Diagnostics and Overdispersion Parameter

William Ruth

30/11/2021

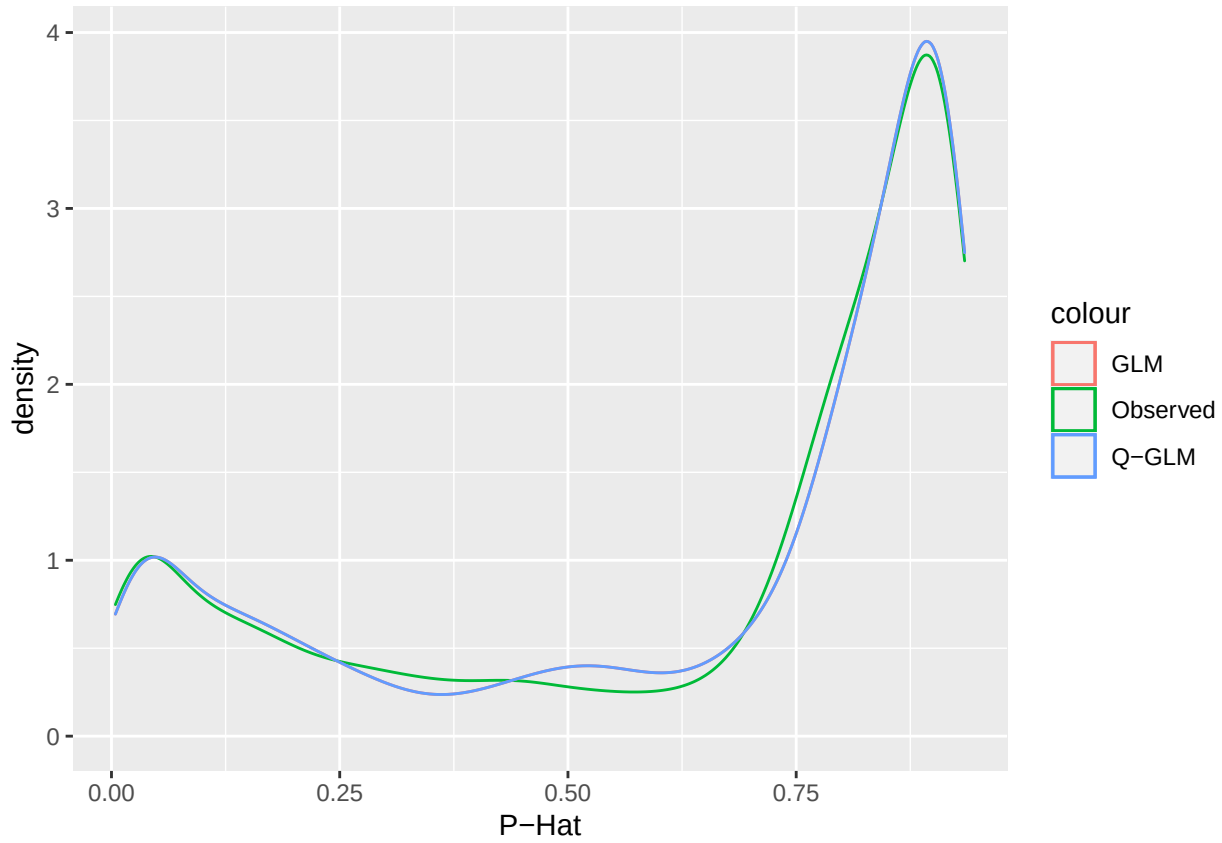
Model Diagnostics

We fit logistic regression models to predict the CII in our simulations using class size threshold and various epidemic parameters. We use both ordinary logistic regression and the overdispersed quasi-binomial model. The purpose of this document is to investigate model diagnostics. We also estimate the overdispersion parameter as described in McCullagh and Nelder to compare with the value given in the quasi-binomial GLM fit.

Density Comparisons I - Observed and Fitted Values

First, we plot the density of the observed and fitted proportions. Note that the quasibinomial model gives the same fitted values as the ordinary logistic fit.

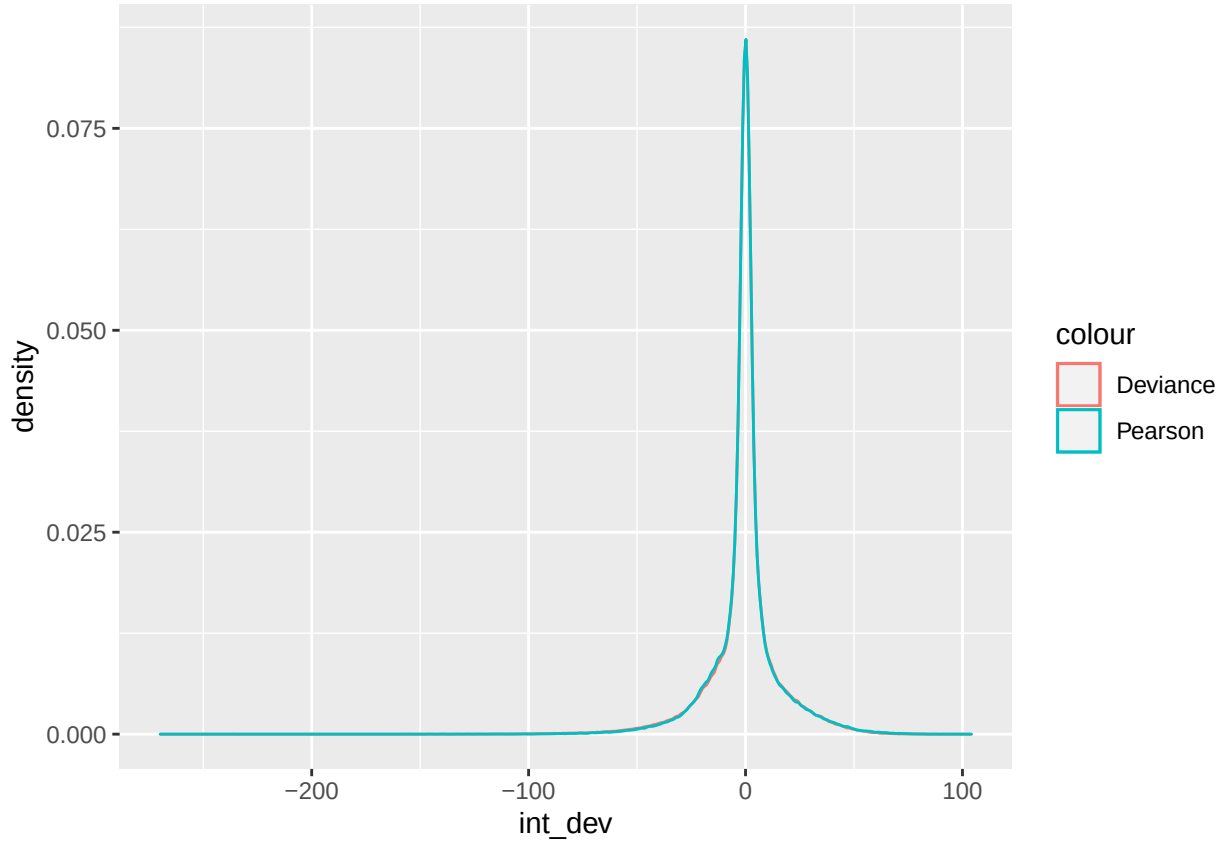
```
########################################
### Compare observed p_hat with GLM fit ###
########################################
hist_all = ggplot(all_SDs_obs) +
  geom_density(aes(x = p_hat_obs, color = "Observed")) +
  geom_density(aes(x = p_hat_int, color = "GLM")) +
  geom_density(aes(x = p_hat_disp, color = "Q-GLM")) +
  xlab("P-Hat")
plot(hist_all)
```



Density Comparisons II - Deviance and Pearson Residuals

Next, we plot densities of the two types of residuals (i.e. deviance and Pearson).

```
##############################################
### Investigate distribution of residuals ###
##############################################

resid_int_dev <- residuals(fit_glm_int, "deviance")
resid_int_pea <- residuals(fit_glm_int, "pearson")

resid_disp_dev <- residuals(fit_glm_int_disper, "deviance")
resid_disp_pea <- residuals(fit_glm_int_disper, "pearson")



data_residuals <- data.frame(int_dev = resid_int_dev,
  int_pea = resid_int_pea, disp_dev = resid_disp_dev,
  disp_pea = resid_disp_pea, threshold = data$threshold)


density_residuals <- ggplot(data_residuals) +
  geom_density(aes(x = int_dev, color = "Deviance")) +
  geom_density(aes(x = int_pea, color = "Pearson"))
plot(density_residuals)
```

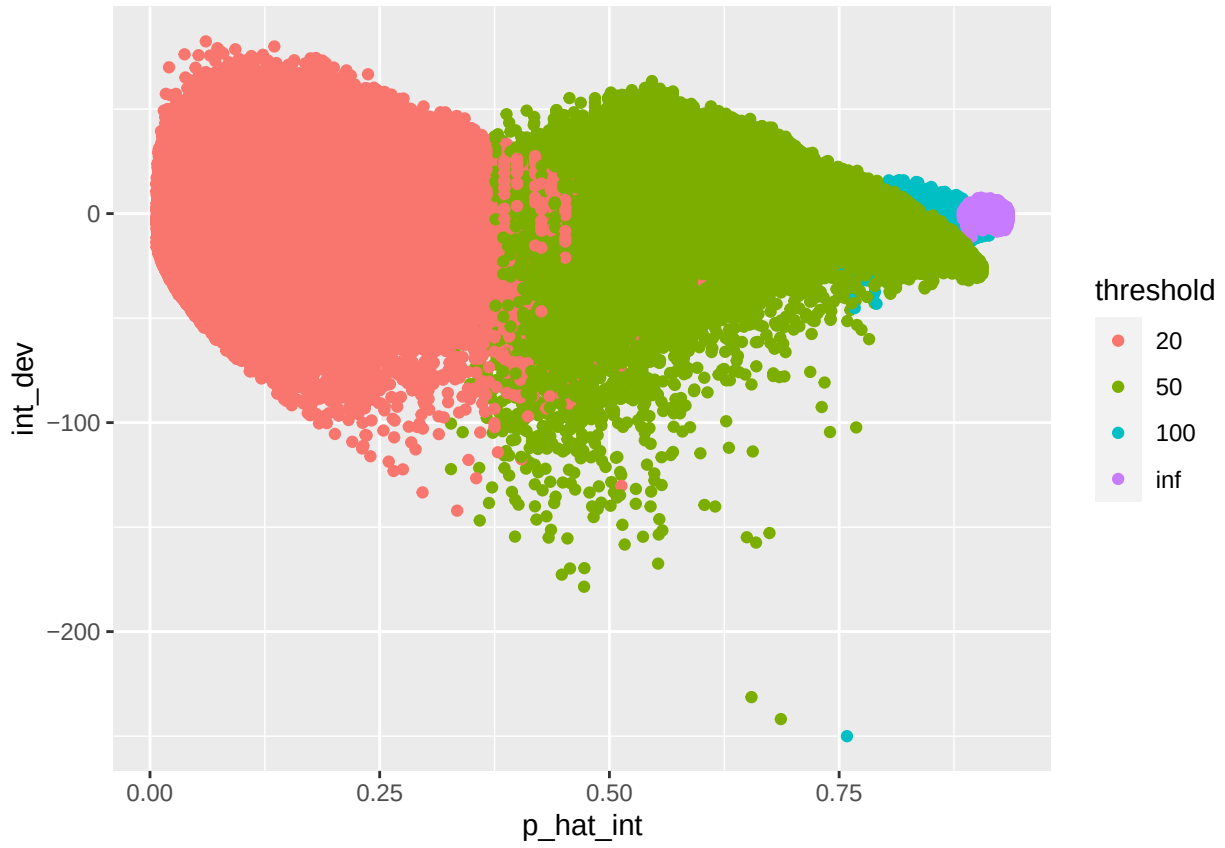



```
# summaries <- map(data_residuals, summary)
```

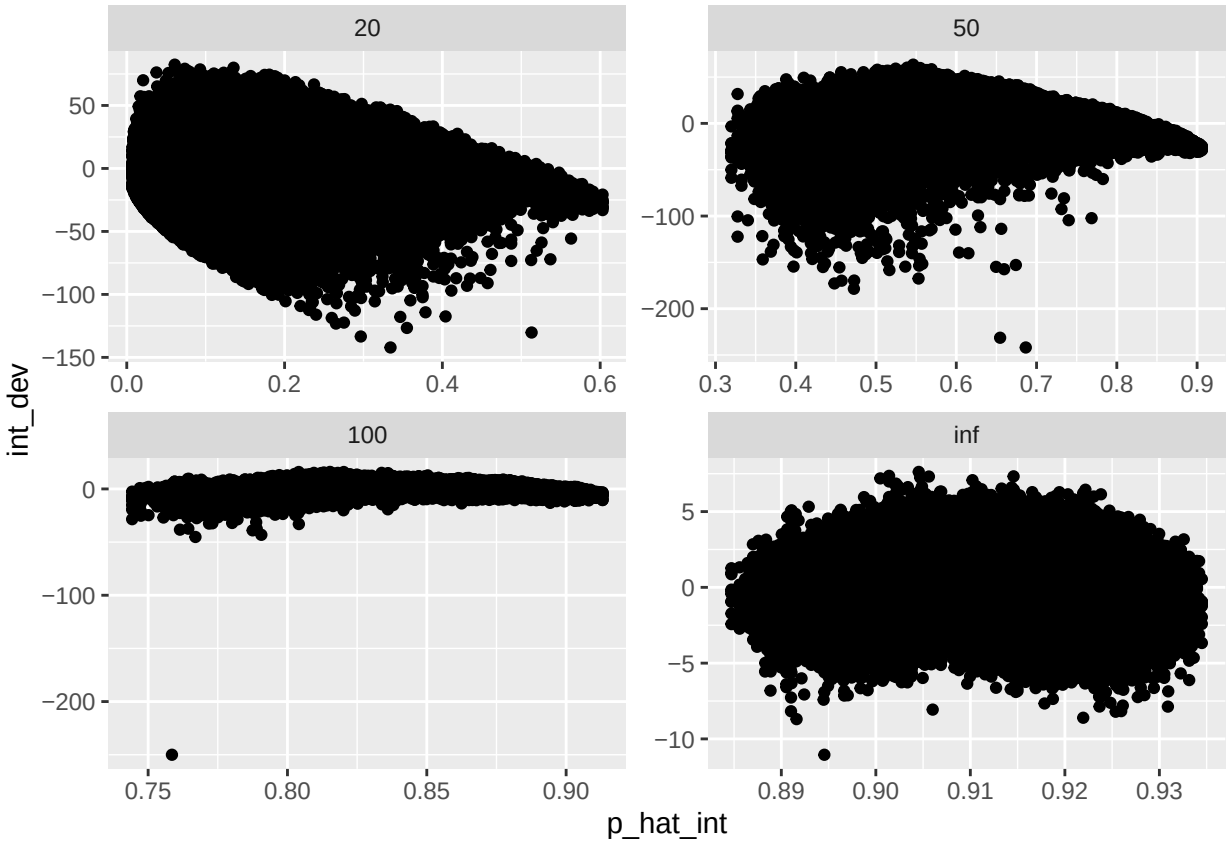
Residuals vs Fitted Values

Now we make the classic diagnostic plot of residual vs fitted value.

```
#######################################
### Plot residuals vs fitted values ###
#######################################

## Add fitted values to residual dataset
data_residuals$p_hat_int <- p_hat_int


plot_res_dev <- ggplot(data_residuals,
  aes(x = p_hat_int, y = int_dev, color = threshold)) +
  geom_point()
plot(plot_res_dev)
```




```
facet_res_dev <- ggplot(data_residuals,
  aes(x = p_hat_int, y = int_dev)) +
  geom_point() + facet_wrap(~threshold, scales = "free")
plot(facet_res_dev)
```

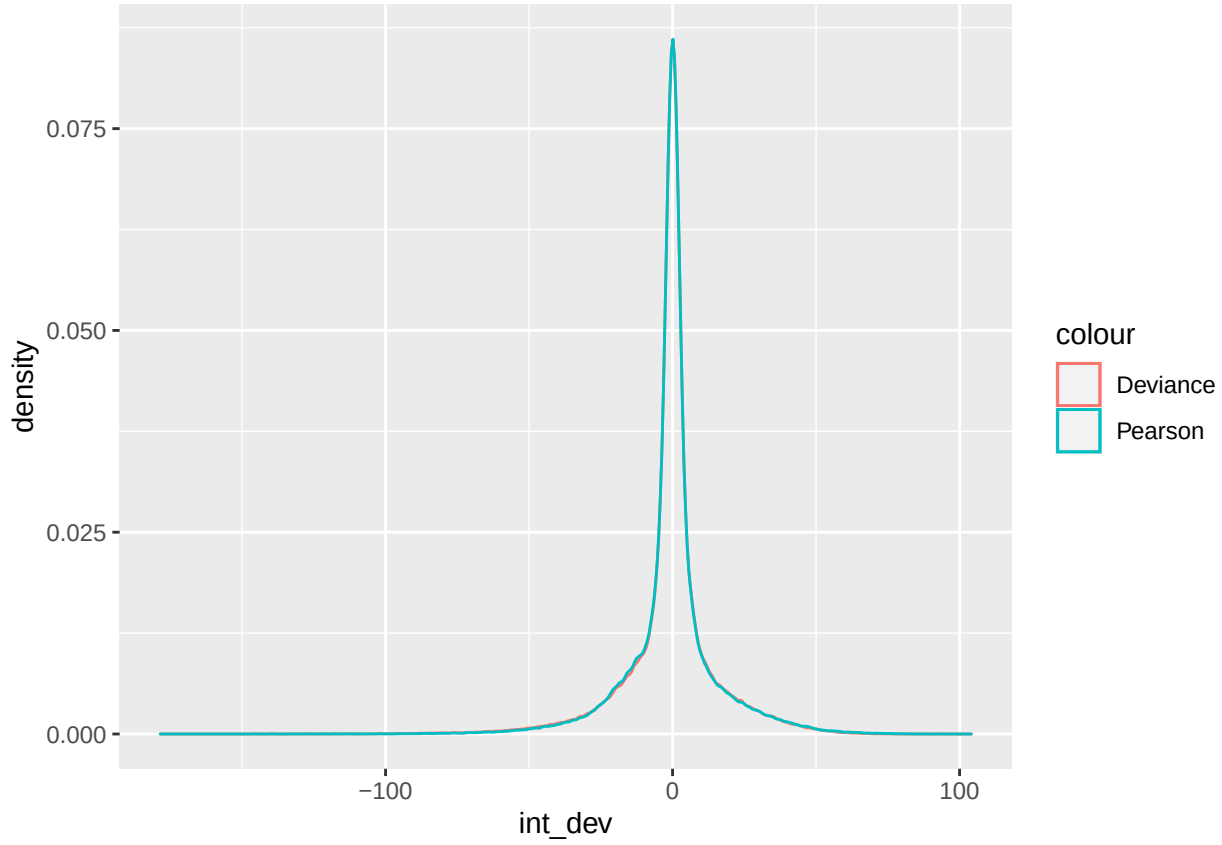


Redux - Removed Outliers

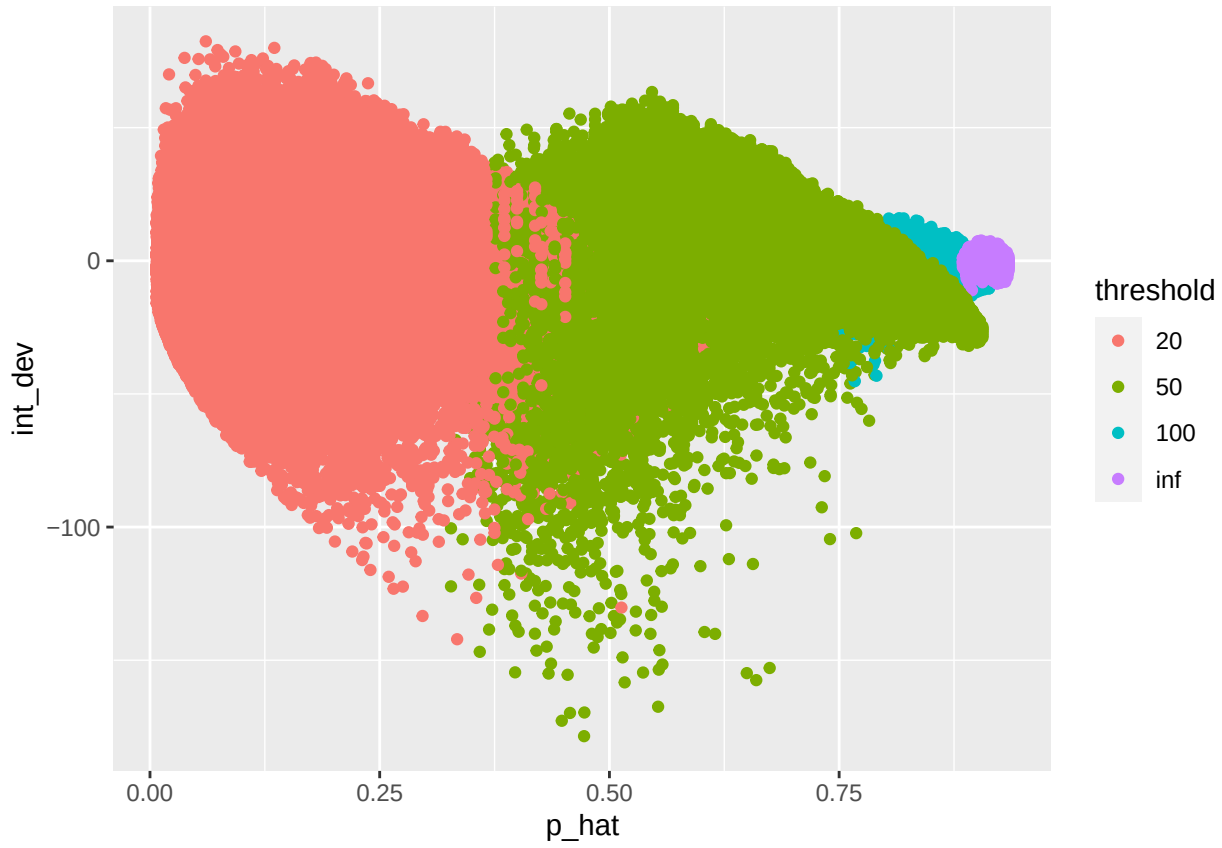
There are a few extreme outliers, so we remove these points, refit our models and reproduce the above plots.

```
### Remove points with extreme residuals (i.e. < -200), then refit models
inds_remove <- which(data_residuals$int_dev < -200)
data_clean <- data_logit[-inds_remove,]
N_clean = num_trials - length(inds_remove)

fit_clean_int <- glm(form, family = binomial(),
  data = data_clean, weights = rep(num_students, times = N_clean))
fit_clean_disper <- glm(form, family = quasibinomial(),
  data = data_clean, weights = rep(num_students, times = N_clean))


### Build residual dataset ###

## Extract residuals
resid_int_dev_clean <- residuals(fit_clean_int, "deviance")
resid_int_pea_clean <- residuals(fit_clean_int, "pearson")

resid_disp_dev_clean <- residuals(fit_clean_disper, "deviance")
resid_disp_pea_clean <- residuals(fit_clean_disper, "pearson")


## Extract fitted values
```

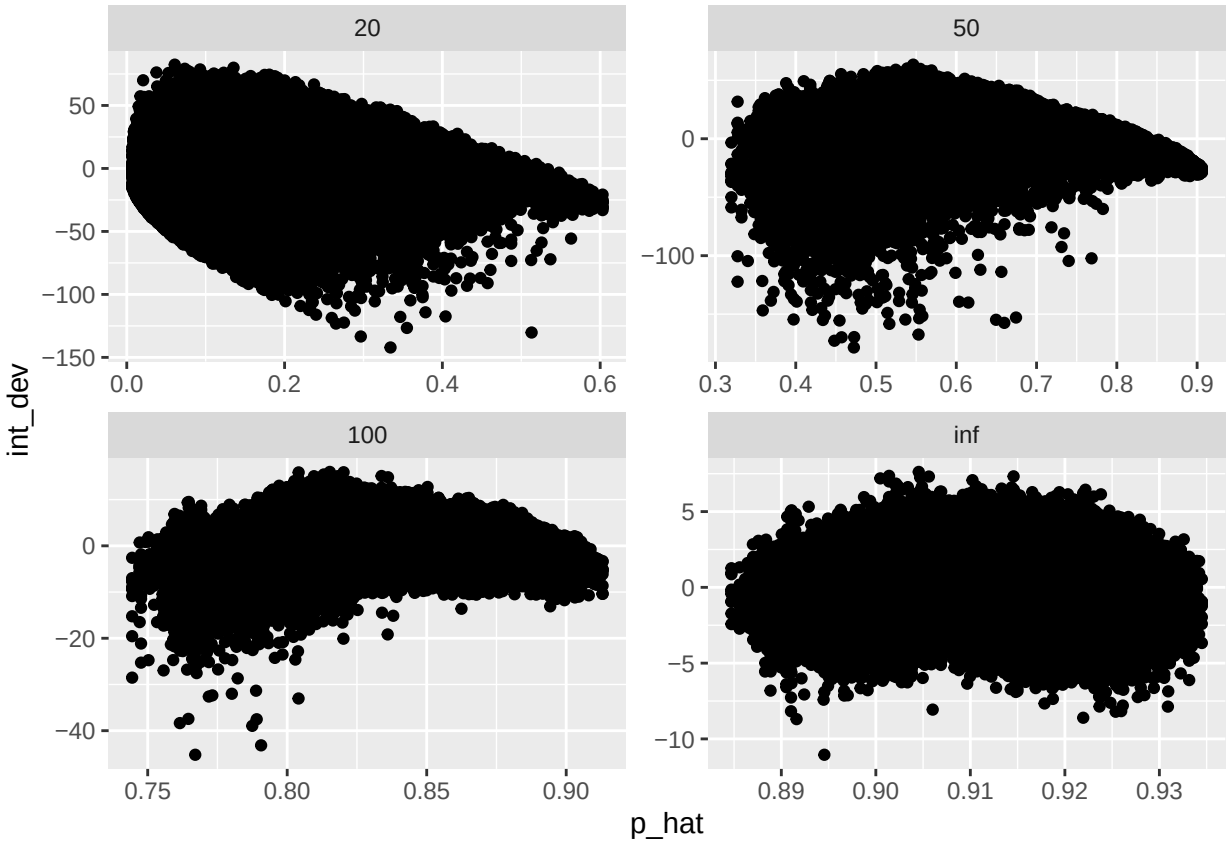
```
p_hat_int_clean <- predict(fit_clean_int, type = "response")


## Build data frame
data_res_clean <- data.frame(int_dev = resid_int_dev_clean,
  int_pea = resid_int_pea_clean, disp_dev = resid_disp_dev_clean,
  disp_pea = resid_disp_pea_clean, threshold = data_clean$threshold,
  p_hat = p_hat_int_clean)


### Make plots with clean dataset ###
plot_res_dens_clean <- ggplot(data_res_clean) +
  geom_density(aes(x = int_dev, color = "Deviance")) +
  geom_density(aes(x = int_pea, color = "Pearson"))
plot(plot_res_dens_clean)
```




```
plot_res_dev_clean <- ggplot(data_res_clean,
  aes(x = p_hat, y = int_dev, color = threshold)) +
  geom_point()
plot(plot_res_dev_clean)
```



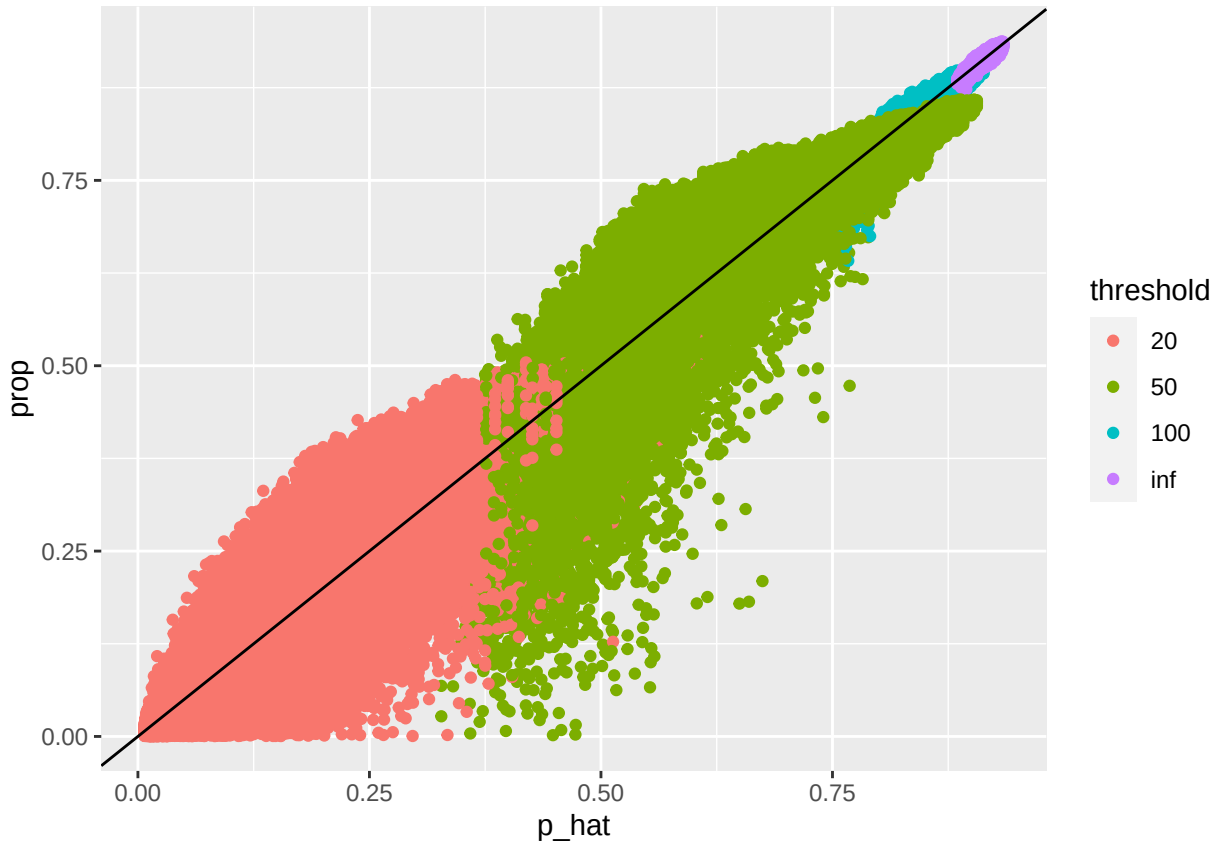

```
facet_res_dev_clean <- ggplot(data_res_clean,
  aes(x = p_hat, y = int_dev)) +
  geom_point() + facet_wrap(~threshold, scales = "free")
plot(facet_res_dev_clean)
```



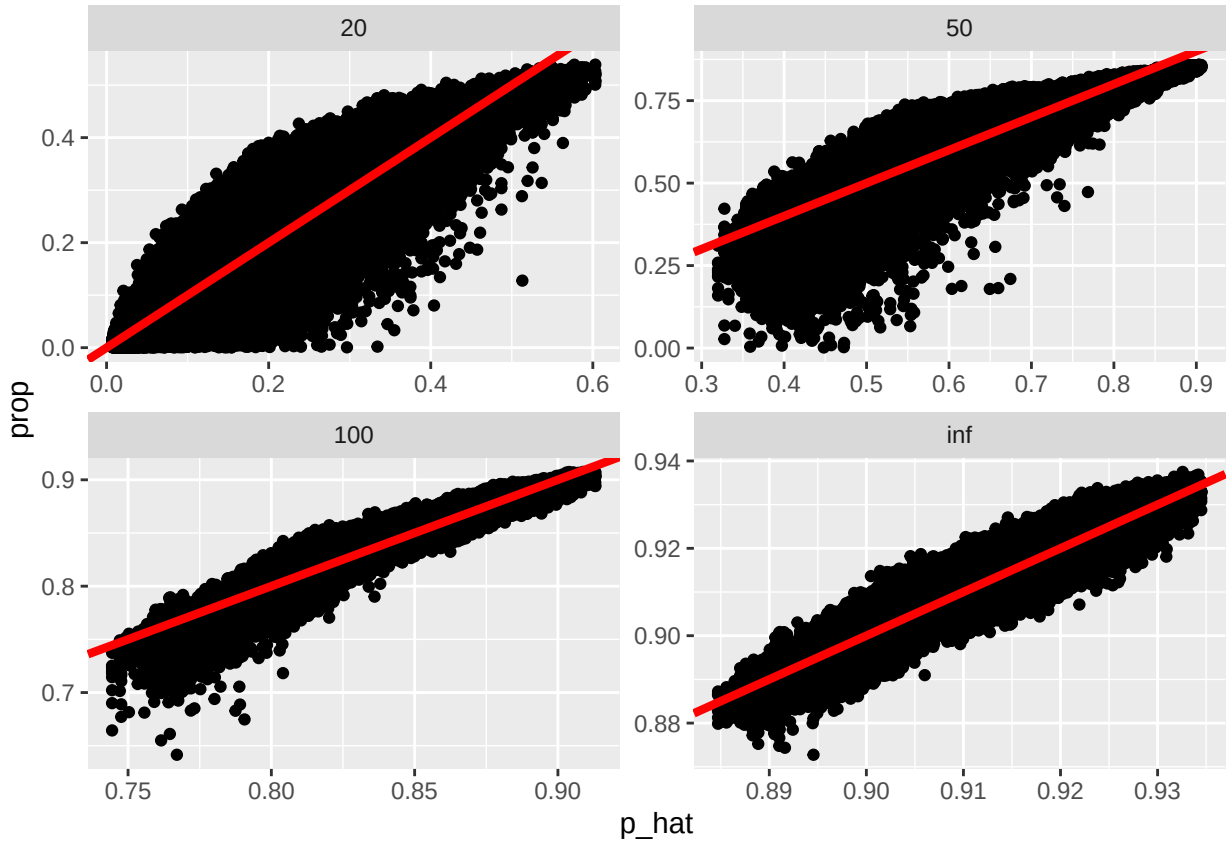
Response vs Fitted Values

Finally, we plot the response vs the fitted values.

```
## Plot response vs fitted value
data_res_clean$prop <- data_clean$prop

plot_obs_vs_fit_clean <- ggplot(data_res_clean,
  aes(x = p_hat, y = prop, color = threshold)) +
  geom_point() + geom_abline(slope = 1, intercept = 0)
plot(plot_obs_vs_fit_clean)
```




```
facet_obs_vs_fit_clean <- ggplot(data_res_clean,
  aes(x = p_hat, y = prop)) + geom_point() +
  geom_abline(slope = 1, intercept = 0, color = "red", lwd = 1.5) +
  facet_wrap(~threshold, scales = "free")
plot(facet_obs_vs_fit_clean)
```

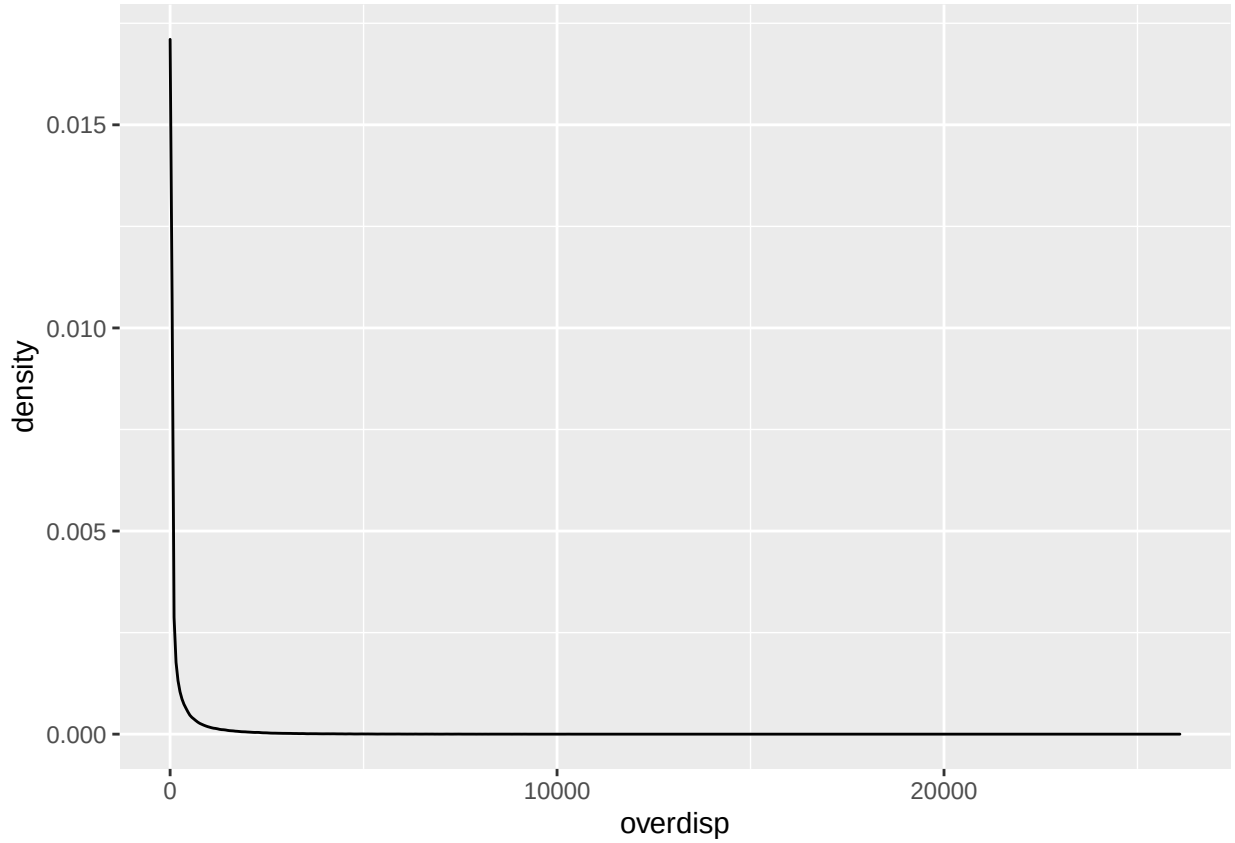


This concludes our investigation of model diagnostics.

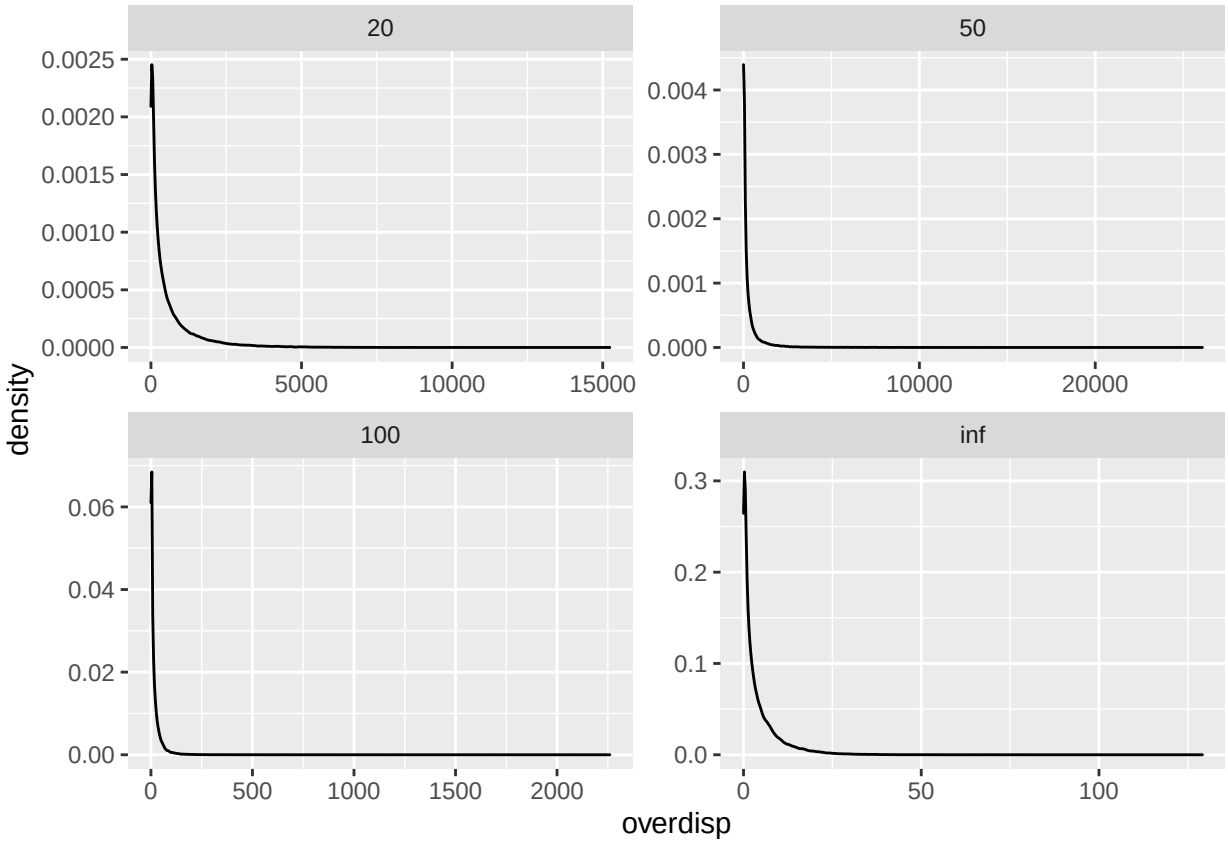
Overdispersion Parameter Investigation

Next, we explore the overdispersion parameter, ϕ. The glm function gives an estimate of 237.622971, which is much larger than I expected based on my investigation of the cell-wise observed standard deviations (in another write-up).

```
### Compute the estimate of the overdispersion parameter manually
data_res_clean %<>% mutate(overdisp =
    (prop - p_hat)^2 / (p_hat * (1 - p_hat)/num_students))

plot_overdisp <- ggplot(data_res_clean, aes(x = overdisp)) +
  geom_density()
plot(plot_overdisp)
```




```
facet_overdisp <- plot_overdisp + facet_wrap(~threshold, scales = "free")
plot(facet_overdisp)
```




```
N_pars <- length(fit_clean_int$coef)

phi_hat_global <- sum(data_res_clean$overdisp) / (N_clean - N_pars)
phi_hat_global
```

```
## [1] 237.6195
```

```
phi_hat_groups <- data_res_clean %>%
  group_by(threshold) %>%
  mutate(overdisp = (prop - p_hat)^2 / (p_hat * (1 - p_hat)/num_students)) %>%
  summarise(phi_hat = mean(overdisp),.groups = "drop")
phi_hat_groups
```

```
## # A tibble: 4 x 2
##   threshold phi_hat
##   <fct>       <dbl>
## 1 20          564.
## 2 50          365.
## 3 100          17.7
## 4 inf           3.97
```

Note that both the global and group-wise overdispersion parameters are wildly different from the value computed by the glm function.