Choosing an appropriate scalar transformation to normalise wpd

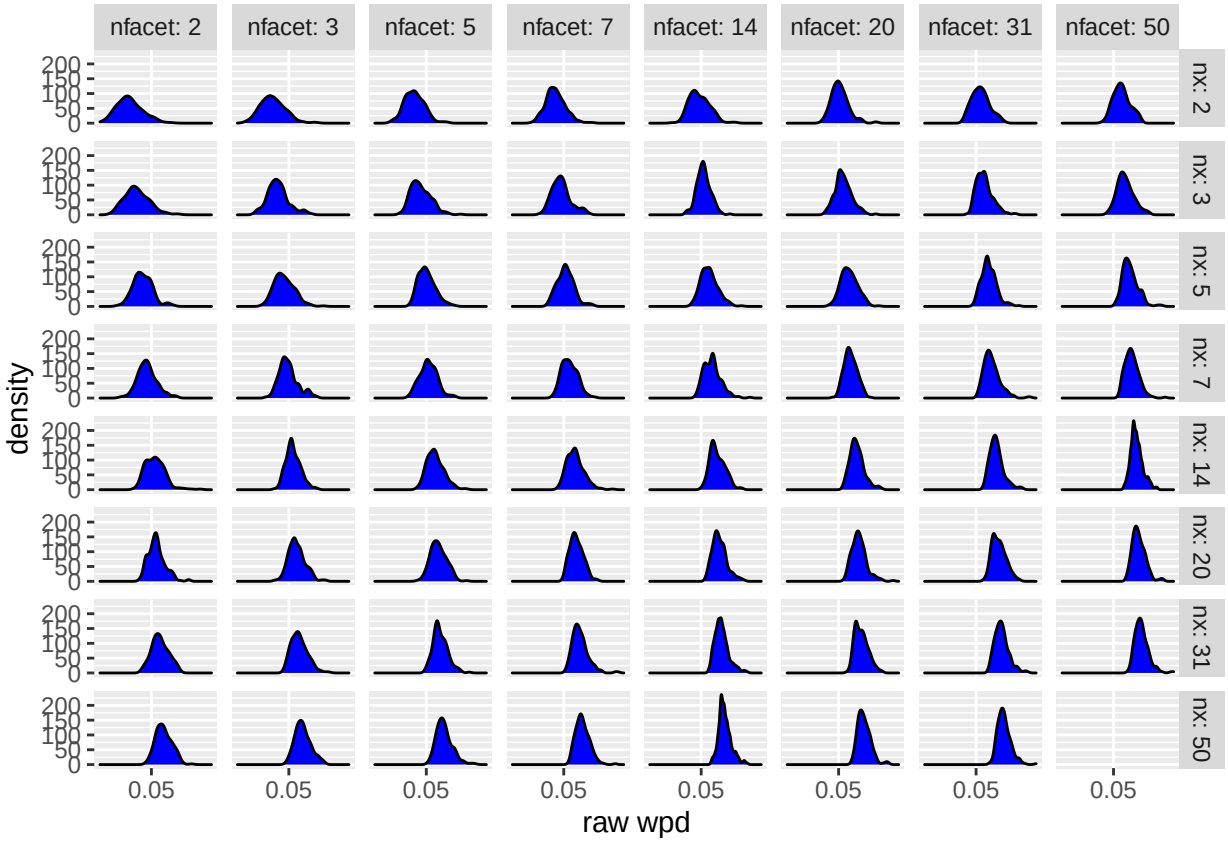
Sayani Gupta

28/01/2021

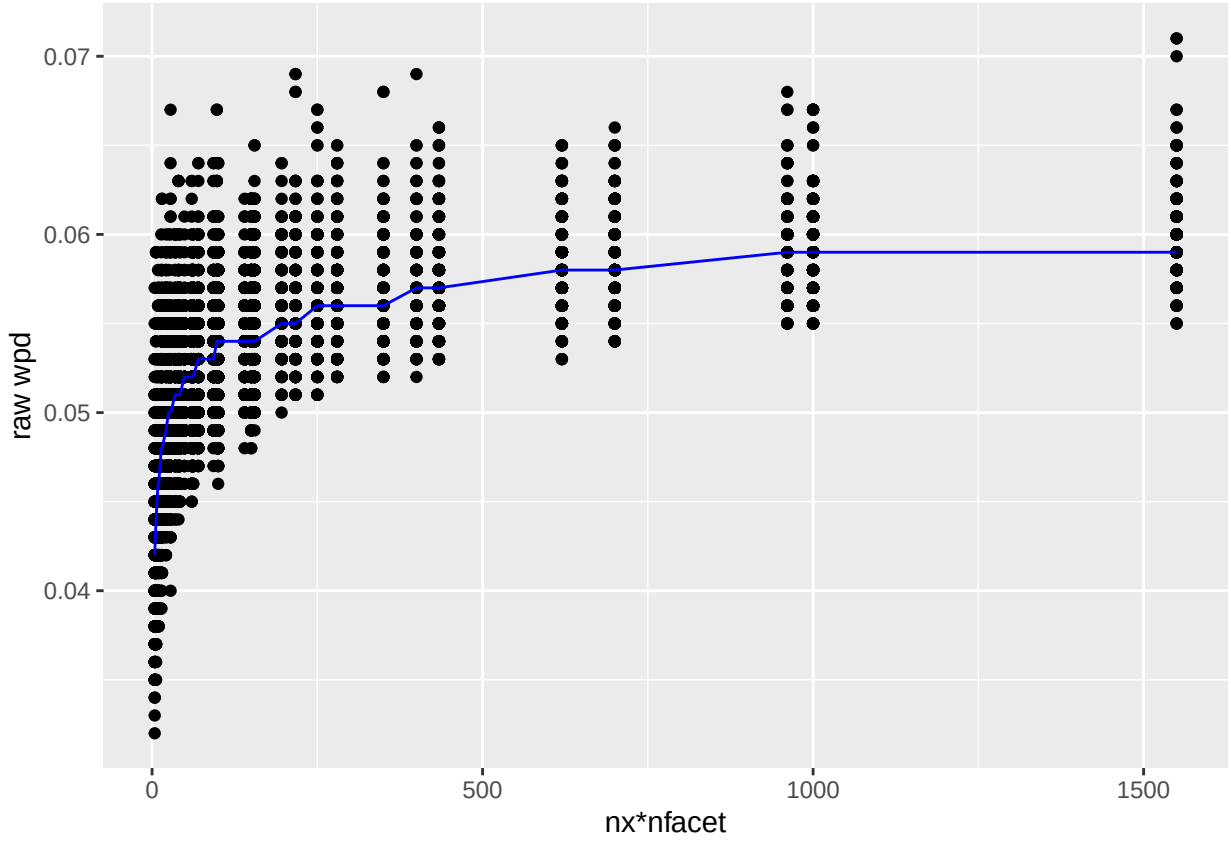
- Data presented

Observations are generated from a N(0,1) distribution for each combination of nx and $nfacet$ from the following sets: $nx = nfacet = \{2, 3, 5, 7, 14, 20, 31, 50\}$ to cover a wide range of levels from very low to moderately high. Each combination is being referred to as a *panel*. That is, data is being generated for each of the panels $\{nx = 2, nfacet = 2\}, \{nx = 2, nfacet = 3\}, \{nx = 2, nfacet = 5\}, \ldots, \{nx = 50, nfacet = 31\}, \{nx = 50, nfacet = 50\}$. For each of the 64 panels, $ntimes = 500$ observations are drawn for each combination of the categories. That is, if we consider the panel $\{nx = 2, nfacet = 2\}$, 500 observations are generated for each of the combination of categories from the panel, namely, $\{(1, 1), (1, 2), (2, 1), (2, 2)\}$. The values of λ is set to 0.67 and values of raw wpd is obtained.

- How the distribution of the raw wpd (without any transformation for normalization) looks across nfacets and nx? Both shape and scale of the distribution changes for different nx and nfacet categories.



- Plot the values of raw wpd against nx*nfacet to see the rough relationship



Current attempt

- Fit a log-linear relationship of median(values) to log(nx*nfacet) as discussed today. The fit is pretty good.

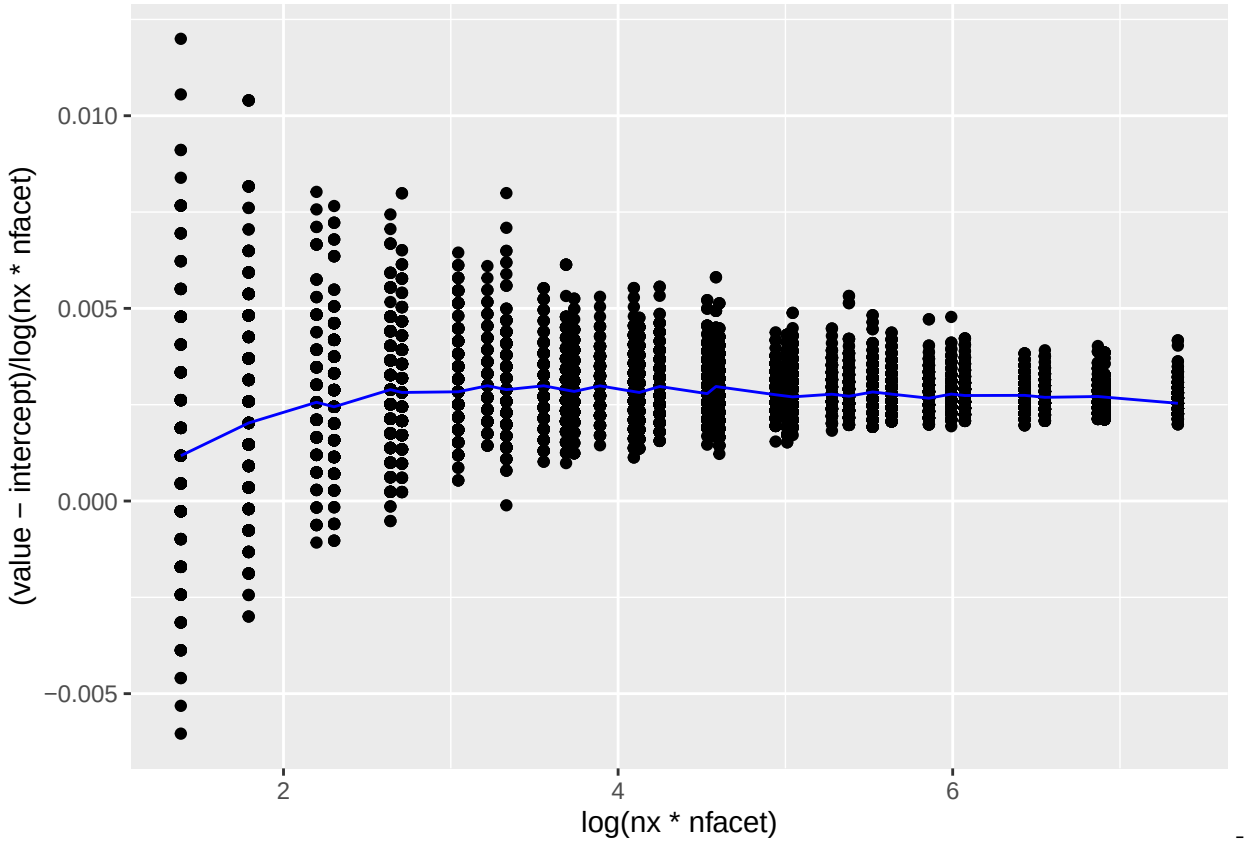
```
N01_median <- N01 %>%
  group_by(nx*nfacet) %>%
  summarise(actual = median(value))

fit_lm2 <- lm(actual ~ poly(log(`nx * nfacet`) ,1, raw=TRUE), data = N01_median)
summary(fit_lm2)
```

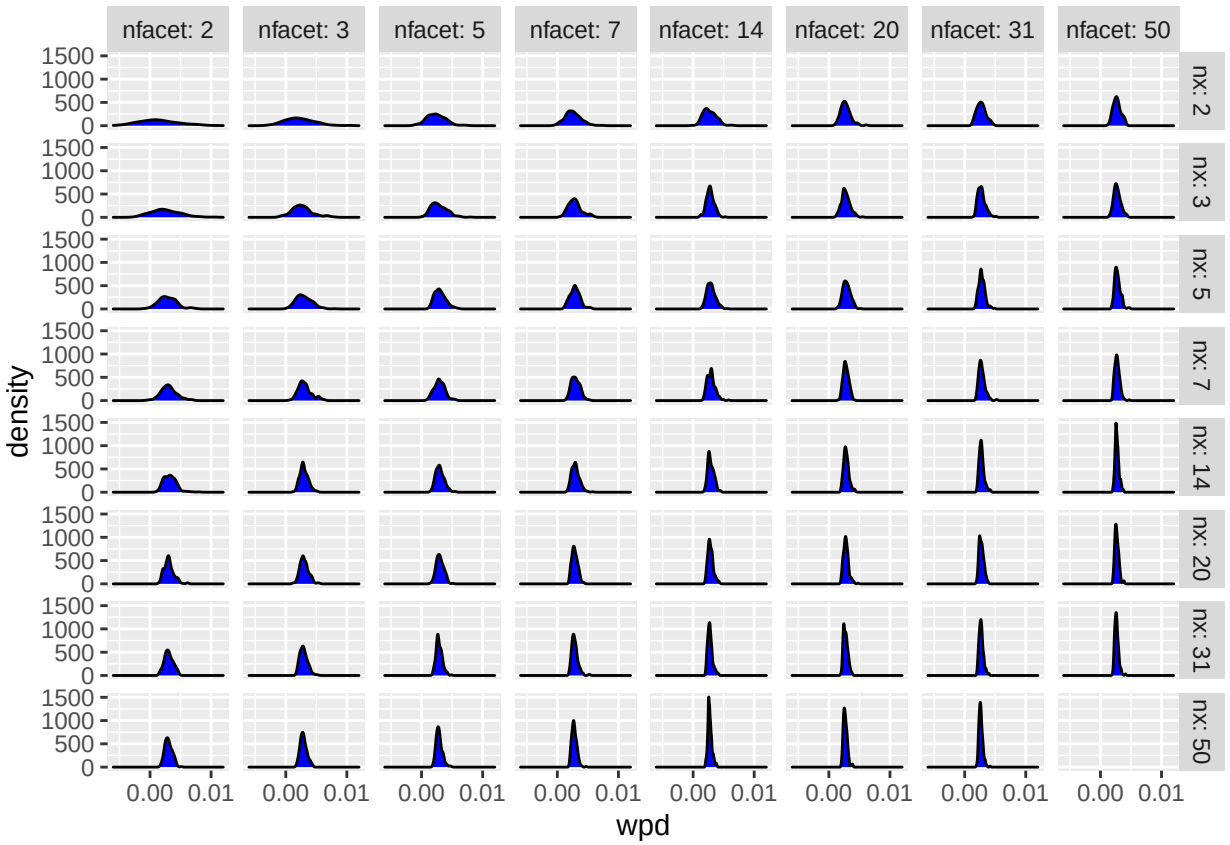
```
##
## Call:
## lm(formula = actual ~ poly(log(`nx * nfacet`), 1, raw = TRUE),
##     data = N01_median)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0021910 -0.0002941  0.0001063  0.0003966  0.0009917
##
## Coefficients:
##                                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)                           4.037e-02  3.814e-04  105.85   <2e-16
```

```
## poly(log(`nx * nfacet`), 1, raw = TRUE) 2.757e-03  8.043e-05   34.27   <2e-16
##
## (Intercept)                              ***
## poly(log(`nx * nfacet`), 1, raw = TRUE) ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0007206 on 32 degrees of freedom
## Multiple R-squared:  0.9735, Adjusted R-squared:  0.9727
## F-statistic:  1175 on 1 and 32 DF,  p-value: < 2.2e-16
```

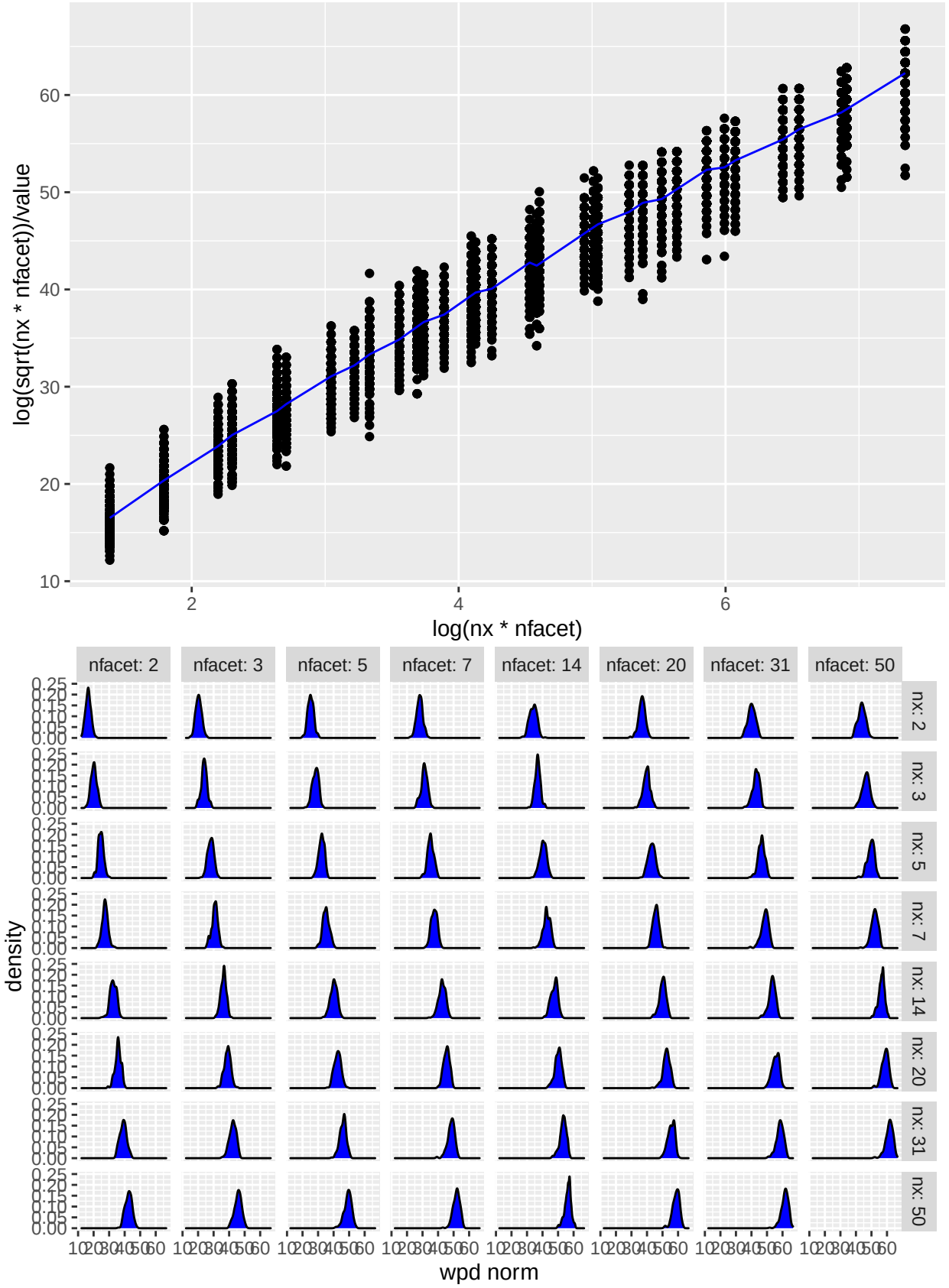
- Make it a horizontal line so that the values are not affected by nx*nfacet by plotting (value - intercept)/log(nx*nfacet) against log(nx*nfacet). The line actually becomes horizontal after this transformation.



See distribution of the transformed variable across nx and nfacets. For higher values of nx and nfacet, the distribution is similar, but not for lower values.

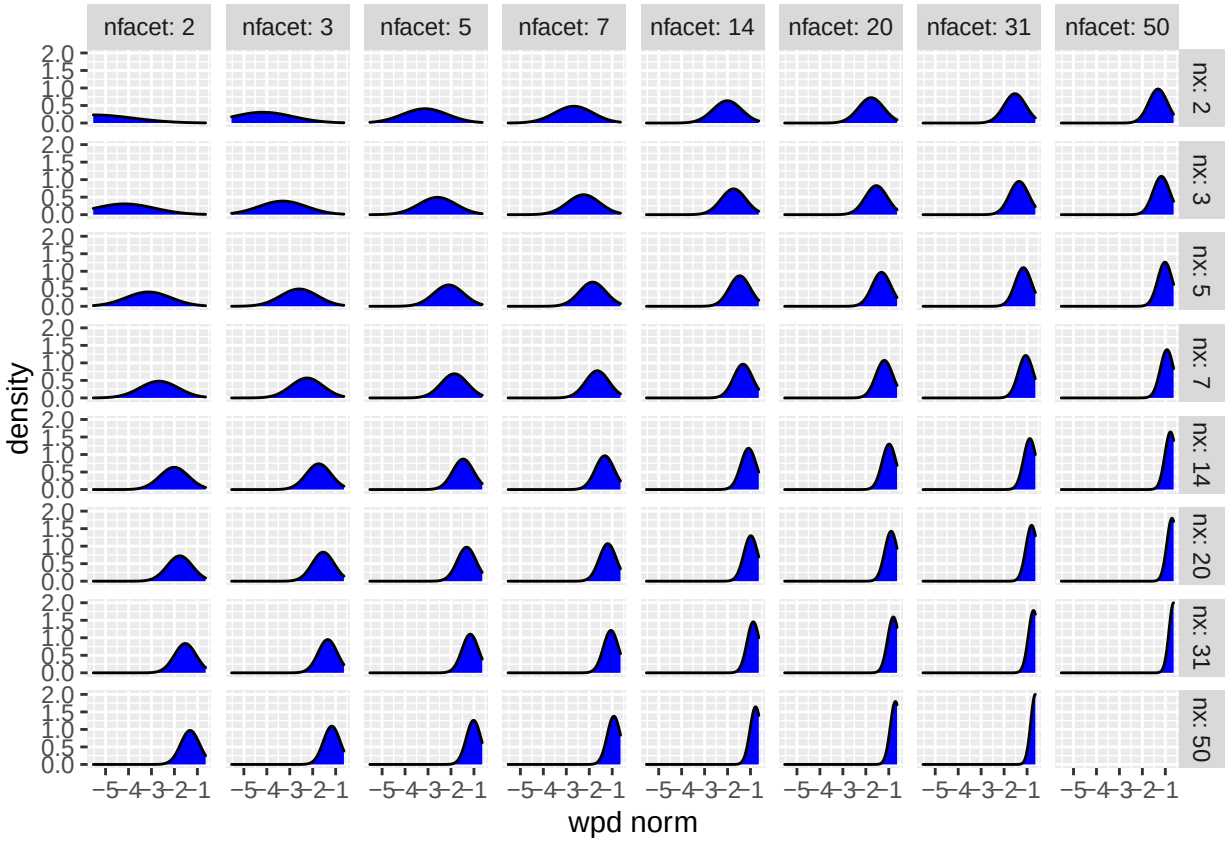


Earlier attempt - The distribution of $wpd_{norm} = log(sqrt(sqrt(nx * nfacet)))/wpd$ is plotted since the plot of log(sqrt(sqrt(nx * nfacet))) against wpd looked linear. The shape and spread look similar but location is shifting to the right. (If we define wpd_{norm} as the inverse of it, the values become too small and the distribution too skewed. Hence the inverse of it is considered.)



What we want is a transformation which will be constant and not linear to obtain similar locations for all

panels.


nfacet: 2
nfacet: 3
nfacet: 5
nfacet: 7
nfacet: 14
nfacet: 20
nfacet: 31
nfacet: 50
nx: 2
nx: 3
nx: 5
nx: 7
nx: 14
nx: 20
nx: 31
nx: 50
density
wpd norm
0.0
0.5
1.0
1.5
2.0
−5
−4
−3
−2
−1