

R Notebook

General results:

```
summarised_df <- results_data_frame |>
  group_by(data_generation, data_fitted) |>
  summarise(mean_point          = mean(point, na.rm = TRUE),
            mean_ci_length_norm = mean(conf_int_normal_upper - conf_int_normal_lower, na.rm = TRUE),
            coverage_ci_norm    = mean((conf_int_normal_lower < 1000) & (1000 < conf_int_normal_upper), na.rm = TRUE),
            mean_ci_length_log_norm = mean(conf_int_log_normal_upper - conf_int_log_normal_lower, na.rm = TRUE),
            coverage_ci_log_norm  = mean((conf_int_log_normal_lower < 1000) & (1000 < conf_int_log_normal_upper), na.rm = TRUE),
            succesful_fits       = mean(!is.na(point)))
```

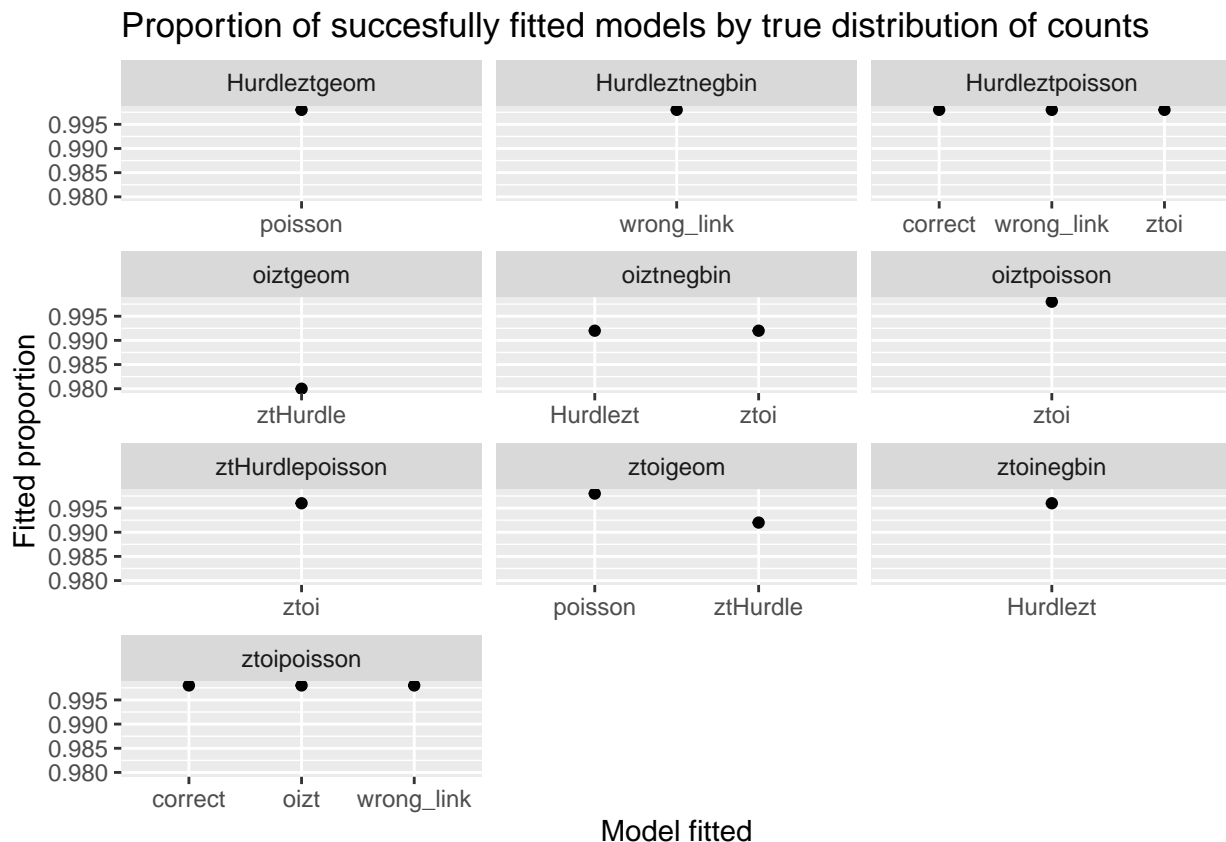
```
## 'summarise()' has grouped output by 'data_generation'. You can override using
## the '.groups' argument.
```

```
print(summarised_df, n=20)
```

```
## # A tibble: 80 x 8
## # Groups:   data_generation [12]
##   data_generation data_fitted mean_point mean_ci_length_norm coverage_ci_norm
##   <chr>           <chr>         <dbl>         <dbl>         <dbl>
## 1 Hurdleztgeom    correct      1.01e 3      1.67e 2      0.93
## 2 Hurdleztgeom    negbin      3.77e 6      1.84e 6      0.896
## 3 Hurdleztgeom    oizt        8.72e 2      6.96e 1      0
## 4 Hurdleztgeom    poisson     6.99e 2      2.40e 1      0
## 5 Hurdleztgeom    wrong_link  1.01e 3      1.67e 2      0.932
## 6 Hurdleztgeom    ztHurdle    1.09e 3      2.25e 2      0.734
## 7 Hurdleztgeom    ztoi        8.73e 2      7.89e 1      0
## 8 Hurdleztnegbin  correct     1.19e15      7.15e14      0.886
## 9 Hurdleztnegbin  geom        1.19e 3      1.80e 2      0
## 10 Hurdleztnegbin oizt        9.95e 2      1.83e 2      0.84
## 11 Hurdleztnegbin poisson     8.16e 2      2.23e 1      0
## 12 Hurdleztnegbin wrong_link  9.15e13      3.95e14      0.840
## 13 Hurdleztnegbin ztHurdle    3.02e 6      1.09e 7      0.896
## 14 Hurdleztnegbin ztoi        3.02e 6      1.09e 7      0.896
## 15 Hurdleztpoisson correct      1.00e 3      1.04e 2      0.956
## 16 Hurdleztpoisson geometric  2.28e 3      8.24e 2      0
## 17 Hurdleztpoisson oizt        9.07e 2      4.23e 1      0
## 18 Hurdleztpoisson wrong_link  1.00e 3      1.04e 2      0.958
## 19 Hurdleztpoisson ztHurdle    1.07e 3      1.44e 2      0.704
## 20 Hurdleztpoisson ztoi        9.08e 2      4.24e 1      0
## # i 60 more rows
## # i 3 more variables: mean_ci_length_log_norm <dbl>,
## #   coverage_ci_log_norm <dbl>, succesful_fits <dbl>
```

```
pp <- summarised_df |>
  subset(succesful_fits < 1) |>
  as.data.frame() |>
  mutate(data_generation = ordered(data_generation)) |>
  ggplot(aes(y = succesful_fits, x = data_fitted)) +
  geom_point() +
  facet_wrap(~data_generation, scales = c("free_x"), ncol = 3) +
  ylab("Fitted proportion") +
  xlab("Model fitted") +
  ggtitle("Proportion of succesfully fitted models by true distribution of counts")
```

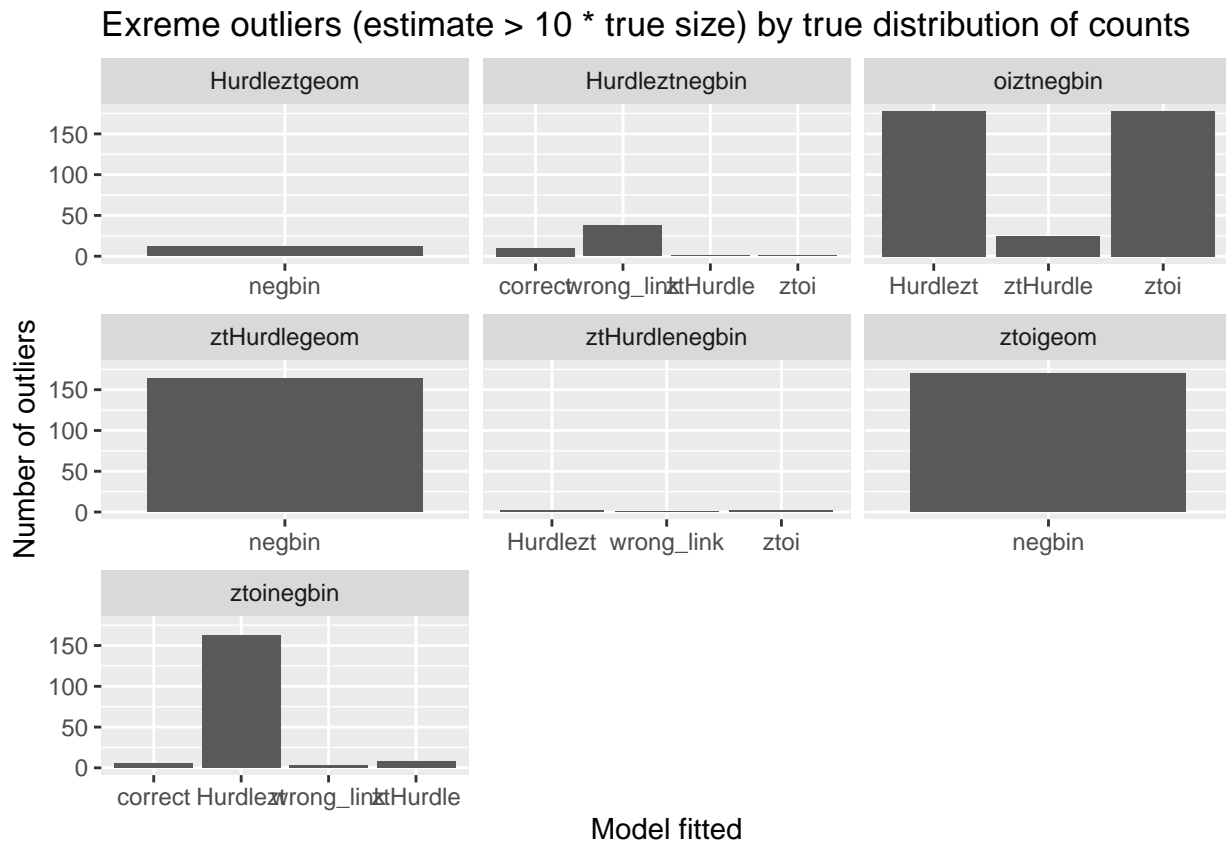
pp



Visualising outliers (i.e. when estimated regression parameters tend to boundary):

```
results_data_frame |>
  subset(!is.na(point)) |>
  subset(point > 10000) |>
  group_by(data_generation, data_fitted) |>
  summarise(n = n()) |>
  ggplot(aes(x = data_fitted, weight = n)) +
  geom_bar() +
  facet_wrap(~data_generation, scales = c("free_x")) +
  ylab("Number of outliers") +
  xlab("Model fitted") +
  ggtitle("Exreme outliers (estimate > 10 * true size) by true distribution of counts")
```

```
## 'summarise()' has grouped output by 'data_generation'. You can override using
## the '.groups' argument.
```

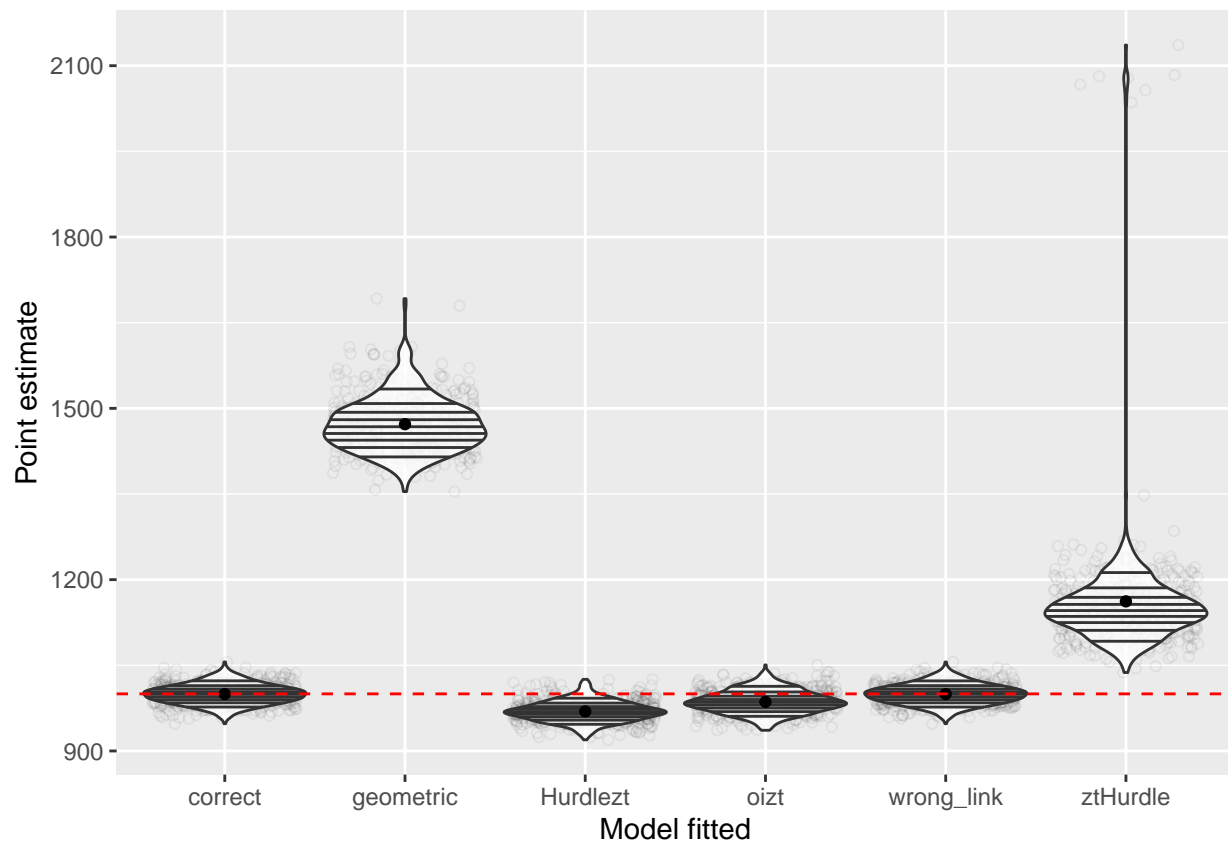


Point estimates

Results for counts generated by ztoipoisson:

```
p1 <- results_data_frame |>
  subset(!is.na(point) & (data_generation == "ztoipoisson")) |>
  subset(point < 25000) |>
  ggplot(aes(x = data_fitted, y = point)) +
  geom_jitter(alpha = 0.05, shape = 1) +
  geom_violin(alpha = 0.8, draw_quantiles = 1:9 / 10, scale = "width") +
  stat_summary(fun = function(x) mean(x, na.rm = TRUE), geom = "point") +
  geom_hline(yintercept = 1000, linetype="dashed", color = "red") +
  ylab("Point estimate") +
  xlab("Model fitted")
```

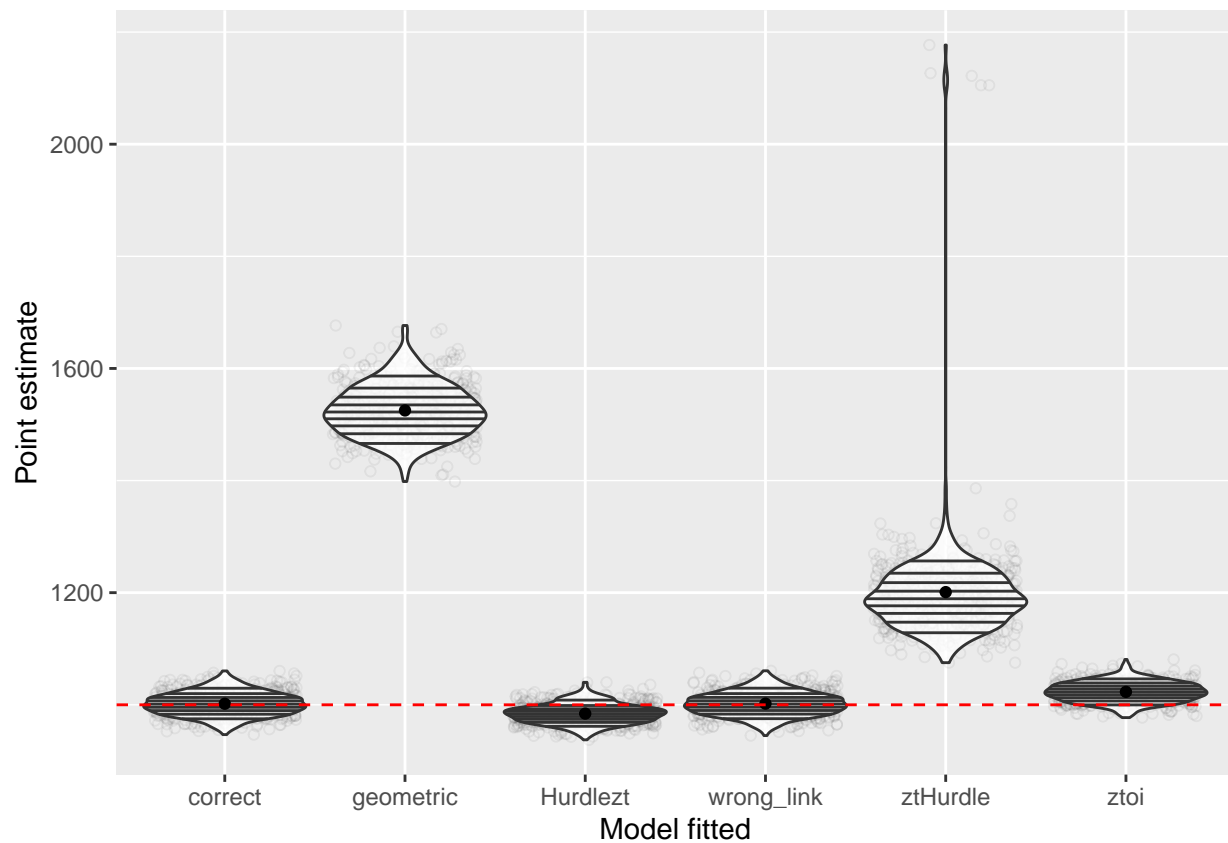
p1



Results for counts generated by oiztpoisson:

```
p2 <- results_data_frame |>
  subset(!is.na(point) & (data_generation == "oiztpoisson")) |>
  subset(point < 25000) |>
  ggplot(aes(x = data_fitted, y = point)) +
    geom_jitter(alpha = 0.05, shape = 1) +
    geom_violin(alpha = 0.8, draw_quantiles = 1:9 / 10, scale = "width") +
    stat_summary(fun = function(x) mean(x, na.rm = TRUE), geom = "point") +
    geom_hline(yintercept = 1000, linetype="dashed", color = "red") +
    ylab("Point estimate") +
    xlab("Model fitted")
```

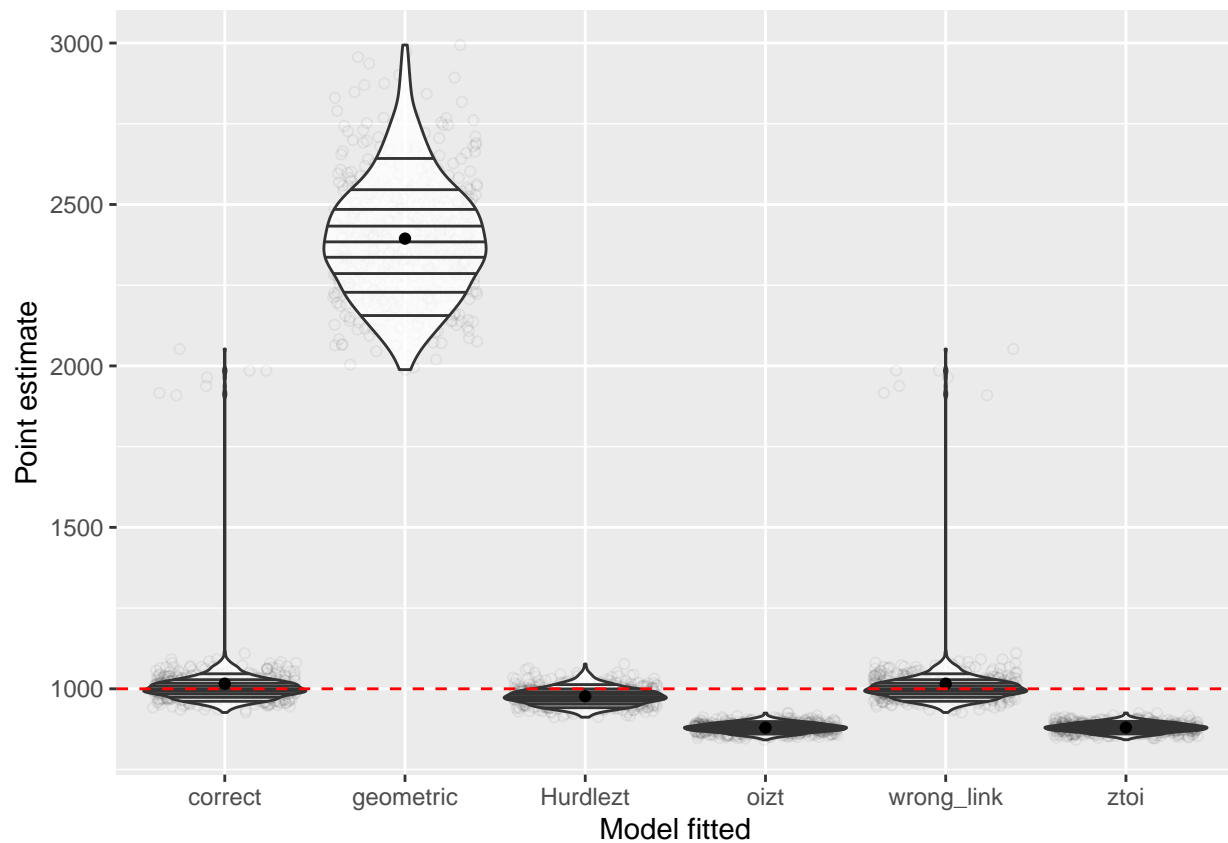
p2



Results for counts generated by ztHurdlepoisson:

```
p3 <- results_data_frame |>
  subset(!is.na(point) & (data_generation == "ztHurdlepoisson")) |>
  subset(point < 25000) |>
  ggplot(aes(x = data_fitted, y = point)) +
    geom_jitter(alpha = 0.05, shape = 1) +
    geom_violin(alpha = 0.8, draw_quantiles = 1:9 / 10, scale = "width") +
    stat_summary(fun = function(x) mean(x, na.rm = TRUE), geom = "point") +
    geom_hline(yintercept = 1000, linetype="dashed", color = "red") +
    ylab("Point estimate") +
    xlab("Model fitted")
```

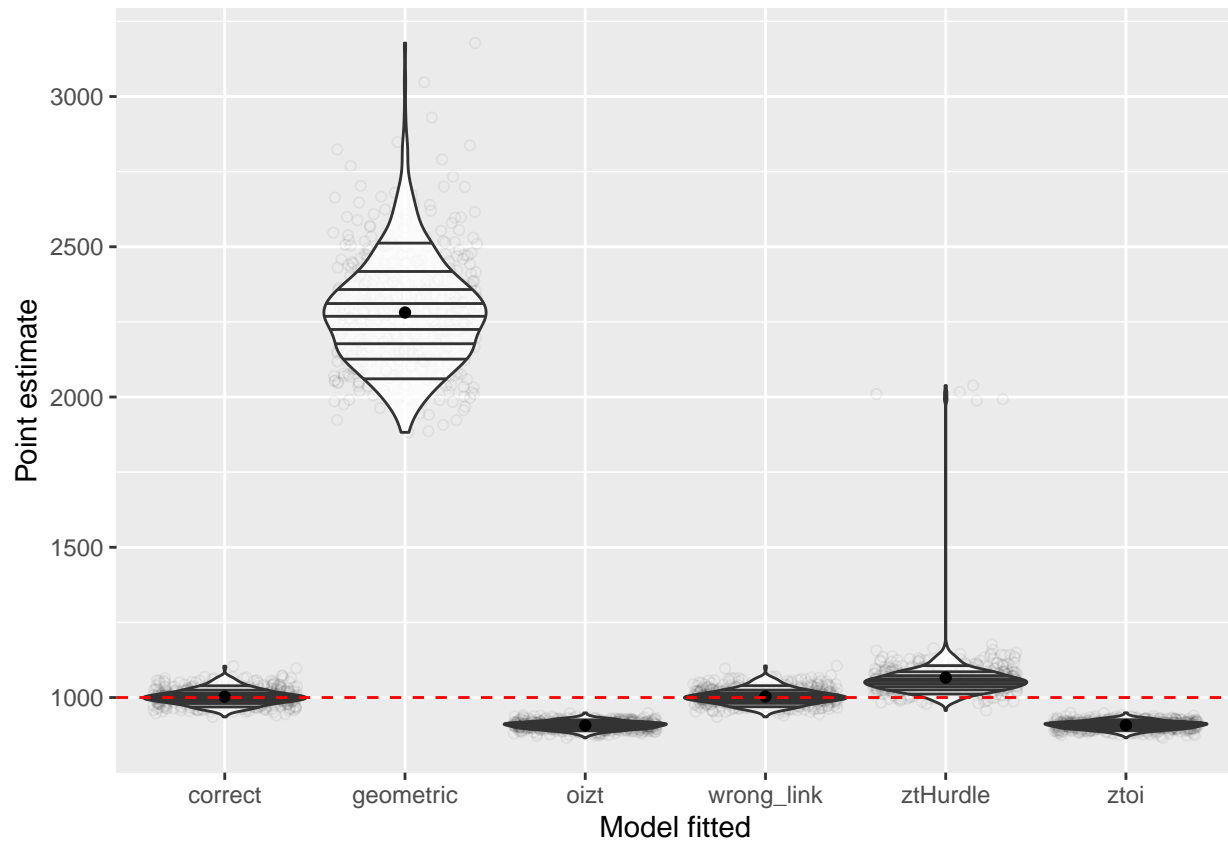
p3



Results for counts generated by hurdleztpoisson:

```
p4 <- results_data_frame |>
  subset(!is.na(point) & (data_generation == "Hurdleztpoisson")) |>
  subset(point < 25000) |>
  ggplot(aes(x = data_fitted, y = point)) +
    geom_jitter(alpha = 0.05, shape = 1) +
    geom_violin(alpha = 0.8, draw_quantiles = 1:9 / 10, scale = "width") +
    stat_summary(fun = function(x) mean(x, na.rm = TRUE), geom = "point") +
    geom_hline(yintercept = 1000, linetype="dashed", color = "red") +
    ylab("Point estimate") +
    xlab("Model fitted")
```

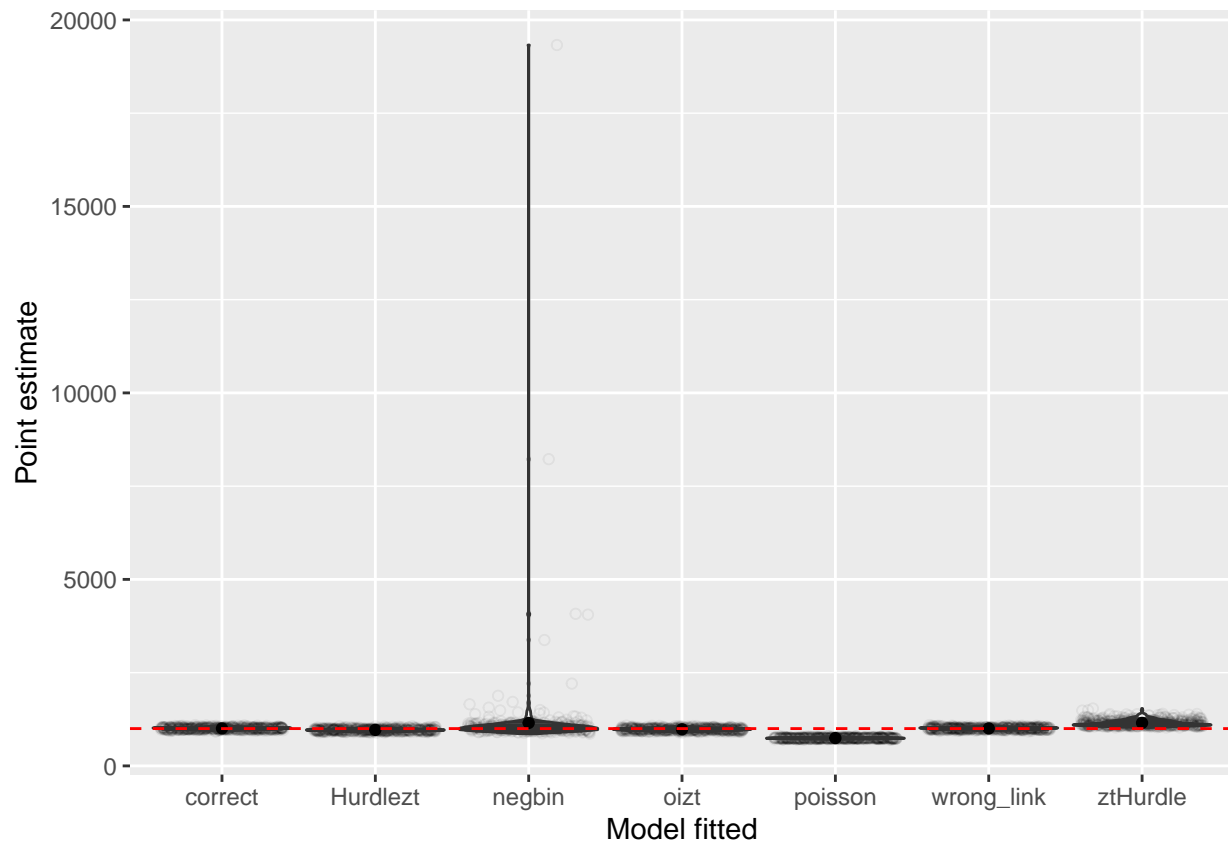
p4



Results for counts generated by ztoigeom:

```
p5 <- results_data_frame |>
  subset(!is.na(point) & (data_generation == "ztoigeom")) |>
  subset(point < 25000) |>
  ggplot(aes(x = data_fitted, y = point)) +
  geom_jitter(alpha = 0.05, shape = 1) +
  geom_violin(alpha = 0.8, draw_quantiles = 1:9 / 10, scale = "width") +
  stat_summary(fun = function(x) mean(x, na.rm = TRUE), geom = "point") +
  geom_hline(yintercept = 1000, linetype="dashed", color = "red") +
  ylab("Point estimate") +
  xlab("Model fitted")
```

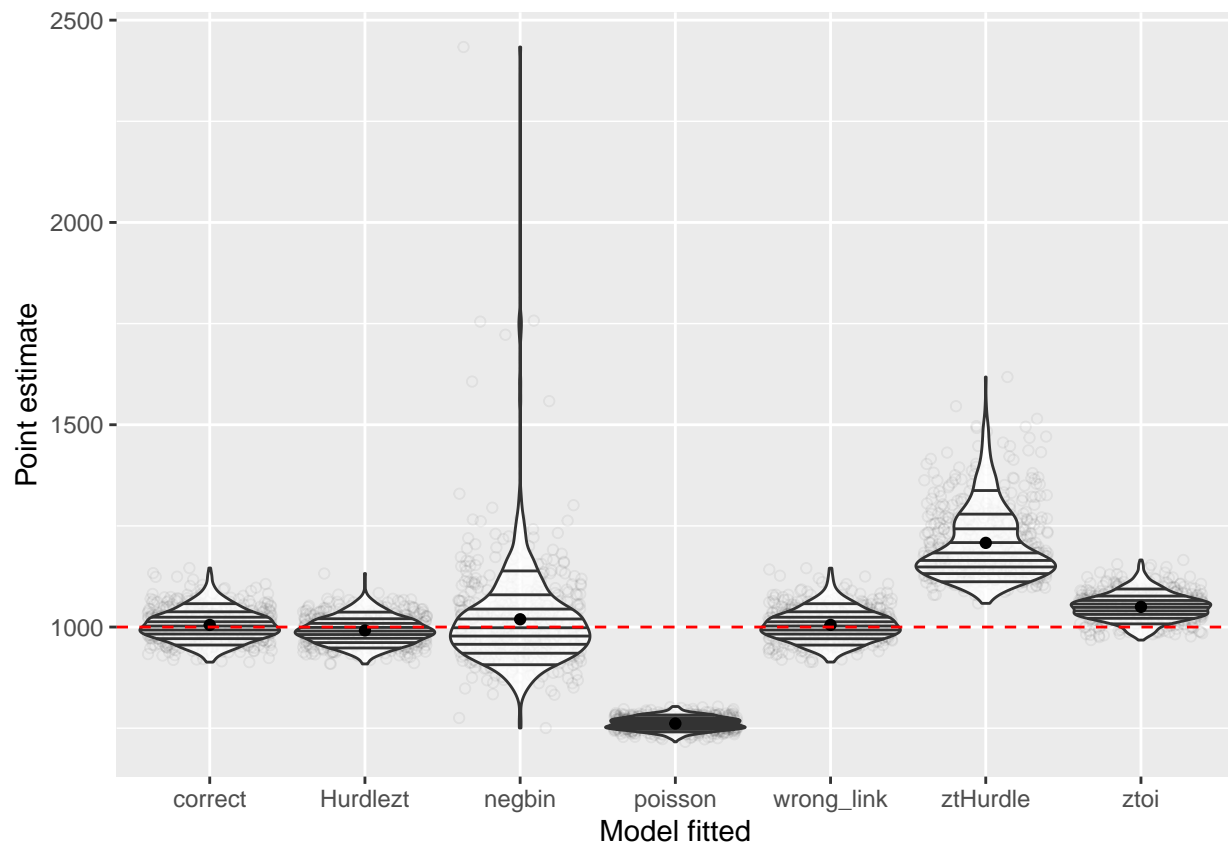
p5



Results for counts generated by oiztgeom:

```
p6 <- results_data_frame |>
  subset(!is.na(point) & (data_generation == "oiztgeom")) |>
  subset(point < 25000) |>
  ggplot(aes(x = data_fitted, y = point)) +
    geom_jitter(alpha = 0.05, shape = 1) +
    geom_violin(alpha = 0.8, draw_quantiles = 1:9 / 10, scale = "width") +
    stat_summary(fun = function(x) mean(x, na.rm = TRUE), geom = "point") +
    geom_hline(yintercept = 1000, linetype="dashed", color = "red") +
    ylab("Point estimate") +
    xlab("Model fitted")
```

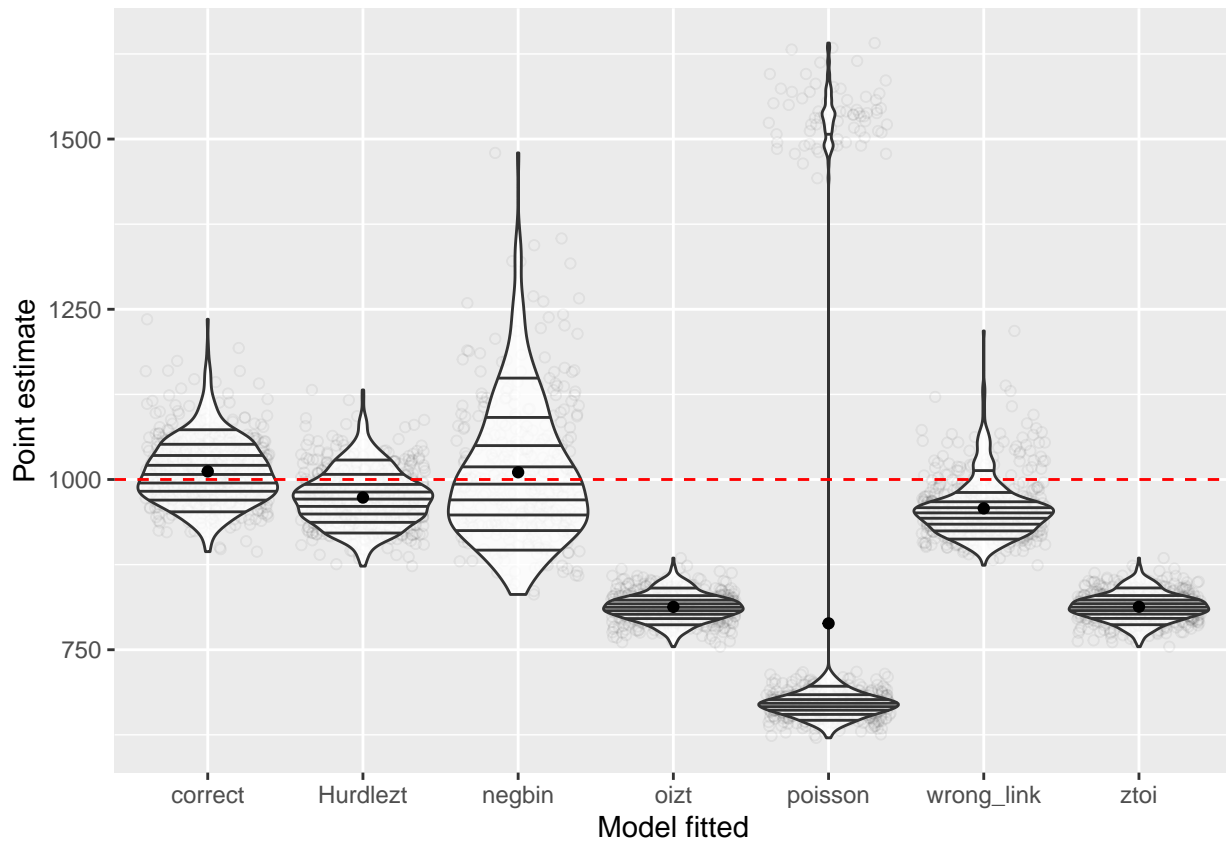
p6



Results for counts generated by ztHurdlegeom:

```
p7 <- results_data_frame |>
  subset(!is.na(point) & (data_generation == "ztHurdlegeom")) |>
  subset(point < 25000) |>
  ggplot(aes(x = data_fitted, y = point)) +
    geom_jitter(alpha = 0.05, shape = 1) +
    geom_violin(alpha = 0.8, draw_quantiles = 1:9 / 10, scale = "width") +
    stat_summary(fun = function(x) mean(x, na.rm = TRUE), geom = "point") +
    geom_hline(yintercept = 1000, linetype="dashed", color = "red") +
    ylab("Point estimate") +
    xlab("Model fitted")
```

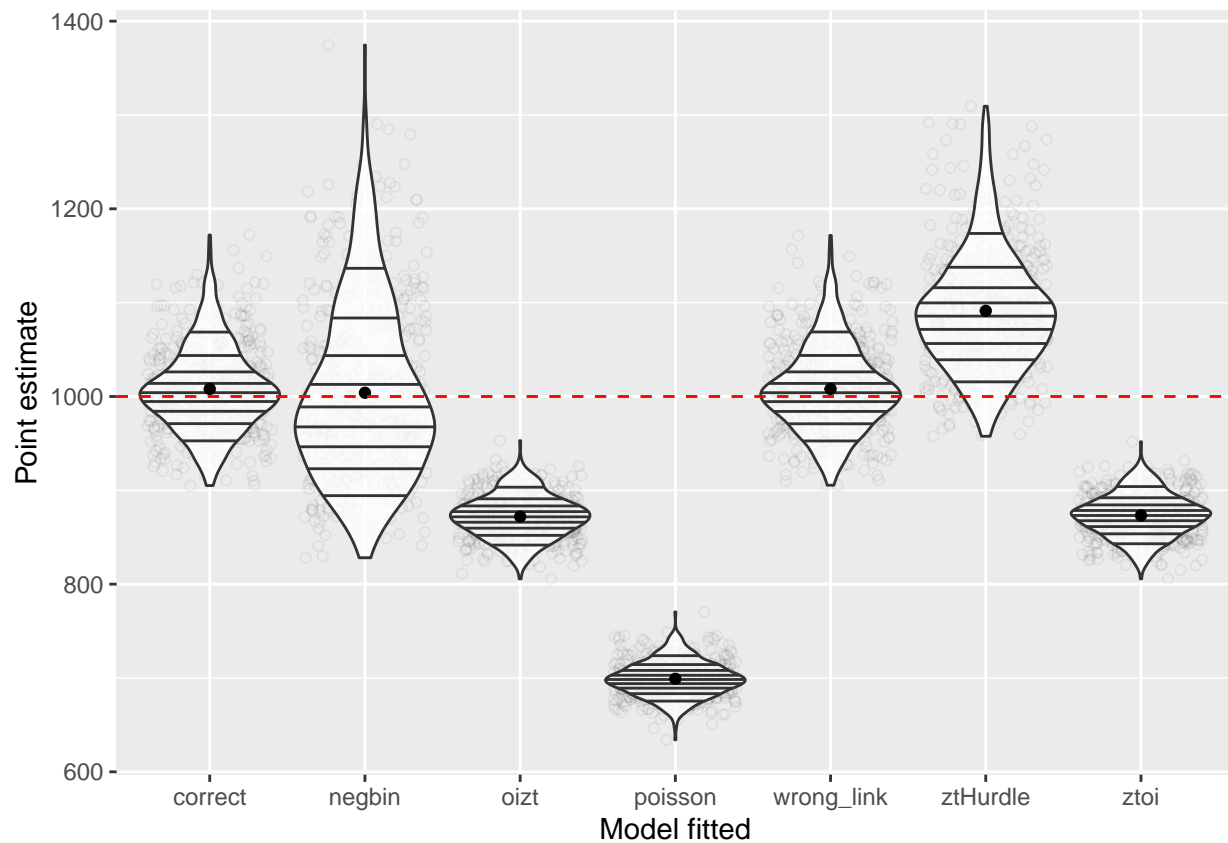
p7



Results for counts generated by hurdleztgeom:

```
p8 <- results_data_frame |>
  subset(!is.na(point) & (data_generation == "Hurdleztgeom")) |>
  subset(point < 25000) |>
  ggplot(aes(x = data_fitted, y = point)) +
    geom_jitter(alpha = 0.05, shape = 1) +
    geom_violin(alpha = 0.8, draw_quantiles = 1:9 / 10, scale = "width") +
    stat_summary(fun = function(x) mean(x, na.rm = TRUE), geom = "point") +
    geom_hline(yintercept = 1000, linetype="dashed", color = "red") +
    ylab("Point estimate") +
    xlab("Model fitted")
```

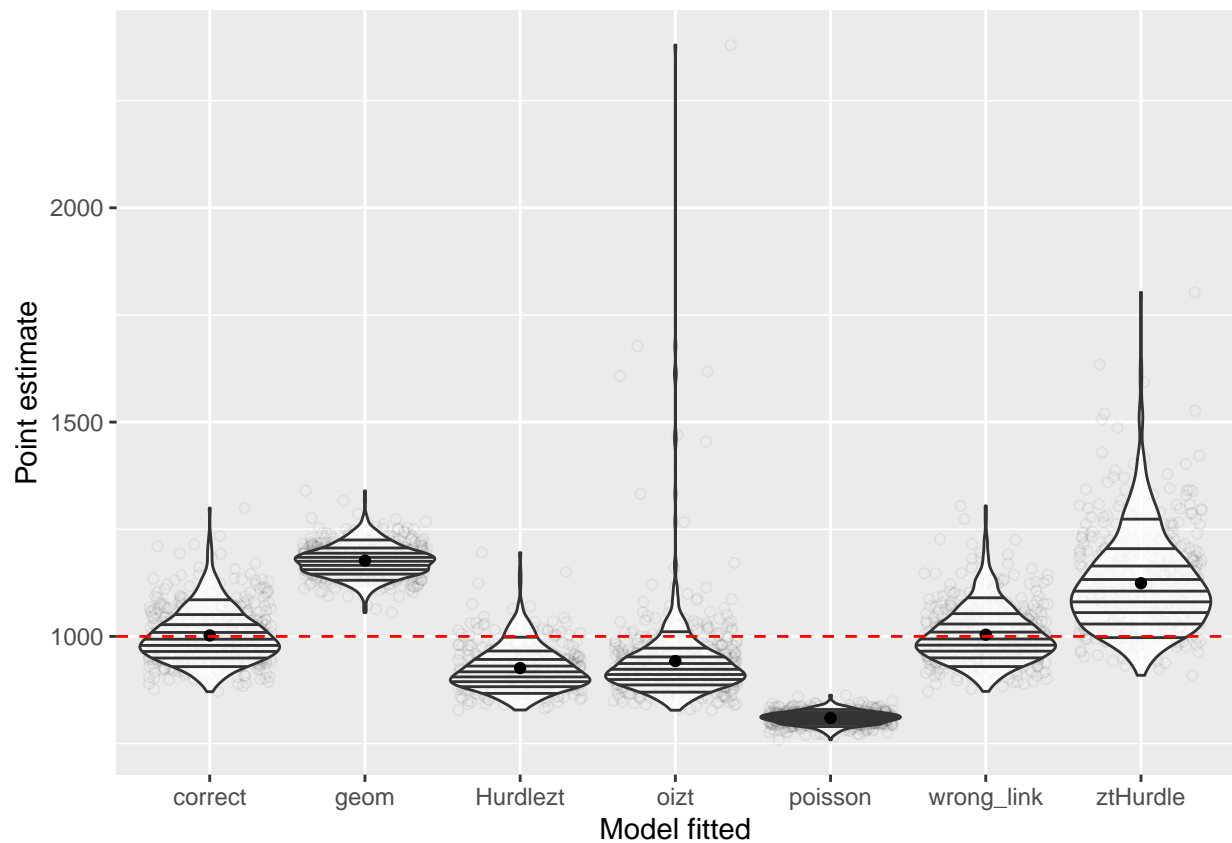
p8



Results for counts generated by ztoinegbin:

```
p9 <- results_data_frame |>
  subset(!is.na(point) & (data_generation == "ztoinegbin")) |>
  subset(point < 10000) |>
  ggplot(aes(x = data_fitted, y = point)) +
  geom_jitter(alpha = 0.05, shape = 1) +
  geom_violin(alpha = 0.8, draw_quantiles = 1:9 / 10, scale = "width") +
  stat_summary(fun = function(x) mean(x, na.rm = TRUE), geom = "point") +
  geom_hline(yintercept = 1000, linetype="dashed", color = "red") +
  ylab("Point estimate") +
  xlab("Model fitted")
```

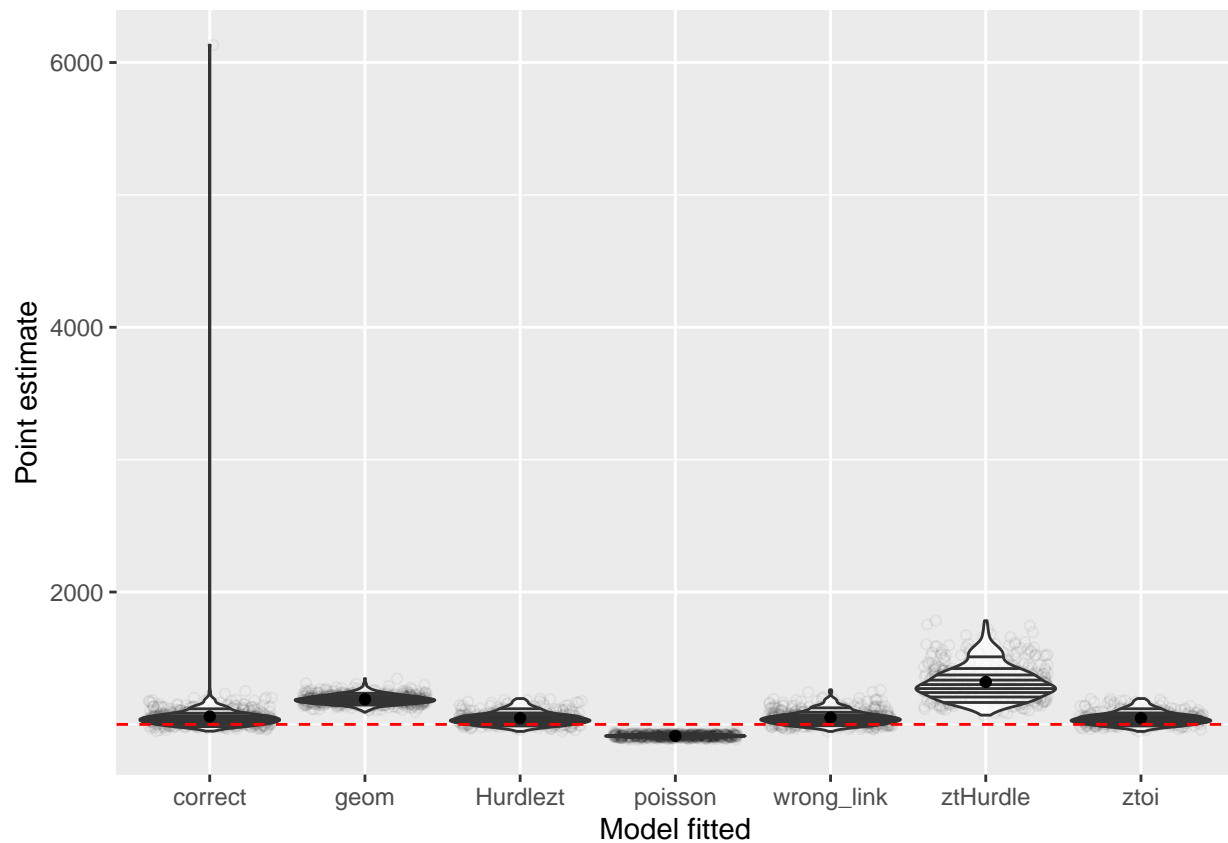
p9



Results for counts generated by oiztnegbin:

```
p10 <- results_data_frame |>
  subset(!is.na(point) & (data_generation == "oiztnegbin")) |>
  subset(point < 10000) |>
  ggplot(aes(x = data_fitted, y = point)) +
  geom_jitter(alpha = 0.05, shape = 1) +
  geom_violin(alpha = 0.8, draw_quantiles = 1:9 / 10, scale = "width") +
  stat_summary(fun = function(x) mean(x, na.rm = TRUE), geom = "point") +
  geom_hline(yintercept = 1000, linetype="dashed", color = "red") +
  ylab("Point estimate") +
  xlab("Model fitted")
```

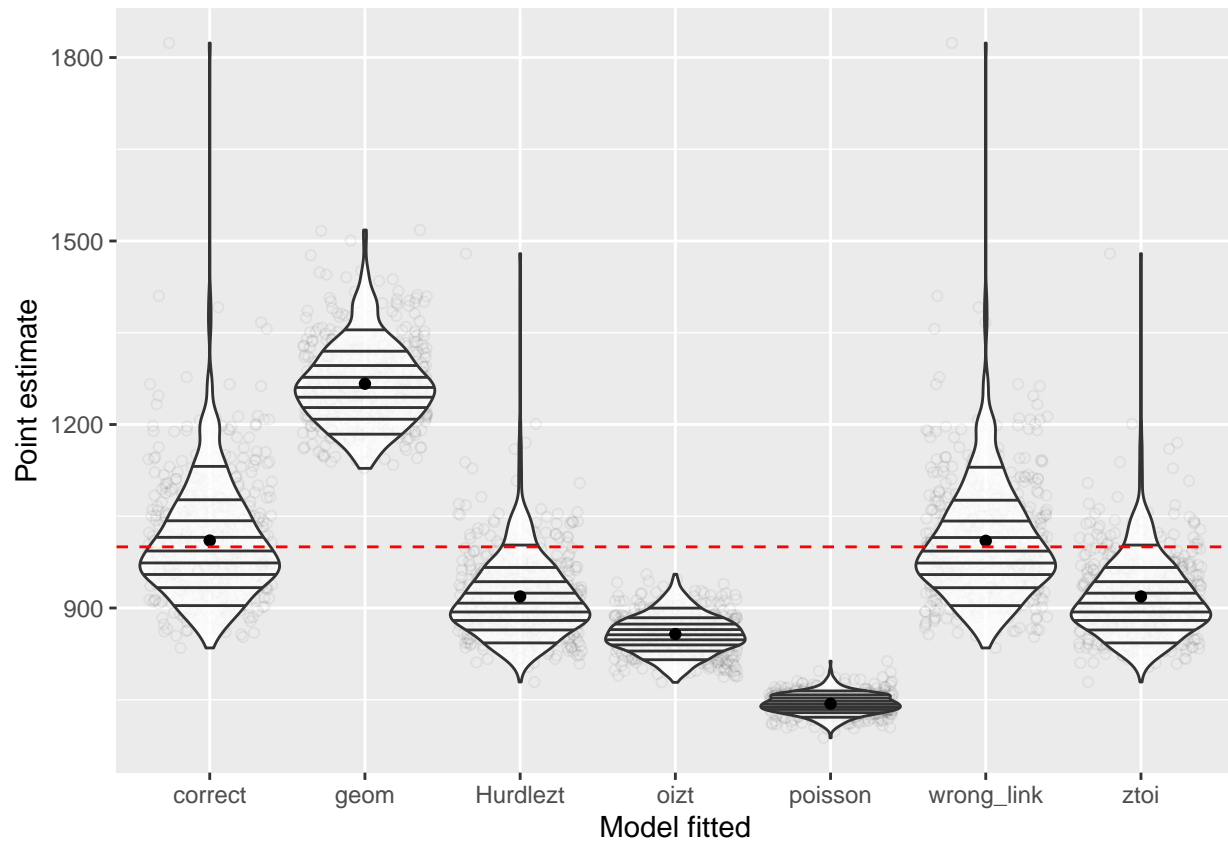
p10



Results for counts generated by ztHurdlenegbin:

```
p11 <- results_data_frame |>
  subset(!is.na(point) & (data_generation == "ztHurdlenegbin")) |>
  subset(point < 25000) |>
  ggplot(aes(x = data_fitted, y = point)) +
  geom_jitter(alpha = 0.05, shape = 1) +
  geom_violin(alpha = 0.8, draw_quantiles = 1:9 / 10, scale = "width") +
  stat_summary(fun = function(x) mean(x, na.rm = TRUE), geom = "point") +
  geom_hline(yintercept = 1000, linetype="dashed", color = "red") +
  ylab("Point estimate") +
  xlab("Model fitted")
```

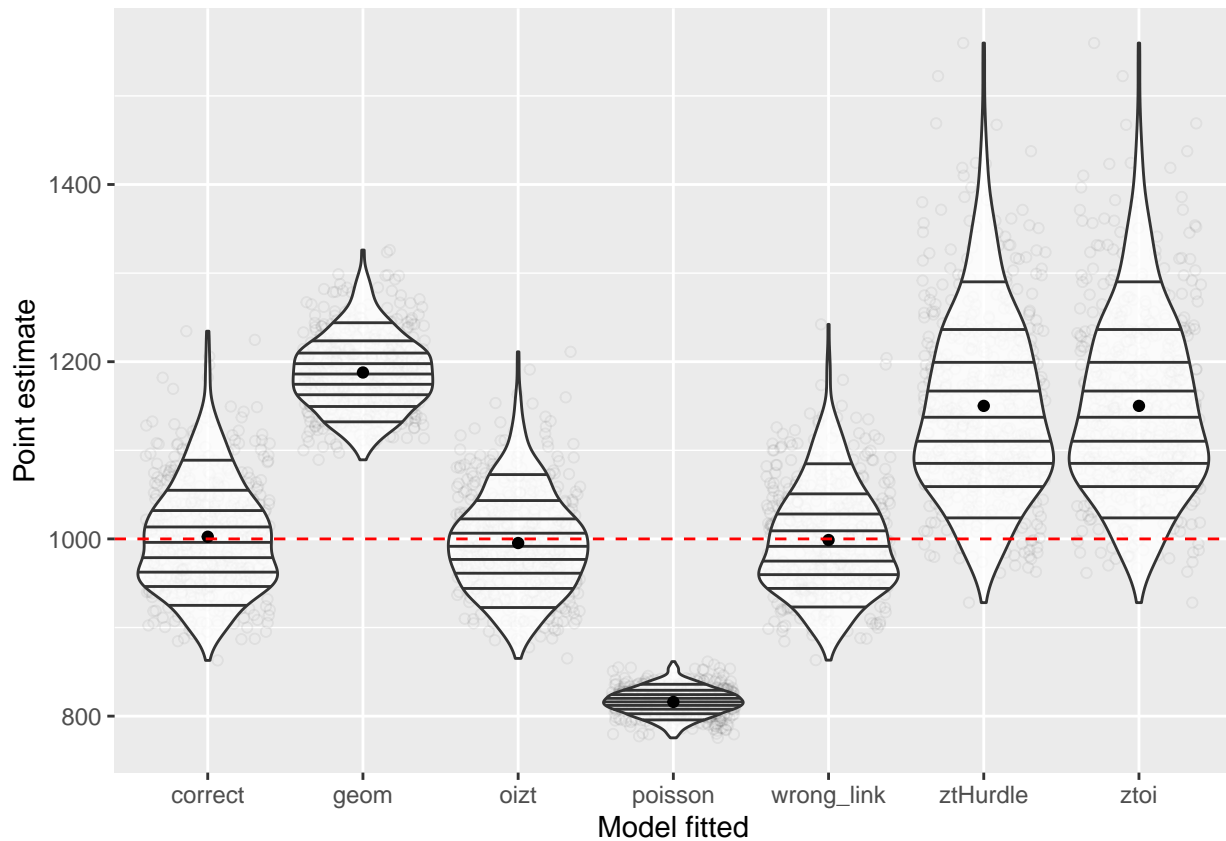
p11



Results for counts generated by hurdleztneqbin:

```
p12 <- results_data_frame |>
  subset(!is.na(point) & (data_generation == "Hurdleztneqbin")) |>
  subset(point < 25000) |>
  ggplot(aes(x = data_fitted, y = point)) +
    geom_jitter(alpha = 0.05, shape = 1) +
    geom_violin(alpha = 0.8, draw_quantiles = 1:9 / 10, scale = "width") +
    stat_summary(fun = function(x) mean(x, na.rm = TRUE), geom = "point") +
    geom_hline(yintercept = 1000, linetype="dashed", color = "red") +
    ylab("Point estimate") +
    xlab("Model fitted")
```

p12



Confidence intervals

Normal

Exact binomial tests for coverage of lognormal confidence intervals with $H_0 : p = 0.95$, $H_1 = \neg H_0$:

```
dd <- results_data_frame |>
  subset(!is.na(point)) |>
  subset(point < 25000) |>
  mutate(covr_norm = (conf_int_normal_lower < 1000) & (conf_int_normal_upper > 1000),
         covr_log = (conf_int_log_normal_lower < 1000) & (conf_int_log_normal_upper > 1000)) |>
  group_by(data_generation, data_fitted) |>
  summarise(n = n(),
            mean = mean(covr_norm, na.rm = TRUE))
```

'summarise()' has grouped output by 'data_generation'. You can override using
the '.groups' argument.

```
dd <- cbind(dd, p_value = NA, lower = NA, upper = NA)

for (x in 1:NROW(dd)) {
  jj <- binom.test(x = as.numeric(dd[x, 4]) * as.integer(dd[x, 3]), n = as.integer(dd[x, 3]), p = .95)
  # this jj object has some very weird interactions with the rest of R ecosystem
  dd[x, 5] <- jj$p.value |> as.numeric()
  dd[x, 6] <- jj[[4]][1]
```

```

dd[x, 7] <- jj[[4]][2]
}

print(dd[, c(1:2, 4, 5)] |> mutate(p_value = round(p_value, digits = 4)),
      n = NROW(dd))

```

```

## # A tibble: 80 x 4
## # Groups:   data_generation [12]
##   data_generation data_fitted   mean p_value
##   <chr>           <chr>       <dbl>   <dbl>
## 1 Hurdleztgeom    correct    0.93    0.0501
## 2 Hurdleztgeom    negbin     0.918   0.0024
## 3 Hurdleztgeom    oizt       0        0
## 4 Hurdleztgeom    poisson    0        0
## 5 Hurdleztgeom    wrong_link 0.932    0.0797
## 6 Hurdleztgeom    ztHurdle   0.734    0
## 7 Hurdleztgeom    ztoi       0        0
## 8 Hurdleztnegbin  correct    0.904    0
## 9 Hurdleztnegbin  geom       0        0
## 10 Hurdleztnegbin  oizt       0.84     0
## 11 Hurdleztnegbin  poisson    0        0
## 12 Hurdleztnegbin  wrong_link 0.896    0
## 13 Hurdleztnegbin  ztHurdle   0.896    0
## 14 Hurdleztnegbin  ztoi       0.896    0
## 15 Hurdleztpoisson correct    0.956   0.608
## 16 Hurdleztpoisson geometric    0        0
## 17 Hurdleztpoisson oizt       0        0
## 18 Hurdleztpoisson wrong_link 0.958    0.473
## 19 Hurdleztpoisson ztHurdle   0.704    0
## 20 Hurdleztpoisson ztoi       0        0
## 21 oiztgeom       Hurdlezt   0.924    0.0132
## 22 oiztgeom       correct    0.906    0
## 23 oiztgeom       negbin     0.834    0
## 24 oiztgeom       poisson    0        0
## 25 oiztgeom       wrong_link 0.9       0
## 26 oiztgeom       ztHurdle   0.0592   0
## 27 oiztgeom       ztoi       0.71     0
## 28 oiztnegbin     Hurdlezt   0.997    0
## 29 oiztnegbin     correct    0.992    0
## 30 oiztnegbin     geom       0        0
## 31 oiztnegbin     poisson    0        0
## 32 oiztnegbin     wrong_link 0.996    0
## 33 oiztnegbin     ztHurdle   0.0505   0
## 34 oiztnegbin     ztoi       0.997    0
## 35 oiztpoisson    Hurdlezt   0.776    0
## 36 oiztpoisson    correct    0.918    0.002
## 37 oiztpoisson    geometric    0        0
## 38 oiztpoisson    wrong_link 0.916    0.0013
## 39 oiztpoisson    ztHurdle   0        0
## 40 oiztpoisson    ztoi       0.772    0
## 41 ztHurdlegeom    Hurdlezt   0.864    0
## 42 ztHurdlegeom    correct    0.972    0.0232
## 43 ztHurdlegeom    negbin     0.940    0.382

```

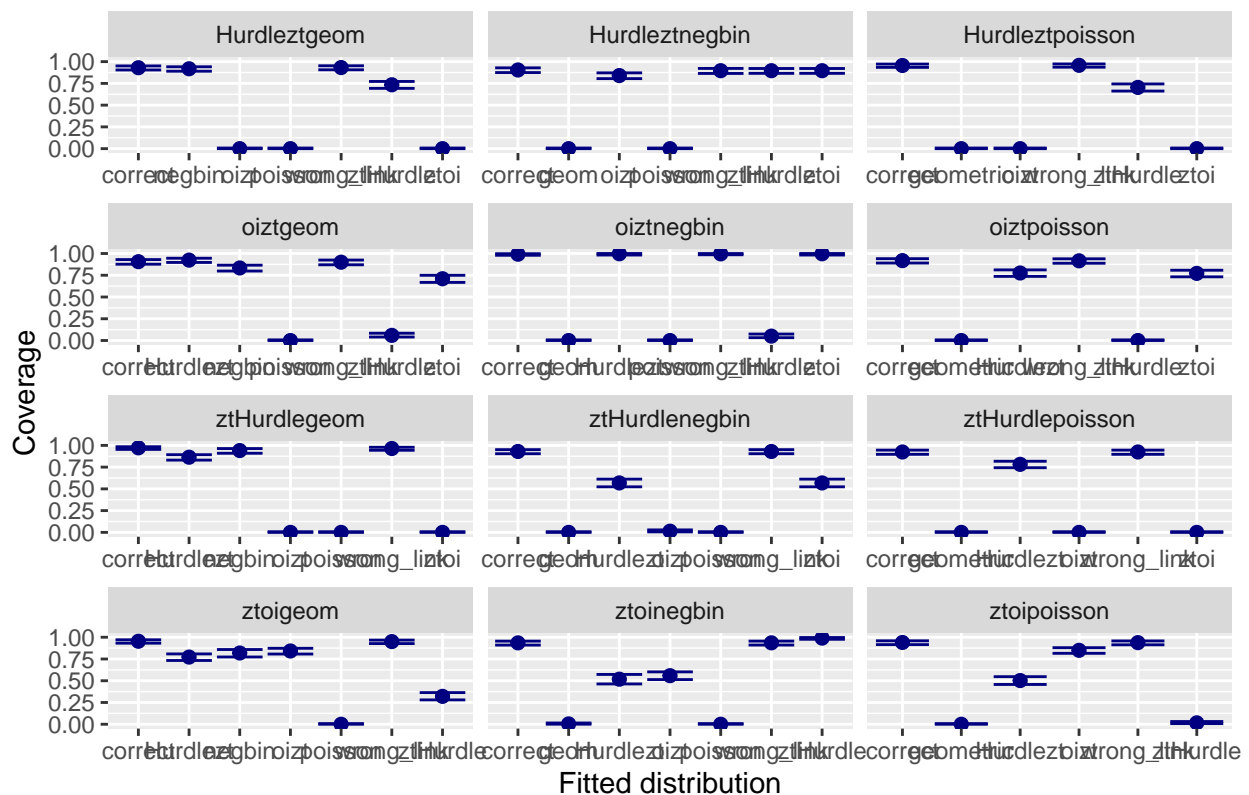

## 44	ztHurdlegeom	oizt	0	0
## 45	ztHurdlegeom	poisson	0	0
## 46	ztHurdlegeom	wrong_link	0.964	0.181
## 47	ztHurdlegeom	ztoi	0	0
## 48	ztHurdlenegbin	Hurdlezt	0.568	0
## 49	ztHurdlenegbin	correct	0.93	0.0501
## 50	ztHurdlenegbin	geom	0	0
## 51	ztHurdlenegbin	oizt	0.014	0
## 52	ztHurdlenegbin	poisson	0	0
## 53	ztHurdlenegbin	wrong_link	0.930	0.0498
## 54	ztHurdlenegbin	ztoi	0.568	0
## 55	ztHurdlepoisson	Hurdlezt	0.782	0
## 56	ztHurdlepoisson	correct	0.924	0.0132
## 57	ztHurdlepoisson	geometric	0	0
## 58	ztHurdlepoisson	oizt	0	0
## 59	ztHurdlepoisson	wrong_link	0.924	0.0132
## 60	ztHurdlepoisson	ztoi	0	0
## 61	ztoigeom	Hurdlezt	0.772	0
## 62	ztoigeom	correct	0.954	0.758
## 63	ztoigeom	negbin	0.819	0
## 64	ztoigeom	oizt	0.842	0
## 65	ztoigeom	poisson	0	0
## 66	ztoigeom	wrong_link	0.95	1
## 67	ztoigeom	ztHurdle	0.321	0
## 68	ztoinegbin	Hurdlezt	0.518	0
## 69	ztoinegbin	correct	0.935	0.147
## 70	ztoinegbin	geom	0.004	0
## 71	ztoinegbin	oizt	0.558	0
## 72	ztoinegbin	poisson	0	0
## 73	ztoinegbin	wrong_link	0.936	0.148
## 74	ztoinegbin	ztHurdle	0.990	0
## 75	ztoipoisson	Hurdlezt	0.502	0
## 76	ztoipoisson	correct	0.940	0.303
## 77	ztoipoisson	geometric	0	0
## 78	ztoipoisson	oizt	0.850	0
## 79	ztoipoisson	wrong_link	0.938	0.217
## 80	ztoipoisson	ztHurdle	0.016	0

Visual results with confidence intervals:

```
qq1 <- dd |>
  ggplot(aes(x = data_fitted)) +
  facet_wrap(~ data_generation, scales = c("free_x"), ncol = 3) +
  geom_point(aes(y = mean), colour = "navy", cex = 2) +
  geom_errorbar(aes(ymin = lower, ymax = upper), colour = "navy") +
  ggtitle("Empirical coverage of studentized confidence intervals by true distribution of counts") +
  xlab("Fitted distribution") +
  ylab("Coverage")

qq1
```

Empirical coverage of studentized confidence intervals by true distribution



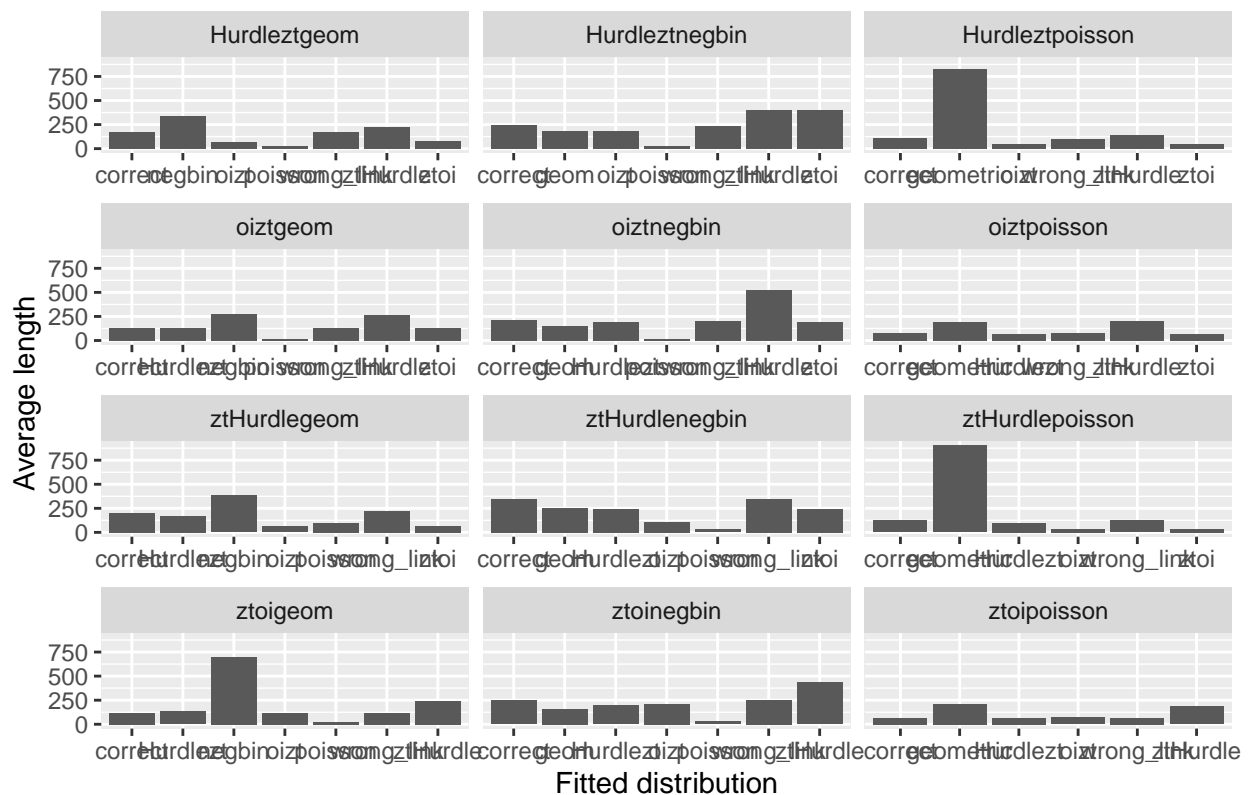
Average sizes of confidence intervals:

```
qq2 <- results_data_frame |>
  subset(!is.na(point)) |>
  subset(point < 25000) |>
  group_by(data_generation, data_fitted) |>
  summarise(len = mean(conf_int_normal_upper - conf_int_normal_lower, na.rm = TRUE)) |>
  ggplot(aes(x = data_fitted, weight = len)) +
  geom_bar() +
  facet_wrap(~ data_generation, scales = c("free_x"), ncol = 3) +
  ylab("Average length") +
  xlab("Fitted distribution") +
  ggtitle("Empirical size of studentized confidence intervals by true distribution of counts")
```

'summarise()' has grouped output by 'data_generation'. You can override using
the '.groups' argument.

```
qq2
```

Empirical size of studentized confidence intervals by true distribution of cou



Logormal

Exact binomial tests for coverage of normal confidence intervals with $H_0 : p = 0.95, H_1 = \neg H_0$:

```
dd <- results_data_frame |>
  subset(!is.na(point)) |>
  subset(point < 25000) |>
  mutate(covr_norm = (conf_int_normal_lower < 1000) & (conf_int_normal_upper > 1000),
         covr_log = (conf_int_log_normal_lower < 1000) & (conf_int_log_normal_upper > 1000)) |>
  group_by(data_generation, data_fitted) |>
  summarise(n = n(),
            mean = mean(covr_log, na.rm = TRUE))
```

'summarise()' has grouped output by 'data_generation'. You can override using
the '.groups' argument.

```
dd <- cbind(dd, p_value = NA, lower = NA, upper = NA)

for (x in 1:NROW(dd)) {
  jj <- binom.test(x = as.numeric(dd[x, 4]) * as.integer(dd[x, 3]), n = as.integer(dd[x, 3]), p = .95)
  # this jj object has some very weird interactions with the rest of R ecosystem
  dd[x, 5] <- jj$p.value |> as.numeric()
  dd[x, 6] <- jj[[4]][1]
  dd[x, 7] <- jj[[4]][2]
}
```

```
print(dd[, c(1:2, 4, 5)] |> mutate(p_value = round(p_value, digits = 4)),
      n = NROW(dd))
```

```
## # A tibble: 80 x 4
## # Groups:   data_generation [12]
##   data_generation data_fitted mean p_value
##   <chr>           <chr>      <dbl> <dbl>
## 1 Hurdleztgeom    correct    0.924 0.0132
## 2 Hurdleztgeom    negbin     0.945 0.602
## 3 Hurdleztgeom    oizt       0      0
## 4 Hurdleztgeom    poisson    0      0
## 5 Hurdleztgeom    wrong_link 0.926 0.0179
## 6 Hurdleztgeom    ztHurdle   0.604 0
## 7 Hurdleztgeom    ztoi       0      0
## 8 Hurdleztnegbin  correct    0.929 0.0373
## 9 Hurdleztnegbin  geom       0      0
## 10 Hurdleztnegbin oizt       0.862 0
## 11 Hurdleztnegbin poisson     0      0
## 12 Hurdleztnegbin wrong_link 0.928 0.0414
## 13 Hurdleztnegbin ztHurdle   0.547 0
## 14 Hurdleztnegbin ztoi       0.547 0
## 15 Hurdleztpoisson correct     0.952 1
## 16 Hurdleztpoisson geometric    0      0
## 17 Hurdleztpoisson oizt       0      0
## 18 Hurdleztpoisson wrong_link 0.948 0.837
## 19 Hurdleztpoisson ztHurdle   0.562 0
## 20 Hurdleztpoisson ztoi       0      0
## 21 oiztgeom        Hurdlezt   0.946 0.681
## 22 oiztgeom        correct     0.9      0
## 23 oiztgeom        negbin     0.832 0
## 24 oiztgeom        poisson    0      0
## 25 oiztgeom        wrong_link 0.894 0
## 26 oiztgeom        ztHurdle   0.0143 0
## 27 oiztgeom        ztoi       0.622 0
## 28 oiztnegbin      Hurdlezt   0.827 0
## 29 oiztnegbin      correct     0.808 0
## 30 oiztnegbin      geom       0      0
## 31 oiztnegbin      poisson    0      0
## 32 oiztnegbin      wrong_link 0.804 0
## 33 oiztnegbin      ztHurdle   0.00211 0
## 34 oiztnegbin      ztoi       0.827 0
## 35 oiztpoisson     Hurdlezt   0.848 0
## 36 oiztpoisson     correct     0.918 0.002
## 37 oiztpoisson     geometric    0      0
## 38 oiztpoisson     wrong_link 0.916 0.0013
## 39 oiztpoisson     ztHurdle   0      0
## 40 oiztpoisson     ztoi       0.675 0
## 41 ztHurdlegeom    Hurdlezt   0.904 0
## 42 ztHurdlegeom    correct     0.958 0.473
## 43 ztHurdlegeom    negbin     0.940 0.382
## 44 ztHurdlegeom    oizt       0      0
## 45 ztHurdlegeom    poisson    0      0
## 46 ztHurdlegeom    wrong_link 0.98    0.0009
```

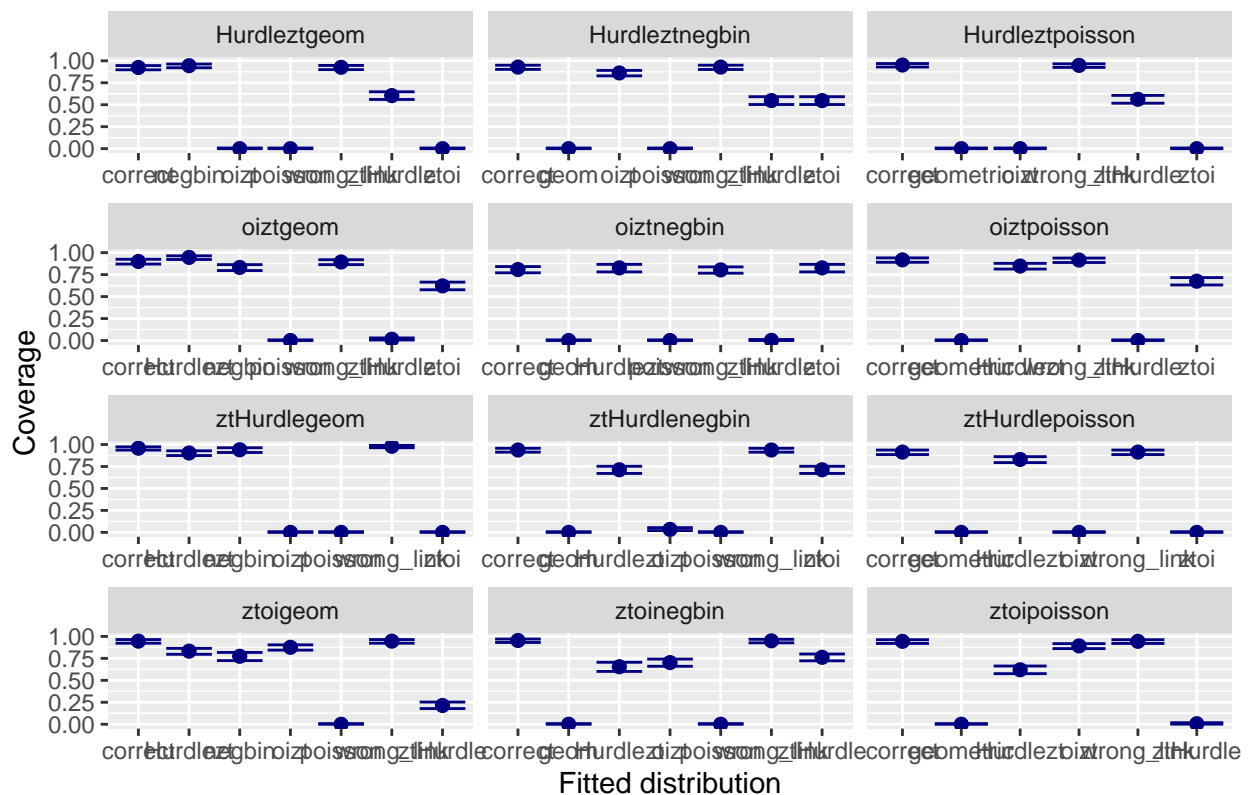
## 47	ztHurdlegeom	ztoi	0	0
## 48	ztHurdlenegbin	Hurdlezt	0.713	0
## 49	ztHurdlenegbin	correct	0.938	0.217
## 50	ztHurdlenegbin	geom	0	0
## 51	ztHurdlenegbin	oizt	0.034	0
## 52	ztHurdlenegbin	poisson	0	0
## 53	ztHurdlenegbin	wrong_link	0.938	0.217
## 54	ztHurdlenegbin	ztoi	0.713	0
## 55	ztHurdlepoisson	Hurdlezt	0.83	0
## 56	ztHurdlepoisson	correct	0.914	0.0006
## 57	ztHurdlepoisson	geometric	0	0
## 58	ztHurdlepoisson	oizt	0	0
## 59	ztHurdlepoisson	wrong_link	0.914	0.0006
## 60	ztHurdlepoisson	ztoi	0	0
## 61	ztoigeom	Hurdlezt	0.832	0
## 62	ztoigeom	correct	0.946	0.681
## 63	ztoigeom	negbin	0.773	0
## 64	ztoigeom	oizt	0.876	0
## 65	ztoigeom	poisson	0	0
## 66	ztoigeom	wrong_link	0.946	0.681
## 67	ztoigeom	ztHurdle	0.214	0
## 68	ztoinegbin	Hurdlezt	0.655	0
## 69	ztoinegbin	correct	0.953	0.836
## 70	ztoinegbin	geom	0	0
## 71	ztoinegbin	oizt	0.702	0
## 72	ztoinegbin	poisson	0	0
## 73	ztoinegbin	wrong_link	0.950	0.918
## 74	ztoinegbin	ztHurdle	0.762	0
## 75	ztoipoisson	Hurdlezt	0.62	0
## 76	ztoipoisson	correct	0.944	0.537
## 77	ztoipoisson	geometric	0	0
## 78	ztoipoisson	oizt	0.892	0
## 79	ztoipoisson	wrong_link	0.944	0.537
## 80	ztoipoisson	ztHurdle	0.006	0

Visual results with confidence intervals:

```
qq3 <- dd |>
  ggplot(aes(x = data_fitted)) +
  facet_wrap(~ data_generation, scales = c("free_x"), ncol = 3) +
  geom_point(aes(y = mean), colour = "navy", cex = 2) +
  geom_errorbar(aes(ymin = lower, ymax = upper), colour = "navy") +
  ggtitle("Empirical coverage of log normal confidence intervals by true distribution of counts") +
  xlab("Fitted distribution") +
  ylab("Coverage")

qq3
```

Empirical coverage of log normal confidence intervals by true distribution o



Average sizes of confidence intervals:

```
qq4 <- results_data_frame |>
  subset(!is.na(point)) |>
  subset(point < 25000) |>
  group_by(data_generation, data_fitted) |>
  summarise(len = mean(conf_int_log_normal_upper - conf_int_log_normal_lower, na.rm = TRUE)) |>
  ggplot(aes(x = data_fitted, weight = len)) +
  geom_bar() +
  facet_wrap(~ data_generation, scales = "free", ncol = 3) +
  ylab("Average length") +
  xlab("Fitted distribution") +
  ggtitle("Empirical size of log normal confidence intervals by true distribution of counts")
```

'summarise()' has grouped output by 'data_generation'. You can override using
the '.groups' argument.

```
qq4
```

Empirical size of log normal confidence intervals by true distribution of cou

