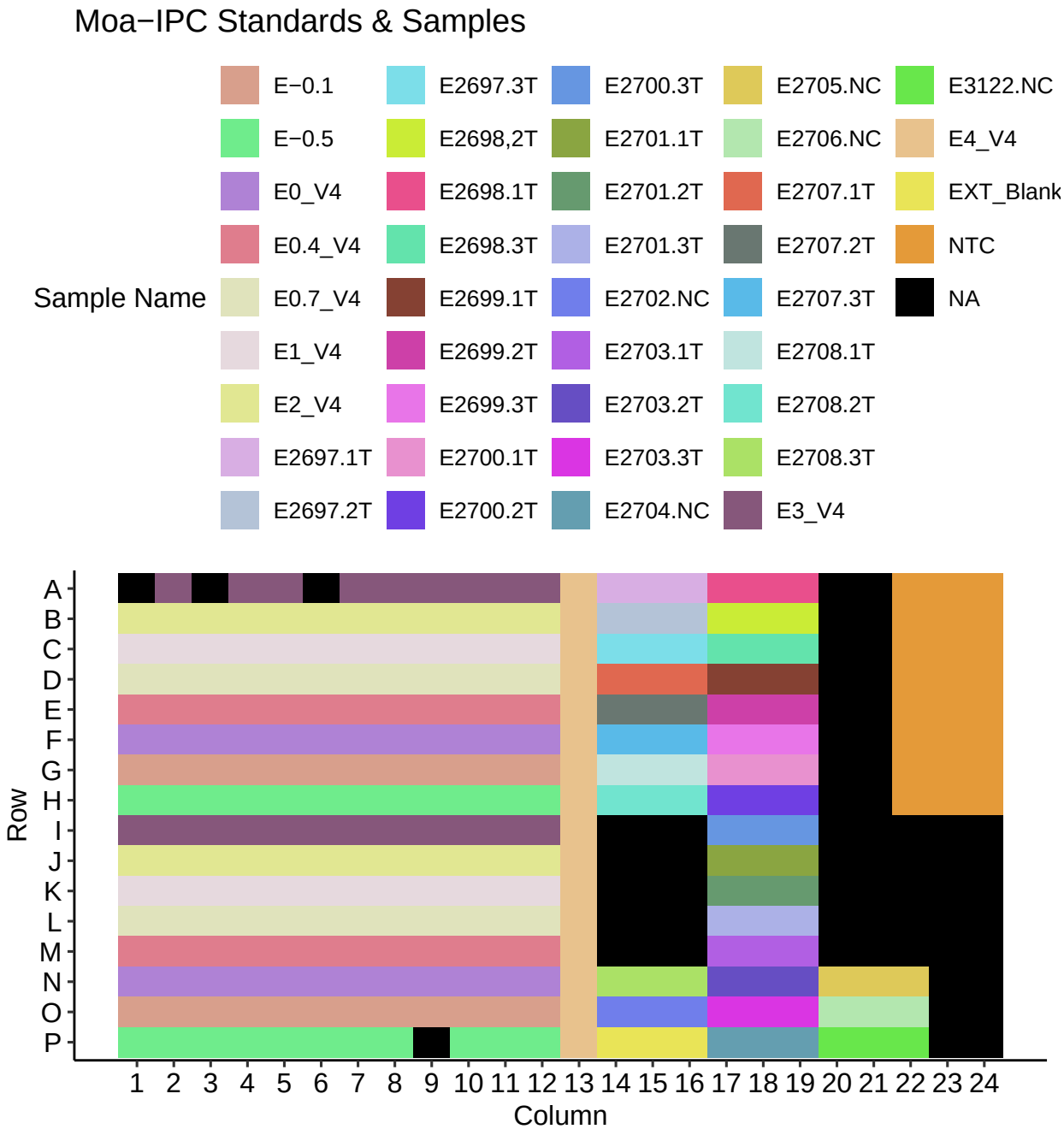
```
amplification_data %>%
  left_join(sample_set_up) -> combined_qPCR_data
```

Now we are ready to start analyzing the data.

First thing we want to do is check to see how our internal positive control is working.

Moa Map

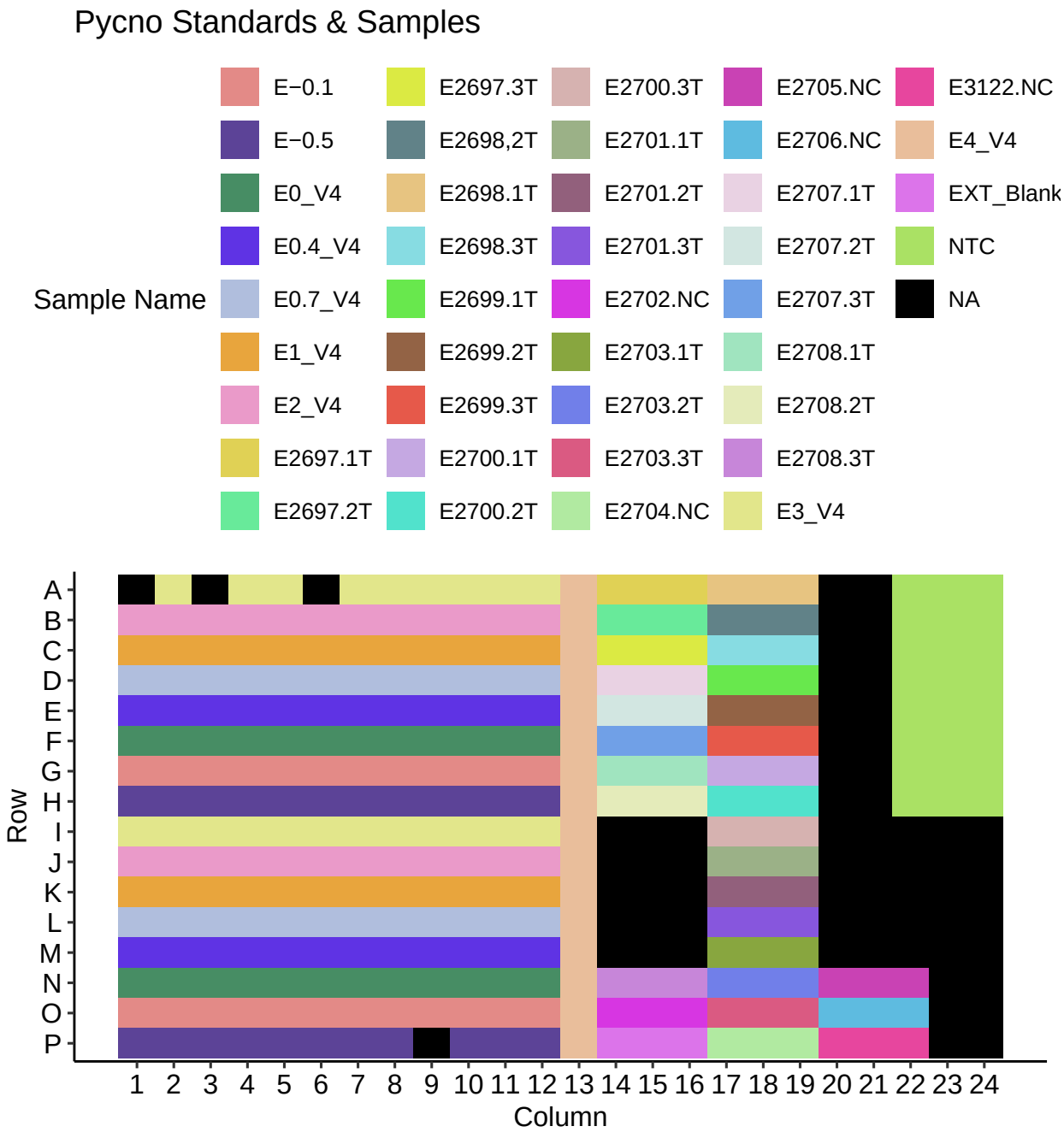
```r
columns = seq(from =1, to=24, by=1)
columns_t= as.tibble(columns) %>%  dplyr::rename("Column"=value)
rows = capitalize(letters[1:16])
rows_t= as.tibble(rows) %>%  dplyr::rename("Row"=value)

cross_join(columns_t, rows_t) -> full_plate

combined_qPCR_data %>%
  filter(., `Target Name`==IPC_name) %>%
  mutate(., Row = str_sub(`Well Position`,start = 1L, end = 1L),
         Column=  as.numeric(str_sub(`Well Position`,start = 2L, end = -1L ))) -> moa_samples

full_plate %>%
  left_join(moa_samples) -> moa_plate


palette <- distinctColorPalette(length(moa_plate$`Sample Name` %>%  unique()))

moa_plate %>%
  mutate(Row = factor(Row, levels = rows)) %>%
  ggplot(aes(x=Column, y=fct_rev(Row), fill=`Sample Name` )) +geom_tile() +scale_x_continuous(breaks=co
```



Pycno Map

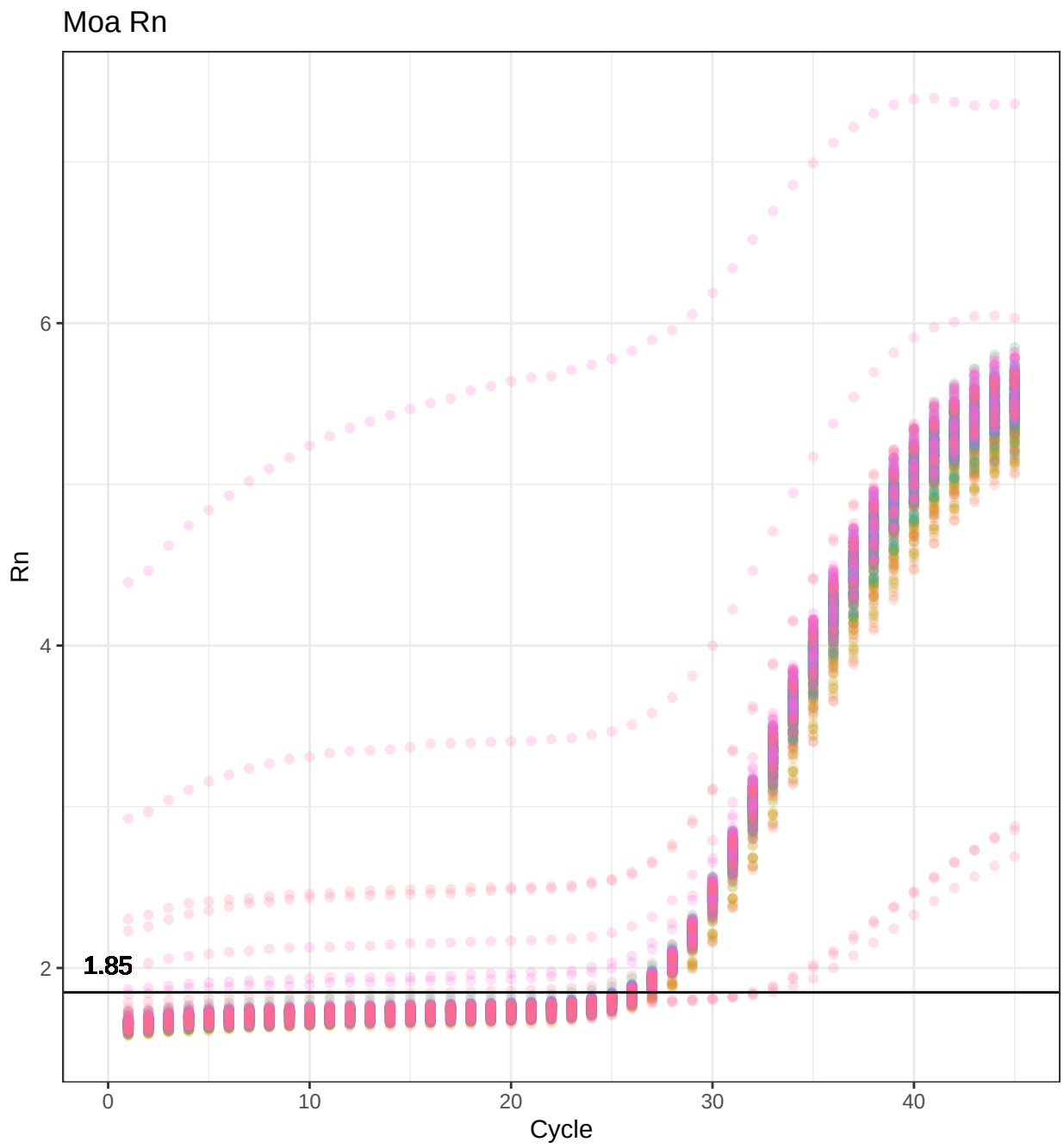
```
combined_qPCR_data %>%
  filter(., `Target Name`==Target_name) %>%
  mutate(., Row = str_sub(`Well Position`,start = 1L, end = 1L),
         Column=  as.numeric(str_sub(`Well Position`,start = 2L, end = -1L ))) -> pycno_samples

full_plate %>%
  left_join(pycno_samples) -> pycno_plate
```

```
palette <- distinctColorPalette(length(pycno_plate$`Sample Name` %>%  unique()))

pycno_plate %>%
  mutate(Row = factor(Row, levels = rows)) %>%
  ggplot(aes(x=Column, y=fct_rev(Row), fill=`Sample Name` )) +geom_tile() +scale_x_continuous(breaks=co
```



Moa Rn

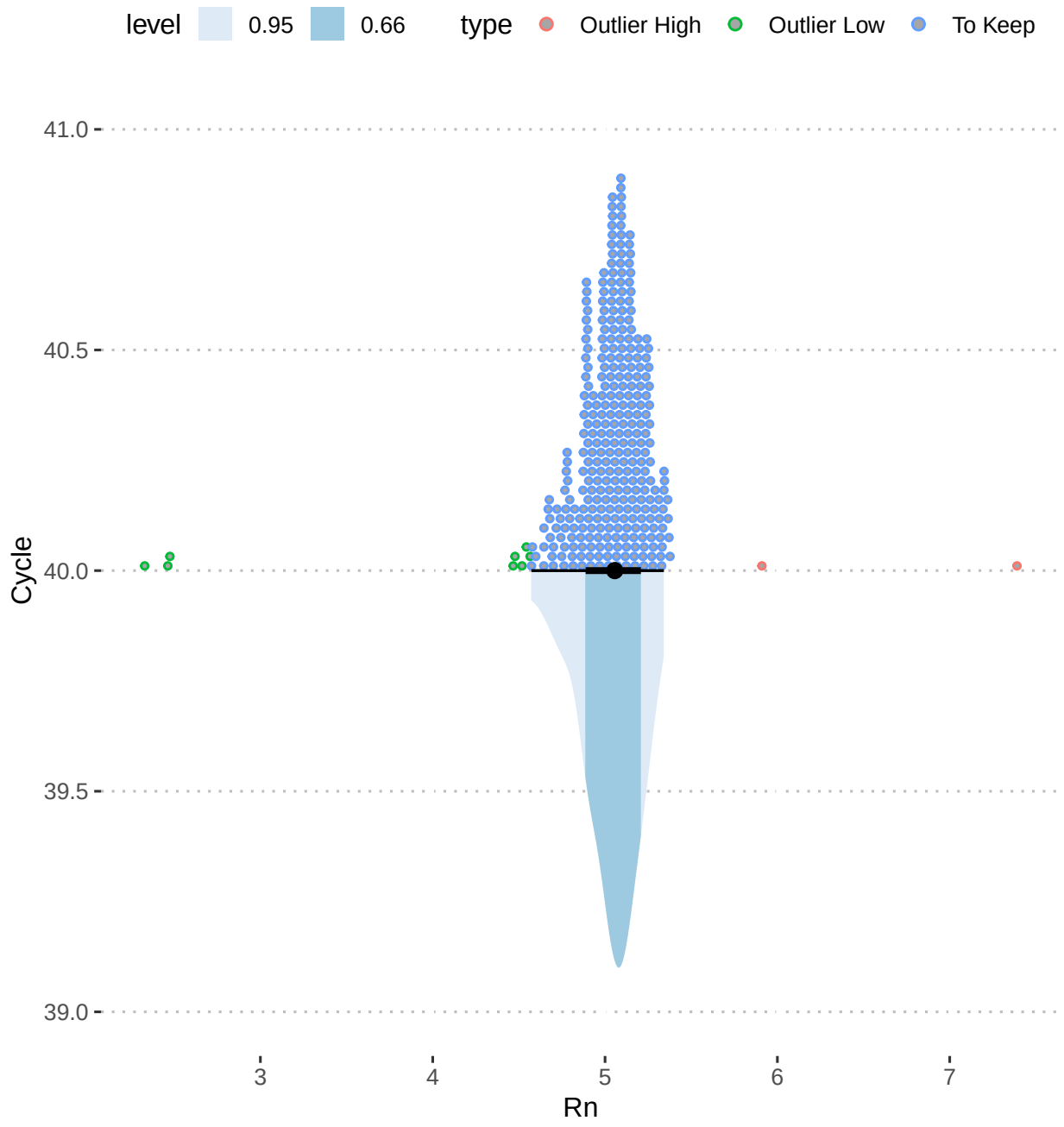
```
combined_qPCR_data %>%
filter(., str_detect(`Target Name`,IPC_name)) -> moa_standard

moa_standard$Rn %>%  max() -> max_Rn_moa
h_moa= max_Rn_moa*0.25

moa_standard %>%
 # filter(., `Well Position`=="A24") %>% A24 did not perform well
ggplot(., aes(y=Rn, x=Cycle, color=`Well Position`)) +
  geom_point(alpha=0.2) + geom_hline(yintercept =h_moa) +
  geom_text(aes(0,h_moa,label = round(h_moa,2), vjust = -1),color="black") +theme_bw() +ggtitle("Moa Rn
```



There is quite a bit of spread in our IPC. Will remove any samples outside normal distribution. #### Remove Samples with Anomolous Outlier IPC values

```
moa_standard %>%
  filter(., Cycle ==40) -> moa_standard_c40

moa_standard_c40 %>%
 dplyr::summarise(mean_Rn = ci(Rn, confidence = 0.99)[1],
                       lower_ci_mpg = ci(Rn, confidence = 0.99)[2],
                       upper_ci_mpg = ci(Rn, confidence = 0.99)[3],
                       sd_Rn = ci (Rn, confidence = 0.99)[4]) -> CI0.95_moa
```

```r
Sum = groupwiseMean(Rn ~ Cycle, data=moa_standard_c40, conf = 0.99)

moa_standard_c40 %>%
summarise(
    Q1 = quantile(Rn, 0.25, na.rm = TRUE),
    Q3 = quantile(Rn, 0.75, na.rm = TRUE),
    IQR = Q3 - Q1,
    lower_bound = Q1 - 1.5 * IQR,
    upper_bound = Q3 + 1.5 * IQR
  ) -> moa_iqr

moa_standard_c40 %>%
  mutate(., type=case_when(Rn > moa_iqr$upper_bound ~"Outlier High",
                           Rn < moa_iqr$lower_bound ~"Outlier Low",
                           TRUE~"To Keep")) -> moa_standard_c40

moa_standard_c40 %>%
  ggplot(., aes(x= Rn, y=Cycle)) +  geom_dots(layout='weave', side = "top", aes(color=type)) + stat_slab
```



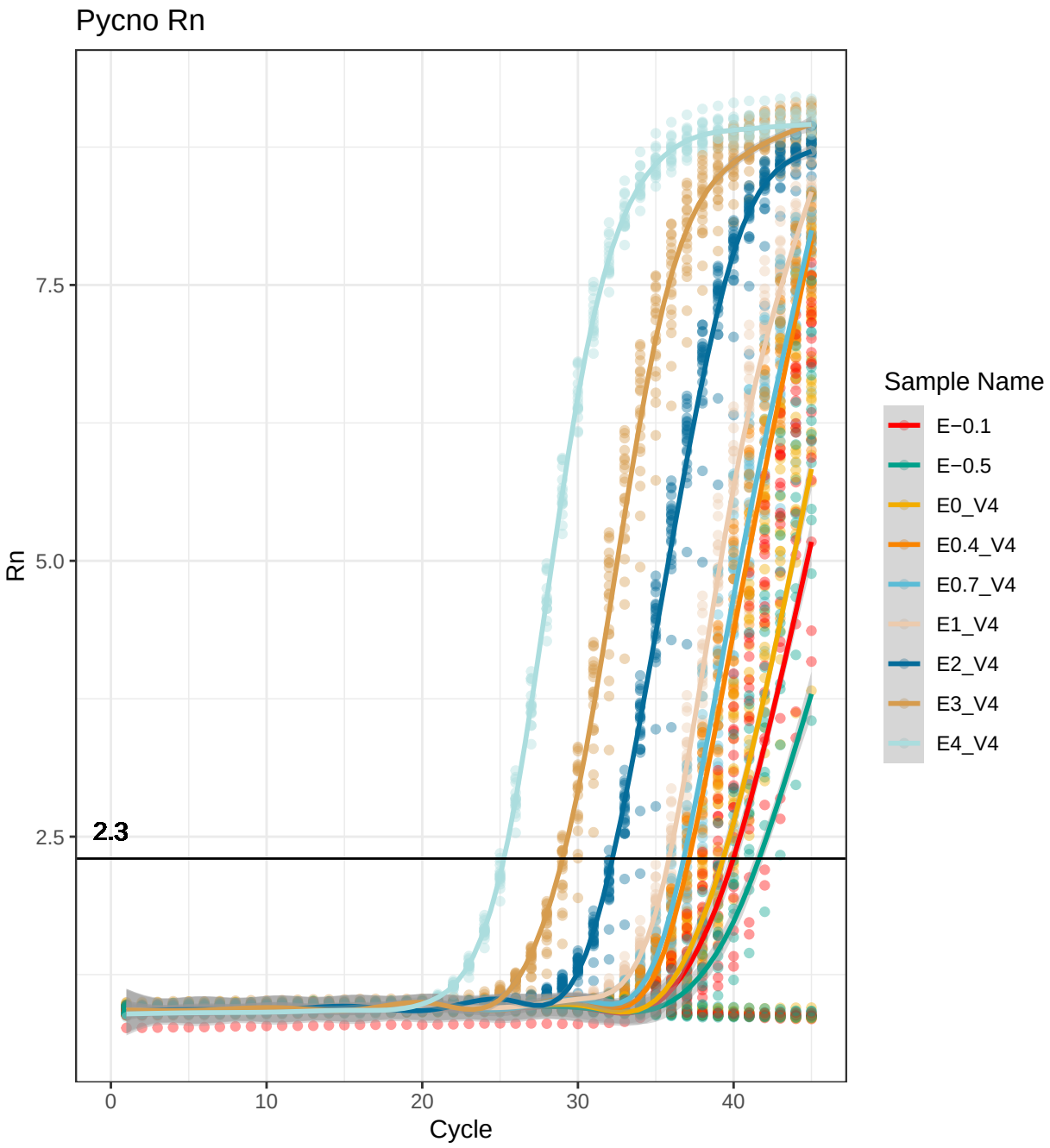

```
moa_standard_c40 %>%
  filter(., type!="To Keep") -> moa_outliers_to_drop
```

```
combined_qPCR_data %>%
  filter(., `Well Position` != moa_outliers_to_drop$`Well Position`) -> combined_qPCR_data_cleaned

results_data %>%
    filter(., `Well Position` != moa_outliers_to_drop$`Well Position`) -> results_data_cleaned
```

Filter Data

Pycno Rn

```
combined_qPCR_data_cleaned %>%
filter(., str_detect(`Target Name`,Target_name)) %>%
  filter(., Task=="STANDARD")-> pycno_standard

pycno_standard$Rn %>%  max() -> max_Rn
h= max_Rn*0.25

pycno_standard %>%
ggplot(., aes(y=Rn, x=Cycle, color=`Sample Name`)) +geom_point(alpha=0.4) +geom_smooth() + geom_hline(y
  geom_text(aes(0,h,label = round(h,2), vjust = -1),color="black") +theme_bw() + ggtitle("Pycno Rn") +s
```

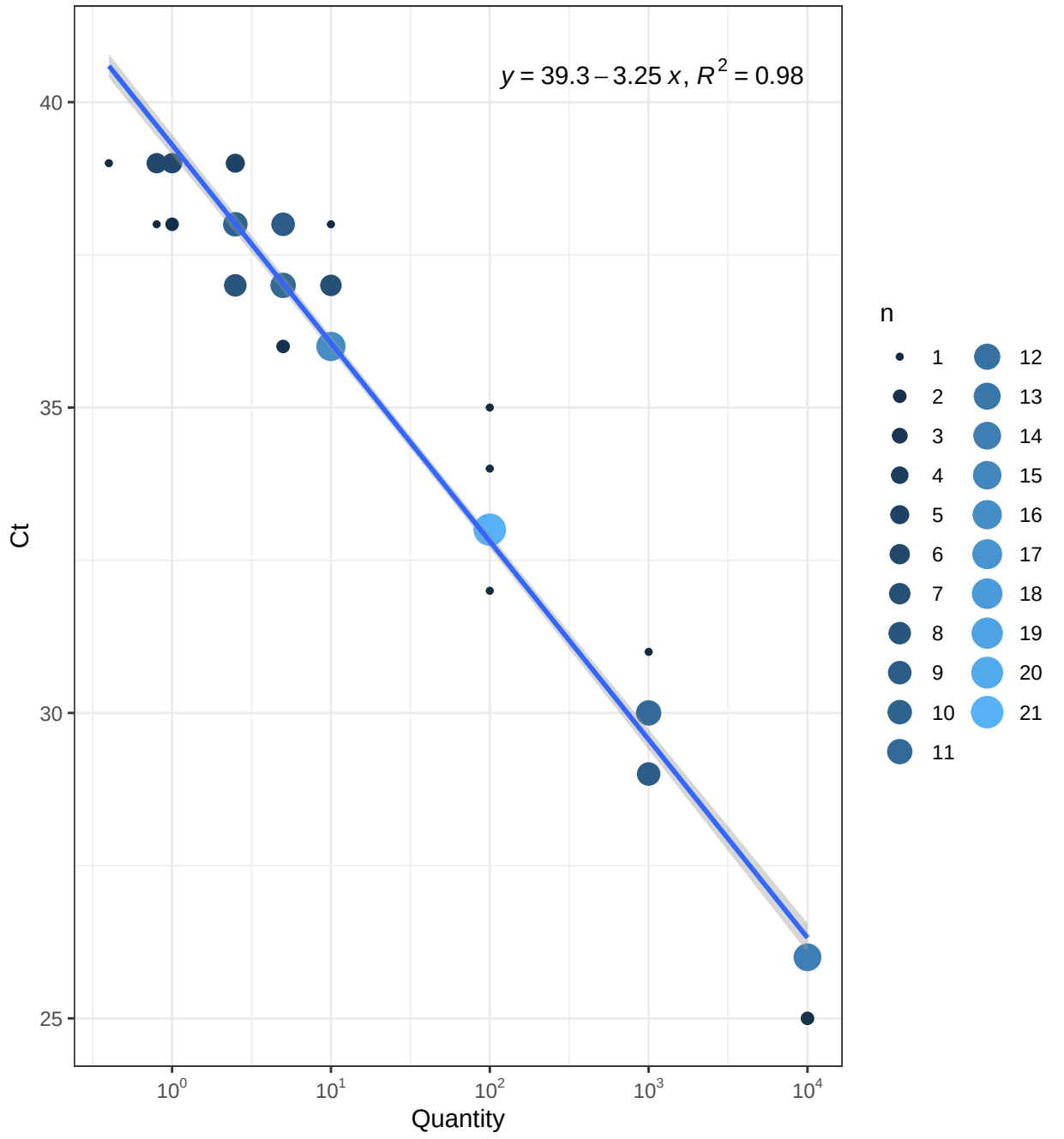


Looks great. We except higher concentration standards to surpass the threshold (here set to 25% of the maximum Rn flourescence value) at fewer cycles and we see that in the data.

Pycno Ct versus Quantity

```
pycno_standard %>%
  mutate(., Threshold = if_else(Rn > h, "Above","Below")) -> pycno_standard

pycno_standard %>%
```

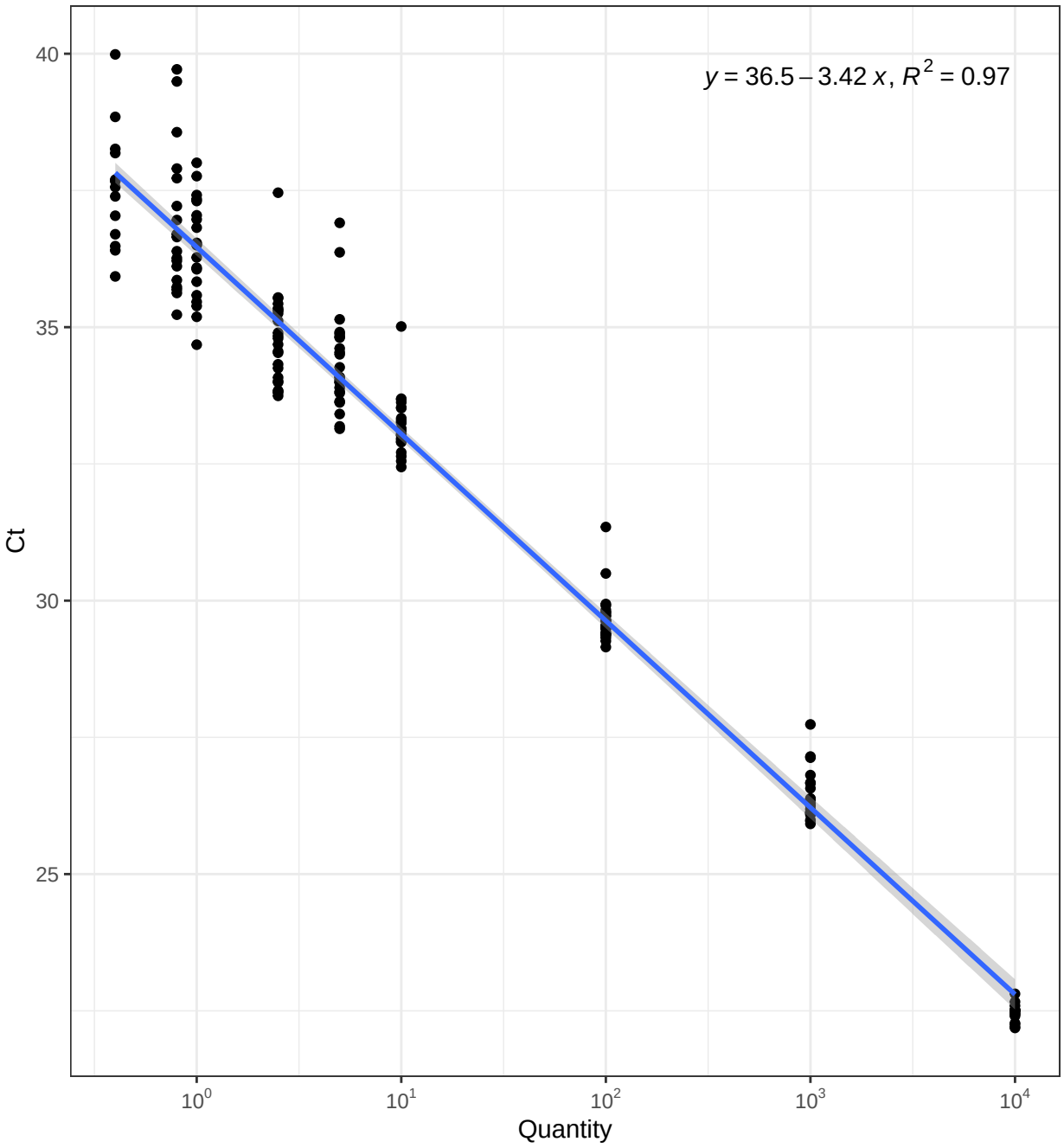
```
  group_by(`Well Position`,`Sample Name`) %>%
  filter(., rank(Threshold, ties.method="first")==1) %>%
  filter(., Threshold =="Above") -> pycno_standard_above

pycno_standard %>%
  ungroup() %>%
  filter(., !`Well Position` %in%  pycno_standard_above$`Well Position`) %>%
  group_by(`Well Position`,`Sample Name`) %>%
  filter(., rank(Threshold, ties.method="last")==1) -> pycno_standard_below

bind_rows(pycno_standard_above, pycno_standard_below) -> pycno_standard_threshold



set_breaks = function(limits) {
     seq(limits[1], limits[2], by = 1)
}

pycno_standard_threshold %>%
  filter(., Cycle < 40 ) %>%
  ggplot(., aes(x=`Quantity`, y = Cycle)) +geom_count(aes(color = ..n.., size = ..n..))  +
    scale_x_log10("Quantity",
        breaks = trans_breaks("log10", function(x) 10^x),
        labels = trans_format("log10", math_format(10^.x))) +
  guides(color = 'legend') +
     scale_size_continuous(breaks = set_breaks)+
     scale_color_continuous(breaks = set_breaks)+
  stat_poly_line() +
  stat_poly_eq(use_label(c("eq", "R2")),label.x = "right",
  label.y = "top") +theme_bw() +ylab("Ct") -> run5_ct_versus_quant

run5_ct_versus_quant
```




```
# View the structure of the generated plot object
# The computed data is stored within the plot object
grob_data_run5 <- ggplot_build(run5_ct_versus_quant)$data[[3]]

# The result is a list; unlist it to get a character vector
coef_char_run5 <- unlist(grob_data_run5$coefs)


standard_curve_coefs_run5 <- enframe(coef_char_run5, name = "Metric", value = "Value") %>%
 mutate( Metric = recode(Metric,
                         "(Intercept)" = "Intercept",
```

```
                                "x" = "Slope"))

# Extract the R^2 value from the 'r.squared' column
r_squared_value_run5 <- grob_data_run5$r.squared
```

```
results_data_cleaned %>%
  filter(., `Target Name`==Target_name) %>%
  filter(., Task=="STANDARD") %>%
   mutate(., CT=as.numeric(CT)) -> pycno_results_standard


 pycno_results_standard %>%
  ggplot(., aes(x=`Quantity`, y = CT)) +geom_point()  +
    scale_x_log10("Quantity",
        breaks = trans_breaks("log10", function(x) 10^x),
        labels = trans_format("log10", math_format(10^.x))) +
  guides(color = 'legend') +
     scale_size_continuous(breaks = set_breaks)+
     scale_color_continuous(breaks = set_breaks)+
  stat_poly_line() +
  stat_poly_eq(use_label(c("eq", "R2")),label.x = "right",
  label.y = "top") +theme_bw() +ylab("Ct")
```



Klymus et al. 2020 LoD + LoQ

LoD

LOD was defined as the lowest concentration at which 95% of the technical replicates exhibited positive amplification.

```
pycno_standard_threshold %>%
  group_by(`Sample Name`, Threshold, Quantity) %>%
```

```
  count()
```

Raw data with R fit

```
## # A tibble: 12 x 4
## # Groups:   Sample Name, Threshold, Quantity [12]
##    `Sample Name` Threshold  Quantity     n
##    <chr>         <chr>         <dbl> <int>
##  1 E-0.1         Above         0.800    18
##  2 E-0.1         Below         0.800     6
##  3 E-0.5         Above         0.400    13
##  4 E-0.5         Below         0.400    10
##  5 E0.4_V4       Above         2.5      24
##  6 E0.7_V4       Above         5        24
##  7 E0_V4         Above         1        20
##  8 E0_V4         Below         1         4
##  9 E1_V4         Above        10        24
## 10 E2_V4         Above       100        24
## 11 E3_V4         Above      1000        21
## 12 E4_V4         Above     10000        16
```

```
pycno_standard_threshold %>%
  group_by(`Sample Name`, Threshold, Quantity) %>%
  count() %>%
  pivot_wider(names_from = Threshold, values_from = n, values_fill=0) %>%
  mutate(., Perc = Above/(Above+Below)) %>%
  arrange(desc(Quantity)) %>% kable()
```

Sample Name	Quantity	Above	Below	Perc
E4_V4	10000.0	16	0	1.0000000
E3_V4	1000.0	21	0	1.0000000
E2_V4	100.0	24	0	1.0000000
E1_V4	10.0	24	0	1.0000000
E0.7_V4	5.0	24	0	1.0000000
E0.4_V4	2.5	24	0	1.0000000
E0_V4	1.0	20	4	0.8333333
E-0.1	0.8	18	6	0.7500000
E-0.5	0.4	13	10	0.5652174

```
# LoD is 2.5 copies/µL
```

Here we see that 95% of the technical replicates exhibited positive amplification between 1 and 2.5 copies per µL.

```
pycno_results_standard %>%
  mutate(., detection = case_when(is.na(CT) ~"Non-detection",
                                  TRUE ~ "Detection")) %>%
  group_by(`Sample Name`,detection, Quantity) %>%
  count()
```

QuantStudio 5 fit

```
## # A tibble: 12 x 4
## # Groups:   Sample Name, detection, Quantity [12]
##    `Sample Name` detection      Quantity     n
##    <chr>         <chr>             <dbl> <int>
##  1 E-0.1         Detection         0.800    18
##  2 E-0.1         Non-detection     0.800     6
##  3 E-0.5         Detection         0.400    13
##  4 E-0.5         Non-detection     0.400    10
##  5 E0.4_V4       Detection         2.5      24
##  6 E0.7_V4       Detection         5        24
##  7 E0_V4         Detection         1        20
##  8 E0_V4         Non-detection     1         4
##  9 E1_V4         Detection        10        24
## 10 E2_V4         Detection       100        24
## 11 E3_V4         Detection      1000        21
## 12 E4_V4         Detection     10000        16
```

```
pycno_results_standard %>%
    mutate(., detection = case_when(is.na(CT) ~"Non-detection",
                                    TRUE ~ "Detection")) %>%
  group_by(`Sample Name`, detection, Quantity) %>%
  count() %>%
  pivot_wider(names_from = detection, values_from = n, values_fill=0) %>%
  mutate(., Perc = Detection/(Detection+`Non-detection`)) %>%
  arrange(desc(Quantity)) %>%  kable()
```

Sample Name	Quantity	Detection	Non-detection	Perc
E4_V4	10000.0	16	0	1.0000000
E3_V4	1000.0	21	0	1.0000000
E2_V4	100.0	24	0	1.0000000
E1_V4	10.0	24	0	1.0000000
E0.7_V4	5.0	24	0	1.0000000
E0.4_V4	2.5	24	0	1.0000000
E0_V4	1.0	20	4	0.8333333
E-0.1	0.8	18	6	0.7500000
E-0.5	0.4	13	10	0.5652174

```
# LoD is 2.5 copies/µL
```

Using the QuantStudio5 estimated quantities, we can also see that our limit of quantification here appears to be between 1 and 2.5 copies.

LoQ

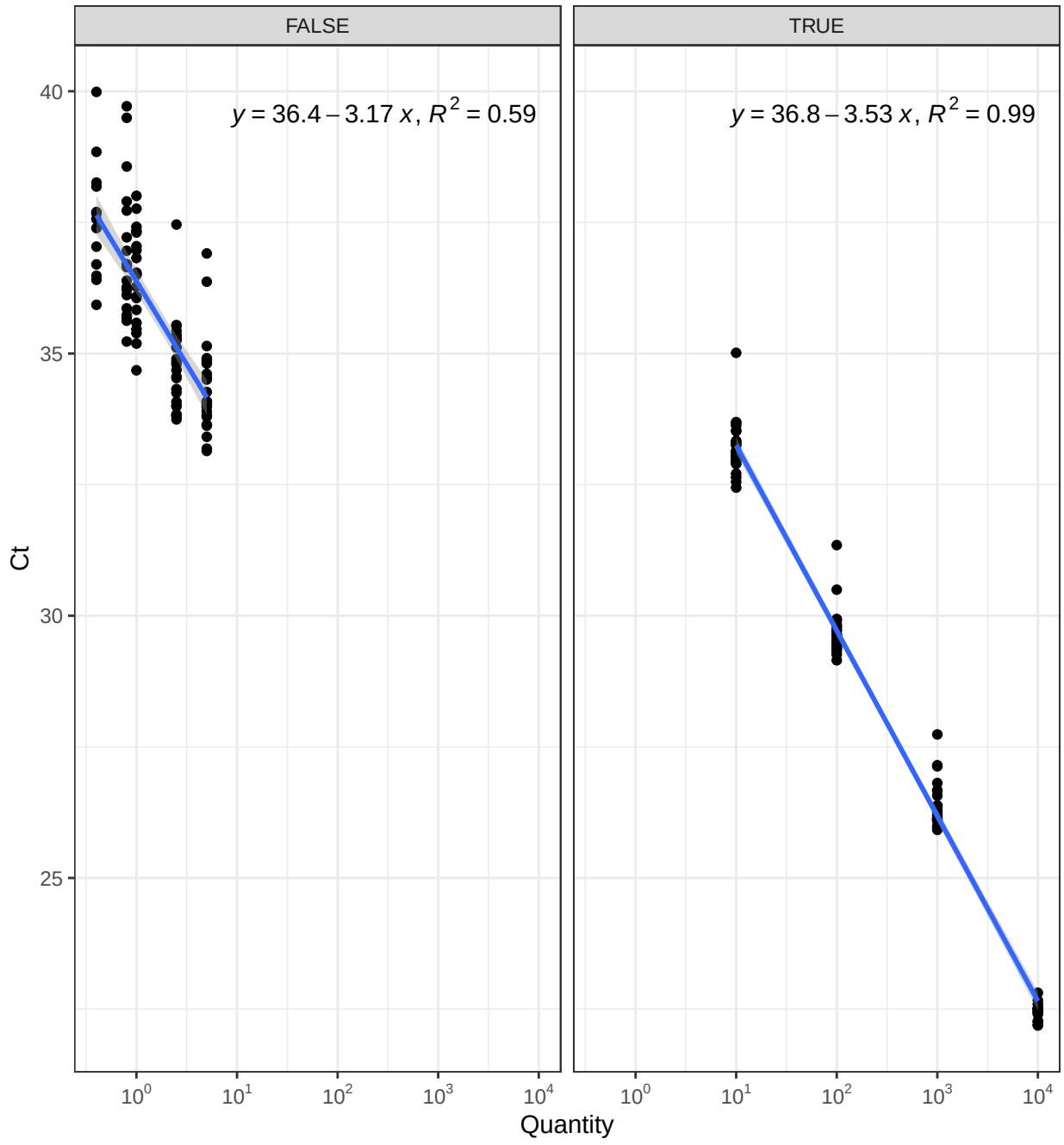
LOQ was determined at the lowest concentration at which the relative standard deviation of back-calculated concentrations was <35%

```
pycno_results_standard %>%
 mutate(., detection = case_when(is.na(CT) ~"Non-detection",
                                 TRUE ~ "Detection")) %>%   group_by(`Well Position`,`Sample Name`) %>%
 mutate(., Observed_Quantity= if_else(detection=="Detection",10^((CT -`Y-Intercept`)/Slope), 0 )) %>%
  mutate(., Concentration_L = Observed_Quantity*100/2 * 1) %>%
  group_by(`Sample Name`,Quantity) %>%
   dplyr::summarise(., mean_conc = round(mean(Concentration_L),2), sd_conc = round(sd(Concentration_L),
```

Sample Name	Quantity	mean_conc	sd_conc	n()	perc_sd_conc
E-0.5	0.4	16.94	20.79	23	122.73
E-0.1	0.8	37.25	34.69	24	93.13
E0_V4	1.0	50.59	41.82	24	82.66
E0.4_V4	2.5	177.66	81.01	24	45.60
E0.7_V4	5.0	238.35	111.48	24	46.77
E1_V4	10.0	474.70	135.31	24	28.50
E2_V4	100.0	5014.39	1147.06	24	22.88
E3_V4	1000.0	45094.94	11707.33	21	25.96
E4_V4	10000.0	634593.60	70796.37	16	11.16

The lowest concentration at which the relative standard deviation of back-calculated concentrations was <35% was between 5 and 10 copies per µL.

```
pycno_results_standard %>%
  ggplot(., aes(x=`Quantity`, y = CT)) +geom_point()  +
    scale_x_log10("Quantity",
        breaks = trans_breaks("log10", function(x) 10^x),
        labels = trans_format("log10", math_format(10^.x))) +
  guides(color = 'legend') +
     scale_size_continuous(breaks = set_breaks)+
     scale_color_continuous(breaks = set_breaks)+
  stat_poly_line() +
  stat_poly_eq(use_label(c("eq", "R2")),label.x = "right",
  label.y = "top") + facet_wrap(~Quantity>9) +theme_bw() +ylab("Ct")
```




```
# Cycle = 41.7-3.52* log10(Quantity)
#  (Cycle -41.7)/-3.52 = log10(Quantity)
```

Lesperance eLowQuant LoD and LoQ

Instructions

This file performs the computations in the publication, 'A Statistical Model for Calibration and Computation of Detection and Quantification Limits for Low Copy Number Environmental DNA samples' by Lesperance,

Allison, Bergman, Hocking, Helbing, *Environmental DNA*, 2021, 00, 1-12 https://doi.org/10.1002/edn3.220. Code obtained from eLowQuant v20220912

- Create a folder call Outputs in your working directory.
- Put your data file in your working directory and put the name of the file in the chunk labeled 'READIN' below.
- Data set csv file requirements. Columns: Target, Lab, Cq, SQ
- For nondetects, set Cq to be empty or NA value
- For negative controls, set Sq to be empty or 0 or NA value
- Include negative controls in the csv file!
- DO NOT DUPLICATE Target names over different Labs!!
- This code uses observations with nonempty data and where phat <1 (num detect<num technical replicates)
- Only the SQ's up to the first one with phat==1 are used.
- Allows for variable numbers of SQ levels per Target.
- Assumes SQs == NA are zero, i.e. are negative controls.
- The code uses the R function optim. A convergence code 0 indicates successful completion.
- Ignore warnings if results are sensible.
- EXECUTE EACH CHUNK LOOKING AT THE OUTPUT. IN PARTICULAR, LOOK AT THE GRAPHS IN THE CHUNK CALLED 'PlotPois' TO DETERMINE IF THE MODEL IS APPROPRIATE. IT IS NOT APPROPRIATE IF 'lm Rsq' is small, i.e. near zero!
- You can send results to files by setting the sink.indicator and Manusink to TRUE and running the code in RStudio. This currently does not work for all results files when knitting.
- If you wish to knit to pdf AND you do NOT have a version of Latex installed on your computer, then run the following in your RStudio console:
  install.packages("tinytex"); tinytex::install_tinytex()

Process/Summarize samples by Lab; Compute the Poisson estimates of SQ

Hindson et al "High-Throughput Droplet Digital PCR System for Absolute Quantitation of DNA Copy Number", Anal. Chem., 2011, 83 (22), pp 8604–8610 use a Poisson approximation for quantitation. Before that, Dube et al. 2008, "Mathematical analysis of copy number variation in a DNA sample using digital PCR on a nanofluidic device", PloS One, Vol 3, Issue 8, e2876, model the number of molecules in each chamber as a Poisson process, giving the relationship between p and λ.

```
## [1] 10  6
```

```
## dim, before negative controls added
```

	Target	SQ	detect	n	Cqmean	Lab
	Pycno:10	Min. : 0.00	Min. : 0.0	Min. :16.00	Min. :22.46	NOAA PMEL OME:10
	NA	1st Qu.: 0.85	1st Qu.:16.5	1st Qu.:23.25	1st Qu.:29.68	NA
	NA	Median : 3.75	Median :20.5	Median :24.00	Median :34.35	NA
	NA	Mean : 1111.97	Mean :18.4	Mean :22.80	Mean :32.42	NA
	NA	3rd Qu.: 77.50	3rd Qu.:24.0	3rd Qu.:24.00	3rd Qu.:36.44	NA
	NA	Max. :10000.00	Max. :24.0	Max. :24.00	Max. :37.55	NA
	NA	NA	NA	NA	NA's :1	NA

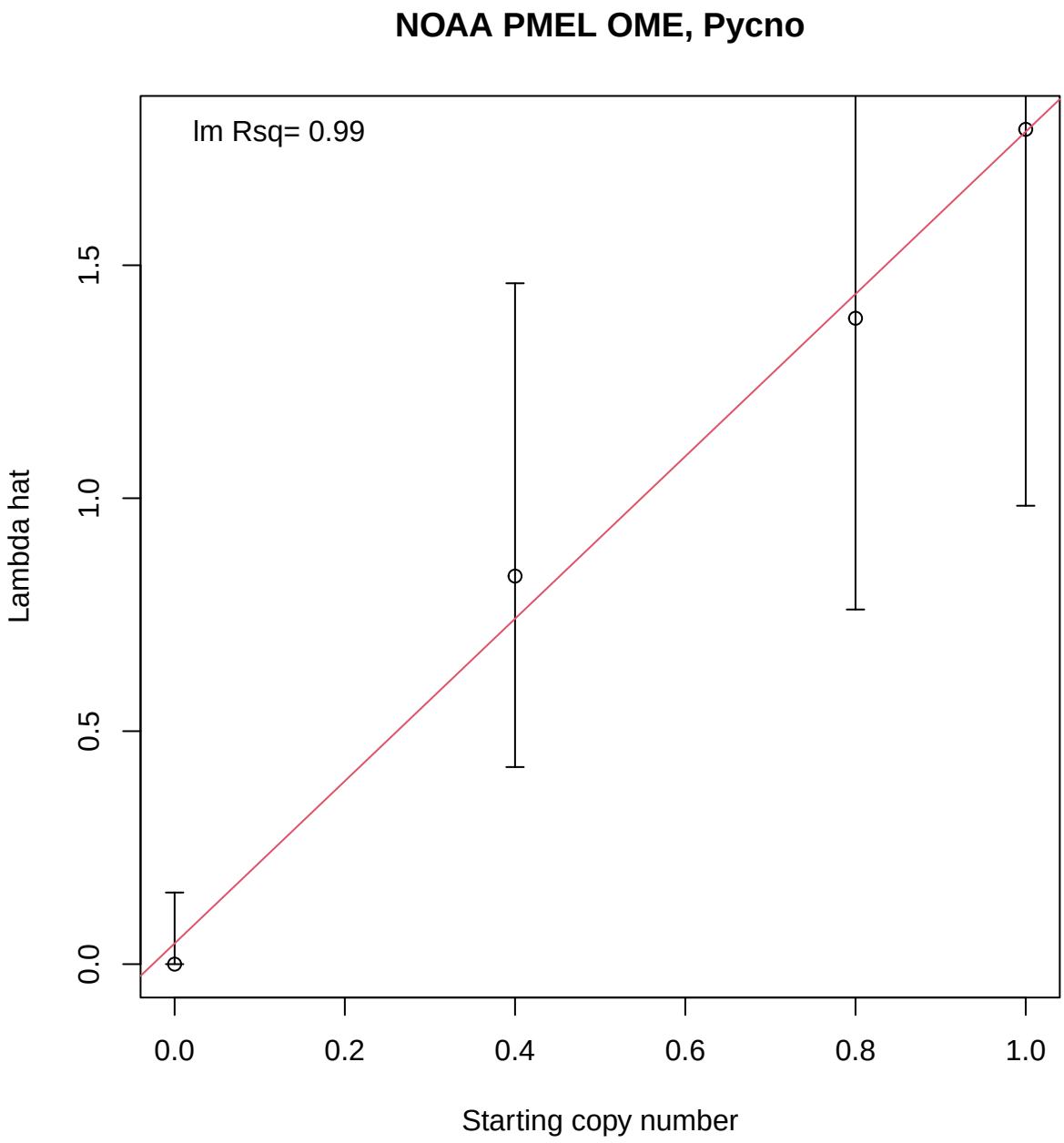
```
##
##
## Table: NOAA PMEL OME
##
## Target          SQ   detect    n    phat
## -------  ---------  -------  ---  ------
## Pycno          0.0         0   24   0.000
## Pycno          0.4        13   23   0.565
## Pycno          0.8        18   24   0.750
## Pycno          1.0        20   24   0.833
## Pycno          2.5        24   24   1.000
## Pycno          5.0        24   24   1.000
## Pycno         10.0        24   24   1.000
## Pycno        100.0        24   24   1.000
## Pycno       1000.0        21   21   1.000
## Pycno      10000.0        16   16   1.000
```

Plot the Poisson estimates (and CI) of SQ for levels that had non-detects

Only the first levels of SQ that had non-detects are analyzed. Red line is least squares linear regression line.

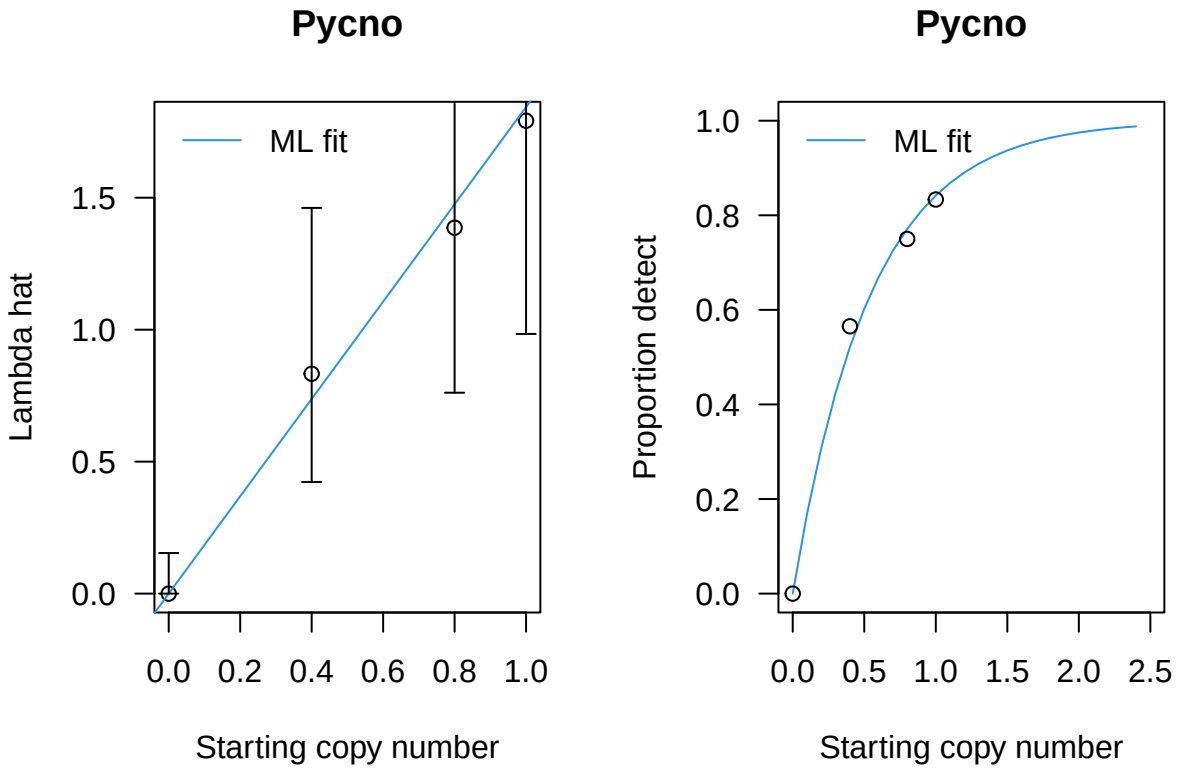
LOOK AT THE GRAPHS IN THE CHUNK CALLED 'PlotPois' TO DETERMINE IF THE MODEL IS APPROPRIATE. IT IS NOT APPROPRIATE IF 'lm Rsq' is small, i.e. near zero!

ML ??will error if all phats==1


NOAA PMEL OME, Pycno
lm Rsq= 0.99
Lambda hat
0.0
0.5
1.0
1.5
0.0
0.2
0.4
0.6
0.8
1.0
Starting copy number

Both the intercept and no intercept models are fit to the data. The 'best' of the two models is the one with the largest Likelihood Ratio test p-value. The 'best' model will be identified in the chunk called *Manuscript*.

Estimate Poisson models - no intercept model




```
##
##
##
##
##  Pycno
## Convergence= 0
##      Estimate Std.Err Z value  Pr(>z)
## beta    1.844   0.282    6.53 6.5e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## LLR test stat= 0.2479441 , df=  3 , p-value= 0.9695018
```

Estimate predicted Sq given number detects and technical replicates - no intercept (not shown)

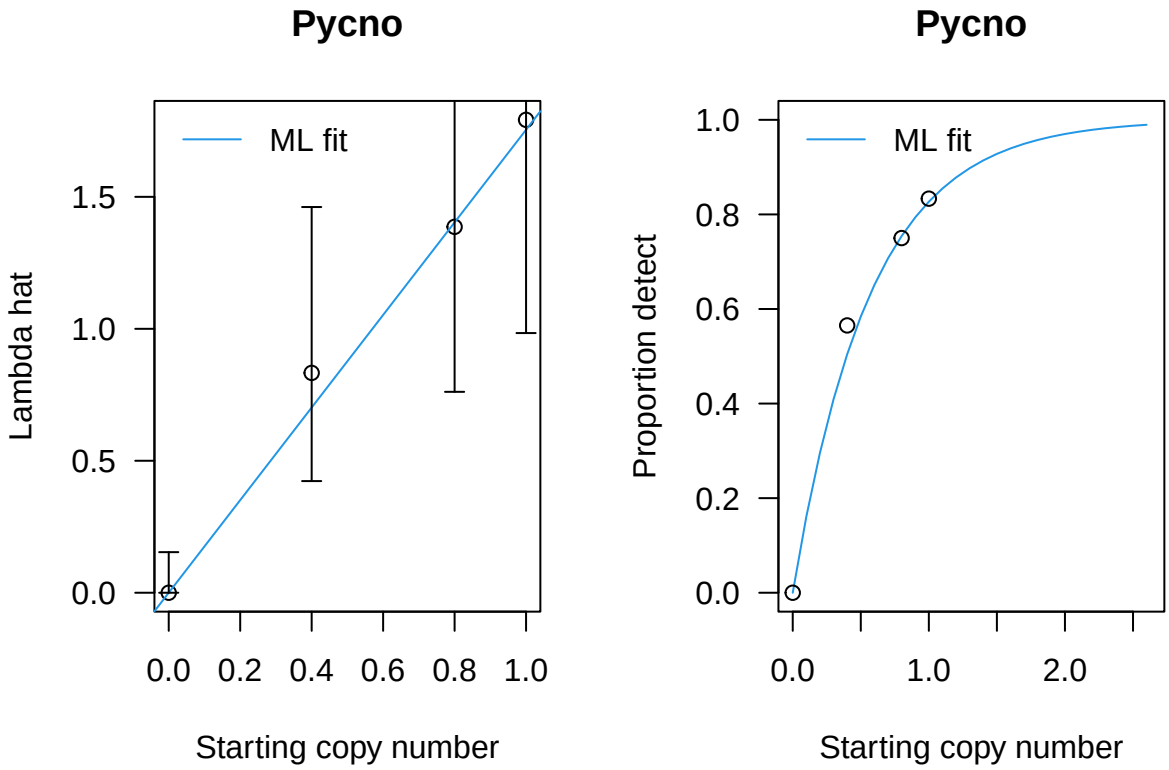
The estimated SQ is easily obtained for given new values of nn0=number of replicates, nd0=number detected and the estimated slope betaS as: Shat <- -(log((nn0 - nd0)/nn0)) / betaS[1] The standard errors are obtained from the Hessian matrix (or via the function CalibS0Or.ddLLik()) .

Estimate predicted Sq given consecutive numbers of detects given number of technical replicates - no intercept

The estimated SQ is easily obtained for given new values of nn0=number of replicates, nd0=number detected and the estimated slope betaS as: Shat <- -(log((nn0 - nd0)/nn0)) / betaS[1] The standard errors are obtained from the Hessian matrix (or via the function CalibS0Or.ddLLik()).

```
##
##  Pycno
##  ML estimate of SQ for numbers of detects and 8 replicates
##
##
##  NumDetects     SQ0   SE_SQ0
## -----------  ------  -------
##           0   0.000      NaN
##           1   0.072    0.073
##           2   0.156    0.113
##           3   0.255    0.154
##           4   0.376    0.200
##           5   0.532    0.261
##           6   0.752    0.352
##           7   1.128    0.536
```

Estimate Poisson models - intercept model




```
##
##
##
##
##  Pycno
## Convergence= 0
##       Estimate  Std.Err Z value Pr(>z)
## alpha 1.49e-08 4.22e-01    0.00  1.000
## beta  1.75e+00 6.94e-01    2.53  0.011 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## LLR test stat= 0.352375 , df=  2 , p-value= 0.8384608
```

Estimate predicted Sq given number detects and technical replicates - intercept model (not shown)

Estimate predicted Sq given consecutive numbers of detects given number of technical replicates - intercept model

The estimated SQ is easily obtained for given new values of nn0=number of replicates, nd0=number detected and the estimated slope betaS as: Shat <- -(log((nn0 - nd0)/nn0) + betaS[1]) / betaS[2] The standard errors are obtained from the Hessian matrix (or via the function CalibS0.ddLLik()).

```
##
##  Pycno
##  ML estimate of SQ for numbers of detects and 8 replicates
##
##
##  NumDetects     SQ0   SE_SQ0
## ------------  ------  -------
##           0   0.000    0.000
##           1   0.056    0.478
##           2   0.146    0.240
##           3   0.250    0.221
##           4   0.376    0.228
##           5   0.537    0.266
##           6   0.769    0.358
##           7   1.165    0.577
```

Determine Lc, Ld, Lq (LOB, LOD, LOQ) - no intercept model

Follows Lavagnini and Magno 2007, Mass Spectrometry Reviews *The notation in the paper was changed from the Lavagnini 2007 paper and is shown in brackets here.*

- Lc (*LOB Limit of blank*) = critical level is the assay signal above which a response is reliably attributed to the presence of analyte
- Ld (*Ld = expected number detects out of NN replicates at concentration LOD*) = signal corresponding to an analyte concentration xd (*=LOD Limit of Detection*) level which may be a priori expected to be recognized
- Lq = quantification limit is a signal with a precision which satisfies an expected value ($= \gamma_Q$)

Lc corresponds to a critical response level or a false positive rate, i.e. critical number of detects given NN replicates, above which we would reject the null hypothesis that the concentration/copy number is zero at alpha = alphaLc ($= \gamma_{FP}$). It is the critical response level corresponding to the false positive rate of alphaLc. Essentially, the test is positive if the Y~Binomial(m, p) > Lc. The False Positive Rate is P(Y > Lc | S=0).

Ld is computed to correspond to the false negative rate, beta = betaLd ($= \gamma_{FN}$) here. It is computed so that the probability of observing a new (unknown concentration) response less than or equal to Lc is less than or equal to betaLd. The probability of observing Lc or less detects if the concentration is xd (*=LOD Limit of Detection*) or more is less than or equal to betaLd. The values of Lc depend on the number of replicates, NN, so xd does as well. Ld is the expected number of detects at values xd (*=LOD Limit of Detection*) and NN replicates. False negative rate Ld computation: P(Y <= Lc | p_xd) <= betaLd, ($= \gamma_{FN}$) and solve for xd (*=LOD Limit of Detection*).

Lq is less well defined. The literature suggests using Lq = beta0 + 10 s.e.(beta0), but this uses the normality assumption. Other literature suggests using the "analyte concentration xq (*=LOQ Limit of Quantification*) for which the experimental relative standard deviation of the responses reaches a fixed level ($= \gamma_Q$), for example, the level 0.1." Lavagnini and Magno 2007. I interpret the term "relative standard deviation" to mean the coefficient of variation, CV = sd/mean.

In the exercise below, we use the fits from the ML models to estimate the Lc, Ld and Lq, for various values of NN replicates for a new observation, i.e. a new (unknown concentration) response number of detects. Both the intercept and no intercept models are considered.

Determine Lc, Ld, Lq (LOB, LOD and LOQ) - intercept model

Estimates, Lc, Ld, Lq (LOB, LOD, LOQ) and confidence limits for a given number of technical reps NN[NNi]

Chooses the model (intercept versus no intercept) with the best LLR test fit, i.e. the largest p-value for the LLR test. A table of values for all assays is printed.

```
## Limits for best choice model for N= 8
```

```
## InterModel      alpha        aSE       beta        bSE         Lc      SdLow
##       0.00       0.00       0.00       1.84       0.28       0.00       0.16
##         Sd       SdUp      SqLow         Sq       SqUp
##       0.20       0.29       0.59       0.77       1.10
```

Tables and Graphs for the manuscript

Revised for general use to use all eligible targets.

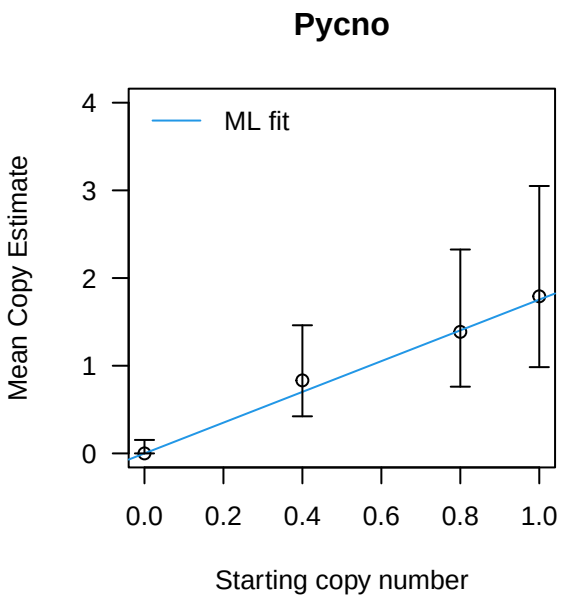
```
##
##
##          alpha   aSE   beta   bSE   LOB   LODL   LOD   LODU   LOQL   LOQ   LOQU
## ------  ------  ----  -----  ----  ----  -----  ----  -----  -----  ----  -----
## Pycno        0   0.4    1.8   0.7     0      0   0.2      1    0.2   0.8    3.6
```

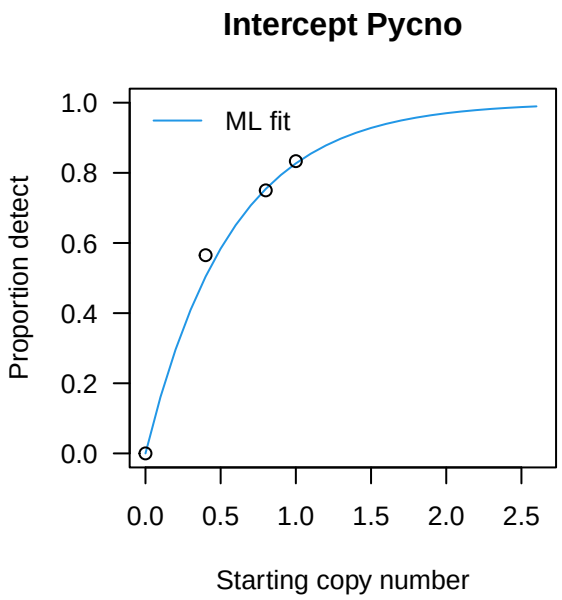
```
##
##
##          alpha   aSE   beta   bSE   LOB   LODL   LOD   LODU   LOQL   LOQ   LOQU
## ------  ------  ----  -----  ----  ----  -----  ----  -----  -----  ----  -----
## Pycno        0     0    1.8   0.3     0    0.2   0.2    0.3    0.6   0.8    1.1
```

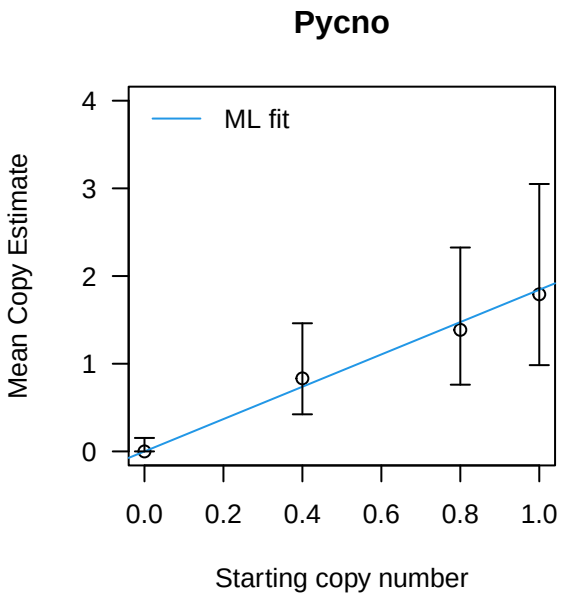
```
##
##
##          alpha   aSE   beta   bSE   LOB   LODL   LOD   LODU   LOQL   LOQ   LOQU
## ------  ------  ----  -----  ----  ----  -----  ----  -----  -----  ----  -----
## Pycno        0     0    1.8   0.3     0    0.2   0.2    0.3    0.6   0.8    1.1
```

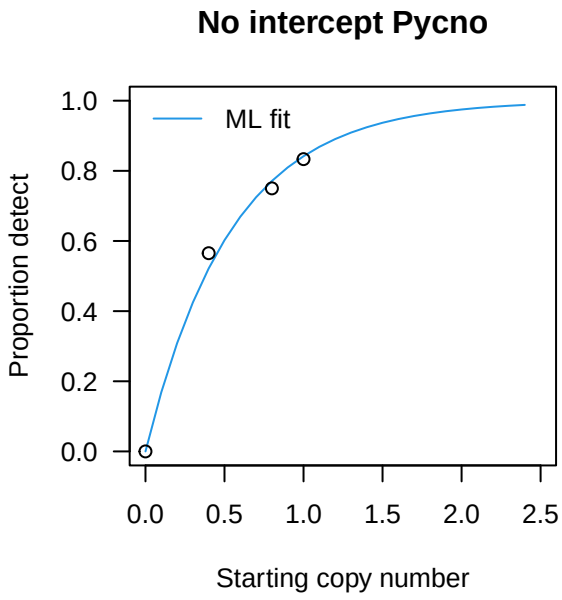
```
##
##
## Table: Pycno
##
##    S   num.detect    n   p.tilde   lambda.tilde
## ----  -----------  ---  --------  -------------
##  0.0            0   24     0.000          0.000
##  0.4           13   23     0.565          0.833
##  0.8           18   24     0.750          1.386
##  1.0           20   24     0.833          1.792
```

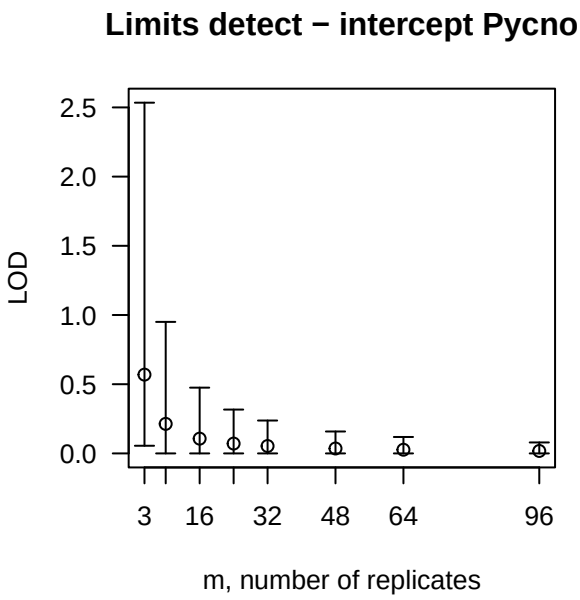
```
##
##  Pycno
## Convergence= 0
##       Estimate  Std.Err Z value Pr(>z)
## alpha 1.49e-08 4.22e-01    0.00  1.000
## beta  1.75e+00 6.94e-01    2.53  0.011 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## LLR test stat= 0.352375 , df=  2 , p-value= 0.8384608
##
##
##  Pycno
## Convergence= 0
##      Estimate Std.Err Z value  Pr(>z)
## beta    1.844   0.282    6.53 6.5e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## LLR test stat= 0.2479441 , df=  3 , p-value= 0.9695018
```

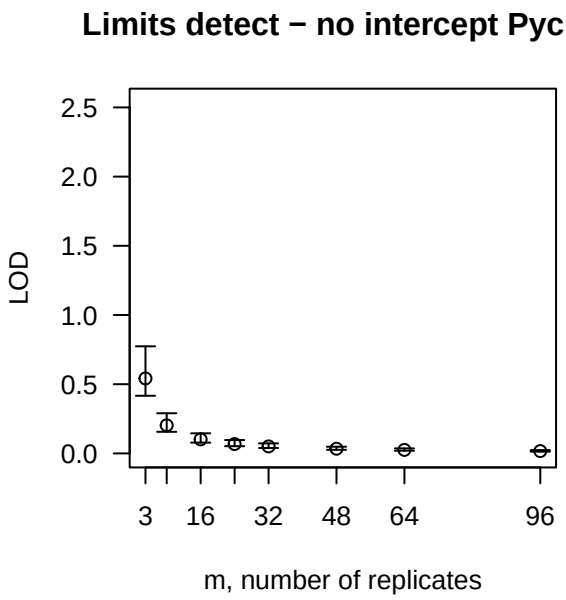

Pycno
ML fit
Mean Copy Estimate
Starting copy number
0
1
2
3
4
0.0
0.2
0.4
0.6
0.8
1.0

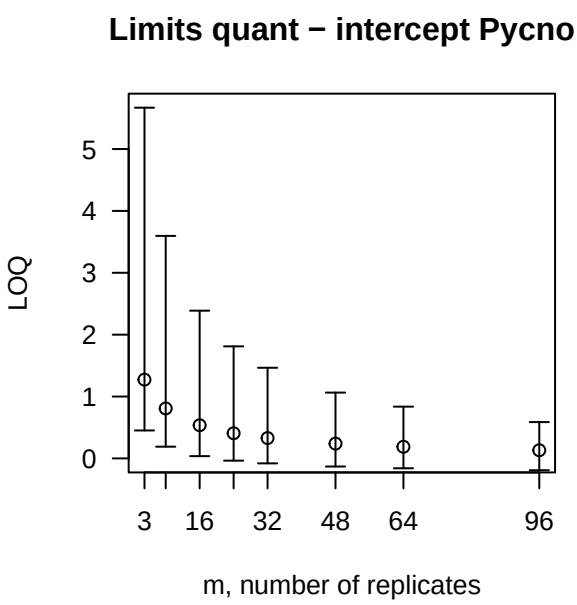

Intercept Pycno
ML fit
Proportion detect
Starting copy number
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.5
1.0
1.5
2.0
2.5

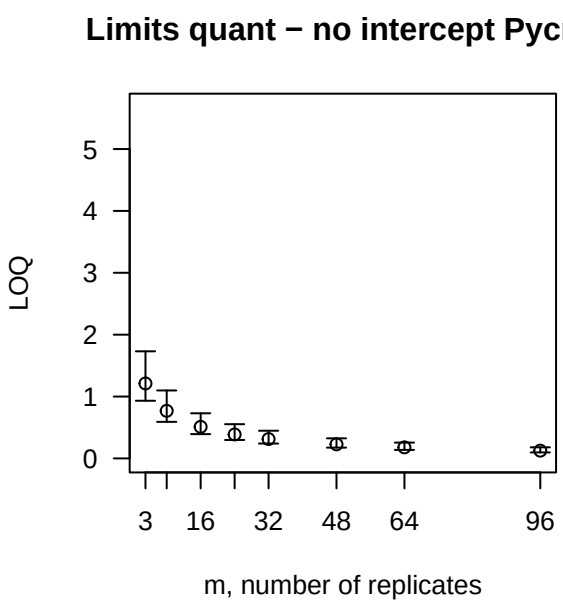

Pycno
ML fit
Mean Copy Estimate
Starting copy number
0
1
2
3
4
0.0
0.2
0.4
0.6
0.8
1.0


No intercept Pycno
ML fit
Proportion detect
Starting copy number
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.5
1.0
1.5
2.0
2.5









FHL Analysis Transplanted Individual Detection

Load Data

```
sample_set_up_run1 <- read_excel(here("2025-05-23_132340_Pycno_test_1.xlsx"), sheet="Sample Setup", ski

amplification_data_run1 <- read_excel(here("2025-05-23_132340_Pycno_test_1.xlsx"), sheet="Amplification
```

Merge Tables

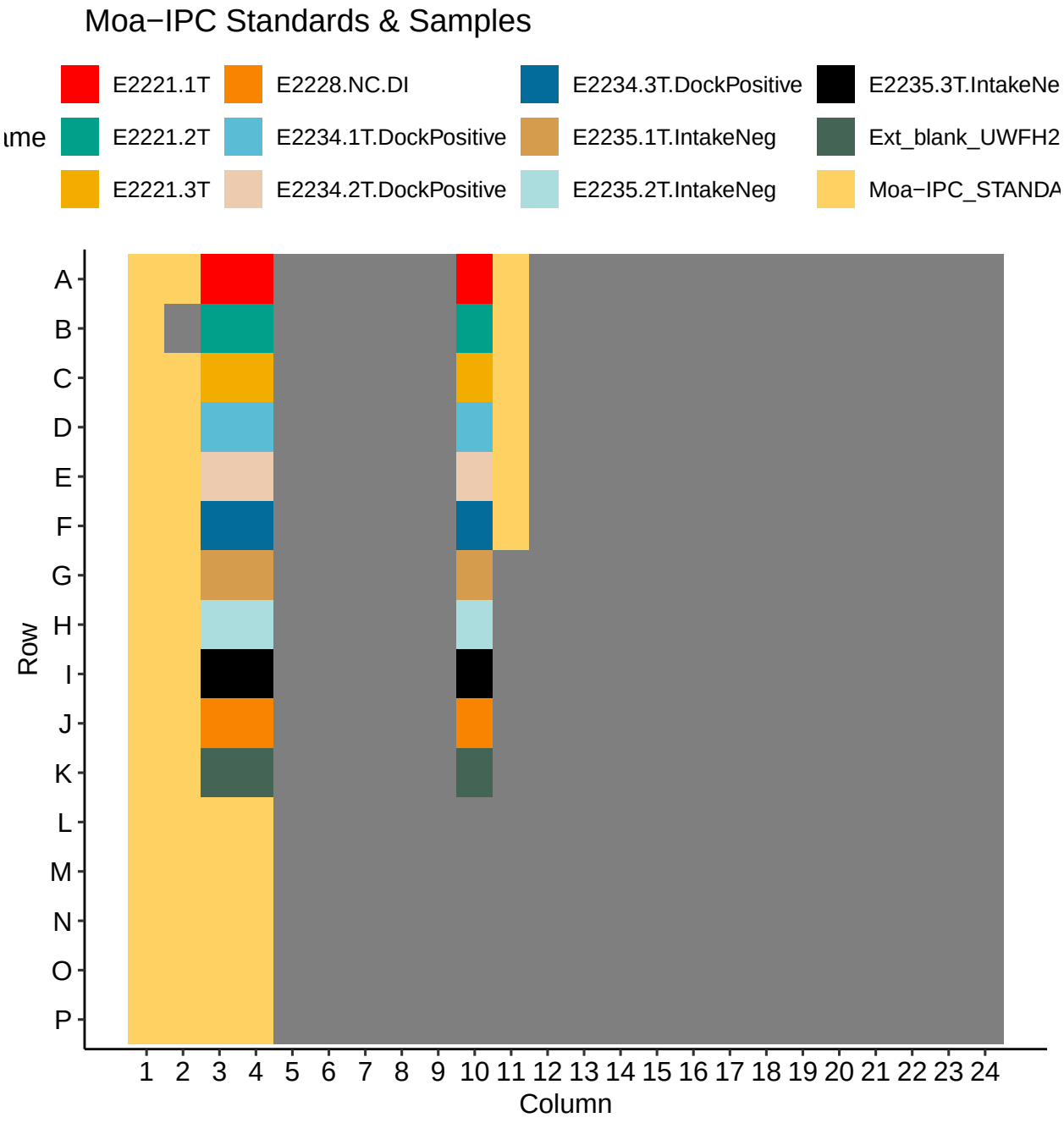
```
amplification_data_run1 %>%
  left_join(sample_set_up_run1) %>%
  mutate(., `Sample Name` = if_else(is.na(`Sample Name`),str_c(`Target Name`,"_",Task,"_", Quantity), `
  mutate(., `Sample Name` = if_else(is.na(`Sample Name`),"Negative Control", `Sample Name` ))-> combine
```

Now we are ready to start analyzing the data.

First thing we want to do is check to see how our internal positive control is working.

Moa Map

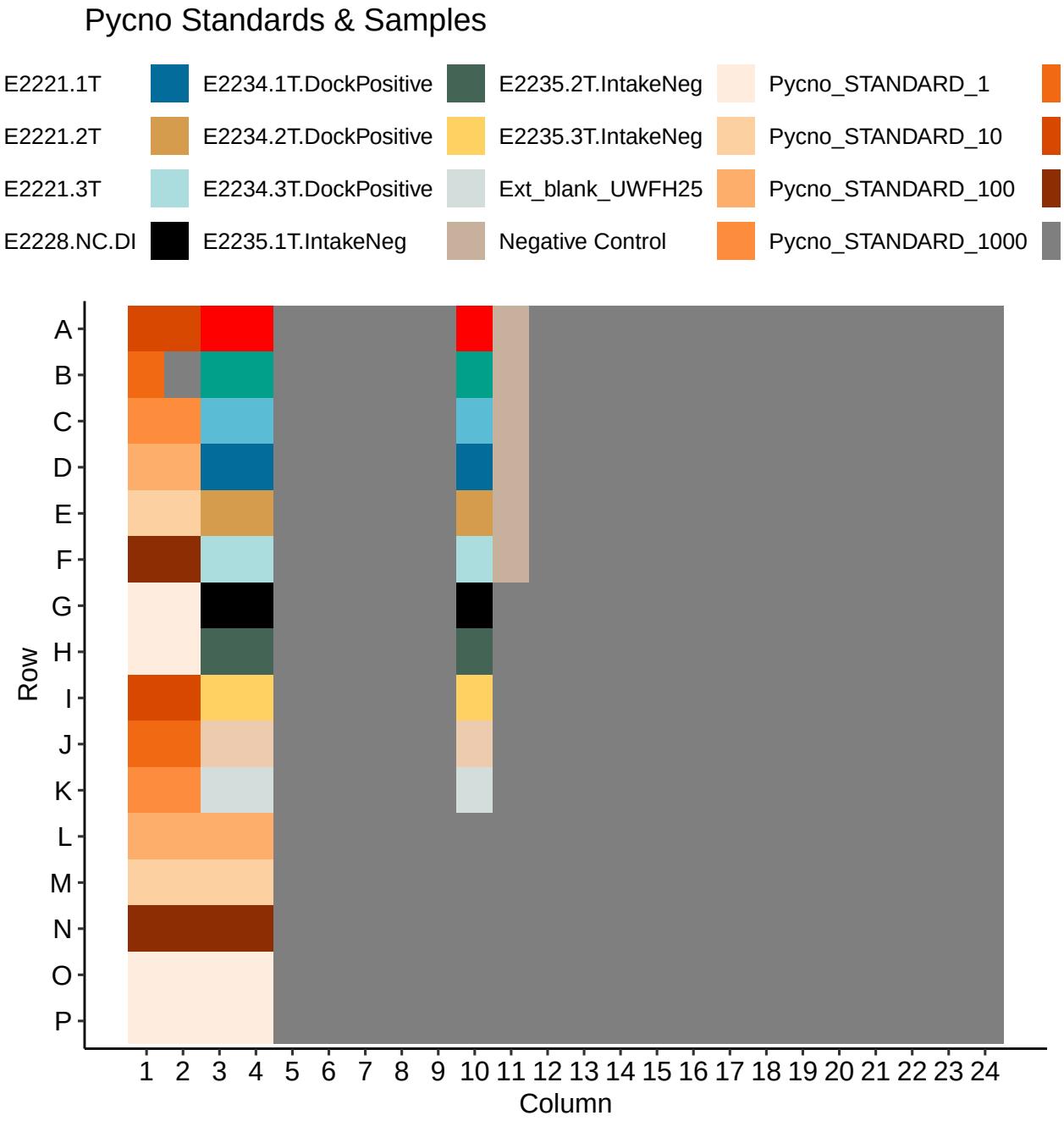
```
columns = seq(from =1, to=24, by=1)
columns_t= as.tibble(columns) %>%  dplyr::rename("Column"=value)
rows = capitalize(letters[1:16])
rows_t= as.tibble(rows) %>%  dplyr::rename("Row"=value)

cross_join(columns_t, rows_t) -> full_plate


combined_qPCR_data_run1 %>%
  filter(., `Target Name`=="Moa-IPC") %>%
  mutate(., Row = str_sub(`Well Position`,start = 1L, end = 1L),
         Column=  as.numeric(str_sub(`Well Position`,start = 2L, end = -1L ))) -> moa_samples_run1

full_plate %>%
  left_join(moa_samples_run1) -> moa_plate_run1


moa_plate_run1 %>%
  mutate(Row = factor(Row, levels = rows)) %>%
  ggplot(aes(x=Column, y=fct_rev(Row), fill=`Sample Name` )) +geom_tile() +scale_x_continuous(breaks=co
```



Pycno Map

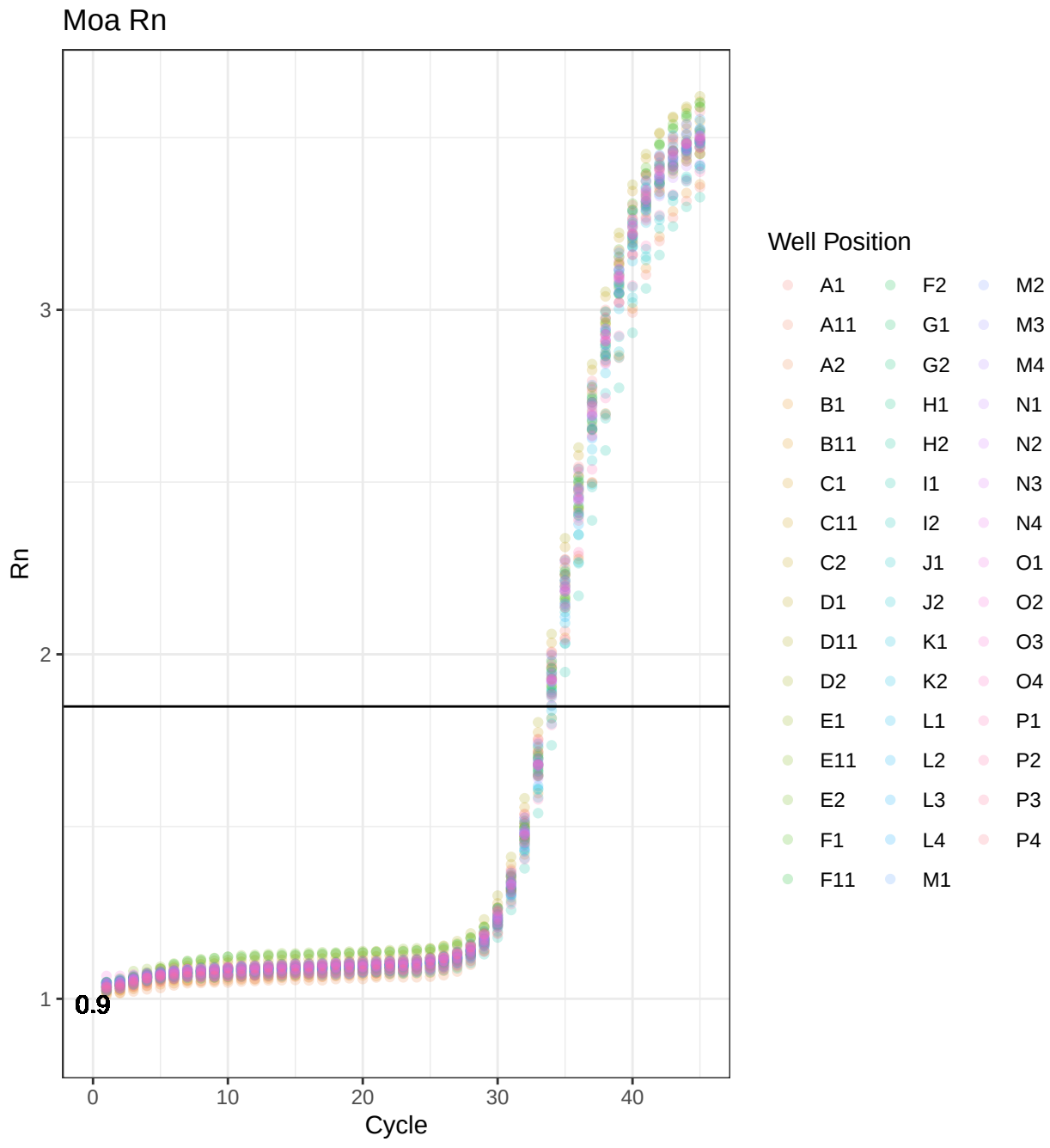
```
combined_qPCR_data_run1 %>%
  filter(., `Target Name`=="Pycno") %>%
  mutate(., Row = str_sub(`Well Position`,start = 1L, end = 1L),
         Column=  as.numeric(str_sub(`Well Position`,start = 2L, end = -1L ))) -> pycno_samples_run1

full_plate %>%
  left_join(pycno_samples_run1) -> pycno_plate_run1
```

```
pycno_plate_run1 %>%
  mutate(Row = factor(Row, levels = rows)) %>%
  ggplot(aes(x=Column, y=fct_rev(Row), fill=`Sample Name` )) +geom_tile() +scale_x_continuous(breaks=co
```



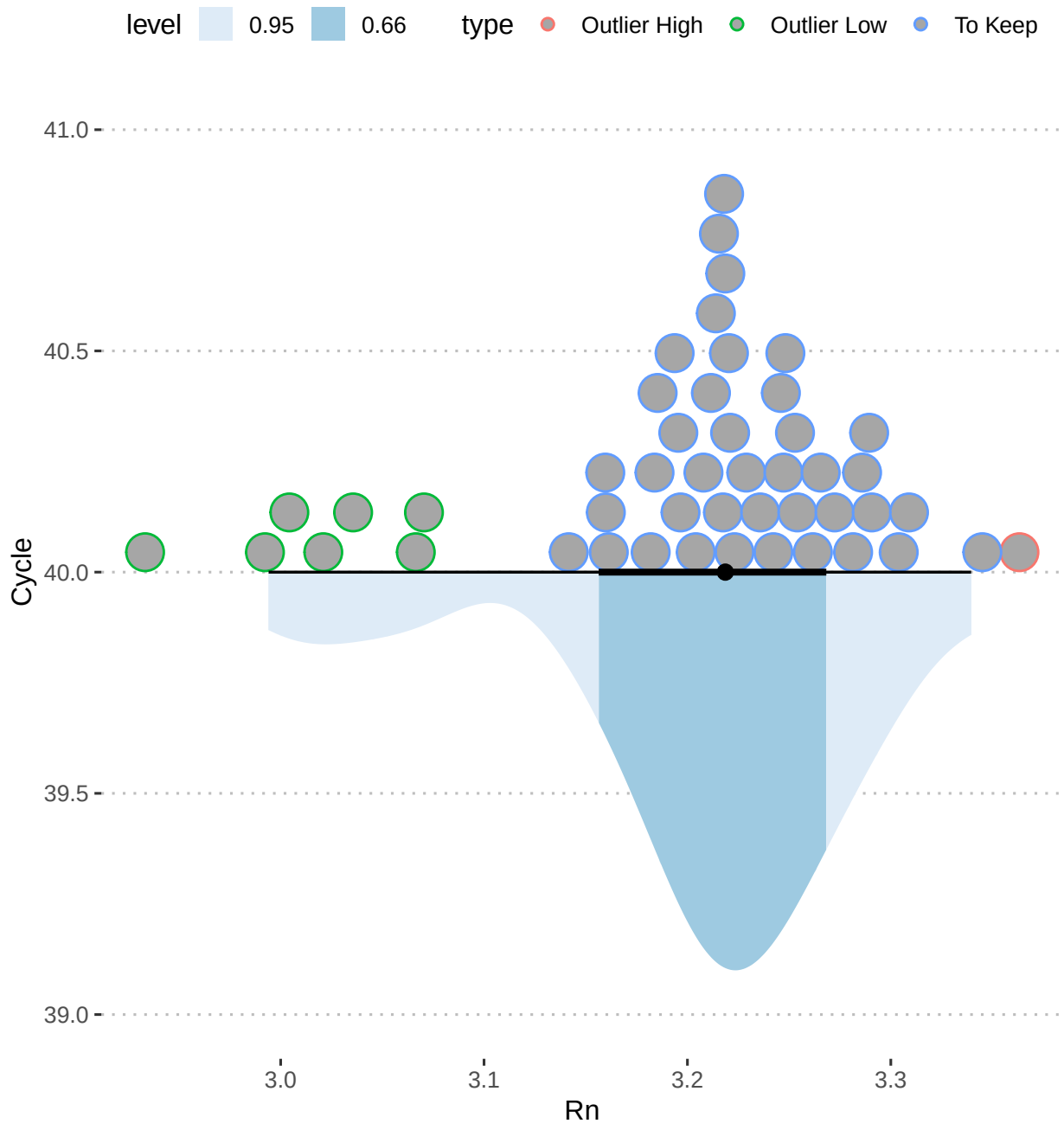
Moa Rn

```
combined_qPCR_data_run1 %>%
filter(., str_detect(`Sample Name`,"Moa-IPC_STANDARD")) -> moa_standard_run1

moa_standard_run1$Rn %>%  max() -> max_Rn_moa_run1
h_moa_run1= max_Rn_moa_run1*0.25

moa_standard_run1 %>%
ggplot(., aes(y=Rn, x=Cycle, color=`Well Position`)) +
  geom_point(alpha=0.2) + geom_hline(yintercept =h_moa) +
  geom_text(aes(0,h_moa_run1,label = round(h_moa_run1,2), vjust = -1),color="black") +theme_bw() +ggtitl
```



There is quite a bit of spread in our IPC. Will remove any samples outside normal distribution. #### Remove Samples with Anomolous Outlier IPC values

```
moa_standard_run1 %>%
  filter(., Cycle ==40) -> moa_standard_c40_run1

moa_standard_c40_run1 %>%
 dplyr::summarise(mean_Rn = ci(Rn, confidence = 0.99)[1],
                       lower_ci_mpg = ci(Rn, confidence = 0.99)[2],
                       upper_ci_mpg = ci(Rn, confidence = 0.99)[3],
                       sd_Rn = ci (Rn, confidence = 0.99)[4]) -> CI0.95_moa_run1

Sum_run1 = groupwiseMean(Rn ~ Cycle, data=moa_standard_c40_run1, conf = 0.99)

moa_standard_c40_run1 %>%
summarise(
    Q1 = quantile(Rn, 0.25, na.rm = TRUE),
    Q3 = quantile(Rn, 0.75, na.rm = TRUE),
    IQR = Q3 - Q1,
    lower_bound = Q1 - 1.5 * IQR,
    upper_bound = Q3 + 1.5 * IQR
  ) -> moa_iqr_run1

moa_standard_c40_run1 %>%
  mutate(., type=case_when(Rn > moa_iqr_run1$upper_bound ~"Outlier High",
                           Rn < moa_iqr_run1$lower_bound ~"Outlier Low",
                           TRUE~"To Keep")) -> moa_standard_c40_run1

moa_standard_c40_run1 %>%
  ggplot(., aes(x= Rn, y=Cycle)) +  geom_dots(layout='weave', side = "top", aes(color=type)) + stat_slab
```



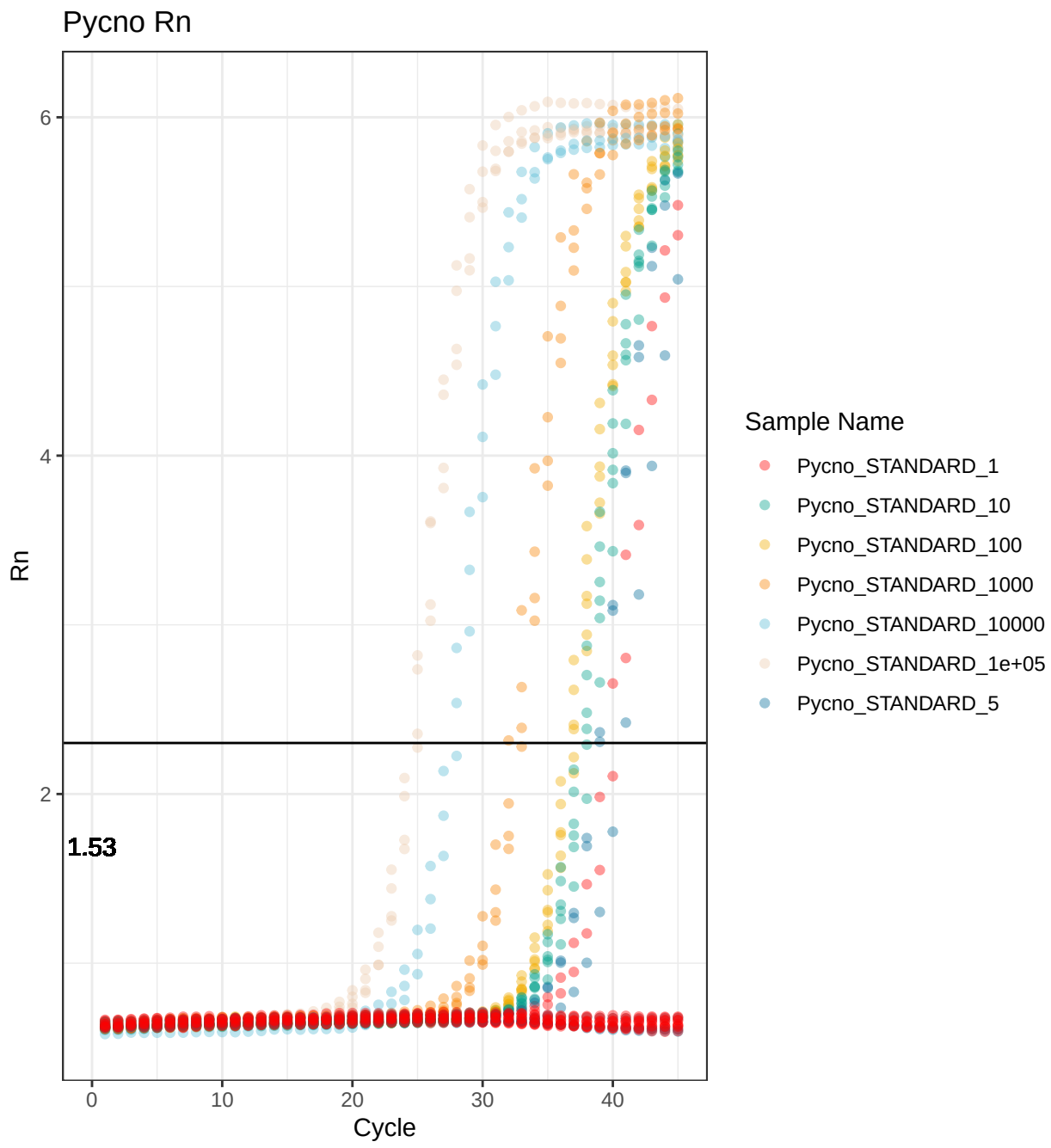

```
moa_standard_c40_run1 %>%
  filter(., type!="To Keep") -> moa_outliers_to_drop_run1
```

```
combined_qPCR_data_run1 %>%
  filter(., `Well Position` != moa_outliers_to_drop_run1$`Well Position`) -> combined_qPCR_data_cleaned
```

Filter Data

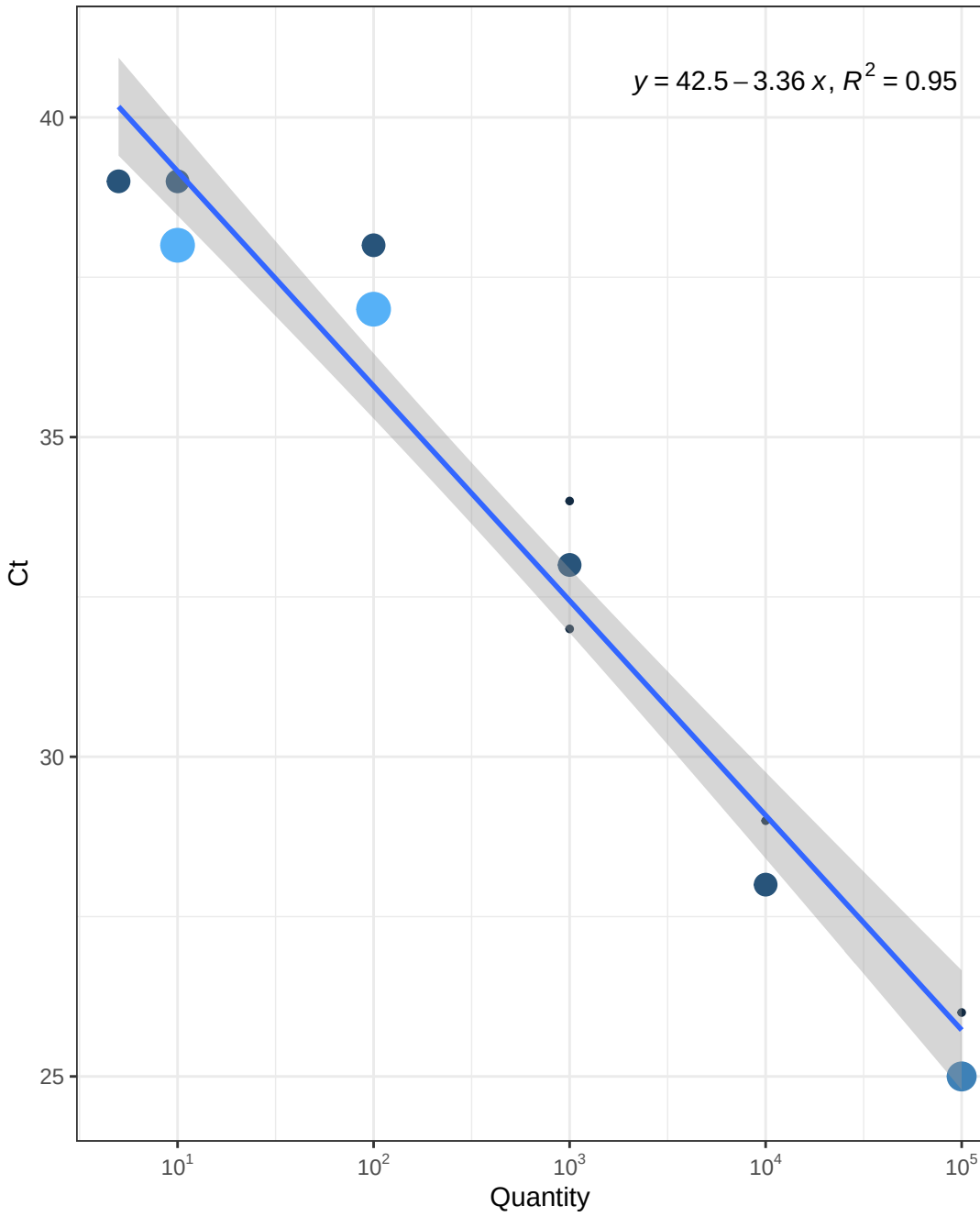
Pycno Rn

```
combined_qPCR_data_run1 %>%
filter(., str_detect(`Sample Name`,"Pycno_STANDARD")) -> pycno_standard_run1

pycno_standard_run1$Rn %>%  max() -> max_Rn_run1
h_run1= max_Rn_run1*0.25

pycno_standard_run1 %>%
ggplot(., aes(y=Rn, x=Cycle, color=`Sample Name`)) +geom_point(alpha=0.4) + geom_hline(yintercept =h) +
  geom_text(aes(0,h_run1,label = round(h_run1,2), vjust = -1),color="black") +theme_bw() + ggtitle("Pyc
```




```
pycno_standard_run1 %>%
  mutate(., Threshold = if_else(Rn > h, "Above","Below")) -> pycno_standard_run1

pycno_standard_run1 %>%
  group_by(`Well Position`,`Sample Name`) %>%
  filter(., rank(Threshold, ties.method="first")==1) %>%
  filter(., Threshold =="Above") -> pycno_standard_above_run1
```

```
pycno_standard_run1 %>%
  ungroup() %>%
  filter(., !`Well Position` %in%  pycno_standard_above$`Well Position`) %>%
  group_by(`Well Position`,`Sample Name`) %>%
  filter(., rank(Threshold, ties.method="last")==1) -> pycno_standard_below_run1

bind_rows(pycno_standard_above_run1, pycno_standard_below_run1) -> pycno_standard_threshold_run1



set_breaks = function(limits) {
     seq(limits[1], limits[2], by = 1)
}

pycno_standard_threshold_run1 %>%
    filter(., Cycle < 40 ) %>%
  ggplot(., aes(x=`Quantity`, y = Cycle)) +geom_count(aes(color = ..n.., size = ..n..))  +
    scale_x_log10("Quantity",
        breaks = trans_breaks("log10", function(x) 10^x),
        labels = trans_format("log10", math_format(10^.x))) +
  guides(color = 'legend') +
     scale_size_continuous(breaks = set_breaks)+
     scale_color_continuous(breaks = set_breaks)+
  stat_poly_line() +
  stat_poly_eq(use_label(c("eq", "R2")),label.x = "right",
  label.y = "top") +theme_bw() +ylab("Ct") -> run1_ct_versus_quant


run1_ct_versus_quant
```



Pycno Ct versus Quantity

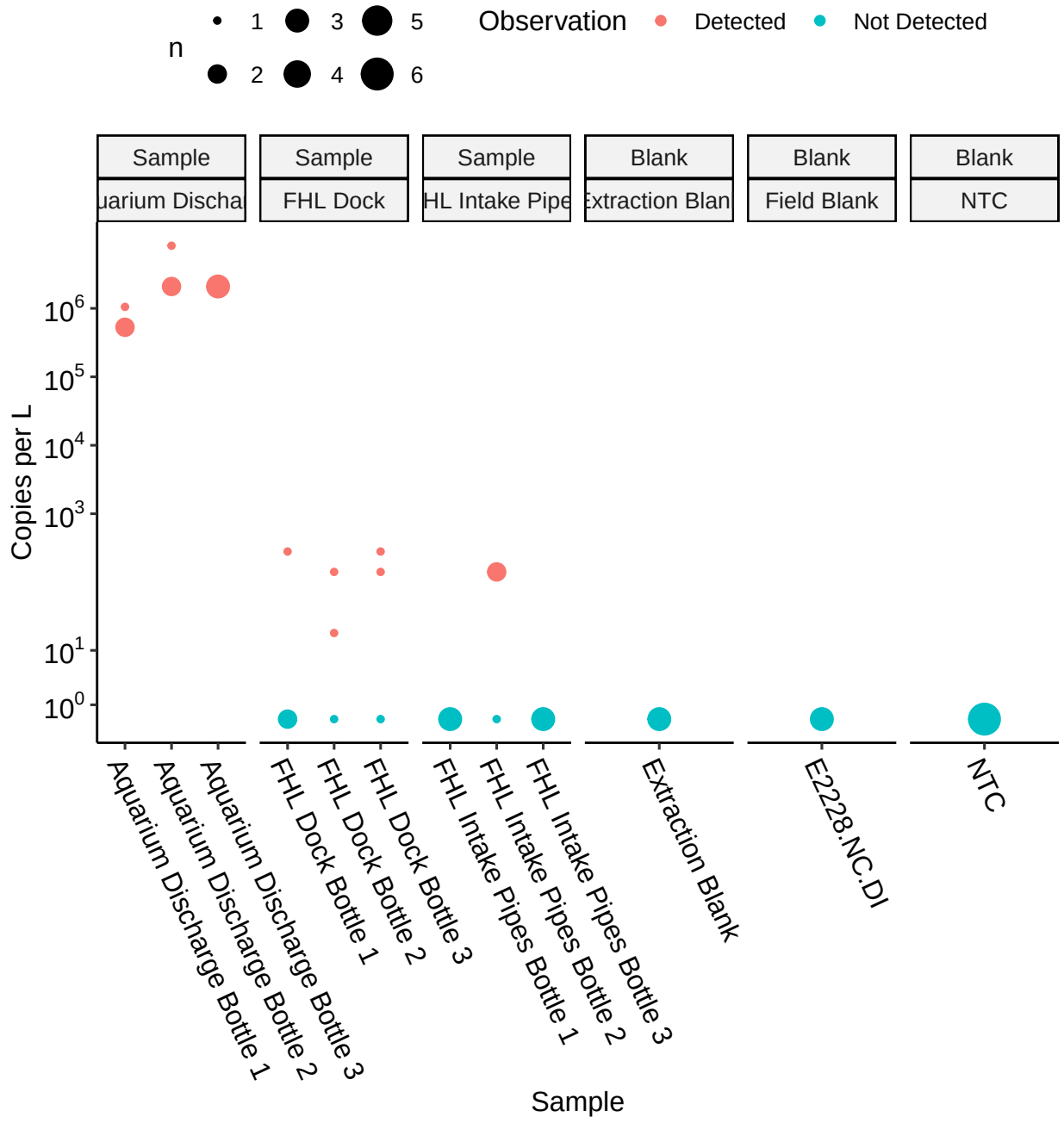
```
# View the structure of the generated plot object
# The computed data is stored within the plot object
grob_data_run1 <- ggplot_build(run1_ct_versus_quant)$data[[3]]

# The result is a list; unlist it to get a character vector
coef_char_run1 <- unlist(grob_data_run1$coefs)


standard_curve_coefs_run1 <- enframe(coef_char_run1, name = "Metric", value = "Value") %>%
 mutate( Metric = recode(Metric,
                         "(Intercept)" = "Intercept",
```

```
                                "x" = "Slope"))

# Extract the R^2 value from the 'r.squared' column
r_squared_value_run1 <- grob_data_run1$r.squared
```

Merge Sample Metadata

```
sample_metadata<- read_excel(here("ElectronicArchive_CruiseSampleMetadata_20250516_FHL.xlsx"), sheet="E

combined_qPCR_data_run1 %>%
   filter(.,`Target Name`=="Pycno") %>%
  filter(., !str_detect(`Task`,"STANDARD")) %>%
      mutate(., Threshold = if_else(Rn > h, "Above","Below")) %>%
  group_by(`Well Position`,`Sample Name`) %>%
  filter(., rank(Threshold, ties.method="first")==1) %>%
  mutate(., Quantity= if_else(Threshold=="Above",10^((Cycle -standard_curve_coefs_run1$Value[[1]])/stan
    mutate(., Quantity=if_else(is.na(Quantity), 0, Quantity)) %>%
  mutate(., Concentration = Quantity*100/2 * 1) %>%
  mutate(., Observation = if_else(Quantity ==0, "Not Detected", "Detected")) %>%
  mutate(., `Sample Name`= if_else(str_detect(`Sample Name`,"Negative Control"), "NTC",`Sample Name`)) %
  ungroup() %>%
  left_join(sample_metadata, by=c("Sample Name"="Field_Name")) %>%
  mutate(., `Sample Name` = if_else(`Sample Name`=="Ext_blank_UWFH25","Extraction Blank",`Sample Name`)
  mutate(., Negative_control = if_else(is.na(Negative_control), TRUE, Negative_control)) %>%
  group_by(`Sample Name`) %>%
  mutate(Technical_Replicate = row_number(`Sample Name`)) %>%
  mutate(., Station = if_else(is.na(Station) & Negative_control==TRUE | Task=="NTC", `Sample Name`, Sta
  mutate(., Name_plotting = str_c(Station," Bottle ", Biological_Replicate," PCR ",Technical_Replicate
  mutate(., Name_plotting = if_else(is.na(Name_plotting),str_c(`Sample Name`, " PCR ",Technical_Replica
   mutate(., Name_plotting_b = str_c(Station," Bottle ", Biological_Replicate)) %>%
  mutate(., Name_plotting_b = if_else(is.na(Name_plotting_b),`Sample Name`,Name_plotting_b)) %>%
  mutate(., Type = if_else(is.na(Station),"Extraction Blank",Station )) %>%
   mutate(., Type = if_else(str_detect(Type,".NC"),"Field Blank",Type )) %>%
  mutate(., Sample_Type = if_else(Negative_control==FALSE,"Sample","Blank")) -> pycno_sample_data_4_plo
```

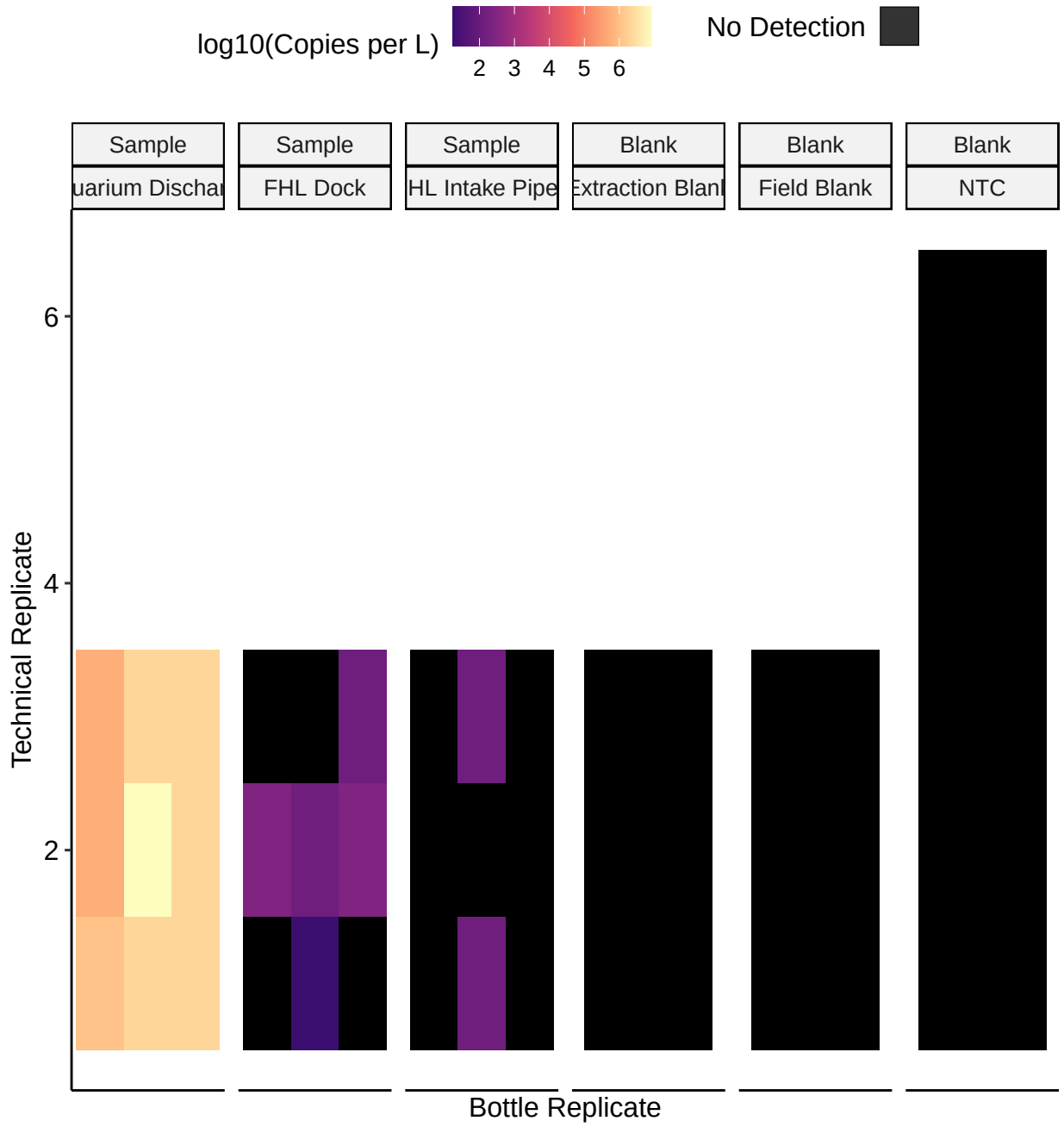
```
pycno_sample_data_4_plotting_run1 %>%
     mutate(across(Sample_Type, ~factor(., levels=c("Sample","Blank")))) %>%
  ggplot(., aes(x=Name_plotting_b, y=Concentration, color=Observation)) + geom_count() +
    scale_y_continuous(
         trans = scales::pseudo_log_trans(base = 10),
         breaks = c(1,10,1000,10000,100000,1000000),
         labels = scales::trans_format("log10", scales::math_format(10^.x))
     )    +theme_pubr()+theme(axis.text.x = element_text(angle = -65, vjust = 1, hjust=0)) +facet_wrap(
```




```
# pycno_sample_data_4_plotting_run1 %>%
#   filter(., str_detect(Name_plotting_b,"Aquarium")) %>%
#   group_by(`Target Name`) %>%
#   dplyr::summarise(mean(Concentration))
```

```
pycno_sample_data_4_plotting_run1 %>%
       mutate(across(Sample_Type, ~factor(., levels=c("Sample","Blank")))) %>%
  group_by(Name_plotting_b) %>%
  mutate(., Replicate = row_number()) %>%
  ungroup() %>%
```

```
  mutate(., Concentration=if_else(Concentration==0,NA,Concentration) ) %>%
  ggplot(., aes(x=Name_plotting_b, y=Replicate, fill=log10(Concentration), color="")) + geom_tile() +th
     guides(colour=guide_legend("No Detection", override.aes=list(colour="black"))) +
  labs(fill = "log10(Copies per L)")
```



Here we see ~2 million copies per L of Pycnopodia target DNA in the Friday Harbor Aquarium discharge water. The discharge is from a flow through system with 39 individual stars of at least 40 cm in diameter.

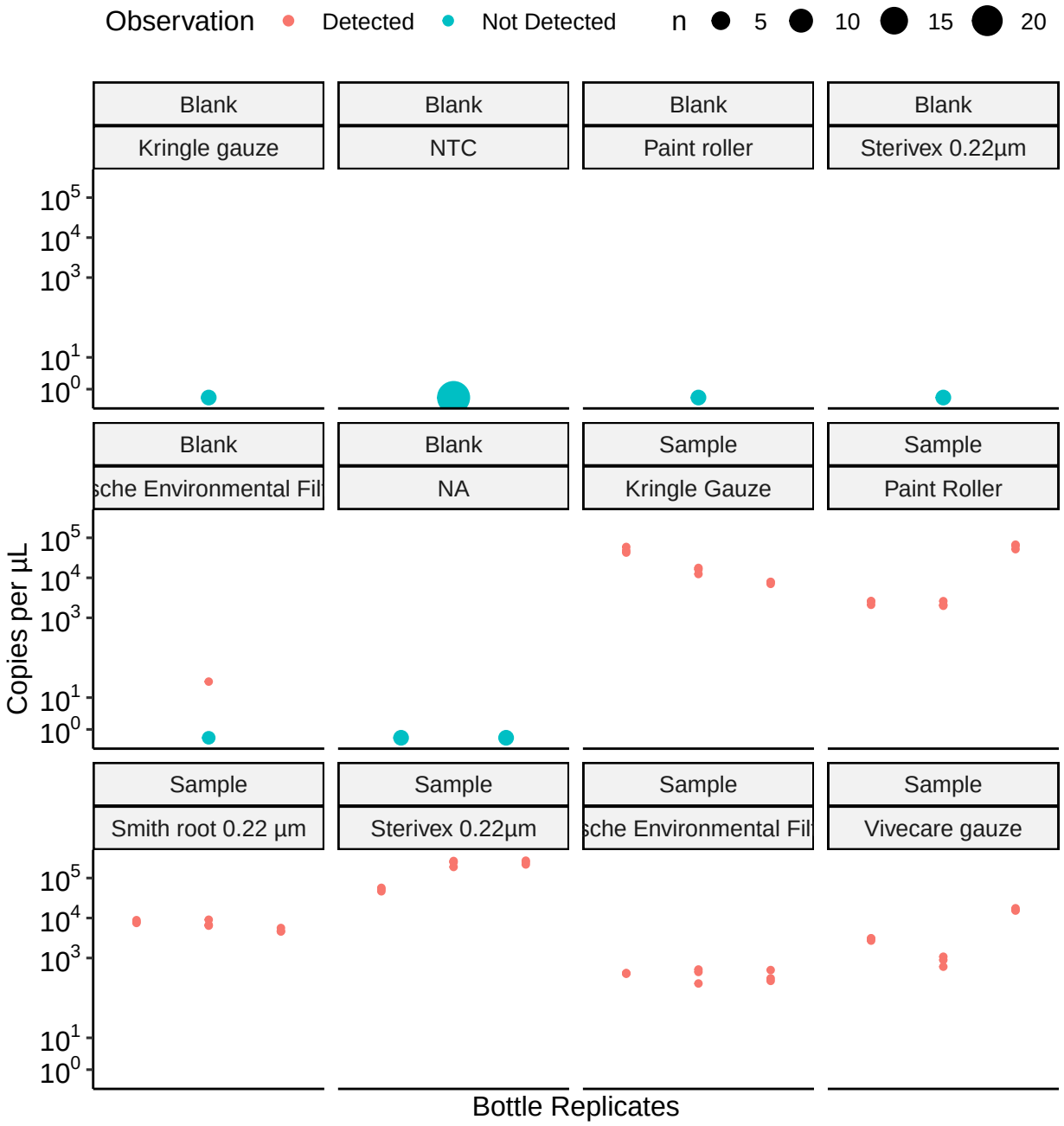
We were able to detect the 10 transplanted stars from at least one technical replicate from each of the three surface eDNA samples taken ~5m above the individuals. We only detected a single individual located at 10m depth from one of three surface samples taken at the Friday Harbor Lab intake pipes.

These results demonstrate that the eDNA qPCR assay can detect Pycnopodia from both mesocosms and the field.

FHL Analysis Passive Versus Active Filtration

```
results_data_cleaned %>%
  filter(.,`Target Name`=="Pycno") %>%
  filter(., !str_detect(`Task`,"STANDARD")) %>%
    mutate(., `Sample Name`=str_replace(`Sample Name`,",",".")) %>%
  left_join(sample_metadata, by=c("Sample Name"="Sample_Name")) %>%
  group_by(`Sample Name`) %>%
    mutate(., Negative_control = if_else(is.na(Negative_control), TRUE, Negative_control)) %>%
  mutate(Technical_Replicate = row_number(`Sample Name`)) %>%
  mutate(., Station = if_else(is.na(Station) & Negative_control=="TRUE" | Task=="NTC", `Sample Name`, S
  mutate(., Name_plotting = str_c(Station," ",Field_collection_method," ", Biological_Replicate," ",Tec
  mutate(., Name_plotting = if_else(is.na(Name_plotting),str_c(`Sample Name`, " ",Technical_Replicate),
   mutate(., Name_plotting_b = str_c(Station," ",Field_collection_method," ", Biological_Replicate)) %>
  mutate(., Name_plotting_b = if_else(is.na(Name_plotting_b),`Sample Name`,Name_plotting_b)) %>%
  mutate(., Quantity=if_else(is.na(Quantity), 0, Quantity)) %>%
  mutate(., Concentration_L = Quantity*100/2 * 1) %>%
  mutate(., Observation = if_else(Quantity ==0, "Not Detected", "Detected")) %>%
  mutate(., Type = if_else(is.na(Station),"Extraction Blank",Station )) %>%
   mutate(., Type = if_else(str_detect(Type,".NC"),"Field Blank",Type )) %>%  mutate(., Sample_Type = i
  mutate(., Field_collection_method = if_else(Task=="NTC","NTC",Field_collection_method)) %>%
mutate(., Field_collection_method = if_else(Type=="Extraction Blank","Extraction Blank",Field_collectio
```

```
pycno_sample_data_4_plotting %>%
  filter(., !str_detect(Name_plotting_b, "FHL")) %>%
  ggplot(., aes(x=Name_plotting_b, y=Concentration_L, color=Observation)) + geom_count() +
    scale_y_continuous(
         trans = scales::pseudo_log_trans(base = 10),
         breaks = c(1,10,1000,10000,100000,1000000,10000000,100000000),
         labels = scales::trans_format("log10", scales::math_format(10^.x))
     ) +theme_pubr()+theme(axis.text.x = element_text(angle = -65, vjust = 1, hjust=-0.05)) +facet_wrap
```



We had minor contamination in the passive filter. It's 2 orders of magnitude less than any of the samples so not a major concern. Currently not adjusting the values here, but could subtract this value from all observed positive results.

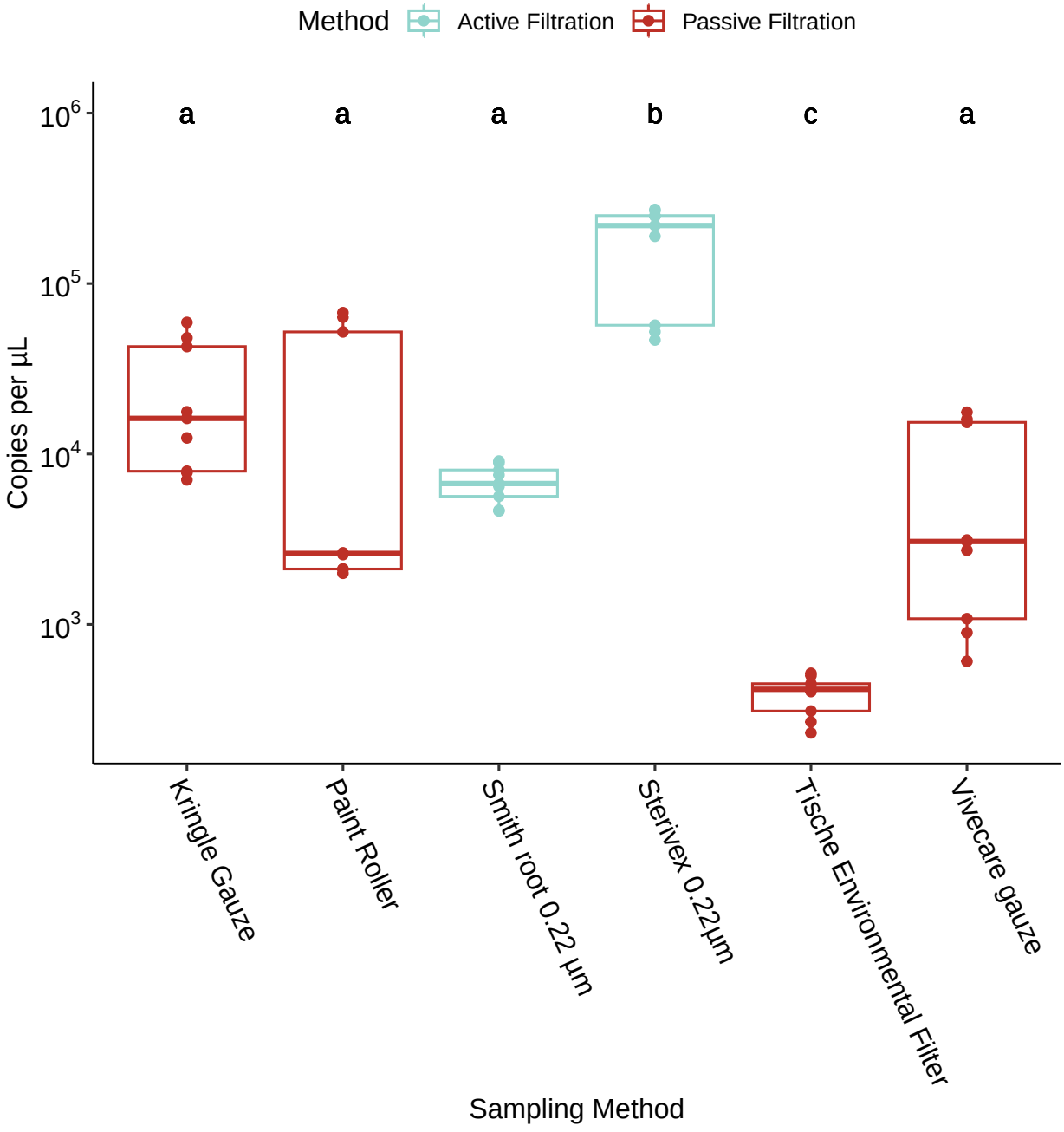
```
pycno_sample_data_4_plotting %>%
  filter(., !str_detect(Name_plotting_b, "FHL")) %>%
  filter(., Sample_Type!="Blank") %>%
  mutate(., Method= case_when(str_detect(Field_collection_method,"Sterivex")~"Active Filtration",str_de
                  TRUE~"Passive Filtration"            )) ->pycno_sample_data_4_plotting_fm
```

Kruskal Wallace test

```
# Perform one-way ANOVA
kw_test_results <-  kruskal.test(Concentration_L ~ Field_collection_method, pycno_sample_data_4_plottin

# Perform post-hoc Tukey HSD test
pairwise.wilcox.test(pycno_sample_data_4_plotting_fm$Concentration_L, pycno_sample_data_4_plotting_fm$F
                p.adjust.method = "bonferroni") -> pairwise_results

full_p_matrix <- fullPTable(pairwise_results$p.value)
multcompLetters(full_p_matrix) -> cld

cld_tibble_optionA <- as_tibble(cld$Letters, rownames = "group") %>%
  dplyr::select(Field_collection_method=group, Letters=value)

pairwise_results
```

```
##
##  Pairwise comparisons using Wilcoxon rank sum exact test
##
## data:  pycno_sample_data_4_plotting_fm$Concentration_L and pycno_sample_data_4_plotting_fm$Field_coll
##
##                             Kringle Gauze Paint Roller Smith root 0.22 µm
## Paint Roller                1.00000       -            -
## Smith root 0.22 µm          0.08453       1.00000      -
## Sterivex 0.22µm             0.00740       0.02777      0.00062
## Tische Environmental Filter 0.00062       0.00062      0.00062
## Vivecare gauze              0.21288       1.00000      1.00000
##                             Sterivex 0.22µm Tische Environmental Filter
## Paint Roller                -               -
## Smith root 0.22 µm          -               -
## Sterivex 0.22µm             -               -
## Tische Environmental Filter 0.00062         -
## Vivecare gauze              0.00062         0.00062
##
## P value adjustment method: bonferroni
```

```
pycno_sample_data_4_plotting_fm %>%
  left_join(cld_tibble_optionA) %>%
  ggplot(., aes(x=Field_collection_method, y=Concentration_L)) +
  geom_boxplot(aes(color=Method)) +
  geom_point(aes(color=Method)) +
  scale_y_continuous(
         trans = scales::pseudo_log_trans(base = 10),
         breaks = c(1,10,1000,10000,100000,1000000,10000000,100000000),
         labels = scales::trans_format("log10", scales::math_format(10^.x))
     ) +theme_pubr()+theme(axis.text.x = element_text(angle = -65, vjust = 1, hjust=0))  + ylab("Copies
```




```
pycno_sample_data_4_plotting %>%
  filter(., !str_detect(Name_plotting_b, "FHL")) %>%
  filter(., Sample_Type!="Blank") %>%
  mutate(., Method= case_when(str_detect(Field_collection_method,"Sterivex")~"Active Filtration",str_de
                  TRUE~"Passive Filtration"            )) %>%
  group_by(Field_collection_method, Method) %>%
  dplyr::summarize(Max=max(Concentration_L), Min=min(Concentration_L), Mean=mean(Concentration_L), media
    IQR = IQR(Concentration_L)) %>% left_join(cld_tibble_optionA) %>% kable()
```

Field_collection_method	Method	Max	Min	Mean	median	IQR	Letters
Kringle Gauze	Passive Filtration	59142.9810	7050.7278	24327.357	16182.516	34791.4581	a
Paint Roller	Passive Filtration	67489.0137	1996.7890	21888.519	2608.771	49932.2369	a
Smith root 0.22 µm	Active Filtration	9090.7272	4630.6812	6851.241	6709.276	2415.3362	a
Sterivex 0.22µm	Active Filtration	272054.760	746670.1111	177984.776	218896.167	193436.0779	b
Tische Environmental Filter	Passive Filtration	516.8524	231.0910	390.684	416.378	138.7463	c
Vivecare gauze	Passive Filtration	17594.0338	606.8627	6715.402	3066.180	14260.6516	a

Here we see that Sterivex 0.22 µm filters captured nearly an order of magnitude more Pycnopodia DNA than either Smith Root active filters or any of the passive filters deployed here. This is notable given that Sterivex have a smaller surface area (10cm^2) than the Smith Root 47mm filter (17.3 cm^2) and suggests that differences in the housing or sample integrity are driving these apparent differences. These results highlight that passive filters can be used for field applications to detect Pycnopodia, albeit at a significant disadvantage in that even the best performing materials captured less than a 10th of the DNA captured by active sterivex filtration methods.