Untitled

Load data

Load datasets and mark DLI patients. Check number of clonotypes in donors and receptients - we have engough clones for statistics everywhere. We group clones by their abundance in donors: 1 (singletons), 2 (doubletons), 3 (tripletons, as good as doubletons, no need to hate them Vanya) and 4+ reads (Large). The choice is dictated by observing the fact that for rare events Poisson distribution shows huge difference in capture probability for $\lambda \in [1, 3]$ while smaller λ values are unlikely to be encountered and quantified. Moreover, for large clones, each hyperexpanded variant has its own history and, likely, its own dynamic, so binning them to different bins based on minor differences in frequency (e.g. 0.1% vs 0.01%) makes little sense.

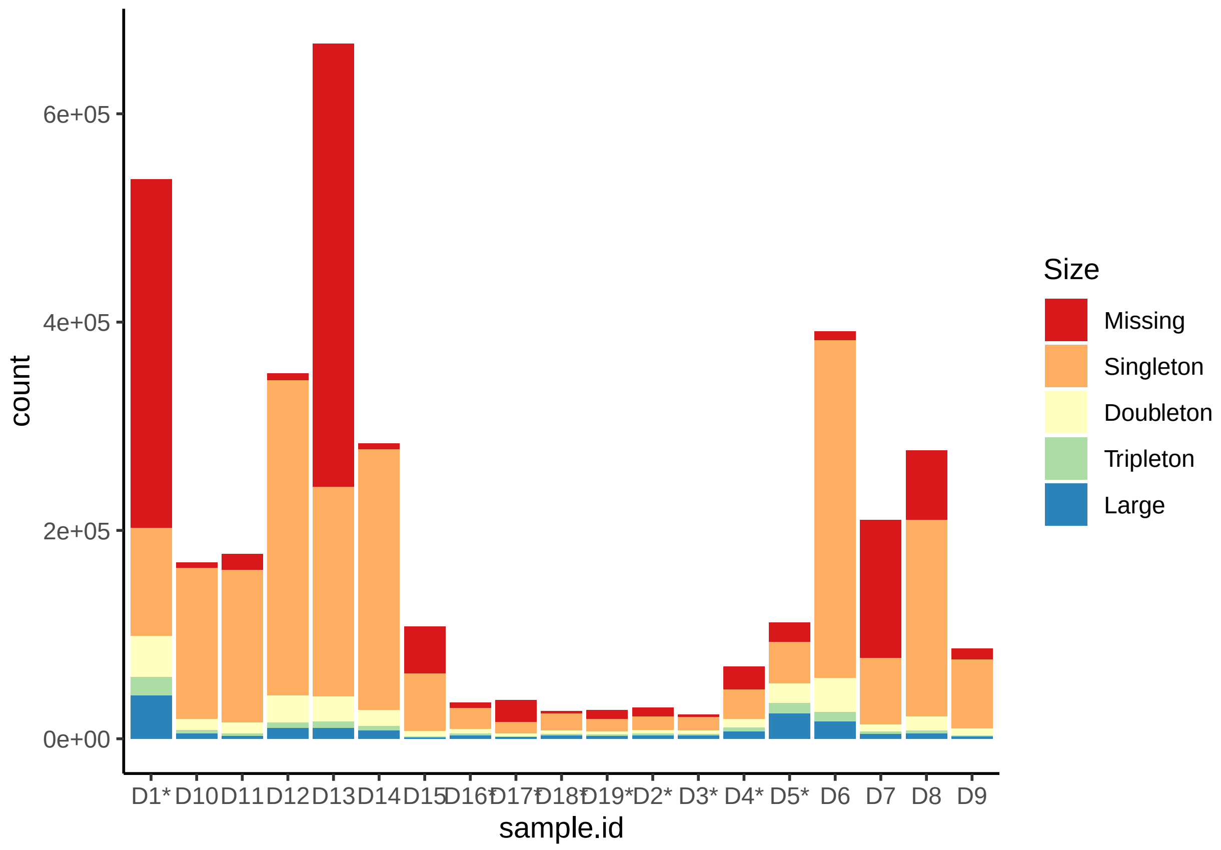
```
data <- list.files("data", full.names = T) %>%
  as.list %>%
  lapply(function(x) read_gz(x) %>% mutate(sample.id = x)) %>%
  rbindlist %>%
  mutate(sample.id.old = sample.id,
         dli = !str_starts(sample.id.old, fixed("data/sh.p")),
         sample.id = paste0("D", sample.id %>% as.factor %>% as.integer, ifelse(dli, "*", ""))) #%>%
```

```
## Taking input= as a system command ('zcat data/sh.Art.txt.gz') and a variable has been used in the ex
## Taking input= as a system command ('zcat data/sh.Bat.txt.gz') and a variable has been used in the ex
## Taking input= as a system command ('zcat data/sh.Hus.txt.gz') and a variable has been used in the ex
## Taking input= as a system command ('zcat data/sh.Kim.txt.gz') and a variable has been used in the ex
## Taking input= as a system command ('zcat data/sh.Make.txt.gz') and a variable has been used in the e
## Taking input= as a system command ('zcat data/sh.p1005.txt.gz') and a variable has been used in the
## Taking input= as a system command ('zcat data/sh.p1321.txt.gz') and a variable has been used in the
## Taking input= as a system command ('zcat data/sh.p1694.txt.gz') and a variable has been used in the
## Taking input= as a system command ('zcat data/sh.p1772.txt.gz') and a variable has been used in the
## Taking input= as a system command ('zcat data/sh.p2768.txt.gz') and a variable has been used in the
## Taking input= as a system command ('zcat data/sh.p2846.txt.gz') and a variable has been used in the
## Taking input= as a system command ('zcat data/sh.p3514.txt.gz') and a variable has been used in the
## Taking input= as a system command ('zcat data/sh.p3570.txt.gz') and a variable has been used in the
## Taking input= as a system command ('zcat data/sh.p3602.txt.gz') and a variable has been used in the
## Taking input= as a system command ('zcat data/sh.p754.txt.gz') and a variable has been used in the e
## Taking input= as a system command ('zcat data/sh.Ser.txt.gz') and a variable has been used in the ex
## Taking input= as a system command ('zcat data/sh.Str.txt.gz') and a variable has been used in the ex
## Taking input= as a system command ('zcat data/sh.Zav.txt.gz') and a variable has been used in the ex
## Taking input= as a system command ('zcat data/sh.Zuk.txt.gz') and a variable has been used in the ex
```

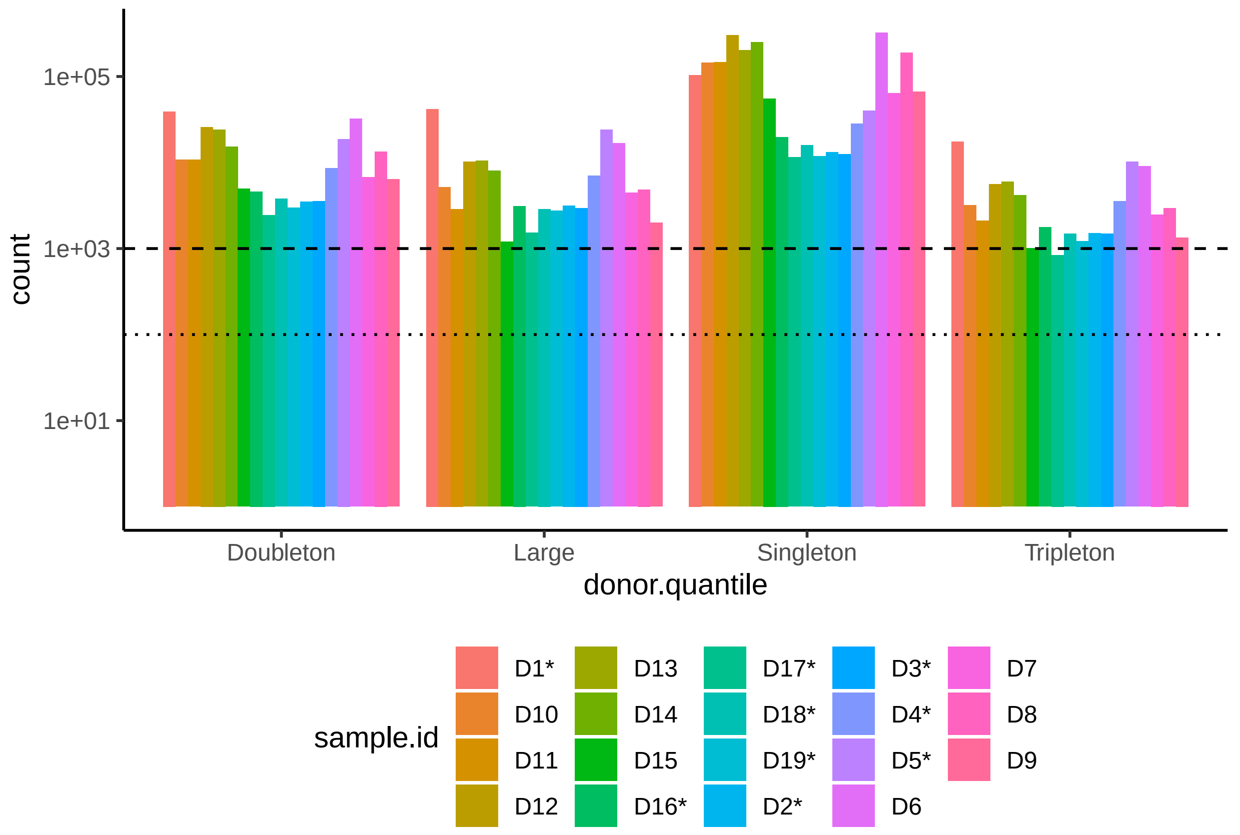
```
  #filter(dli)

data %>%
  select(sample.id.old, dli, sample.id) %>%
  unique
```

```
##            sample.id.old   dli sample.id
##  1:   data/sh.Art.txt.gz  TRUE       D1*
##  2:   data/sh.Bat.txt.gz  TRUE       D2*
##  3:   data/sh.Hus.txt.gz  TRUE       D3*
##  4:   data/sh.Kim.txt.gz  TRUE       D4*
##  5:  data/sh.Make.txt.gz  TRUE       D5*
##  6: data/sh.p1005.txt.gz FALSE        D6
##  7: data/sh.p1321.txt.gz FALSE        D7
##  8: data/sh.p1694.txt.gz FALSE        D8
##  9: data/sh.p1772.txt.gz FALSE        D9
## 10: data/sh.p2768.txt.gz FALSE       D10
## 11: data/sh.p2846.txt.gz FALSE       D11
## 12: data/sh.p3514.txt.gz FALSE       D12
## 13: data/sh.p3570.txt.gz FALSE       D13
## 14: data/sh.p3602.txt.gz FALSE       D14
## 15:  data/sh.p754.txt.gz FALSE       D15
## 16:   data/sh.Ser.txt.gz  TRUE      D16*
## 17:   data/sh.Str.txt.gz  TRUE      D17*
## 18:   data/sh.Zav.txt.gz  TRUE      D18*
## 19:   data/sh.Zuk.txt.gz  TRUE      D19*
```

```
data <- data %>%
  mutate(donor.quantile = case_when(
    is.na(cloneCount.don) ~ "Missing",
    cloneCount.don == 1 ~ "Singleton",
    cloneCount.don == 2 ~ "Doubleton",
    cloneCount.don == 3 ~ "Tripleton",
    T ~ "Large"
    ))

data %>%
  mutate(cloneCount.don = ifelse(is.na(cloneCount.don), 0, cloneCount.don)) %>%
  ggplot(aes(x = sample.id,
             fill = donor.quantile %>%
               fct_reorder(cloneCount.don))) +
  geom_bar() +
  scale_fill_brewer("Size", palette = "Spectral") +
  theme_classic()
```



```r
data %>%
  filter(donor.quantile != "Missing") %>%
  ggplot(aes(x = donor.quantile,
             fill = sample.id)) +
  geom_bar(position = "dodge") +
  scale_y_log10() +
  geom_hline(yintercept = 100, linetype = "dotted") +
  geom_hline(yintercept = 1000, linetype = "dashed") +
  theme_classic() +
  theme(legend.position = "bottom")
```



Modeling probability of "survival" for clones using Beta distribution

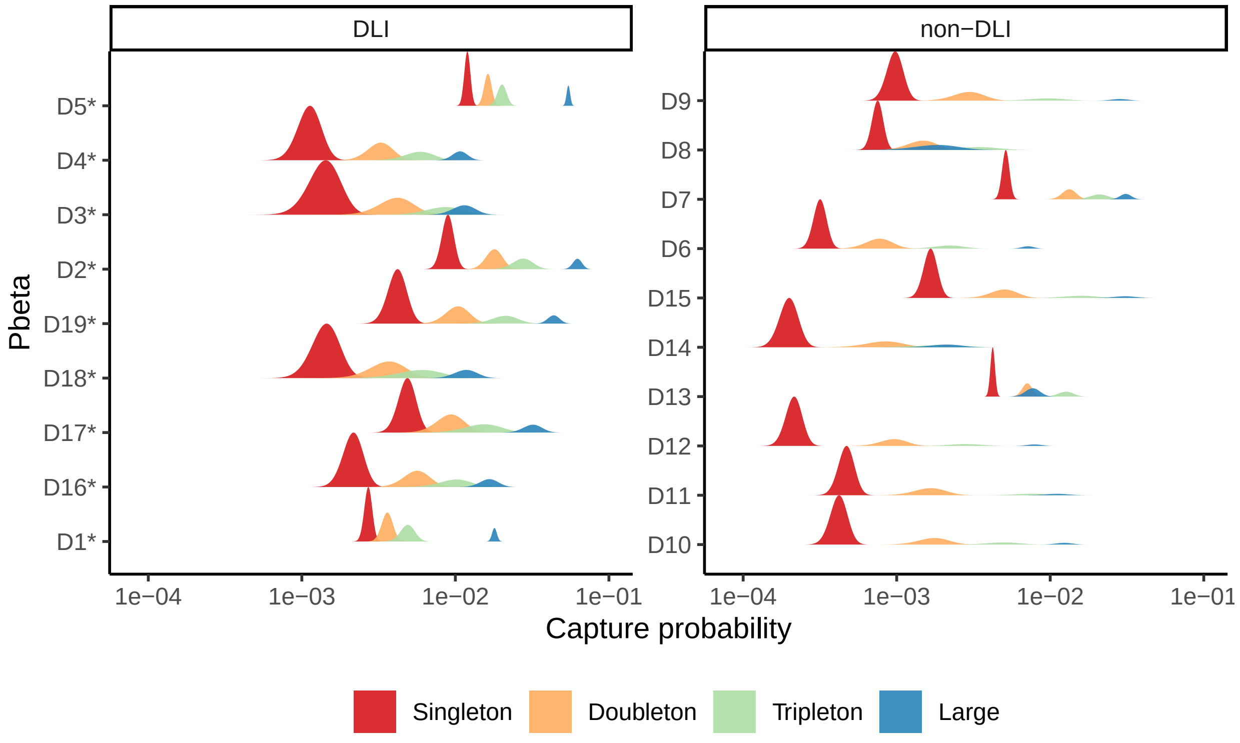
We split our donor dataset into singletons, doubletons, tripletons and higher-order clonotypes. Each of these subsets contains enough clones to reliably estimate the probability of recapturing a clonotype from a given subset of donor clonotypes. Interestingly, the ration between recapturing probabilities of singletons, doubletons and tripletons is in line with exponential difference stemming from Poisson distribution.

```
# summarize & estimate parameters of beta distribution
alpha.prior <- 1
beta.prior <- 1
data.s <- data %>%
  filter(donor.quantile != "Missing") %>%
  group_by(sample.id) %>%
  mutate(total.don = sum(cloneCount.don, na.rm = T),
         clones.don = length(unique(aaSeqCDR3.don)), # note here we count +1 for NA, maybe should modif
         total.rec = sum(cloneCount.rec, na.rm = T),
         clones.rec = length(unique(aaSeqCDR3.rec))) %>%
  group_by(sample.id, donor.quantile) %>%
  mutate(clones.don.quant = length(unique(aaSeqCDR3.don))) %>%
  group_by(dli, sample.id, donor.quantile, total.don, clones.don, total.rec, clones.rec, clones.don.qua
  summarize(alpha = sum(!is.na(cloneCount.rec)) + alpha.prior,
            beta = sum(is.na(cloneCount.rec)) + beta.prior) %>%
  ungroup
```

```
## `summarise()` regrouping output by 'dli', 'sample.id', 'donor.quantile', 'total.don', 'clones.don',
```

```r
data.sp <- data.s %>%
  merge(tibble(p = c(0:1000/1000, 10^(-4000:-1000/1000)))) %>%
  group_by(sample.id, donor.quantile) %>%
  mutate(Pbeta = dbeta(p, alpha, beta)) %>%
  ungroup

data.sp %>%
  mutate(dli = ifelse(dli, "DLI", "non-DLI")) %>%
  group_by(sample.id) %>%
  mutate(height = Pbeta / max(Pbeta)) %>%
  ggplot(aes(x = p, y = sample.id, height = height,
             fill = factor(donor.quantile, levels = c("Singleton",
                                                      "Doubleton",
                                                      "Tripleton",
                                                      "Large"))
  )) +
  geom_ridgeline(color = NA, alpha = 0.9) +
  scale_x_log10("Capture probability", limits = c(0.8e-4, 1e-1)) + ylab("Pbeta") +
  scale_fill_brewer("", palette = "Spectral") +
  facet_wrap(~dli, scales = "free_y") +
  theme_classic() +
  theme(aspect = 1, legend.position = "bottom")
```

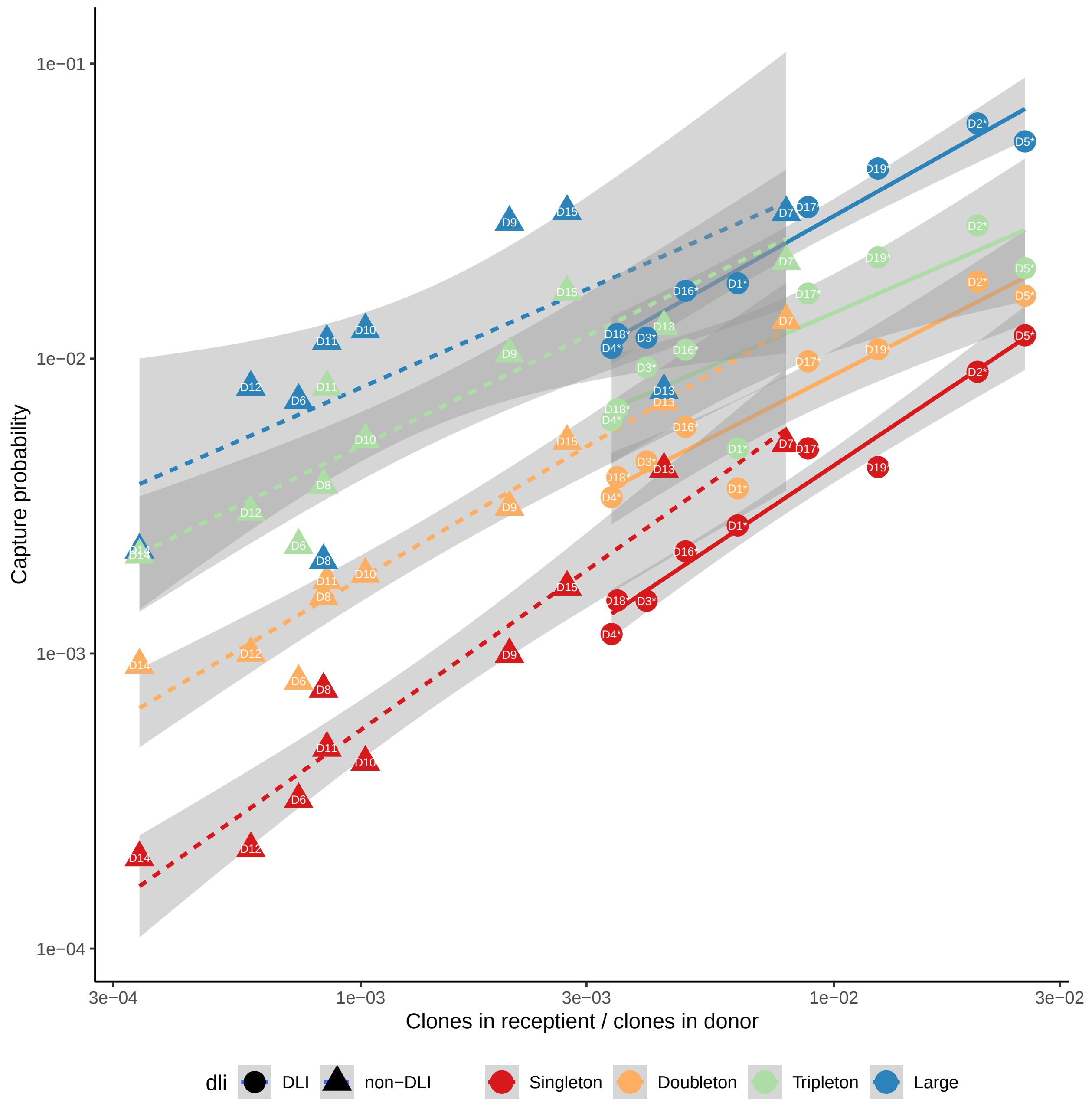
```
## Warning: Transformation introduced infinite values in continuous x-axis
```



Interestingly, the TCR recovery rate is related both to the total number of clones in donor and recipient. It is also different for DLI and non-DLI patients.

```r
data.s %>%
  mutate(dli = ifelse(dli, "DLI", "non-DLI")) %>%
  ggplot(aes(x = clones.rec / clones.don, y = alpha / (alpha + beta),
             group = paste(dli, donor.quantile),
             linetype = dli, shape = dli,
             color = factor(donor.quantile, levels = c("Singleton",
                                                        "Doubleton",
                                                        "Tripleton",
                                                        "Large")))) +
  geom_smooth(method = "lm", aes(), size = 1) +
  geom_point(size = 5) +
  geom_text(aes(label = sample.id), size = 2, color = "white") +
  scale_x_log10("Clones in receptient / clones in donor") +
  scale_y_log10("Capture probability") +
  scale_color_brewer("", palette = "Spectral") +
  theme_classic() +
  theme(aspect = 1, legend.position = "bottom")
```

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



Basic linear modelling

Quantifying the effect of various factors – number of clones detected in donor, number of clones detected in receptient and the frequency quantile of a given clonotype in donor – on the recapture probability. Log-transformed variables show extremely high correlation.

```
data.coord <- data.s %>%
  group_by(donor.quantile) %>%
  mutate(logRecaptureProb = log(alpha / (alpha + beta)),
         logClonesRecepient = log(clones.rec),
         logClonesDonor = log(clones.don)) %>%
  ungroup %>%
  mutate(donor.quantile = factor(donor.quantile, levels = c("Singleton",
                                                       "Doubleton",
                                                       "Tripleton",
                                                       "Large")))
```

Show coefficients of linear model

```
data.coord %>%
  ungroup %>%
  mutate(donor.quantile = as.factor(donor.quantile)) %>%
  do(lm(.$logRecaptureProb ~ .$donor.quantile + .$dli + .$logClonesRecepient + .$logClonesDonor) %>% ti
```

```
## # A tibble: 7 x 5
##   term                        estimate std.error statistic  p.value
##   <chr>                          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)                    1.08     0.753       1.44 1.55e- 1
## 2 .$donor.quantileDoubleton      0.959    0.122       7.88 3.33e-11
## 3 .$donor.quantileTripleton      1.72     0.122      14.2  4.08e-22
## 4 .$donor.quantileLarge          2.28     0.122      18.8  8.52e-29
## 5 .$dliTRUE                     -0.758    0.143      -5.31 1.24e- 6
## 6 .$logClonesRecepient           0.827    0.0541     15.3  7.33e-24
## 7 .$logClonesDonor              -1.04     0.0705    -14.8  4.44e-23
```

Show variance explained (ANOVA)

```
data.coord %>%
  ungroup %>%
  mutate(donor.quantile = as.factor(donor.quantile)) %>%
  do(lm(.$logRecaptureProb ~ .$donor.quantile + .$dli + .$logClonesRecepient + .$logClonesDonor) %>% ao
  mutate(var.explained.pct = sumsq / sum(sumsq) * 100)
```

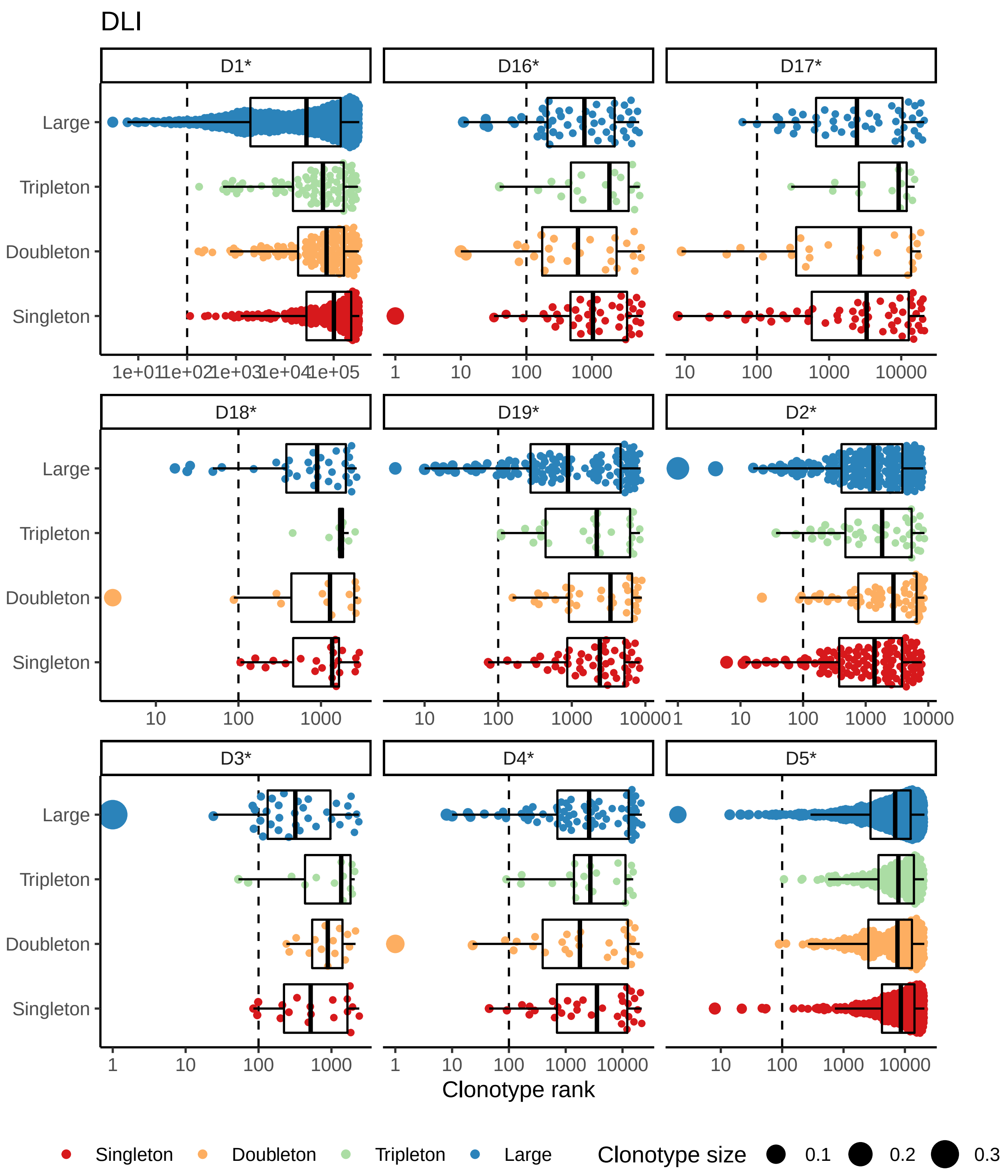
```
## # A tibble: 5 x 7
##   term                      df sumsq meansq statistic   p.value var.explained.pct
##   <chr>                  <dbl> <dbl>  <dbl>     <dbl>     <dbl>             <dbl>
## 1 .$donor.quantile           3 55.8   18.6      132.   1.52e-28             43.9
## 2 .$dli                      1 18.8   18.8      134.   8.55e-18             14.8
## 3 .$logClonesRecepient       1 12.0   12.0       85.3  1.12e-13              9.44
## 4 .$logClonesDonor           1 30.7   30.7      219.   4.44e-23             24.2
## 5 Residuals                 69  9.70   0.141      NA   NA                    7.64
```

Origin of clones found in recepient: number of highly expanded clones that originated from expanded donor clones and rare donor clones varies and depends on donor. In general clonotypes preserve their size, but there is lots of noise here.

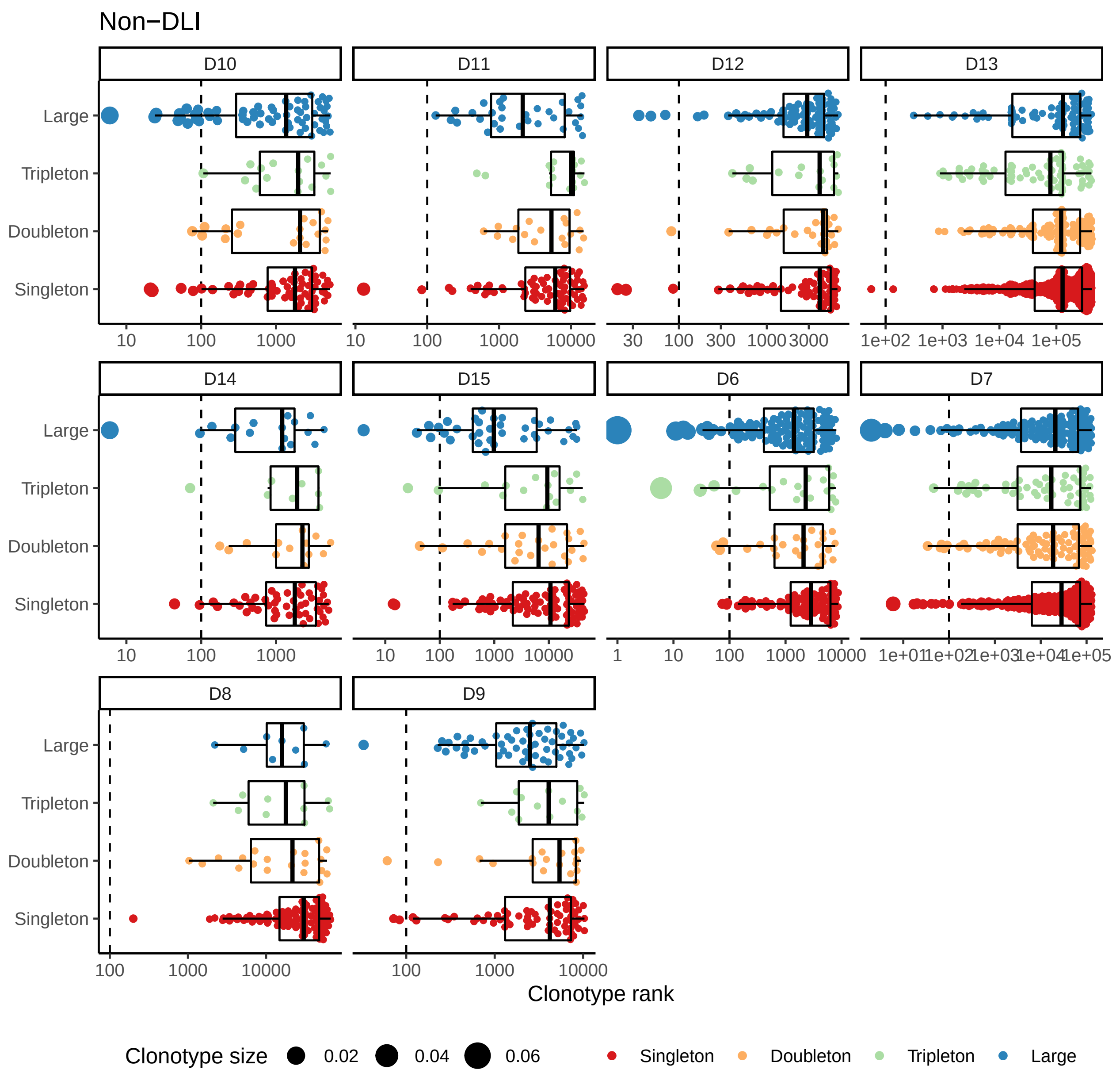
> Open question - show this statistically, that survival prob depends not just on sampling, but is more skewed and depends on clonotype size.

```
data %>%
  mutate(donor.quantile = factor(donor.quantile, levels = c("Singleton",
                                                       "Doubleton",
                                                       "Tripleton",
                                                       "Large"))) %>%
  filter(dli & !is.na(cloneCount.rec)) %>%
  group_by(sample.id) %>%
  mutate(rank = rank(-cloneCount.rec, ties.method   = "first"),
         freq.rec = cloneCount.rec / sum(cloneCount.rec)) %>%
  filter(donor.quantile != "Missing") %>%
  ggplot(aes(x = donor.quantile, y = rank)) +
```

```r
  geom_hline(yintercept = 100, linetype = "dashed") +
  geom_quasirandom(aes(size = freq.rec, color = donor.quantile)) +
  geom_boxplot(fill = NA, color = "black", outlier.colour = NA) +
  coord_flip() +
  scale_y_log10("Clonotype rank") +
  xlab("") +
  scale_size_continuous("Clonotype size") +
  scale_color_brewer("", palette = "Spectral") +
  facet_wrap(.~sample.id, scales = "free_x") +
  theme_classic() +
  theme(aspect = 1, legend.position = "bottom") +
  ggtitle("DLI")
```



```r
data %>%
  mutate(donor.quantile = factor(donor.quantile, levels = c("Singleton",
                                                             "Doubleton",
                                                             "Tripleton",
                                                             "Large"))) %>%
  filter(!dli & !is.na(cloneCount.rec)) %>%
  group_by(sample.id) %>%
  mutate(rank = rank(-cloneCount.rec, ties.method   = "first"),
         freq.rec = cloneCount.rec / sum(cloneCount.rec)) %>%
  filter(donor.quantile != "Missing") %>%
  ggplot(aes(x = donor.quantile, y = rank)) +
  geom_hline(yintercept = 100, linetype = "dashed") +
  geom_quasirandom(aes(size = freq.rec, color = donor.quantile)) +
  geom_boxplot(fill = NA, color = "black", outlier.colour = NA) +
  coord_flip() +
  scale_y_log10("Clonotype rank") +
  xlab("") +
  scale_size_continuous("Clonotype size") +
  scale_color_brewer("", palette = "Spectral") +
  facet_wrap(.~sample.id, scales = "free_x") +
  theme_classic() +
  theme(aspect = 1, legend.position = "bottom") +
  ggtitle("Non-DLI")
```



Modeling data and covariate analysis examples

Here is an example on how we can correct for sampling probability based on sample diversities and clonotype size and compare between DLI and non-DLI donors. Here we recompute a single $\Delta p = p_{observed} - p_{predicted}$ for every sample, i.e. if donor sample contains (by percent) ϕ_1 singletons, ϕ_2 doubletons, etc and difference for singletons is Δp_i - we compute a weighted sum $\Delta p = \sum_i \phi_i \Delta p_i$.

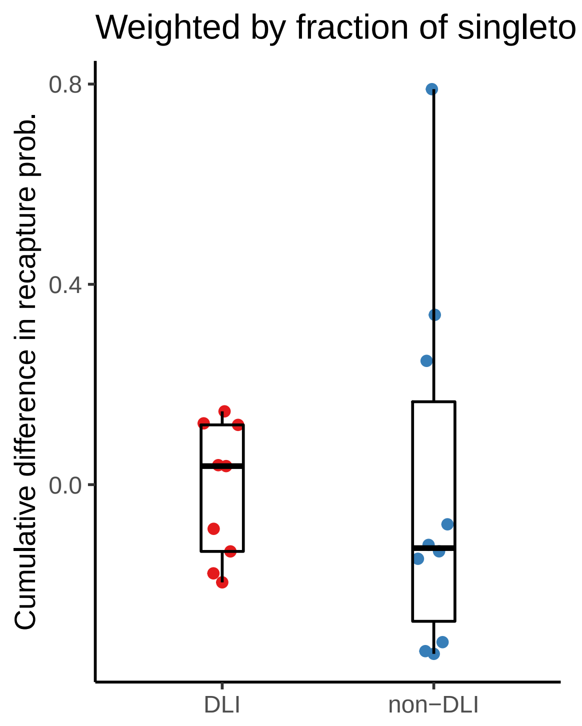
Actually works not that great, however, if we treat Δp_i separately for each sample we are artificially boosting the number of “samples” for statistical testing. An alternative would be to look separately at singletons, doubletons, etc (I think multiple testing can be omitted here as we do like 2-3 tests at most).

This is just an example of what can be done “manually” without using proper statistical methods like building several models, estimating P-values for various covariates and comparing models using ANOVA, etc.

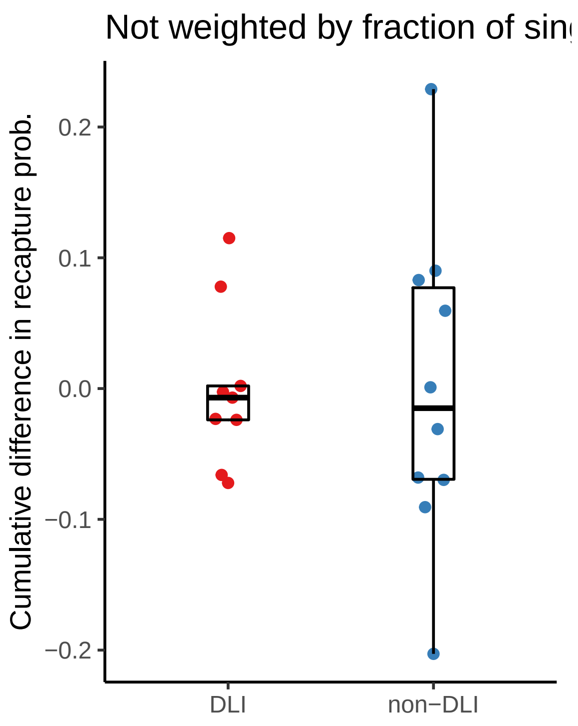
```
data.dli.pred <- data.coord %>% filter(dli)
lm(logRecaptureProb ~ donor.quantile + logClonesRecepient + logClonesDonor,
   data = data.dli.pred) -> lm.dli
data.dli.pred$logRecaptureProbPred <- predict(lm.dli, data.dli.pred)

data.ndli.pred <- data.coord %>% filter(!dli)
lm(logRecaptureProb ~ donor.quantile + logClonesRecepient + logClonesDonor,
   data = data.ndli.pred) -> lm.ndli
data.ndli.pred$logRecaptureProbPred <- predict(lm.ndli, data.ndli.pred)

rbind(data.dli.pred,
      data.ndli.pred) %>%
  group_by(sample.id) %>%
  mutate(clones.don.quant.frac = clones.don.quant / sum(clones.don.quant),
         delta = logRecaptureProb - logRecaptureProbPred) %>%
  group_by(sample.id, dli) %>%
  summarise(diff = sum(delta * clones.don.quant.frac),
            diff.unweighted = mean(delta),
            diff.singl = sum(delta * (donor.quantile == "Singleton"))) -> data.delta.summ
```

```
## `summarise()` regrouping output by 'sample.id' (override with `.groups` argument)
```

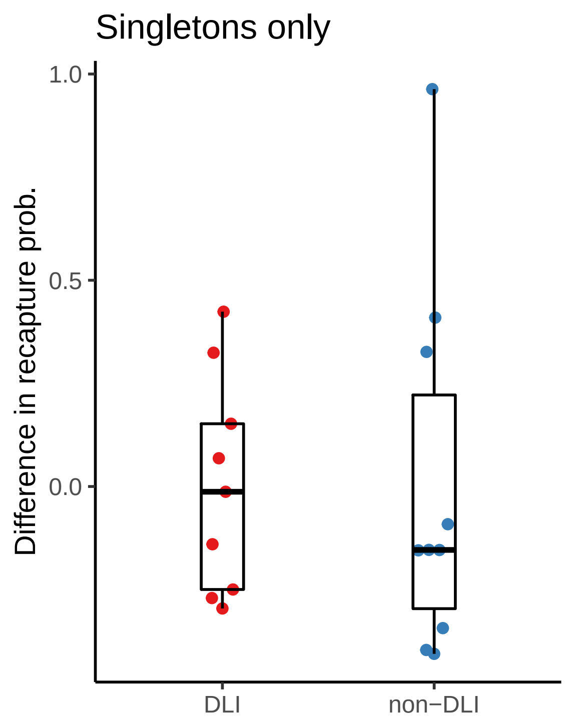
```
data.delta.summ %>%
  mutate(dli = ifelse(dli, "DLI", "non-DLI")) %>%
  ggplot(aes(x = dli, y = diff, color = dli)) +
  geom_quasirandom(width = 0.1) +
  geom_boxplot(width = 0.2, color = "black", fill = NA, outlier.colour = NA) +
  scale_color_brewer(guide = F, palette = "Set1") +
  xlab("") + ylab("Cumulative difference in recapture prob.") +
  ggtitle("Weighted by fraction of singletons, doubletons, etc") +
  theme_classic()
```



```r
data.delta.summ %>%
  mutate(dli = ifelse(dli, "DLI", "non-DLI")) %>%
  ggplot(aes(x = dli, y = diff.unweighted, color = dli)) +
  geom_quasirandom(width = 0.1) +
  geom_boxplot(width = 0.2, color = "black", fill = NA, outlier.colour = NA) +
  scale_color_brewer(guide = F, palette = "Set1") +
  xlab("") + ylab("Cumulative difference in recapture prob.") +
  ggtitle("Not weighted by fraction of singletons, doubletons, etc") +
  theme_classic()
```



```
data.delta.summ %>%
  mutate(dli = ifelse(dli, "DLI", "non-DLI")) %>%
  ggplot(aes(x = dli, y = diff.singl, color = dli)) +
  geom_quasirandom(width = 0.1) +
  geom_boxplot(width = 0.2, color = "black", fill = NA, outlier.colour = NA) +
  scale_color_brewer(guide = F, palette = "Set1") +
  xlab("") + ylab("Difference in recapture prob.") +
  ggtitle("Singletons only") +
  theme_classic()
```



Now lets build some linear models using `lm` and compare them using ANOVA

```
mdl.1 <- lm(logRecaptureProb ~ donor.quantile + logClonesRecepient + logClonesDonor,
            data = data.coord)
mdl.1
```

```
##
## Call:
## lm(formula = logRecaptureProb ~ donor.quantile + logClonesRecepient +
##     logClonesDonor, data = data.coord)
##
## Coefficients:
##            (Intercept)  donor.quantileDoubleton  donor.quantileTripleton
##                -1.9379                   0.9585                   1.7241
##    donor.quantileLarge       logClonesRecepient           logClonesDonor
##                 2.2817                   0.7008                  -0.7445
```

```
mdl.2 <- lm(logRecaptureProb ~ donor.quantile + logClonesRecepient + logClonesDonor + dli,
            data = data.coord)
mdl.2
```

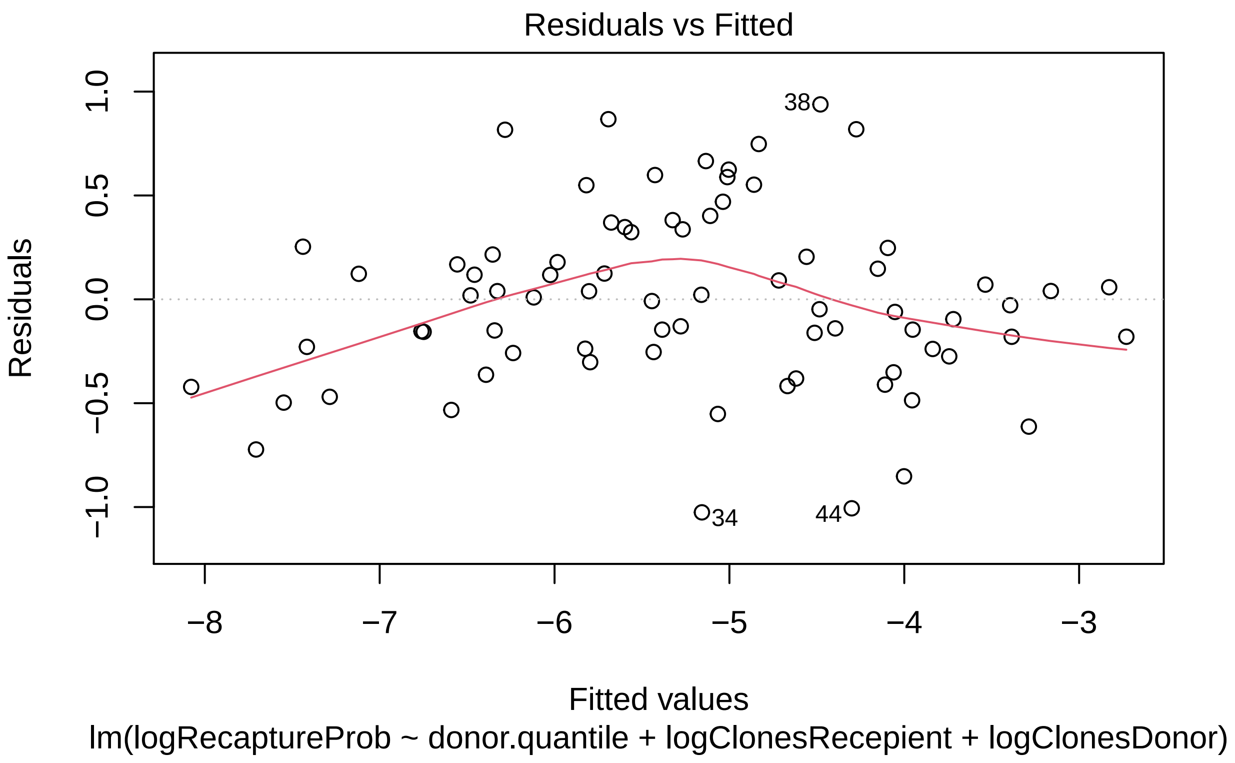
```
##
## Call:
## lm(formula = logRecaptureProb ~ donor.quantile + logClonesRecepient +
##     logClonesDonor + dli, data = data.coord)
##
## Coefficients:
##            (Intercept)  donor.quantileDoubleton  donor.quantileTripleton
##                 1.0820                   0.9585                   1.7241
##    donor.quantileLarge       logClonesRecepient           logClonesDonor
##                 2.2817                   0.8273                  -1.0423
##                dliTRUE
##                -0.7584
```

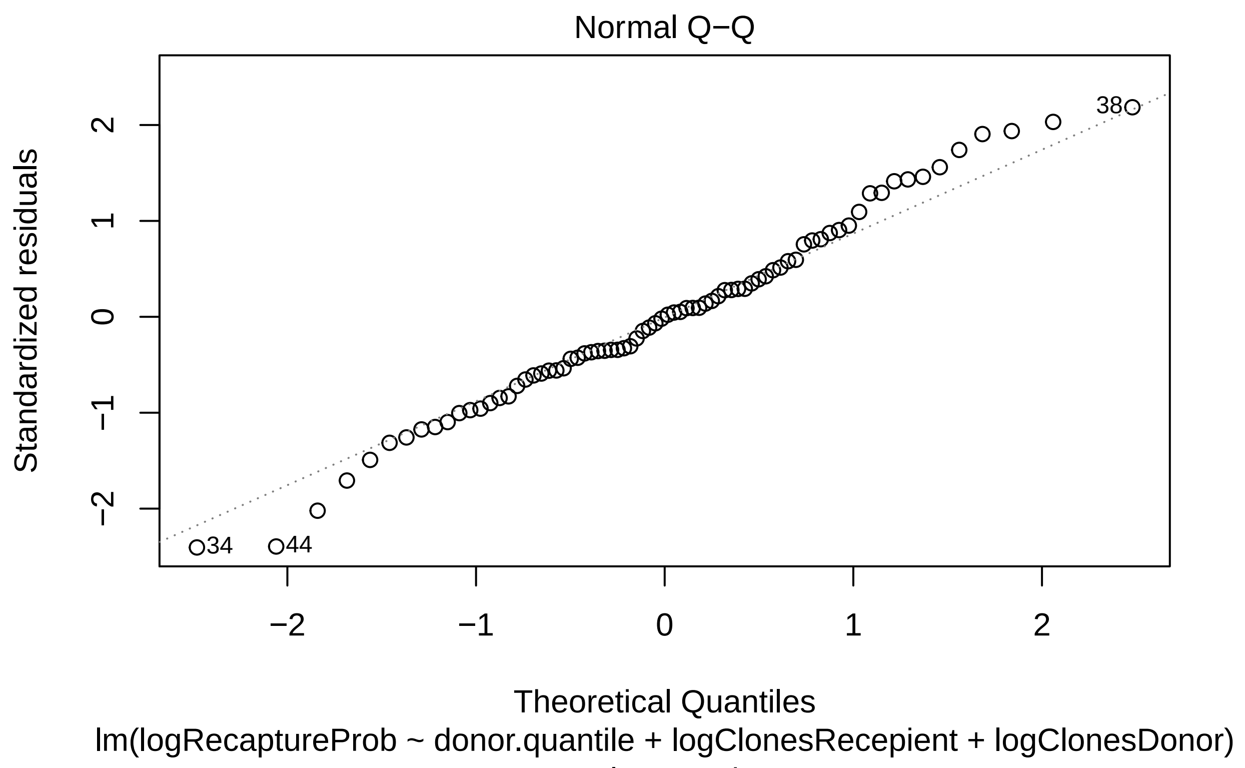
Summary of model with DLI covariate

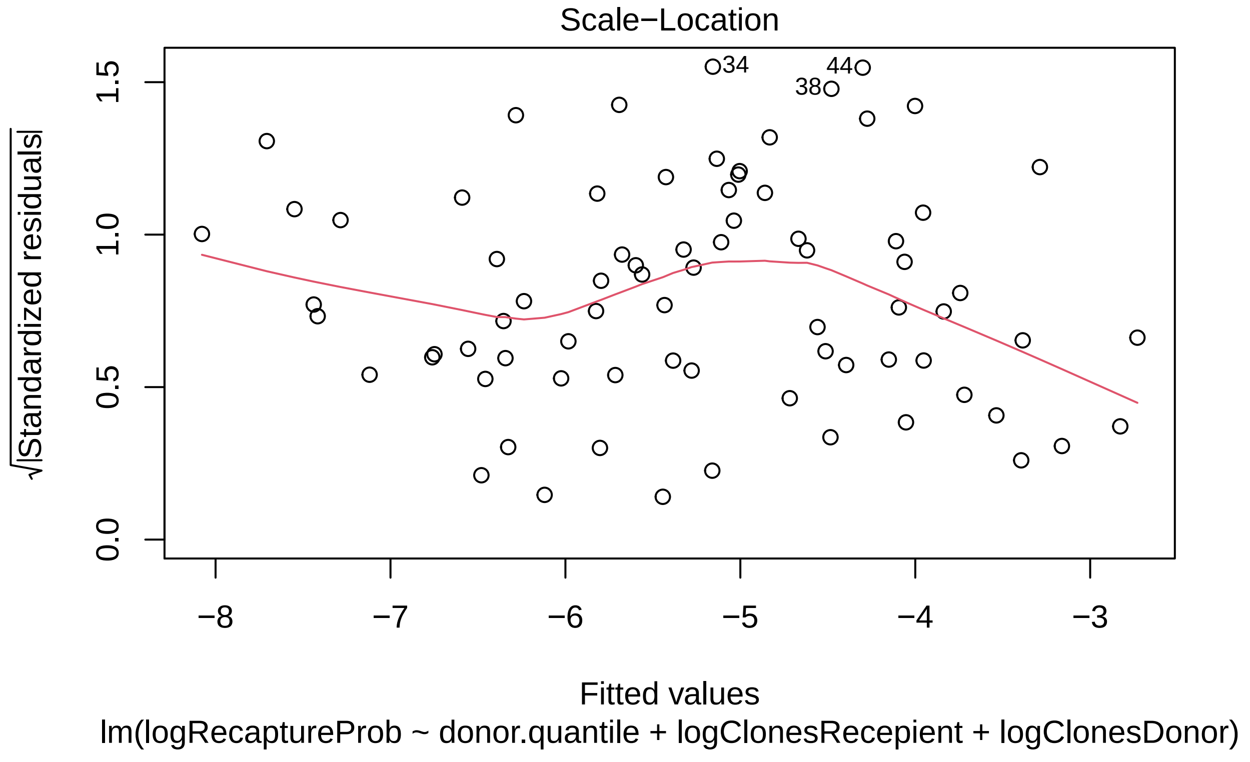
```r
summary(mdl.1)
```

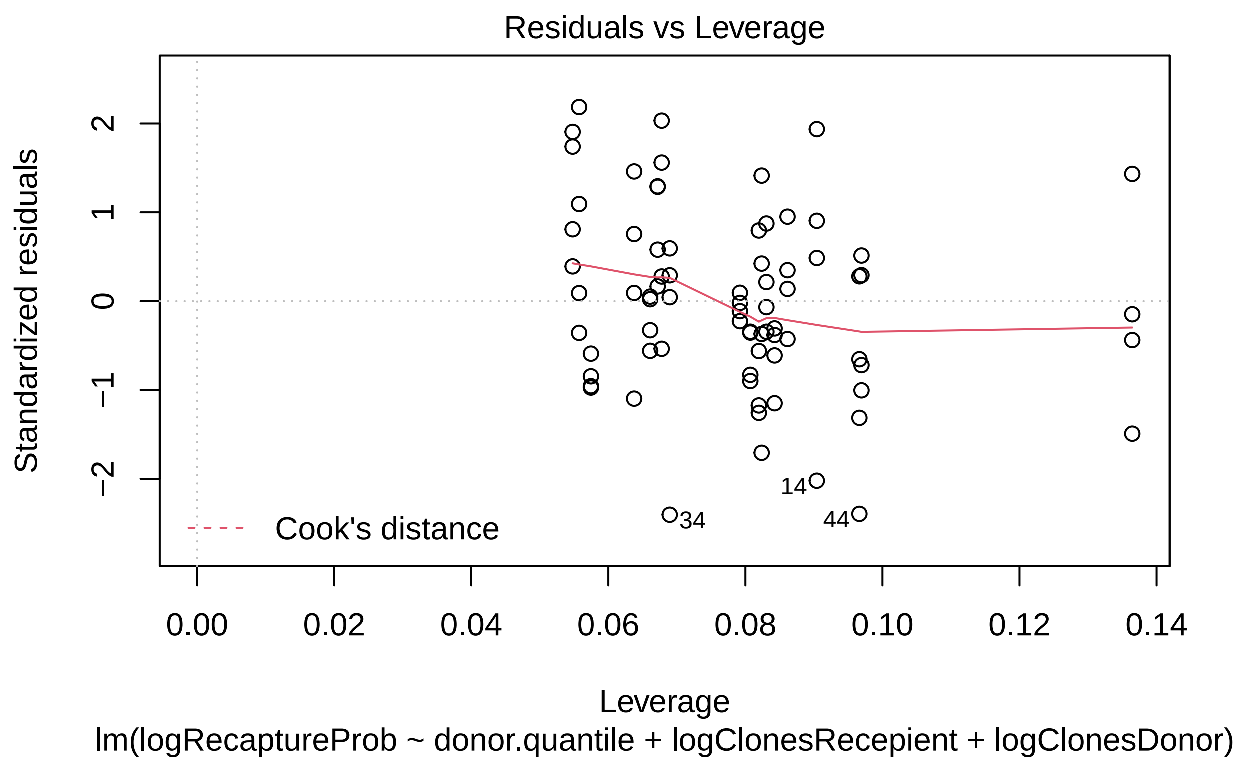
```
##
## Call:
## lm(formula = logRecaptureProb ~ donor.quantile + logClonesRecepient +
##     logClonesDonor, data = data.coord)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.02557 -0.25491  0.00041  0.24889  0.93843
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)              -1.93788    0.58151  -3.332  0.00138 **
## donor.quantileDoubleton  0.95852    0.14340   6.684 4.67e-09 ***
## donor.quantileTripleton  1.72408    0.14340  12.023  < 2e-16 ***
## donor.quantileLarge      2.28169    0.14340  15.912  < 2e-16 ***
## logClonesRecepient       0.70081    0.05724  12.244  < 2e-16 ***
## logClonesDonor          -0.74447    0.05038 -14.776  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.442 on 70 degrees of freedom
## Multiple R-squared:  0.8923, Adjusted R-squared:  0.8846
## F-statistic:   116 on 5 and 70 DF,  p-value: < 2.2e-16
```

```r
plot(mdl.1)
```




Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(logRecaptureProb ~ donor.quantile + logClonesRecepient + logClonesDonor)
34
44
38
−2
−1
0
1
2


Scale−Location
$\sqrt{|\text{Standardized residuals}|}$
Fitted values
lm(logRecaptureProb ~ donor.quantile + logClonesRecepient + logClonesDonor)
34
44
38
0.0
0.5
1.0
1.5
−8
−7
−6
−5
−4
−3

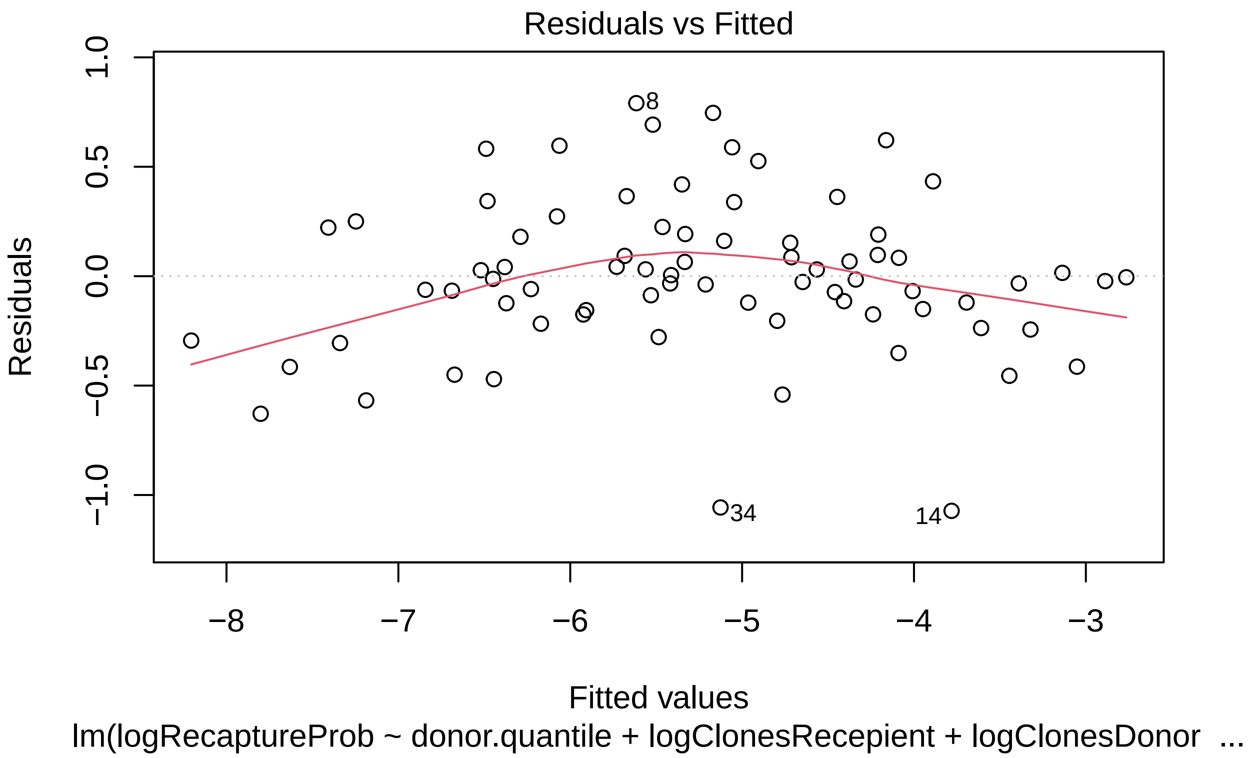


And model without one

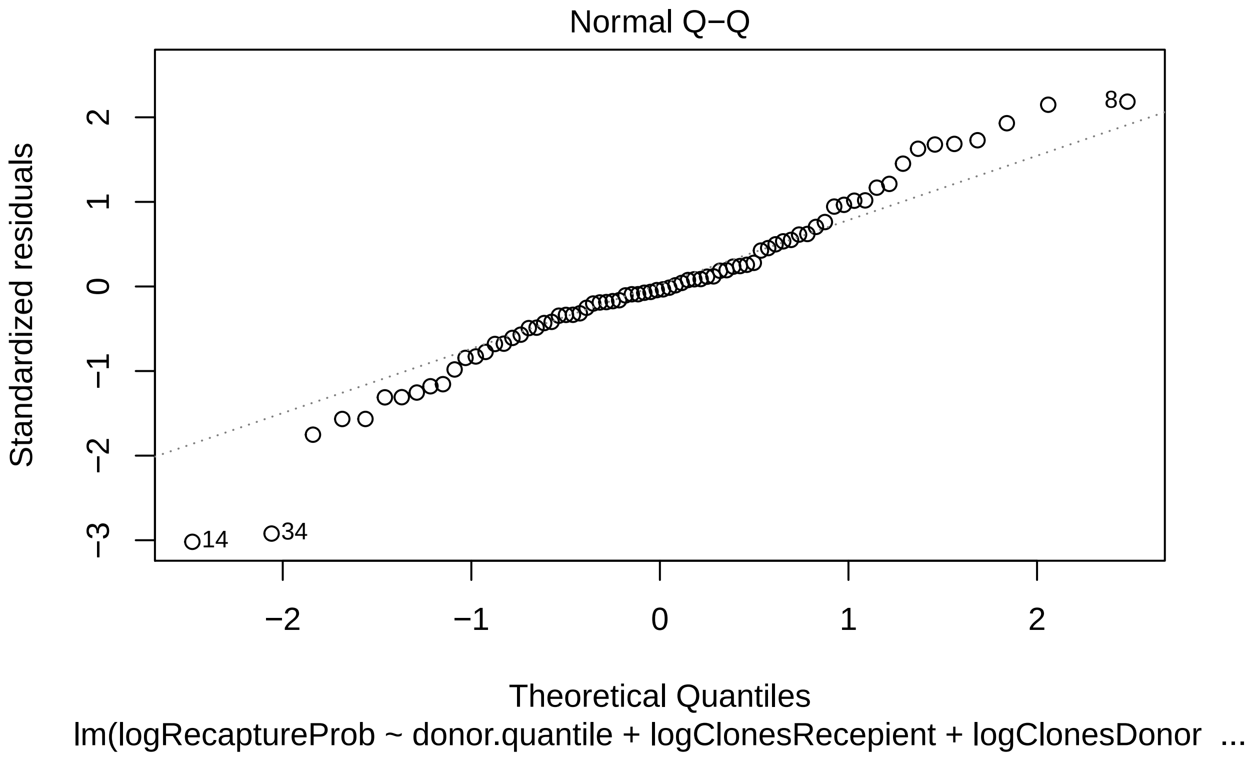
```r
summary(mdl.2)
```

```
##
## Call:
## lm(formula = logRecaptureProb ~ donor.quantile + logClonesRecepient +
##     logClonesDonor + dli, data = data.coord)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.07236 -0.17453 -0.01391  0.19072  0.79057
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)               1.08201    0.75252   1.438    0.155
## donor.quantileDoubleton  0.95852    0.12166   7.879 3.33e-11 ***
## donor.quantileTripleton  1.72408    0.12166  14.171  < 2e-16 ***
## donor.quantileLarge      2.28169    0.12166  18.755  < 2e-16 ***
## logClonesRecepient       0.82734    0.05408  15.298  < 2e-16 ***
## logClonesDonor          -1.04233    0.07049 -14.788  < 2e-16 ***
## dliTRUE                 -0.75841    0.14270  -5.315 1.24e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.375 on 69 degrees of freedom
## Multiple R-squared:  0.9236, Adjusted R-squared:  0.917
## F-statistic:   139 on 6 and 69 DF,  p-value: < 2.2e-16
```

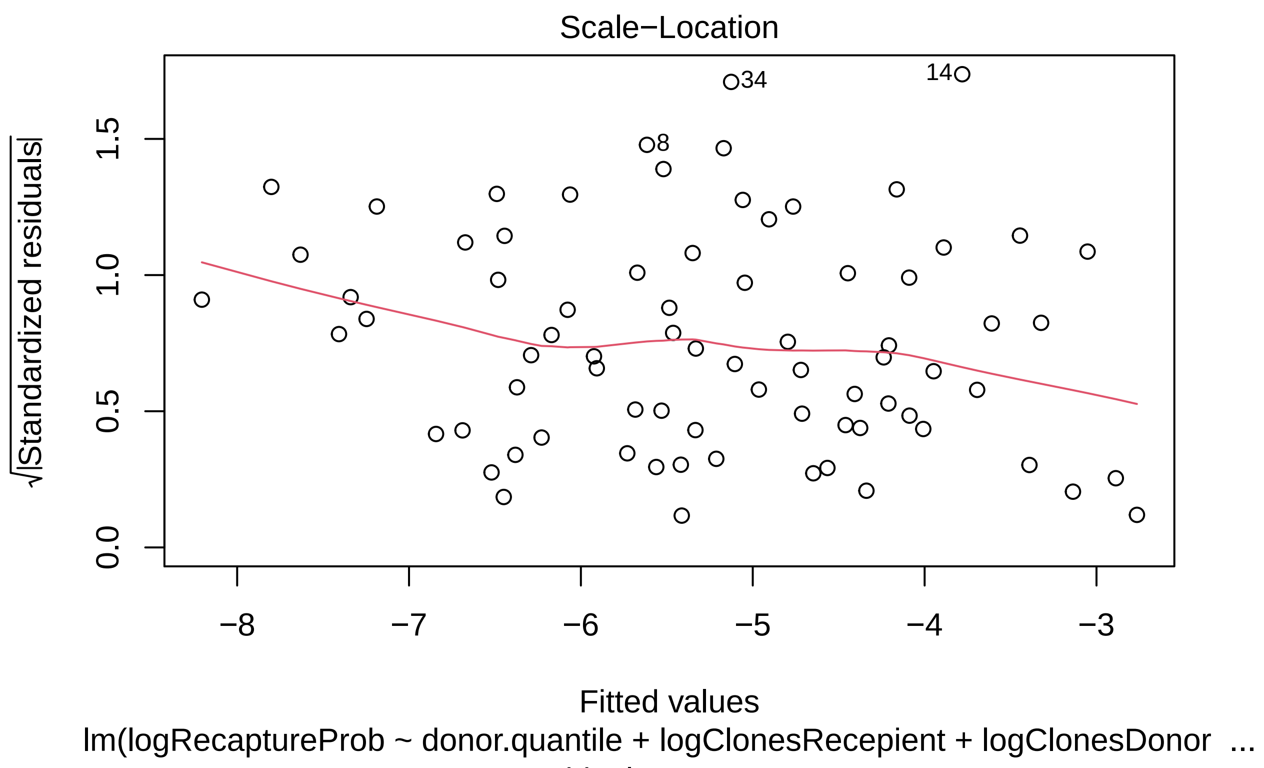
```r
plot(mdl.2)
```

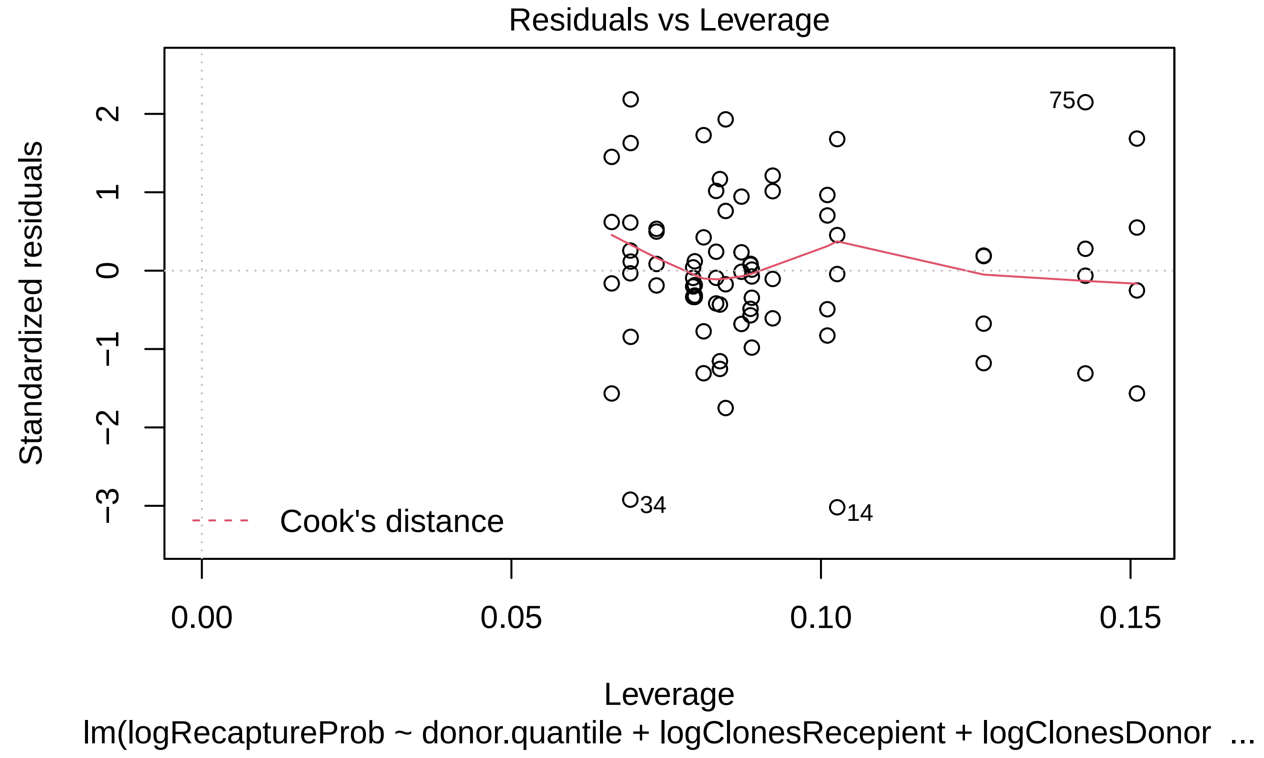


Residuals vs Fitted
Residuals
1.0
0.5
0.0
−0.5
−1.0
8
34
14
−8
−7
−6
−5
−4
−3
Fitted values
lm(logRecaptureProb ~ donor.quantile + logClonesRecepient + logClonesDonor ...



Normal Q−Q
Standardized residuals
2
1
0
−1
−2
−3
8
14
34
−2
−1
0
1
2
Theoretical Quantiles
lm(logRecaptureProb ~ donor.quantile + logClonesRecepient + logClonesDonor ...





Compare two models with ANOVA, the second one has smaller residual sum of squares

```
anova(mdl.1, mdl.2)
```

```
## Analysis of Variance Table
##
## Model 1: logRecaptureProb ~ donor.quantile + logClonesRecepient + logClonesDonor
## Model 2: logRecaptureProb ~ donor.quantile + logClonesRecepient + logClonesDonor +
##     dli
##   Res.Df     RSS Df Sum of Sq      F    Pr(>F)
## 1     70 13.6739
## 2     69  9.7022  1    3.9717 28.246 1.242e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Note that we can also do modelling using generalized linear models, GLMs. Here we remove log-transform as the we use binomial distribution

```
mdl.glm <- glm(found ~ donor.quantile + clones.don + clones.rec + dli,
               family = "binomial",
               data = data %>%
                 filter(donor.quantile != "Missing") %>%
                 mutate(found = ifelse(is.na(cloneCount.rec), 0, 1)) %>%
                 merge(data.coord %>%
                         select(sample.id, clones.don, clones.rec, dli) %>%
                         unique))

summary(mdl.glm)
```

```
##
## Call:
## glm(formula = found ~ donor.quantile + clones.don + clones.rec +
##     dli, family = "binomial", data = data %>% filter(donor.quantile !=
##     "Missing") %>% mutate(found = ifelse(is.na(cloneCount.rec),
##     0, 1)) %>% merge(data.coord %>% select(sample.id, clones.don,
##     clones.rec, dli) %>% unique))
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.3706  -0.0705  -0.0538  -0.0324   3.9278
##
## Coefficients:
##                           Estimate Std. Error  z value Pr(>|z|)
## (Intercept)             -4.922e+00  4.602e-02 -106.955  < 2e-16 ***
## donor.quantileLarge      1.271e+00  3.586e-02   35.457  < 2e-16 ***
## donor.quantileSingleton -7.267e-01  3.699e-02  -19.644  < 2e-16 ***
## donor.quantileTripleton  4.442e-01  4.959e-02    8.959  < 2e-16 ***
## clones.don              -6.400e-06  1.576e-07  -40.596  < 2e-16 ***
## clones.rec               8.073e-04  1.675e-05   48.202  < 2e-16 ***
## dliTRUE                 -2.551e-01  3.571e-02   -7.145 9.03e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 106905  on 2472071  degrees of freedom
## Residual deviance:  91400  on 2472065  degrees of freedom
## AIC: 91414
##
## Number of Fisher Scoring iterations: 9
```

Bayes modelling

```r
brm(logRecaptureProb ~ donor.quantile + logClonesRecepient + logClonesDonor,
               data = data.coord)
```

```
## Compiling Stan program...

## Trying to compile a simple C file

## Running /usr/lib/R/bin/R CMD SHLIB foo.c
## gcc -std=gnu99 -I"/usr/share/R/include" -DNDEBUG   -I"/home/mikesh/.lib/R/library/Rcpp/include/"  -I
## In file included from /home/mikesh/.lib/R/library/RcppEigen/include/Eigen/Core:88:0,
##                  from /home/mikesh/.lib/R/library/RcppEigen/include/Eigen/Dense:1,
##                  from /home/mikesh/.lib/R/library/StanHeaders/include/stan/math/prim/mat/fun/Eigen.hp
##                  from <command-line>:0:
## /home/mikesh/.lib/R/library/RcppEigen/include/Eigen/src/Core/util/Macros.h:613:1: error: unknown typ
##  namespace Eigen {
##  ^~~~~~~~~
## /home/mikesh/.lib/R/library/RcppEigen/include/Eigen/src/Core/util/Macros.h:613:17: error: expected ‘
##  namespace Eigen {
##                  ^
## In file included from /home/mikesh/.lib/R/library/RcppEigen/include/Eigen/Dense:1:0,
##                  from /home/mikesh/.lib/R/library/StanHeaders/include/stan/math/prim/mat/fun/Eigen.hp
##                  from <command-line>:0:
## /home/mikesh/.lib/R/library/RcppEigen/include/Eigen/Core:96:10: fatal error: complex: No such file o
##  #include <complex>
##           ^~~~~~~~~
## compilation terminated.
## /usr/lib/R/etc/Makeconf:172: recipe for target 'foo.o' failed
## make: *** [foo.o] Error 1

## Start sampling

##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: logRecaptureProb ~ donor.quantile + logClonesRecepient + logClonesDonor
##    Data: data.coord (Number of observations: 76)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##                         Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
## Intercept                  -1.92      0.59    -3.05    -0.76 1.00     6312
## donor.quantileDoubleton     0.96      0.15     0.68     1.25 1.00     3697
## donor.quantileTripleton     1.72      0.15     1.43     2.01 1.00     3530
## donor.quantileLarge         2.28      0.15     2.00     2.57 1.00     3455
## logClonesRecepient          0.70      0.06     0.59     0.81 1.00     5372
## logClonesDonor             -0.75      0.05    -0.84    -0.65 1.00     5403
##                         Tail_ESS
## Intercept                   3322
## donor.quantileDoubleton     3213
## donor.quantileTripleton     3218
## donor.quantileLarge         2956
## logClonesRecepient          2954
## logClonesDonor              3180
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
```

```
## sigma     0.45      0.04     0.38     0.53 1.00     5829     2661
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

Bayesian modelling (can also use smth like http://mjskay.github.io/tidybayes/articles/tidy-brms.html). Note that no complex models here (e.g. with dependence on parameters)

```
mdl.b.1 <- brm(logRecaptureProb ~ donor.quantile + logClonesRecepient + logClonesDonor,
               data = data.coord)
```

```
## Compiling Stan program...

## recompiling to avoid crashing R session

## Trying to compile a simple C file

## Running /usr/lib/R/bin/R CMD SHLIB foo.c
## gcc -std=gnu99 -I"/usr/share/R/include" -DNDEBUG   -I"/home/mikesh/.lib/R/library/Rcpp/include/"  -I
## In file included from /home/mikesh/.lib/R/library/RcppEigen/include/Eigen/Core:88:0,
##                  from /home/mikesh/.lib/R/library/RcppEigen/include/Eigen/Dense:1,
##                  from /home/mikesh/.lib/R/library/StanHeaders/include/stan/math/prim/mat/fun/Eigen.hp
##                  from <command-line>:0:
## /home/mikesh/.lib/R/library/RcppEigen/include/Eigen/src/Core/util/Macros.h:613:1: error: unknown type
##  namespace Eigen {
##  ^~~~~~~~~
## /home/mikesh/.lib/R/library/RcppEigen/include/Eigen/src/Core/util/Macros.h:613:17: error: expected ‘=
##  namespace Eigen {
##                  ^
## In file included from /home/mikesh/.lib/R/library/RcppEigen/include/Eigen/Dense:1:0,
##                  from /home/mikesh/.lib/R/library/StanHeaders/include/stan/math/prim/mat/fun/Eigen.hp
##                  from <command-line>:0:
## /home/mikesh/.lib/R/library/RcppEigen/include/Eigen/Core:96:10: fatal error: complex: No such file or
##  #include <complex>
##           ^~~~~~~~~
## compilation terminated.
## /usr/lib/R/etc/Makeconf:172: recipe for target 'foo.o' failed
## make: *** [foo.o] Error 1

## Start sampling
```

```
mdl.b.2 <- brm(logRecaptureProb ~ donor.quantile + logClonesRecepient + logClonesDonor + dli,
               data = data.coord)
```

```
## Compiling Stan program...

## recompiling to avoid crashing R session

## Trying to compile a simple C file

## Running /usr/lib/R/bin/R CMD SHLIB foo.c
## gcc -std=gnu99 -I"/usr/share/R/include" -DNDEBUG   -I"/home/mikesh/.lib/R/library/Rcpp/include/"  -I
## In file included from /home/mikesh/.lib/R/library/RcppEigen/include/Eigen/Core:88:0,
##                  from /home/mikesh/.lib/R/library/RcppEigen/include/Eigen/Dense:1,
##                  from /home/mikesh/.lib/R/library/StanHeaders/include/stan/math/prim/mat/fun/Eigen.hp
##                  from <command-line>:0:
## /home/mikesh/.lib/R/library/RcppEigen/include/Eigen/src/Core/util/Macros.h:613:1: error: unknown type
##  namespace Eigen {
##  ^~~~~~~~~
```

```
## /home/mikesh/.lib/R/library/RcppEigen/include/Eigen/src/Core/util/Macros.h:613:17: error: expected ‘=
##  namespace Eigen {
##                  ^
## In file included from /home/mikesh/.lib/R/library/RcppEigen/include/Eigen/Dense:1:0,
##                  from /home/mikesh/.lib/R/library/StanHeaders/include/stan/math/prim/mat/fun/Eigen.hp
##                  from <command-line>:0:
## /home/mikesh/.lib/R/library/RcppEigen/include/Eigen/Core:96:10: fatal error: complex: No such file o
##  #include <complex>
##           ^~~~~~~~~
## compilation terminated.
## /usr/lib/R/etc/Makeconf:172: recipe for target 'foo.o' failed
## make: *** [foo.o] Error 1
```

```
## Start sampling
```

Plot models

```
summary(mdl.b.1)
```

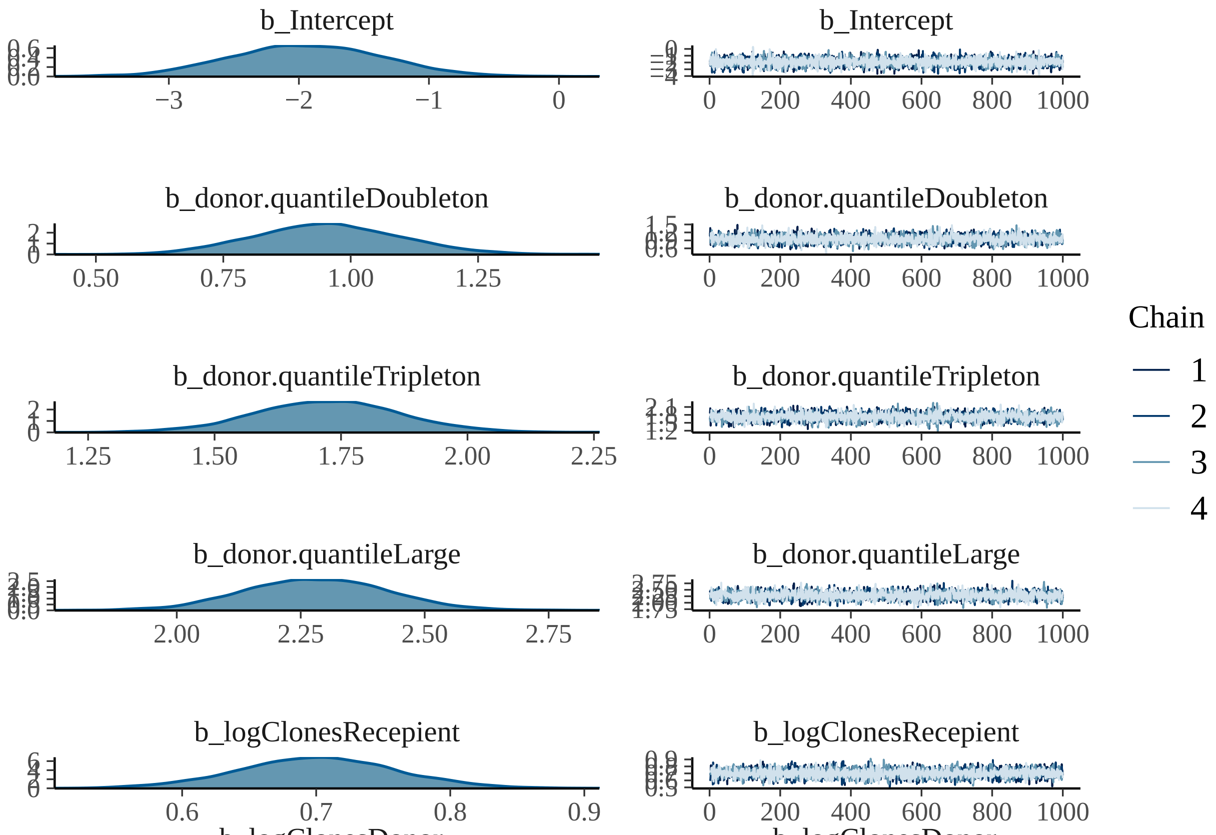
```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: logRecaptureProb ~ donor.quantile + logClonesRecepient + logClonesDonor
##    Data: data.coord (Number of observations: 76)
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##                         Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
## Intercept                  -1.94      0.58    -3.06    -0.81 1.00     6385
## donor.quantileDoubleton     0.96      0.14     0.68     1.24 1.00     3942
## donor.quantileTripleton     1.72      0.14     1.44     2.01 1.00     4356
## donor.quantileLarge         2.28      0.15     2.00     2.57 1.00     4251
## logClonesRecepient          0.70      0.06     0.58     0.82 1.00     6438
## logClonesDonor             -0.74      0.05    -0.84    -0.64 1.00     6021
##                         Tail_ESS
## Intercept                   3053
## donor.quantileDoubleton     2944
## donor.quantileTripleton     3531
## donor.quantileLarge         3267
## logClonesRecepient          2908
## logClonesDonor              3153
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma     0.45      0.04     0.38     0.53 1.00     5223     2939
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

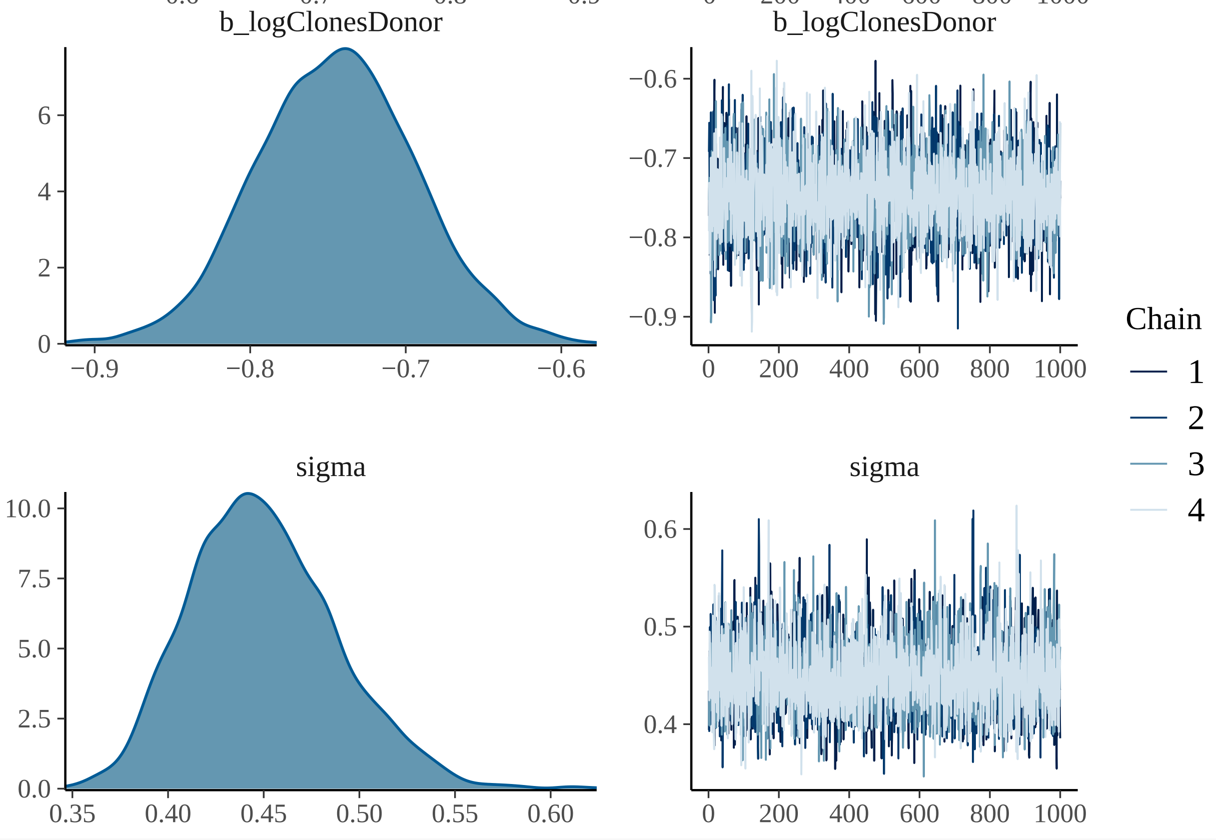
```
summary(mdl.b.2)
```

```
##  Family: gaussian
##   Links: mu = identity; sigma = identity
## Formula: logRecaptureProb ~ donor.quantile + logClonesRecepient + logClonesDonor + dli
##    Data: data.coord (Number of observations: 76)
```

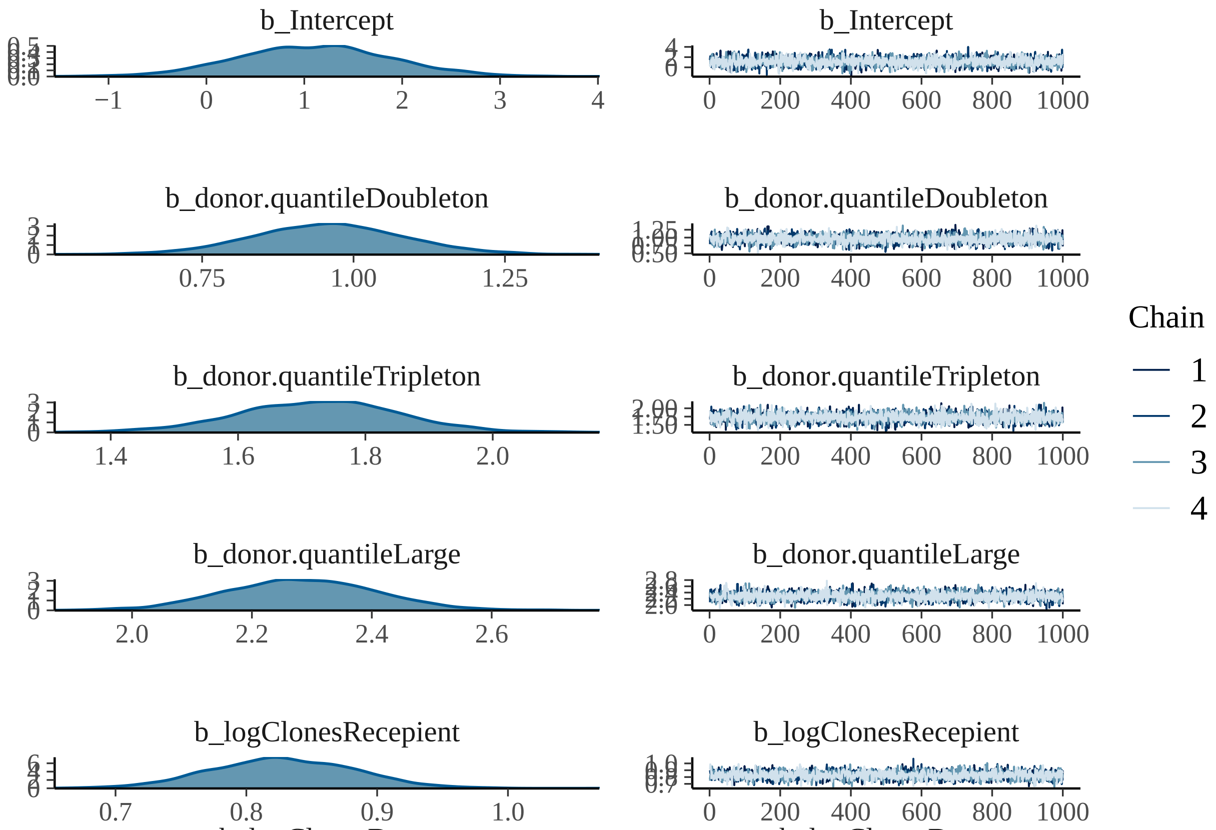
```
## Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup samples = 4000
##
## Population-Level Effects:
##                         Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS
## Intercept                   1.10      0.78    -0.42     2.64 1.00     2672
## donor.quantileDoubleton     0.96      0.12     0.71     1.20 1.00     2983
## donor.quantileTripleton     1.73      0.13     1.47     1.97 1.00     3458
## donor.quantileLarge         2.28      0.13     2.04     2.52 1.00     3061
## logClonesRecepient          0.83      0.05     0.72     0.94 1.00     3008
## logClonesDonor             -1.04      0.07    -1.19    -0.90 1.00     2153
## dliTRUE                    -0.76      0.15    -1.04    -0.48 1.00     2255
##                         Tail_ESS
## Intercept                   2965
## donor.quantileDoubleton     3191
## donor.quantileTripleton     3391
## donor.quantileLarge         3082
## logClonesRecepient          2958
## logClonesDonor              2511
## dliTRUE                     2537
##
## Family Specific Parameters:
##       Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma     0.38      0.03     0.32     0.45 1.00     4664     3215
##
## Samples were drawn using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

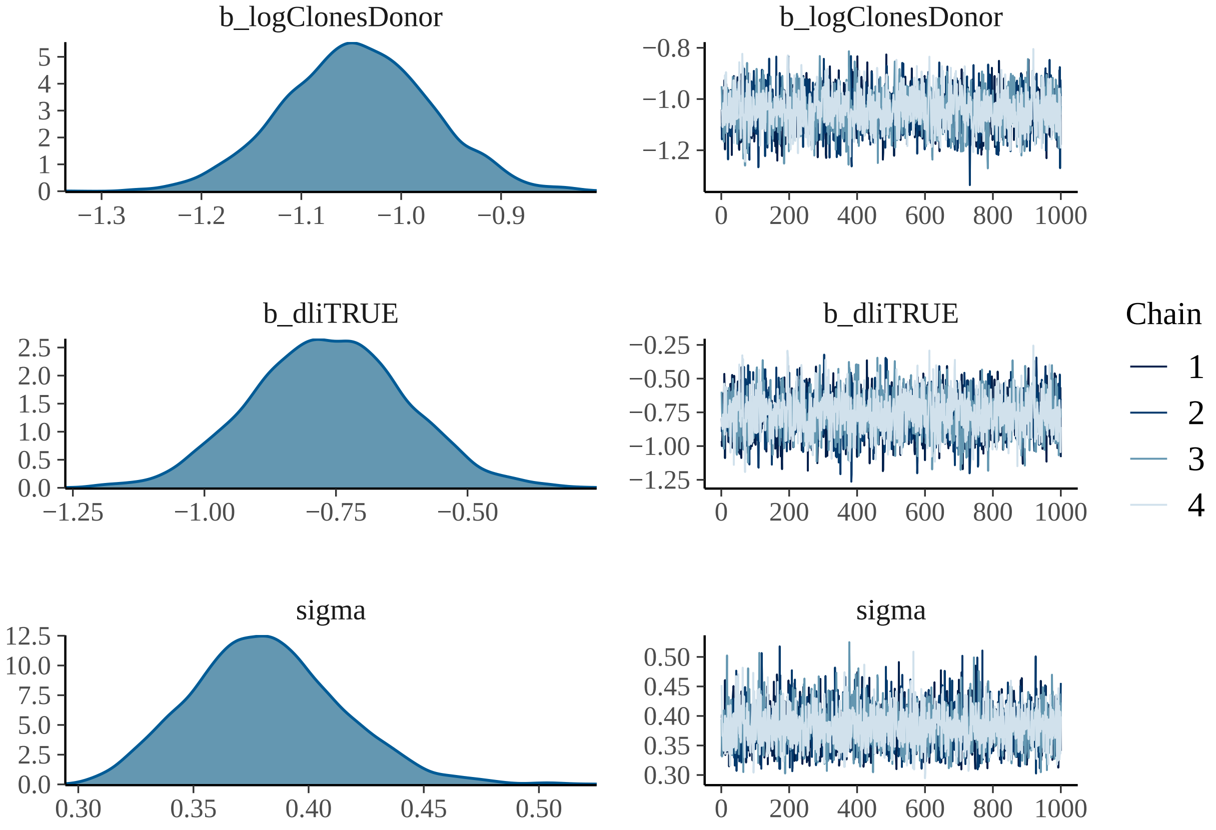
```r
plot(mdl.b.1, ask = F)
```





```
plot(mdl.b.2, ask = F)
```





```r
LOO(mdl.b.1, mdl.b.2)
```

```
## Output of model 'mdl.b.1':
##
## Computed from 4000 by 76 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo    -49.8  6.0
## p_loo         6.4  1.0
## looic        99.7 12.1
## ------
## Monte Carlo SE of elpd_loo is 0.0.
##
## All Pareto k estimates are good (k < 0.5).
## See help('pareto-k-diagnostic') for details.
##
## Output of model 'mdl.b.2':
##
## Computed from 4000 by 76 log-likelihood matrix
##
##          Estimate   SE
## elpd_loo    -38.6  7.7
## p_loo         8.2  1.7
## looic        77.1 15.3
## ------
## Monte Carlo SE of elpd_loo is 0.1.
##
## Pareto k diagnostic values:
##                          Count Pct.    Min. n_eff
## (-Inf, 0.5]   (good)     75    98.7%   707
##  (0.5, 0.7]   (ok)        1     1.3%   2201
##    (0.7, 1]   (bad)       0     0.0%   <NA>
##    (1, Inf)   (very bad)  0     0.0%   <NA>
##
## All Pareto k estimates are ok (k < 0.7).
## See help('pareto-k-diagnostic') for details.
##
## Model comparisons:
##         elpd_diff se_diff
## mdl.b.2   0.0       0.0
## mdl.b.1 -11.3       5.1
```