HW-6

AUTHOR
Damini Vadrevu

1 (a)

```
data <- read.csv("C:/Users/vadre/Downloads/deforest.csv", header=TRUE)
```

```
library(ggplot2)
library(dplyr)
```

```
Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```
library(car)
```

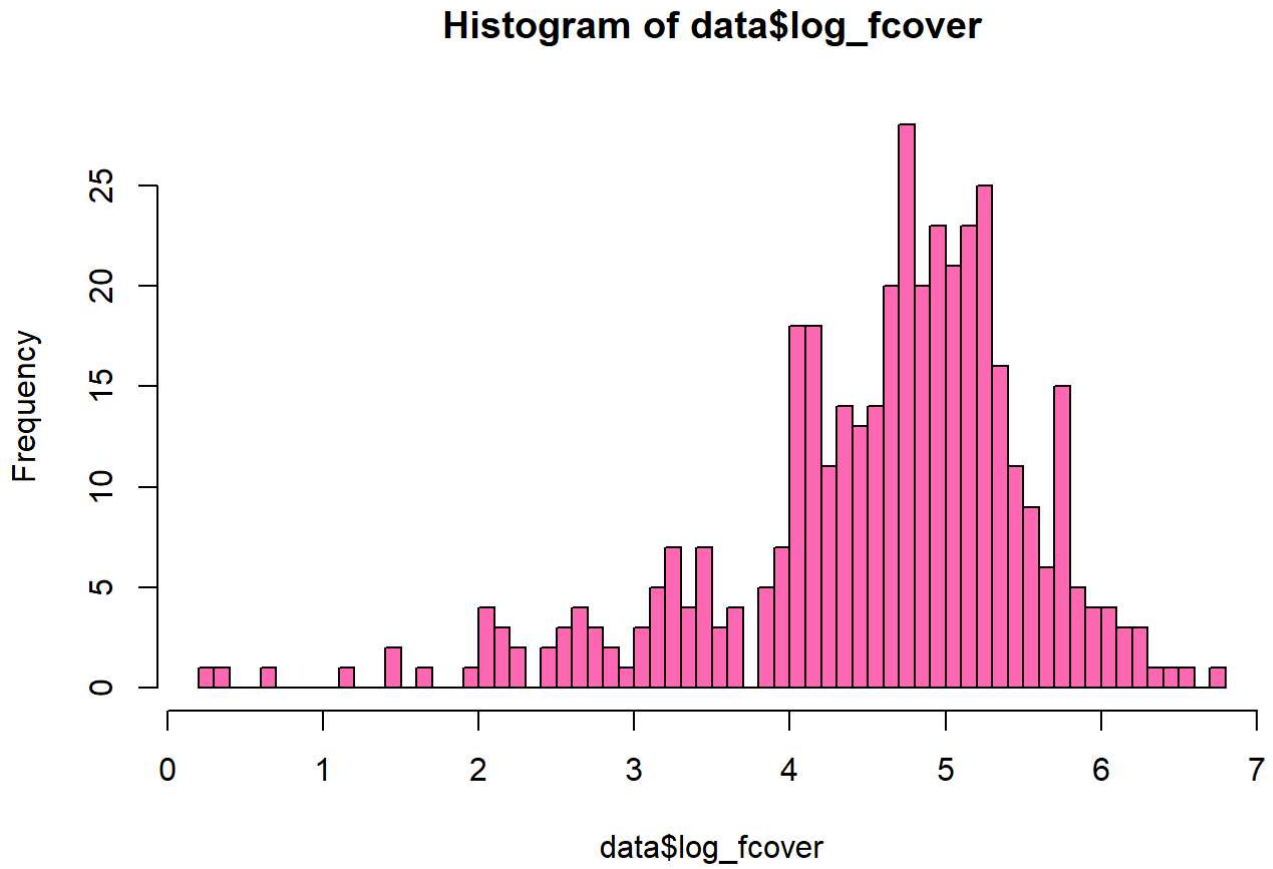
```
Loading required package: carData


Attaching package: 'car'

The following object is masked from 'package:dplyr':

    recode
```

```
head(data)
```

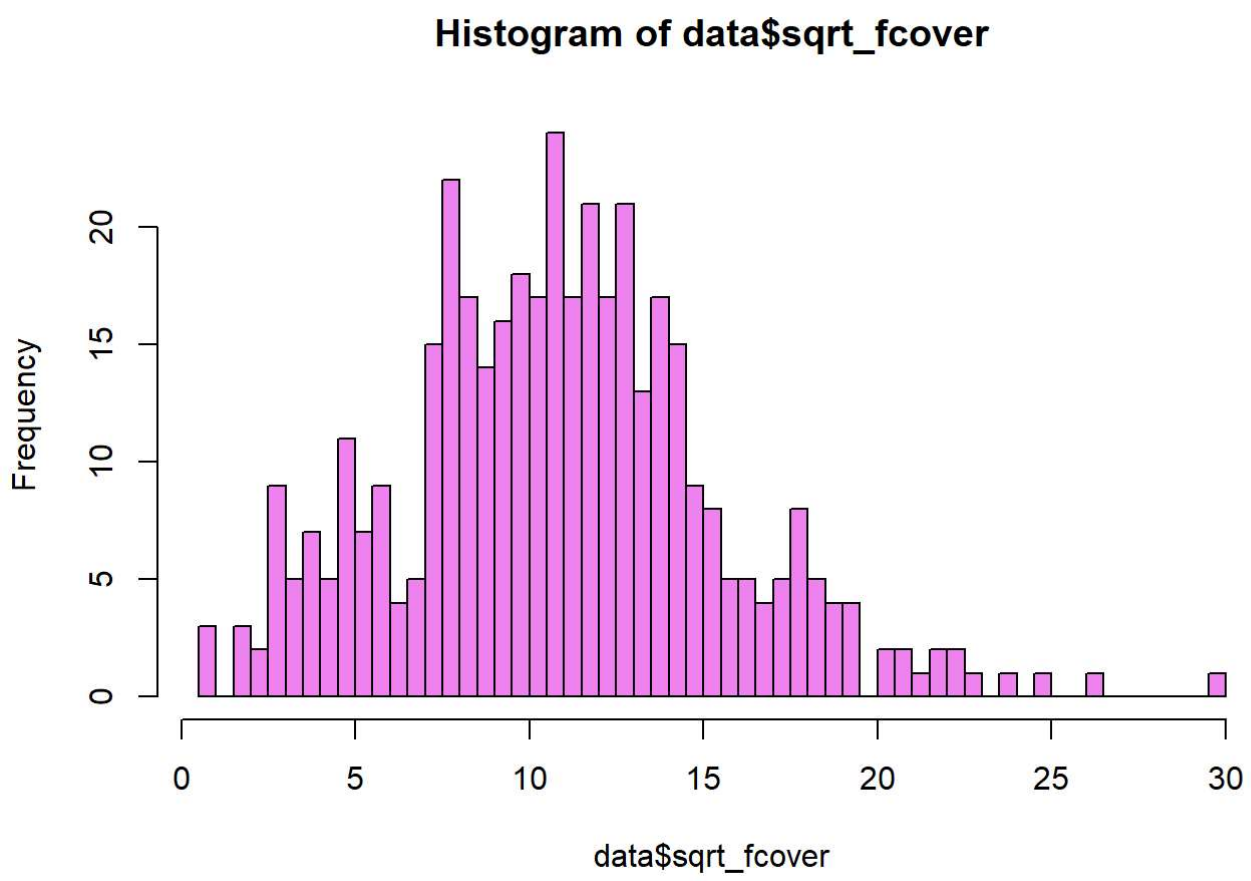
```
  id   code        ename     fcover tpop apop     gdp1     gdp2     gdp3
1  1 360121 nanchangxian   3.101124   93   74 2820.376 9743.925 3898.430
2  2 360122      xinjian  39.285467   67   59 2819.388 6300.552 5167.343
3  3 360123         anyi  23.665299   26   19 2101.154 3967.923 4868.308
4  4 360124      jinxian  38.658192   77   65 2386.662 6313.792 3502.143
5  5 360222      fuliang 265.250505   27   23 2199.148 3453.926 2632.704
6  6 360281       leping 100.085463   82   66 1416.134 4476.183 3060.061
    distpvc distport   hwaylen    explen    plain elev prec slope temp  temp0
1  21260.86 435763.2  28749.55 21419.860 0.922216   19 1569  0.07 17.5 6384.1
2  33245.34 453378.0  13618.60  2173.066 0.666383   30 1521  0.33 17.2 6290.0
3  37650.58 489738.7  26491.12  8121.573 0.379204   70 1571  1.49 16.8 6136.3
```

```
4  46479.27 407609.2  54049.56 16306.070 0.512833   28 1619  0.28 17.5 6423.5
5 168570.30 366528.5 129787.80 11673.830 0.087788  190 1758  2.91 16.0 5856.9
6 134398.80 350229.9 136160.30 13042.650 0.354071   76 1758  1.17 17.2 6293.6
    area policy
1 188635      0
2 234934      1
3  65605      0
4 194959      1
5 329200      0
6 197469      0
```

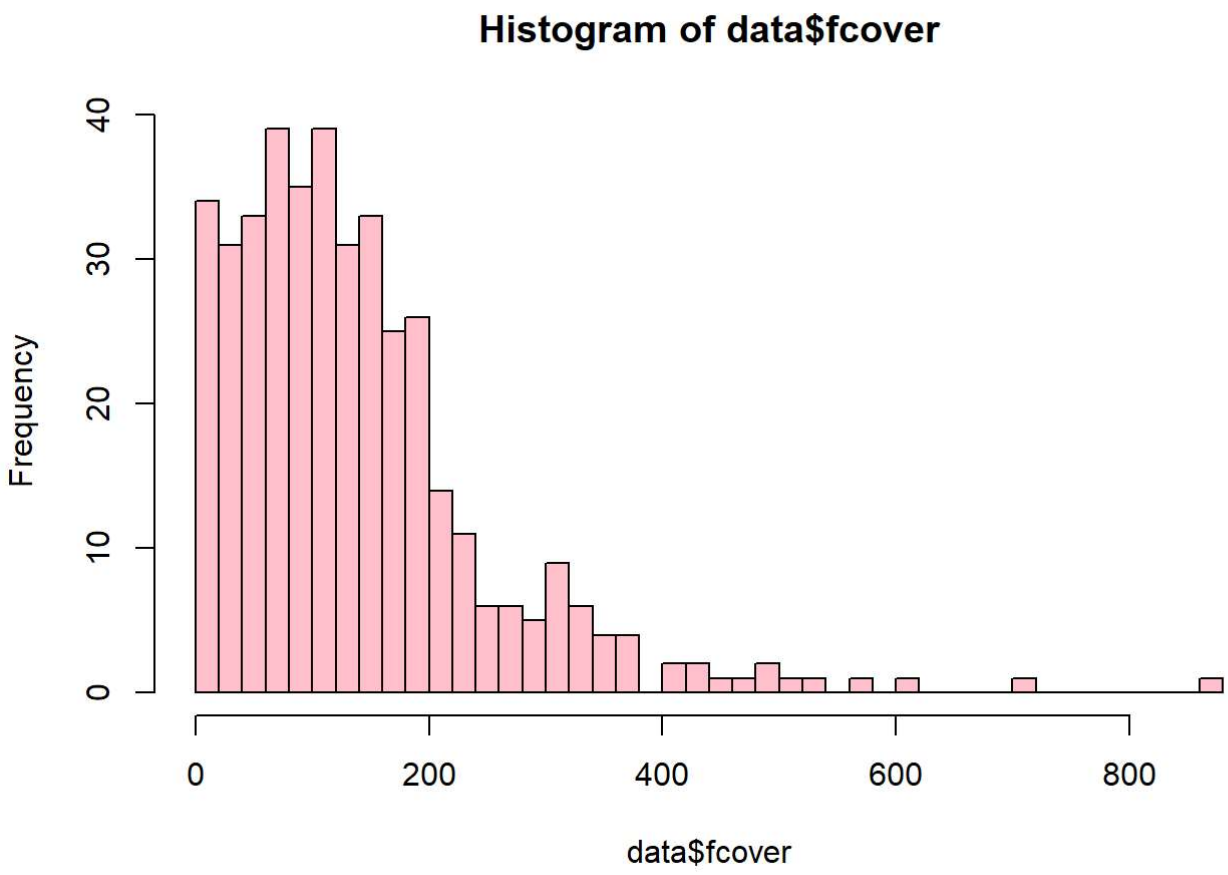
```r
#initial plots

# Create new columns for log(fcover) and sqrt(fcover)
data$log_fcover <- log(data$fcover + 1)  # Adding 1 to handle cases where fcover might be 0
data$sqrt_fcover <- sqrt(data$fcover)

hist(data$log_fcover, breaks=50, col="hotpink")
```




```r
hist(data$sqrt_fcover, breaks=50, col="violet")
```




```
hist(data$fcover, breaks=50, col ="pink")
```



The first histogram shows a logarithmic transformation (log_fcover) and exhibits a bimodal distribution with two prominent peaks around values 4 and 5. The second histogram represents a square root transformation (sqrt_fcover) and displays a slightly skewed distribution with a dominant peak around the value of 10. The third histogram represents the raw data (fcover) and is right-skewed, showing a significant concentration of data points between 0 and 200 with a rapid decline in frequency as values increase.

Based on the provided visualizations, the histogram of the square root transformation $\sqrt{fcover}$ seems to have a more normalized or bell-shaped distribution compared to the other two. This implies that the square root transformation might be more effective in stabilizing the variance and making the data more symmetric

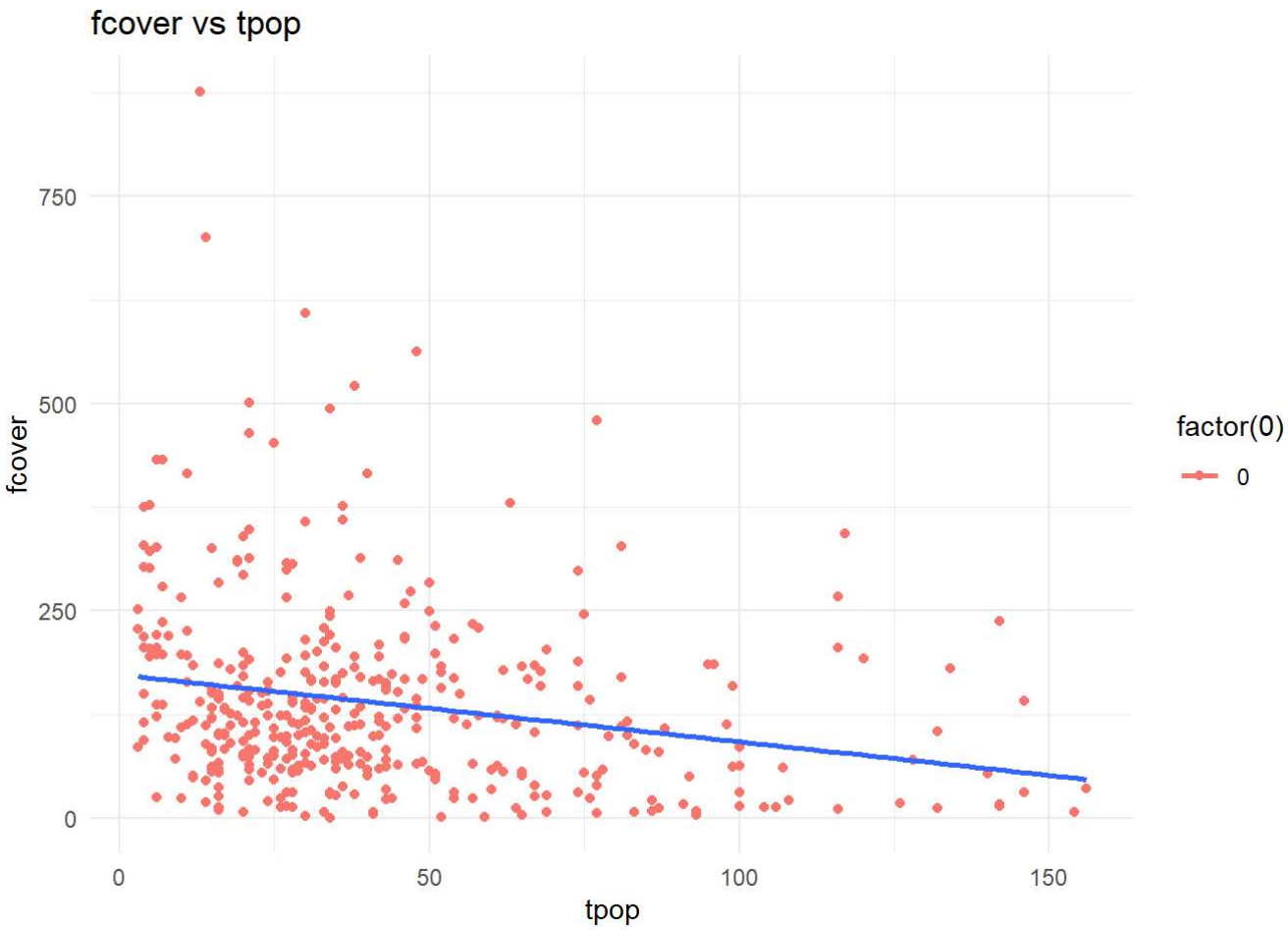
> More Initial Plots

```
library(ggplot2)
library(dplyr)

data$log_fcover <- log(data$fcover + 1)
data$sqrt_fcover <- sqrt(data$fcover)

# List of predictor variables
predictor_vars <- setdiff(names(data), c("fcover", "log_fcover", "sqrt_fcover", "id", "code", "ena

# Generate plots
```

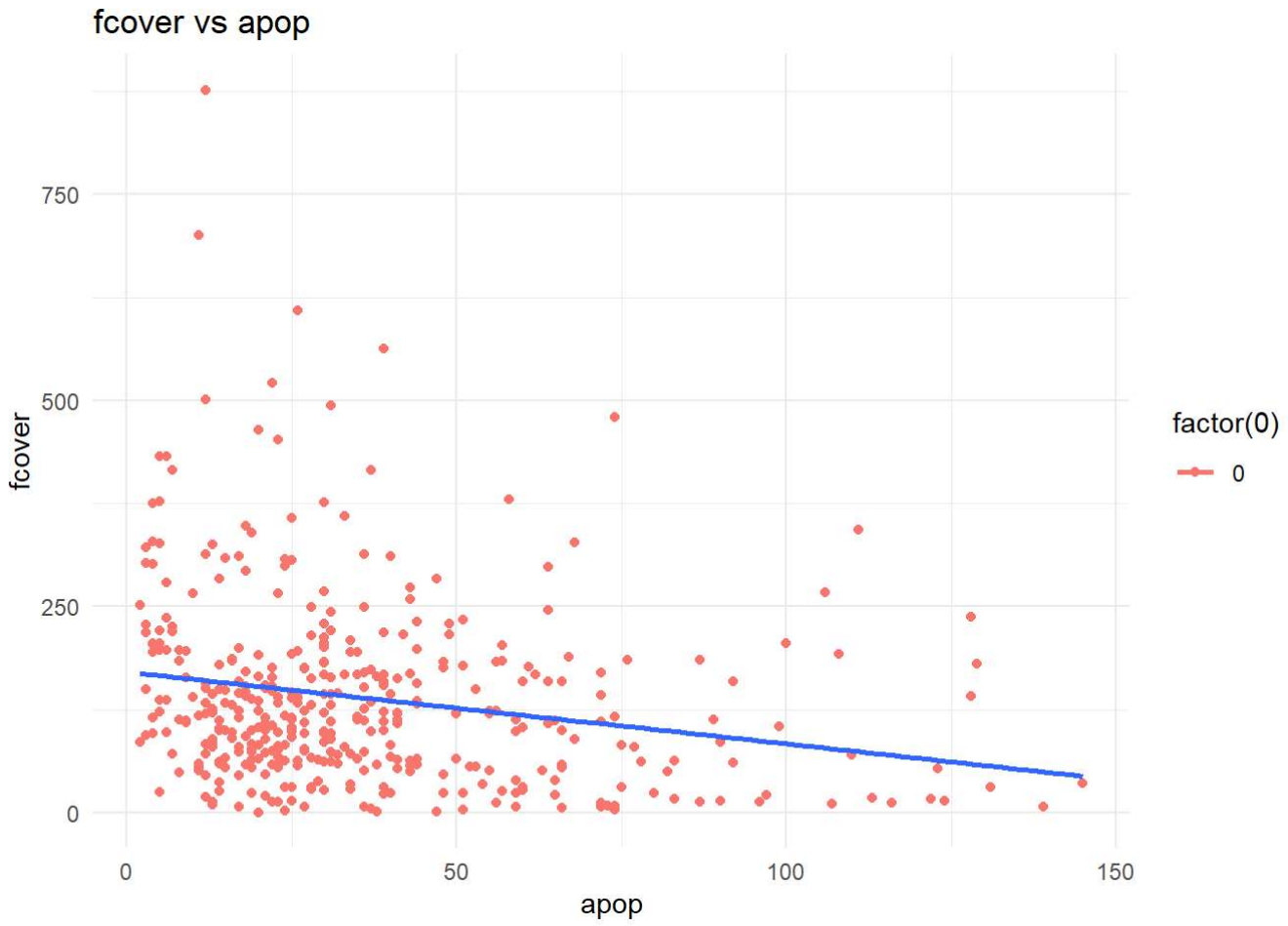
```
for (response_var in c("fcover", "log_fcover", "sqrt_fcover")) {
  for (predictor_var in predictor_vars) {
    p <- ggplot(data, aes_string(x=predictor_var, y=response_var)) +
         geom_point(aes(color=factor(0))) +  # The factor(0) is a hack to get a single color
         geom_smooth(method="lm", se=FALSE, aes(group=1, color=NULL)) +  # Linear fit
         labs(title=paste(response_var, "vs", predictor_var),
              x=predictor_var,
              y=response_var) +
         theme_minimal()
    print(p)
  }
}
```

```
Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
ℹ Please use tidy evaluation idioms with `aes()`.
ℹ See also `vignette("ggplot2-in-packages")` for more information.
```

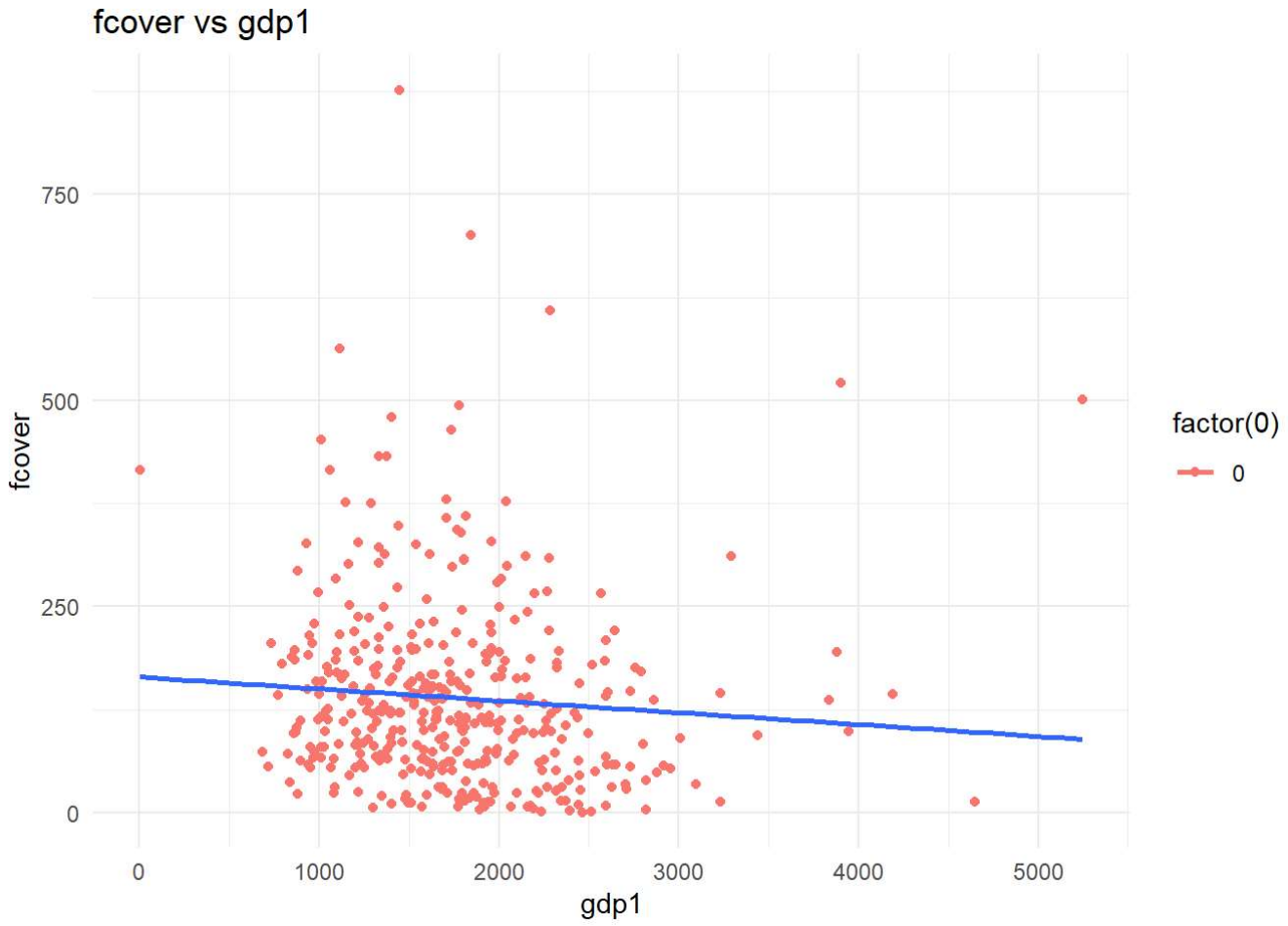
```
`geom_smooth()` using formula = 'y ~ x'
```



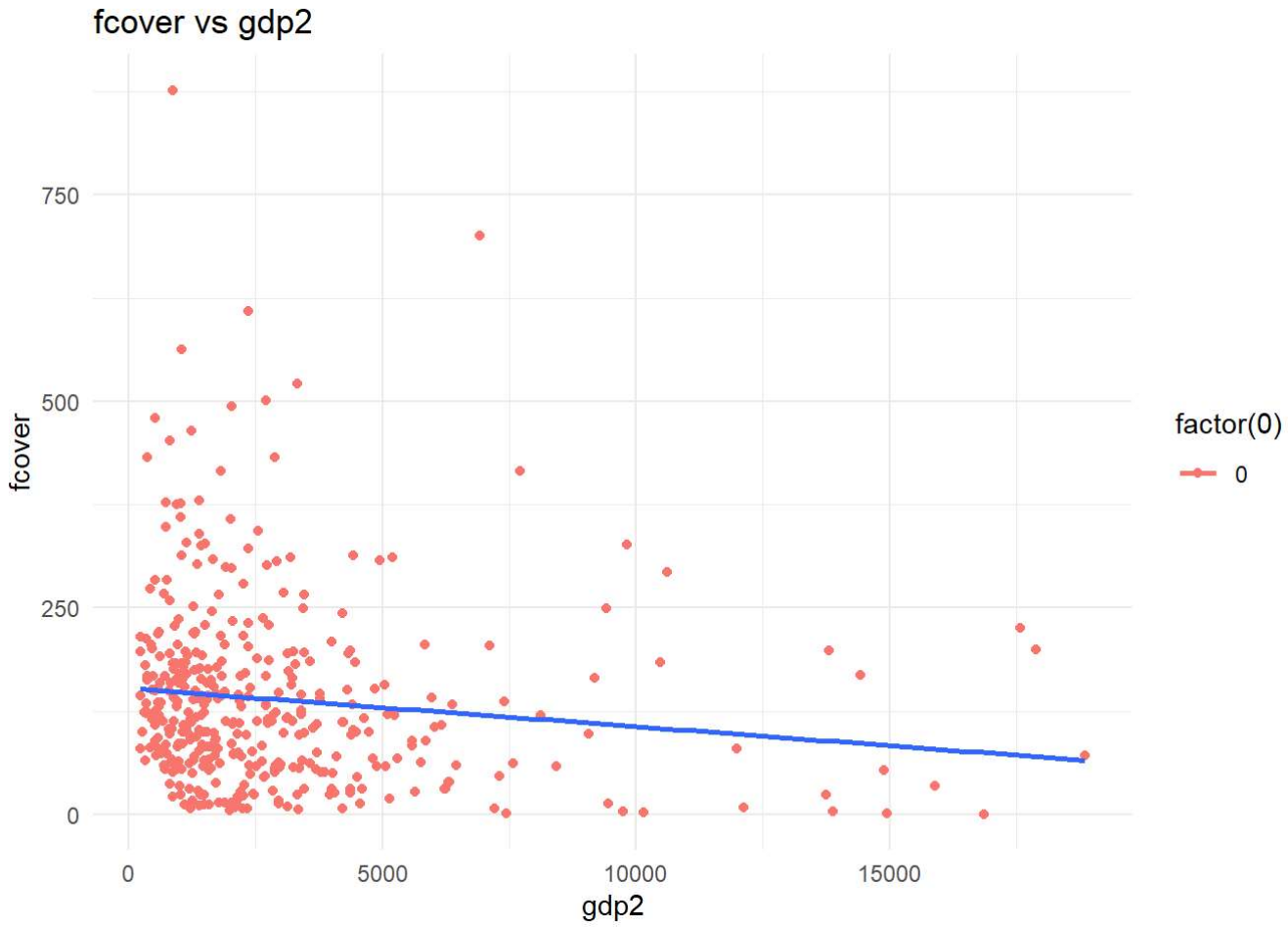

```
`geom_smooth()` using formula = 'y ~ x'
```



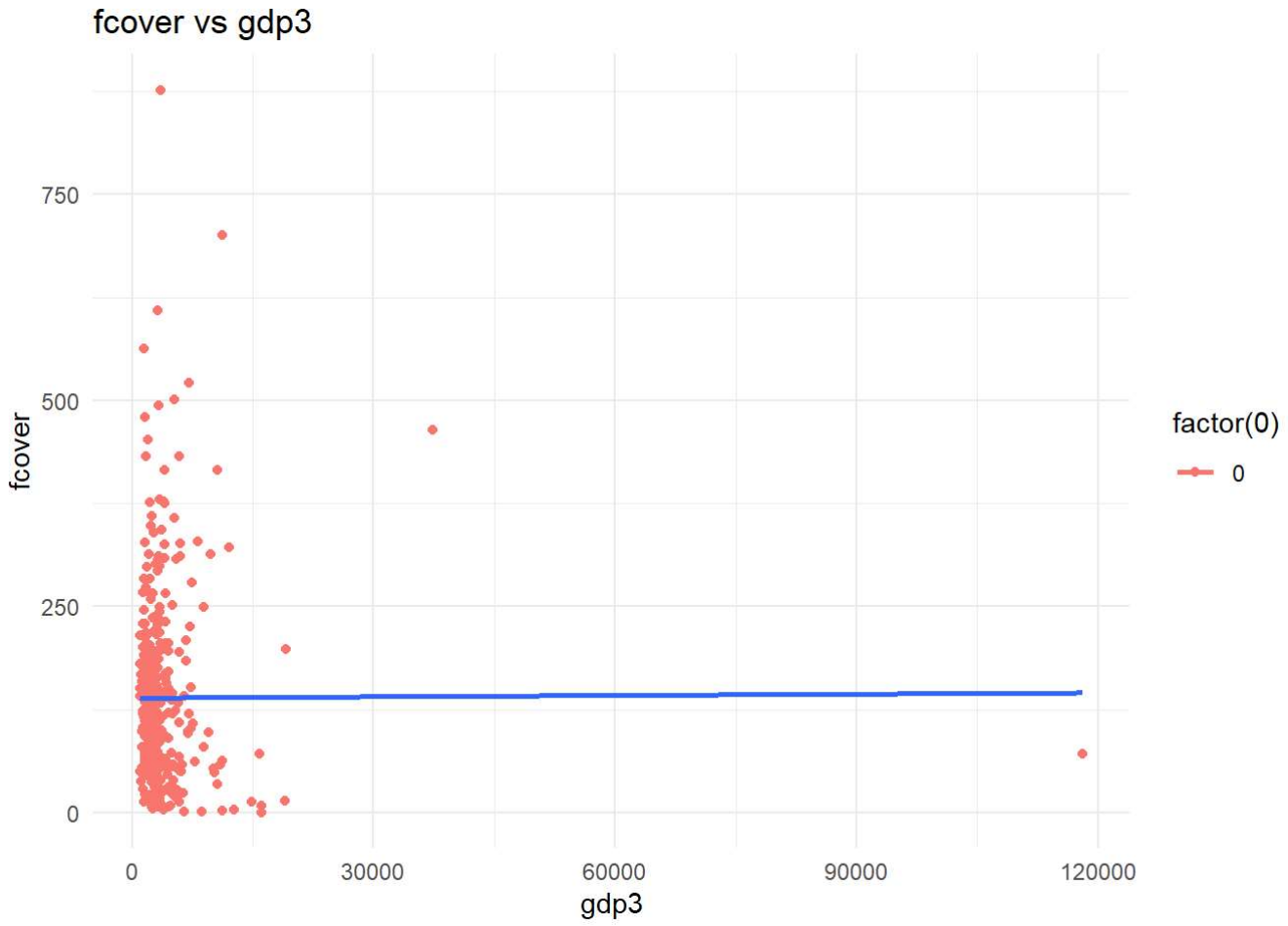

```
`geom_smooth()` using formula = 'y ~ x'
```



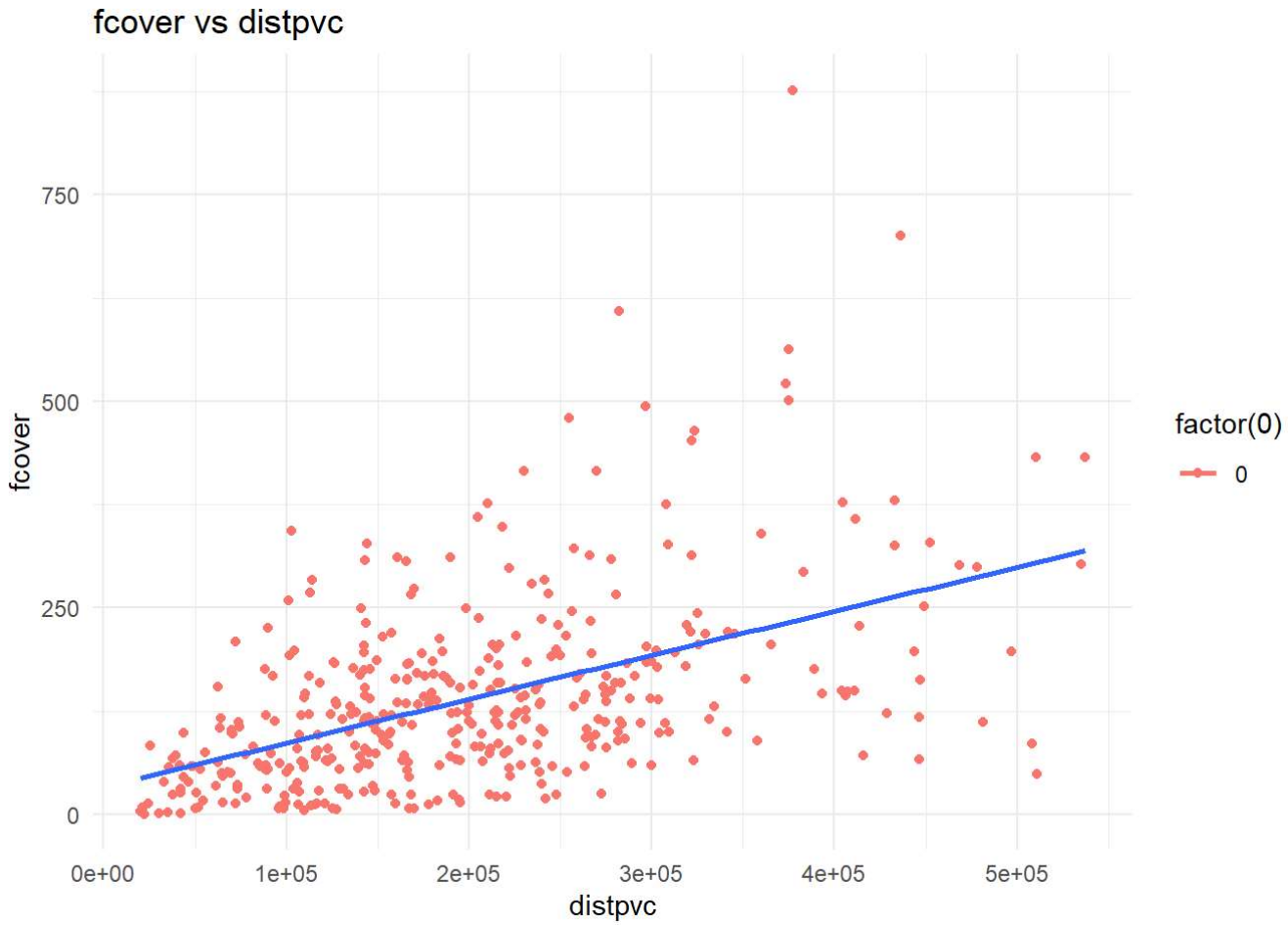

```
`geom_smooth()` using formula = 'y ~ x'
```



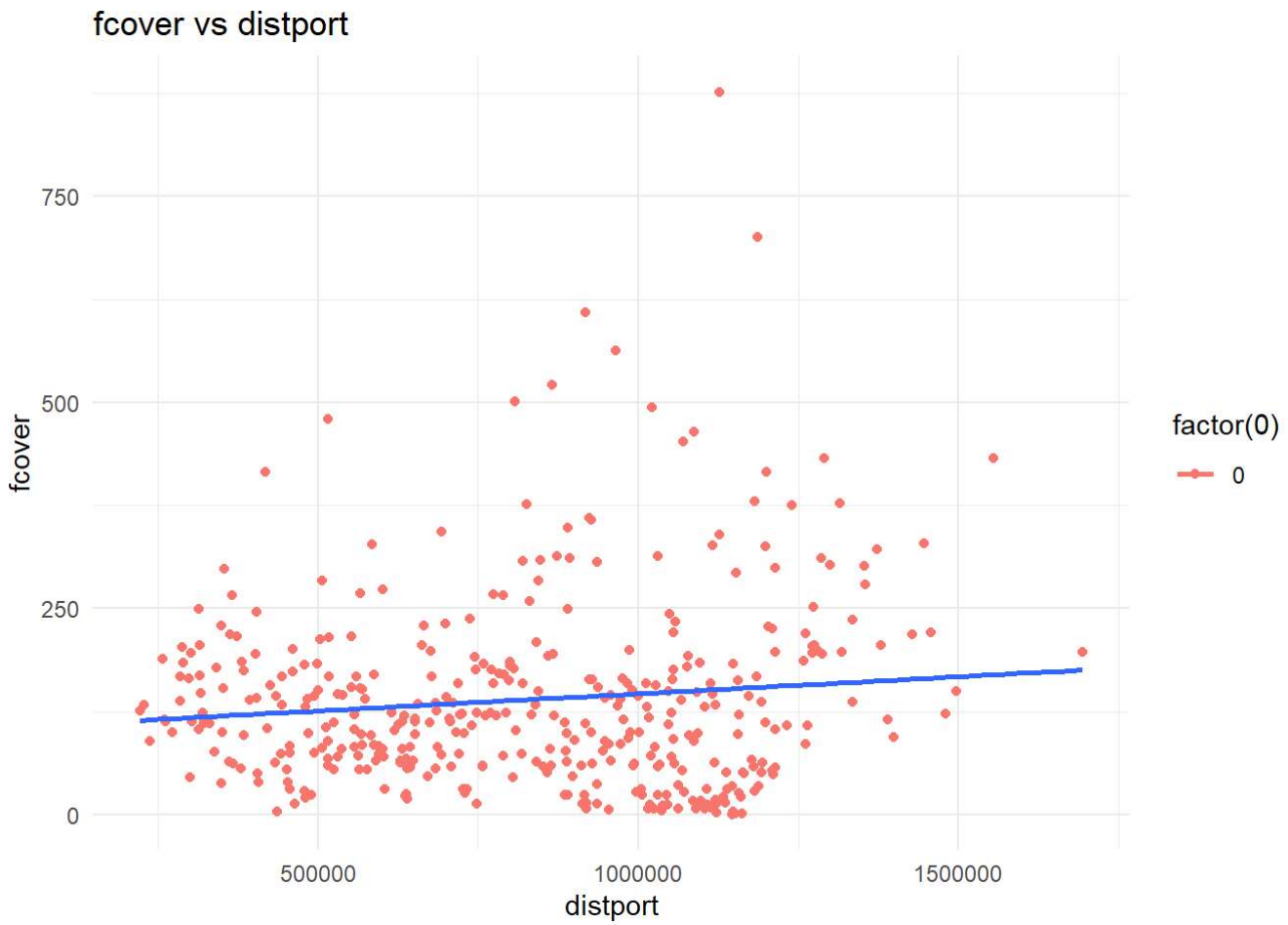

```
`geom_smooth()` using formula = 'y ~ x'
```



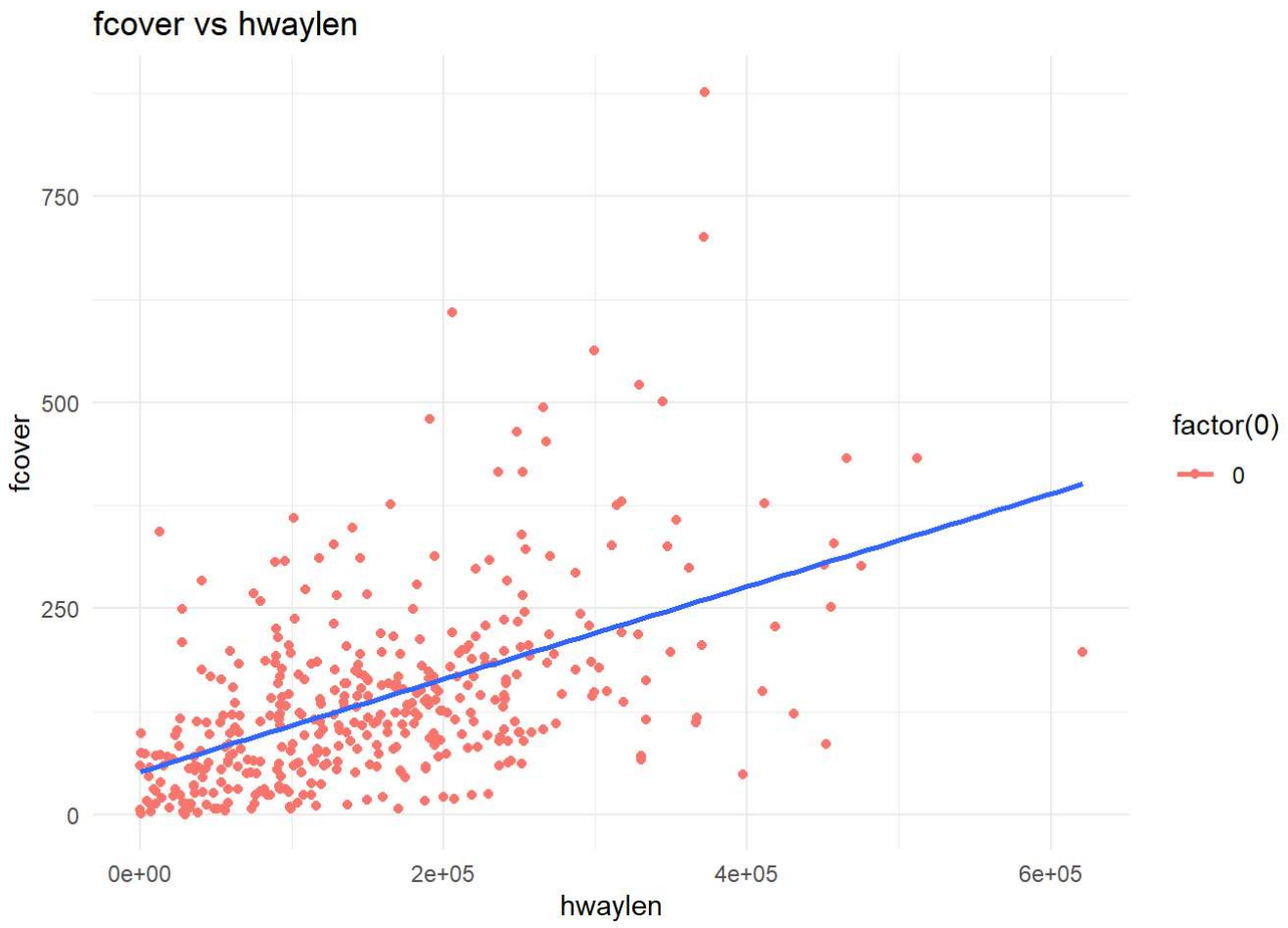

```
`geom_smooth()` using formula = 'y ~ x'
```



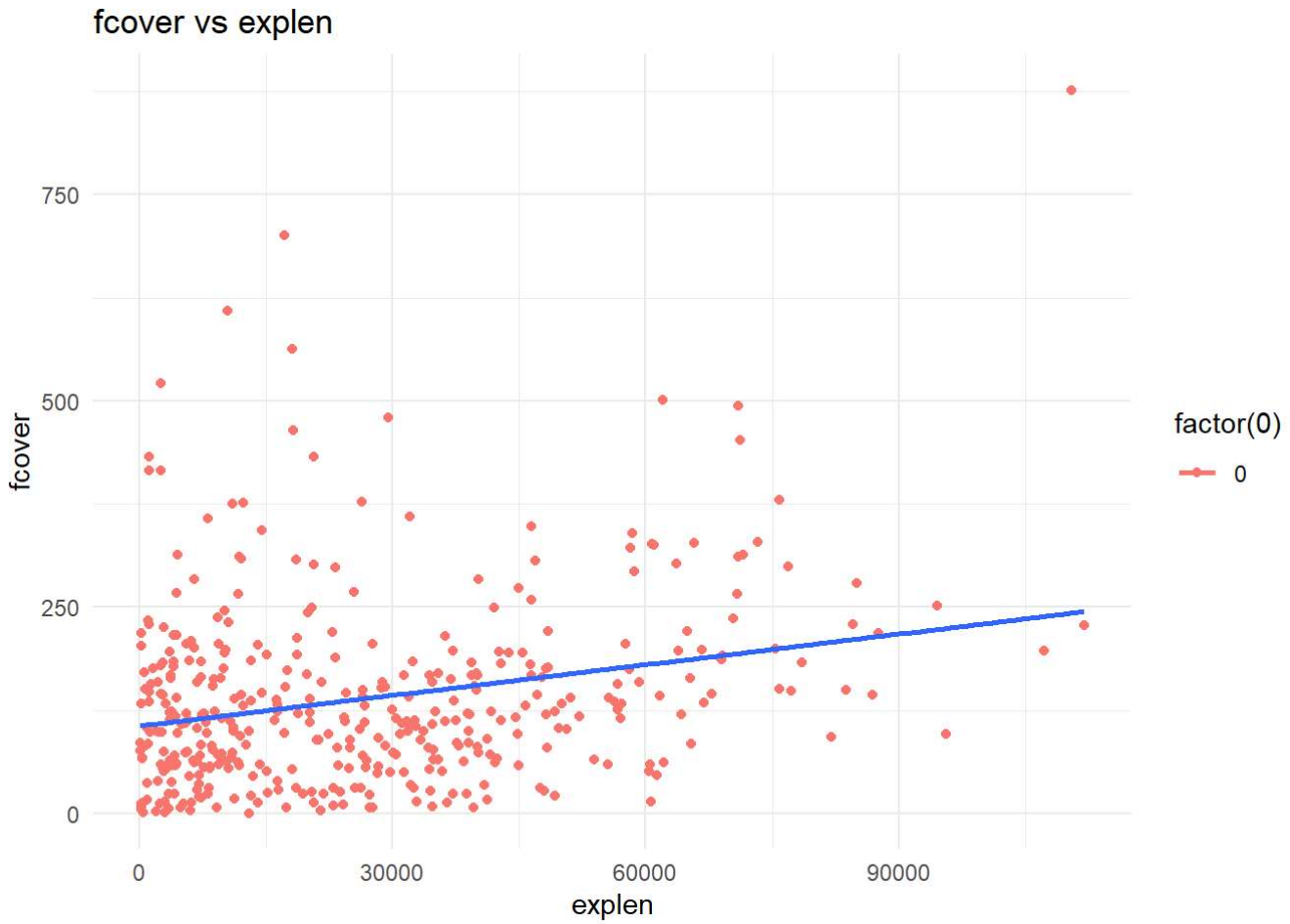

```
`geom_smooth()` using formula = 'y ~ x'
```



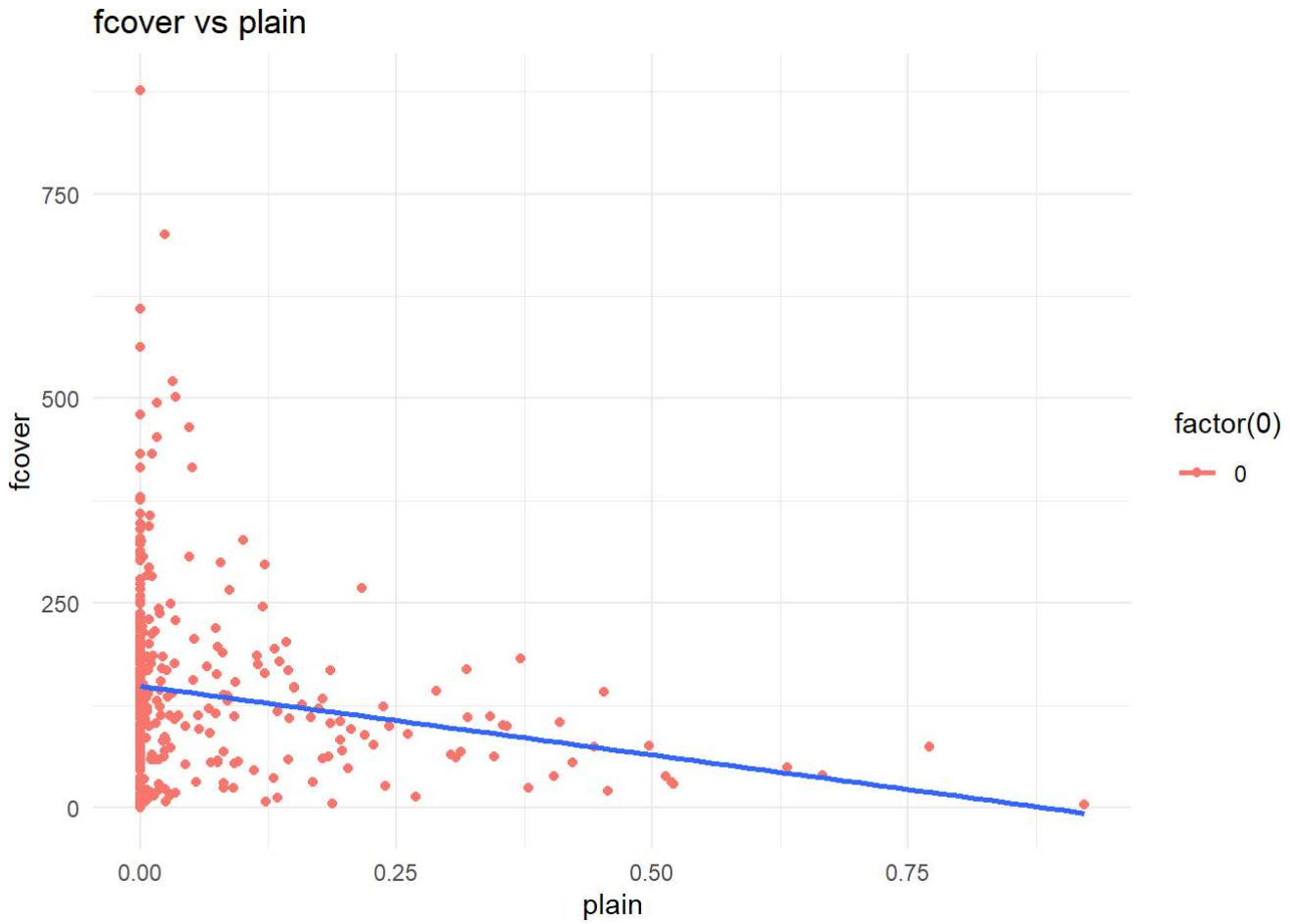

```
`geom_smooth()` using formula = 'y ~ x'
```



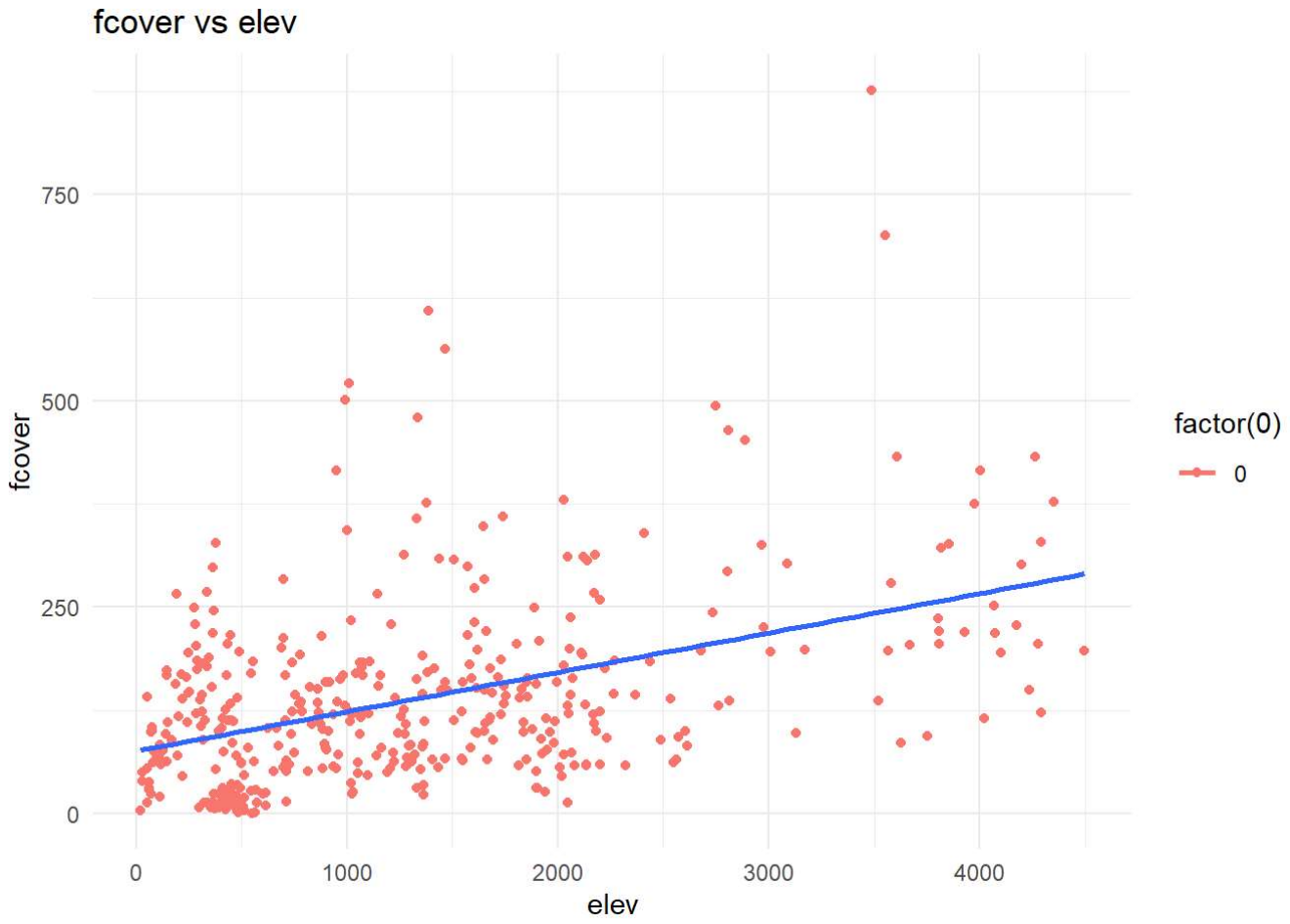

```
`geom_smooth()` using formula = 'y ~ x'
```



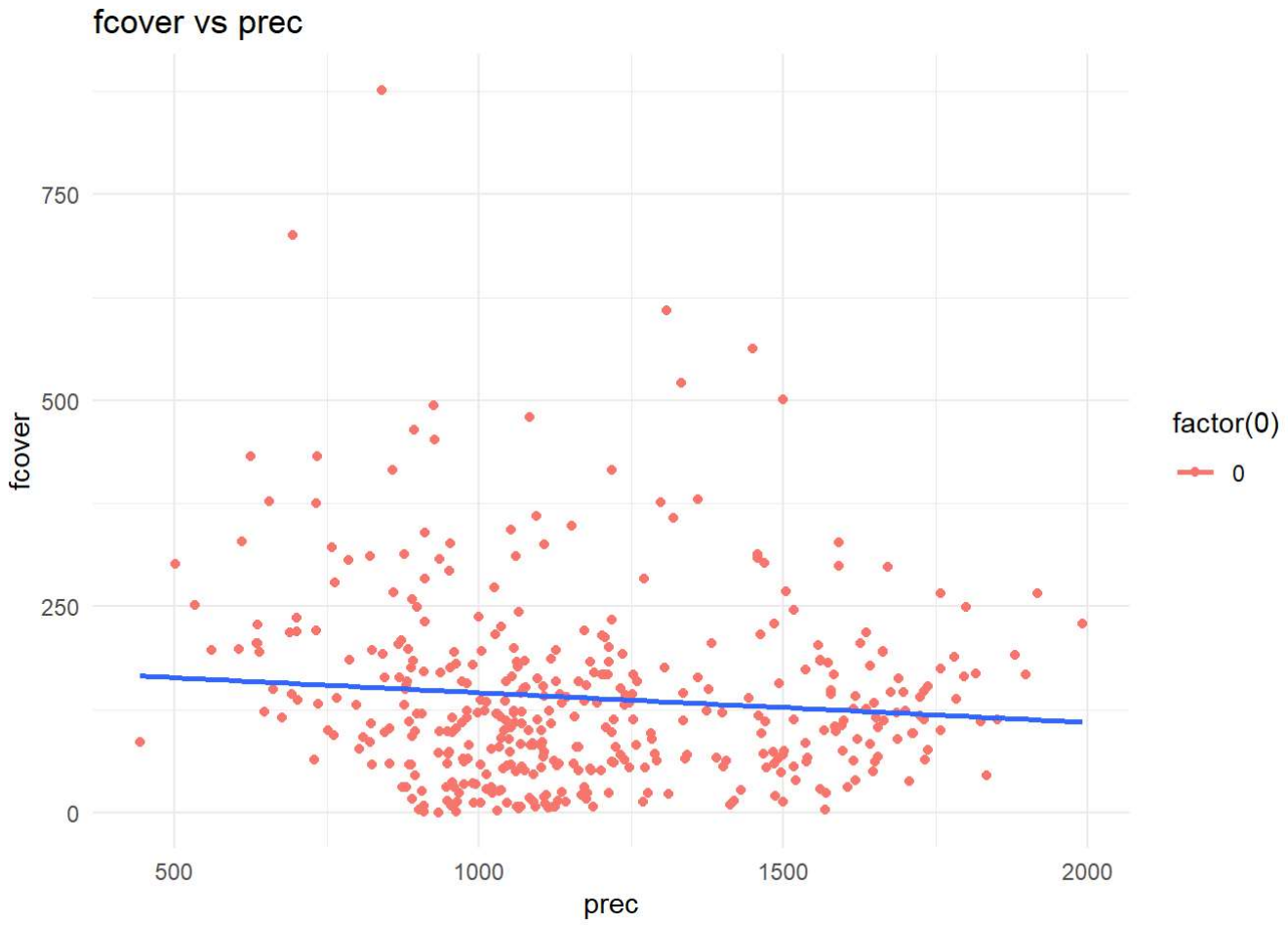

```
`geom_smooth()` using formula = 'y ~ x'
```



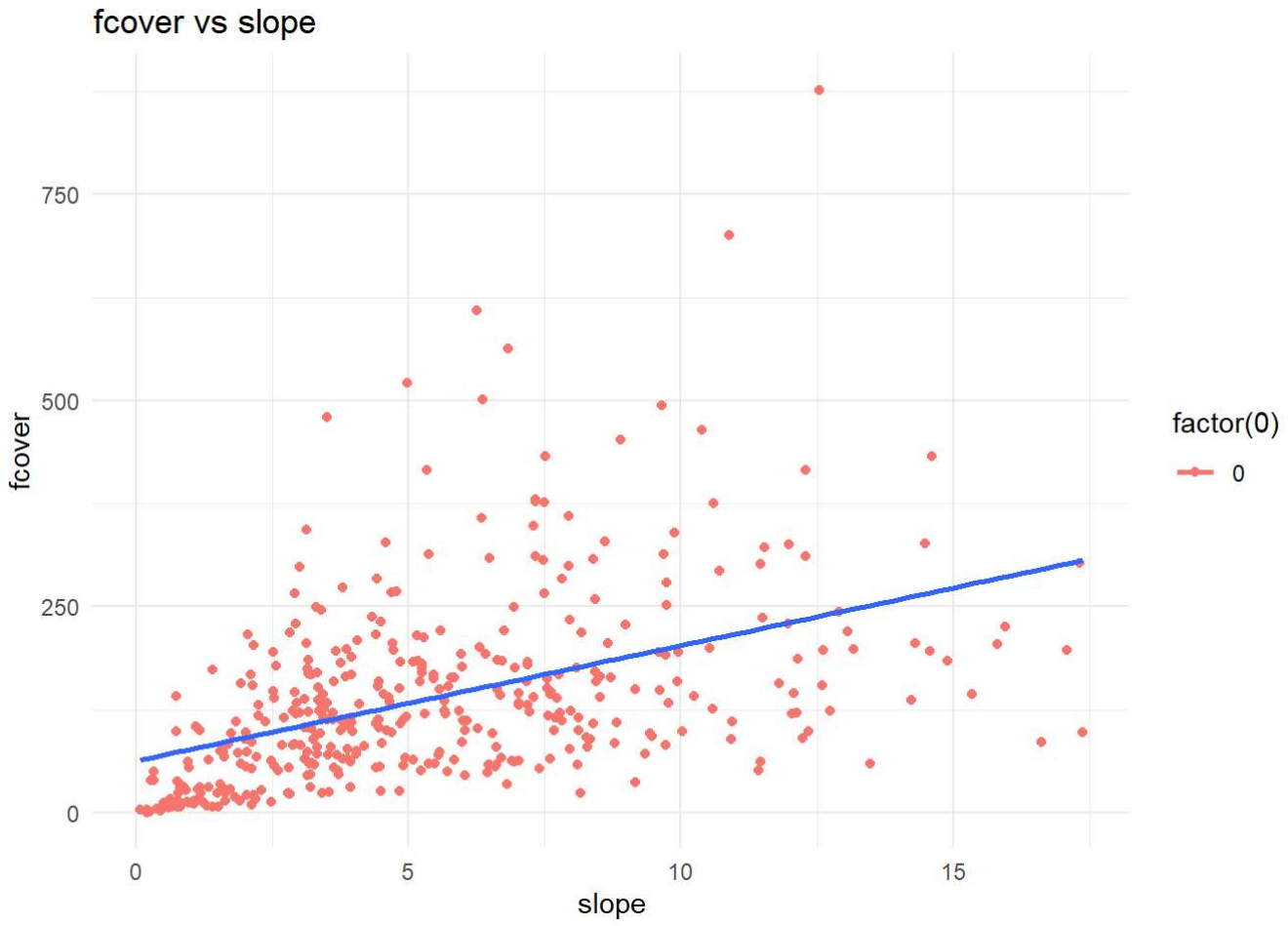

```
`geom_smooth()` using formula = 'y ~ x'
```



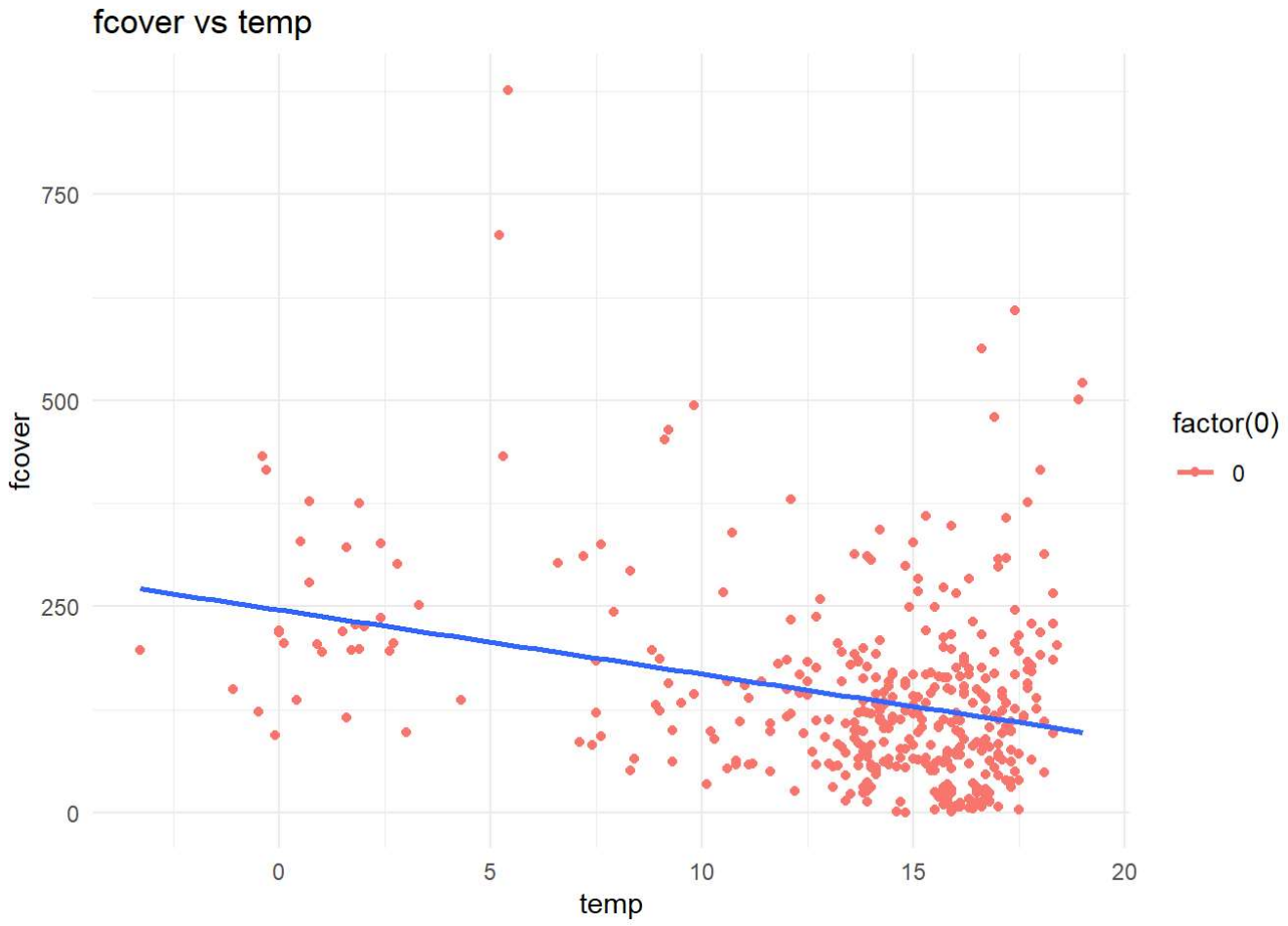

```
`geom_smooth()` using formula = 'y ~ x'
```



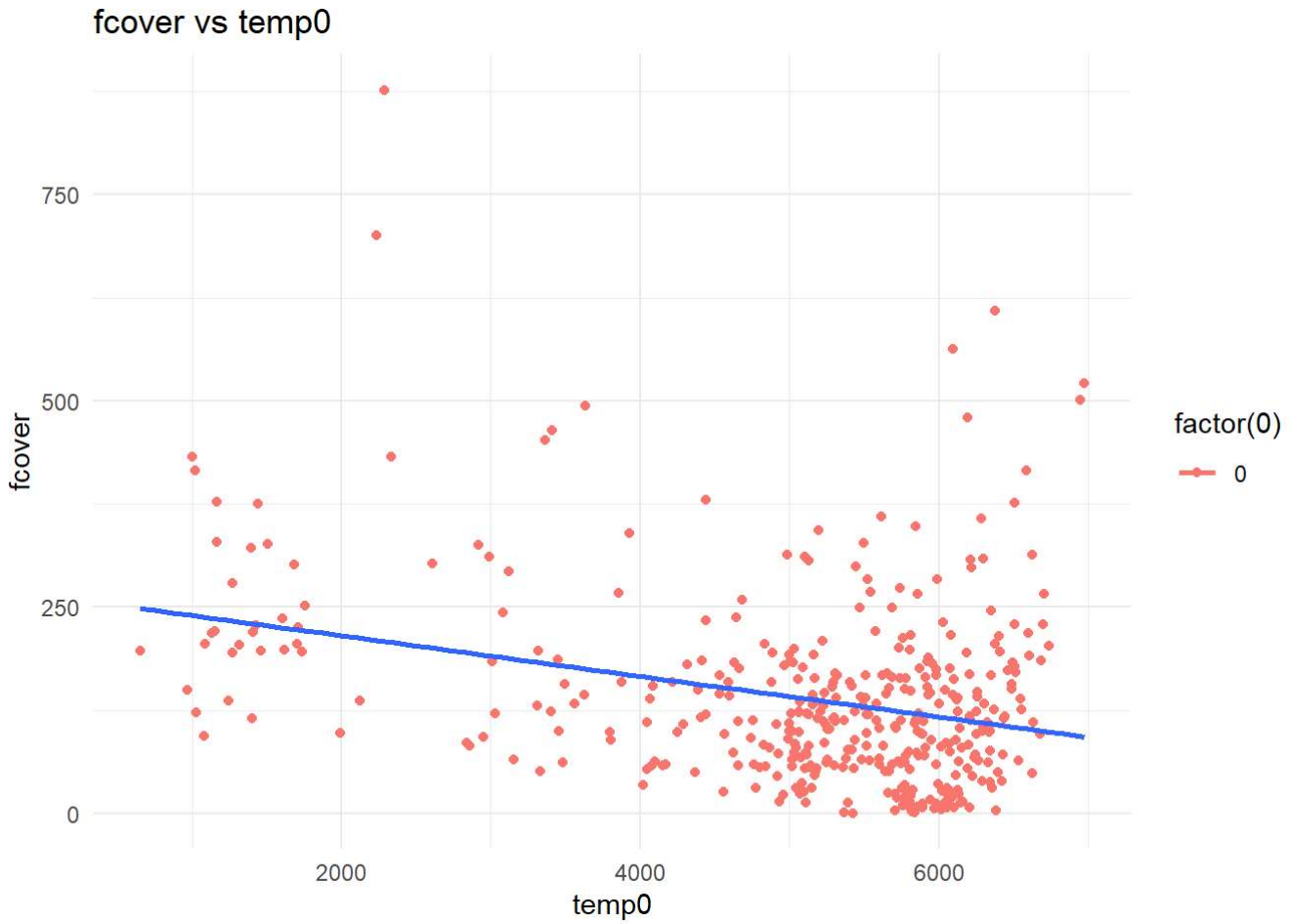

```
`geom_smooth()` using formula = 'y ~ x'
```



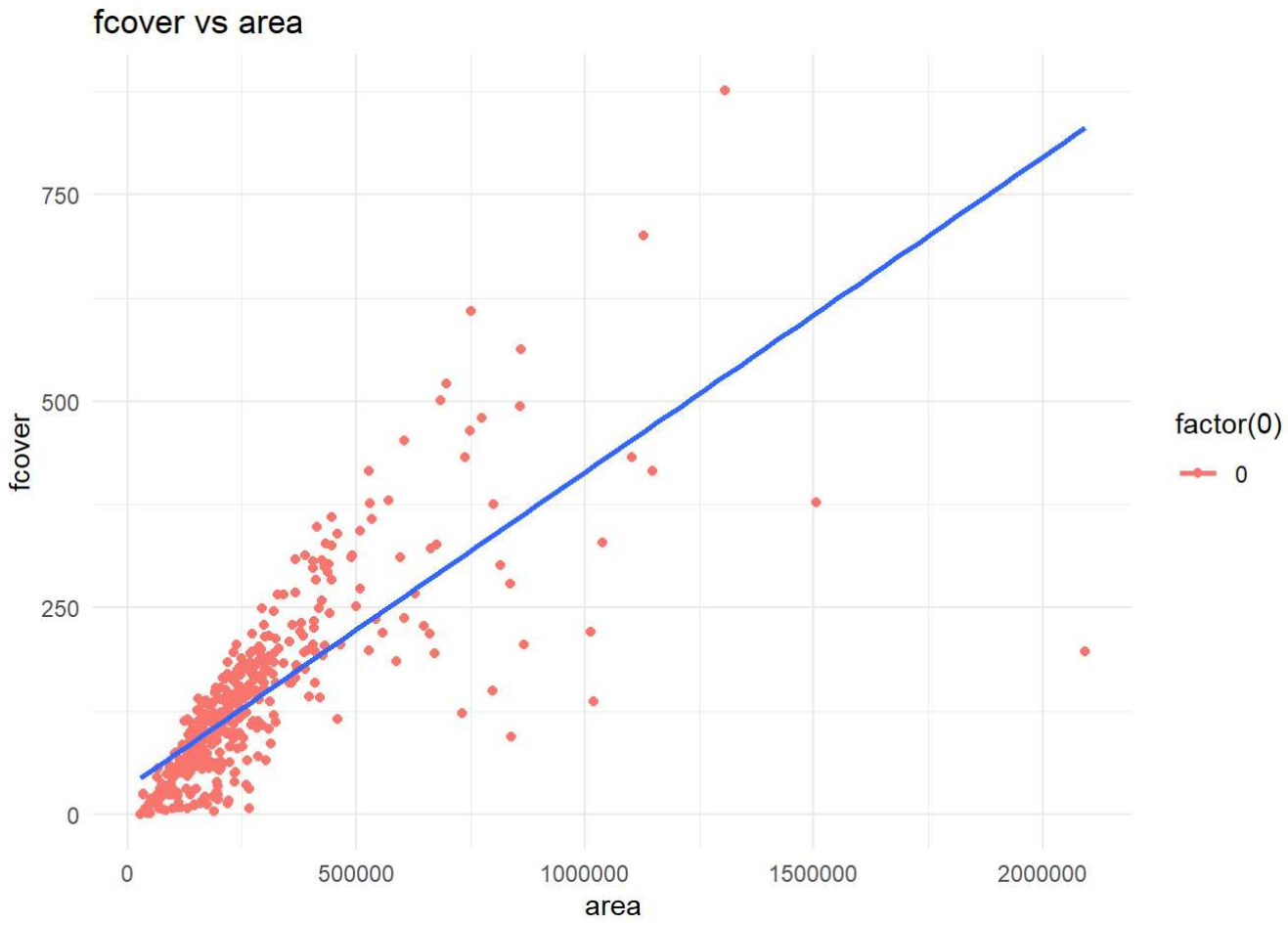

```
`geom_smooth()` using formula = 'y ~ x'
```



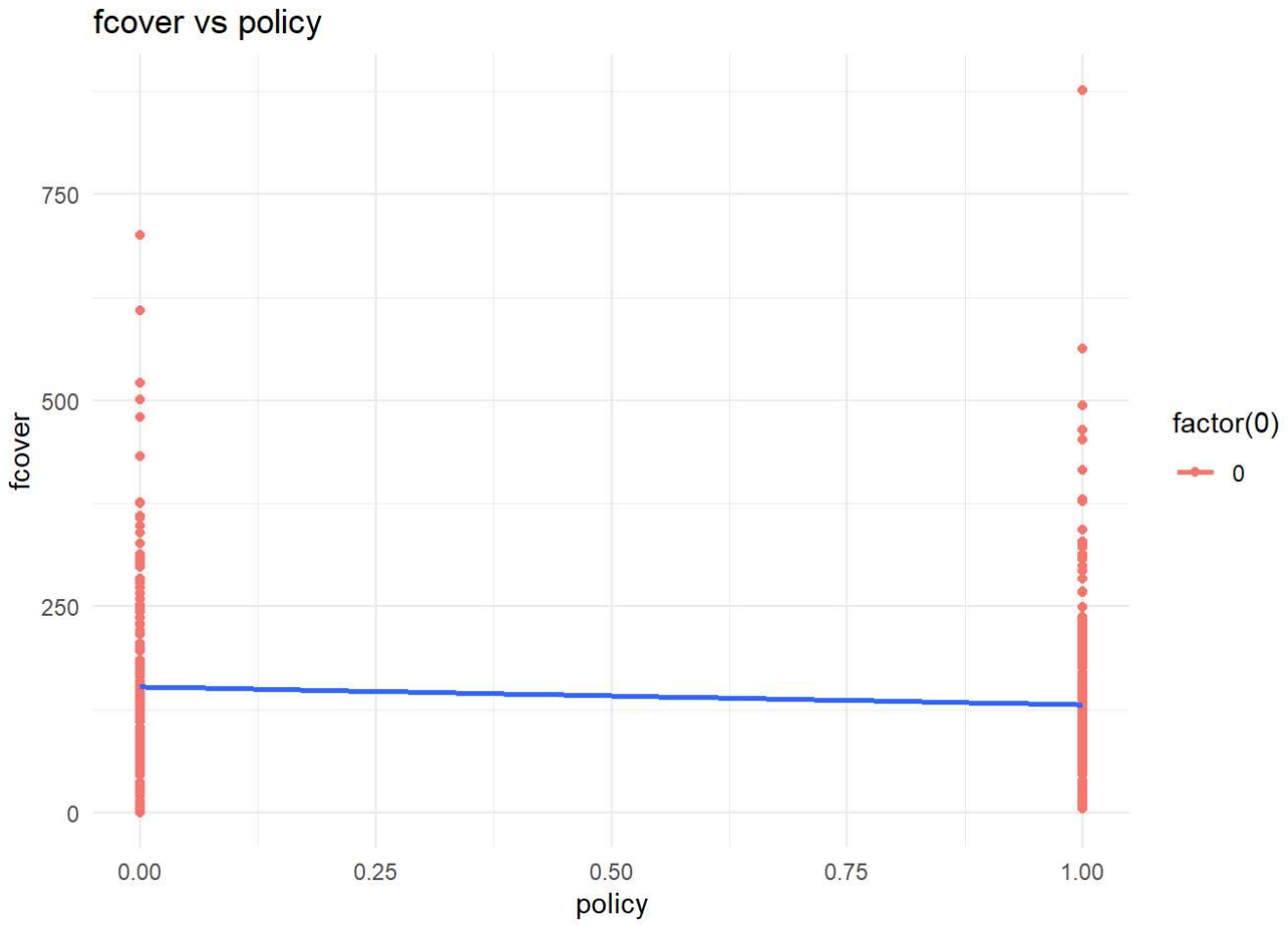

```
`geom_smooth()` using formula = 'y ~ x'
```



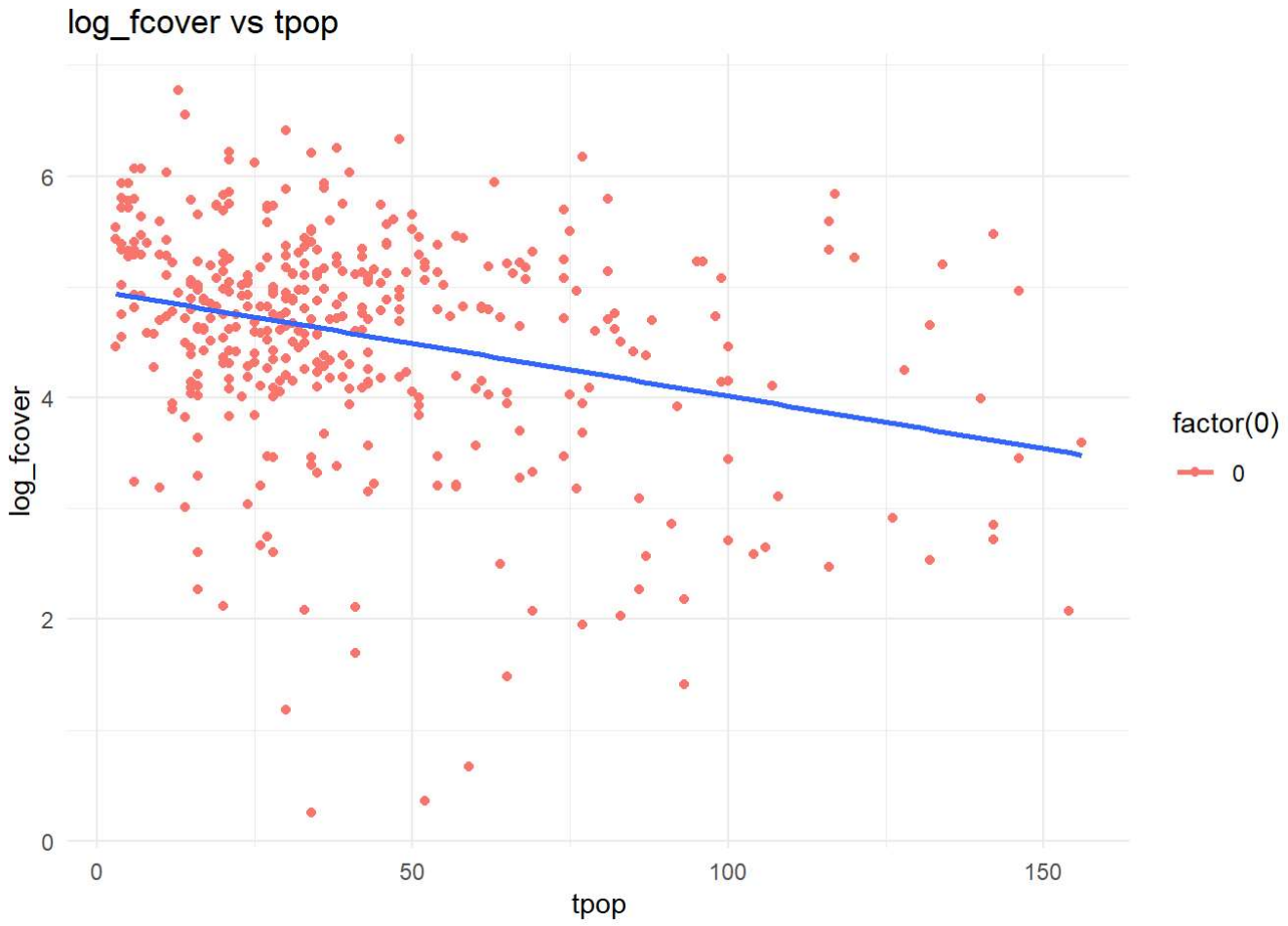

```
`geom_smooth()` using formula = 'y ~ x'
```



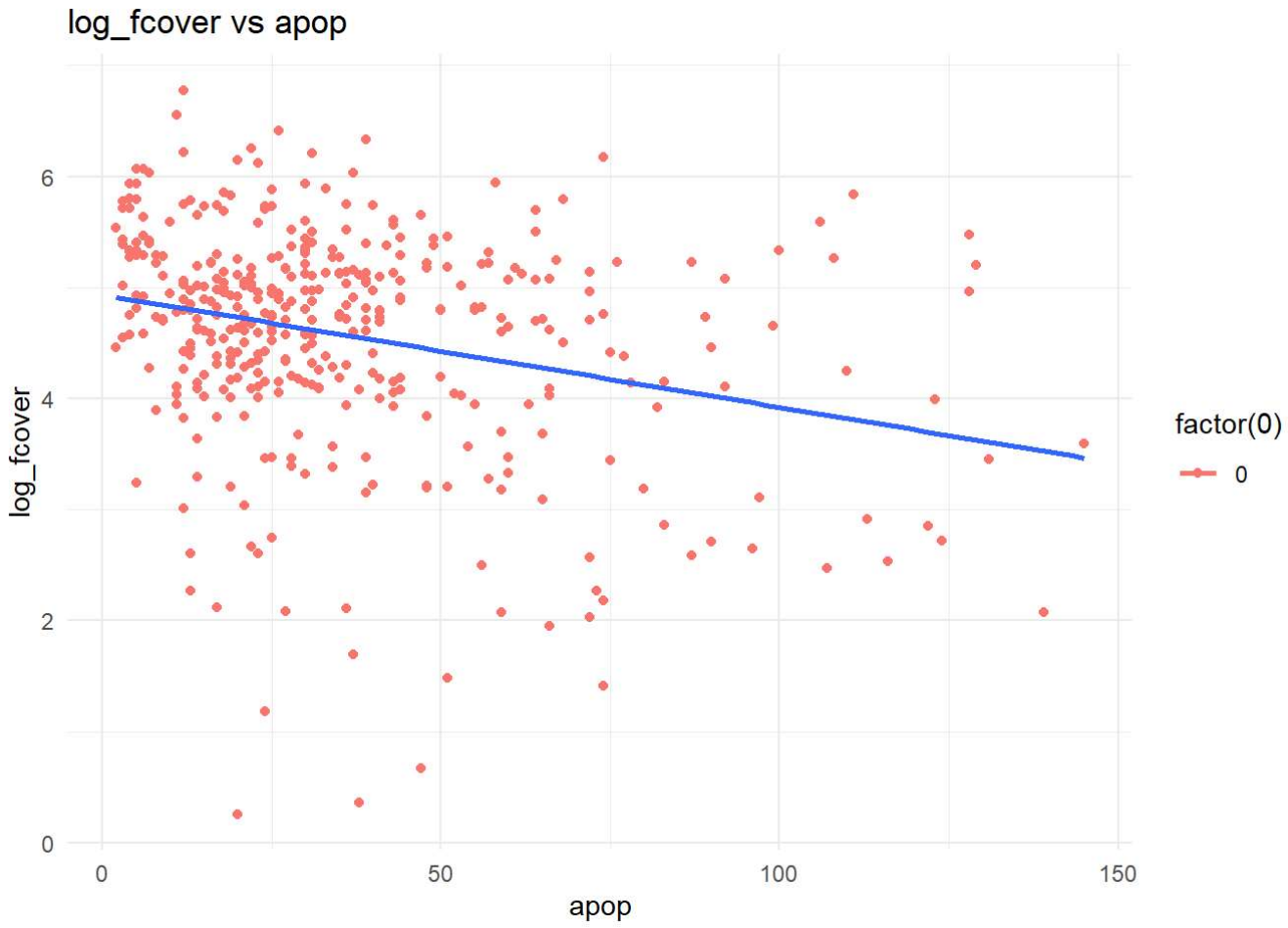

```
`geom_smooth()` using formula = 'y ~ x'
```



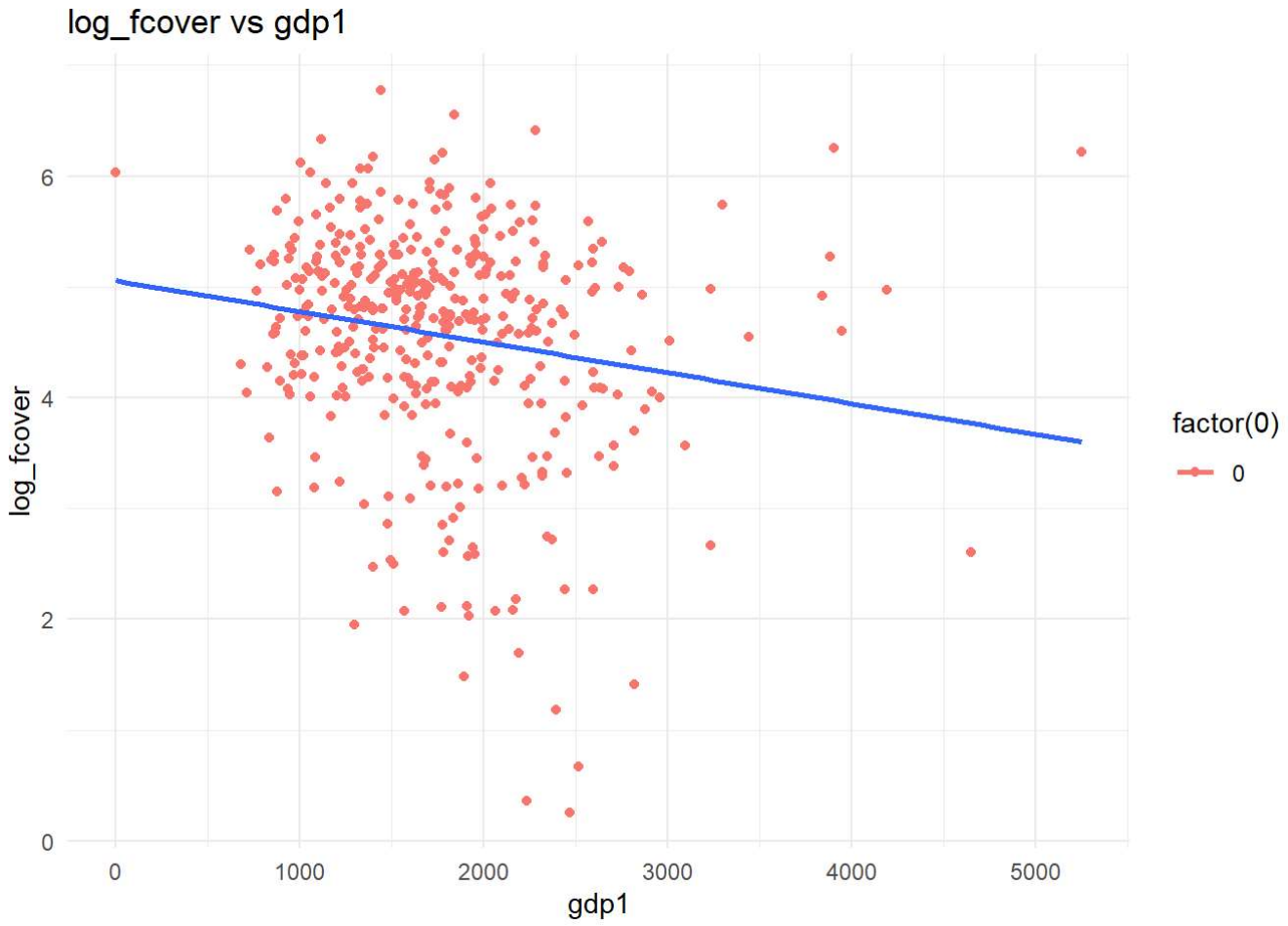

```
`geom_smooth()` using formula = 'y ~ x'
```



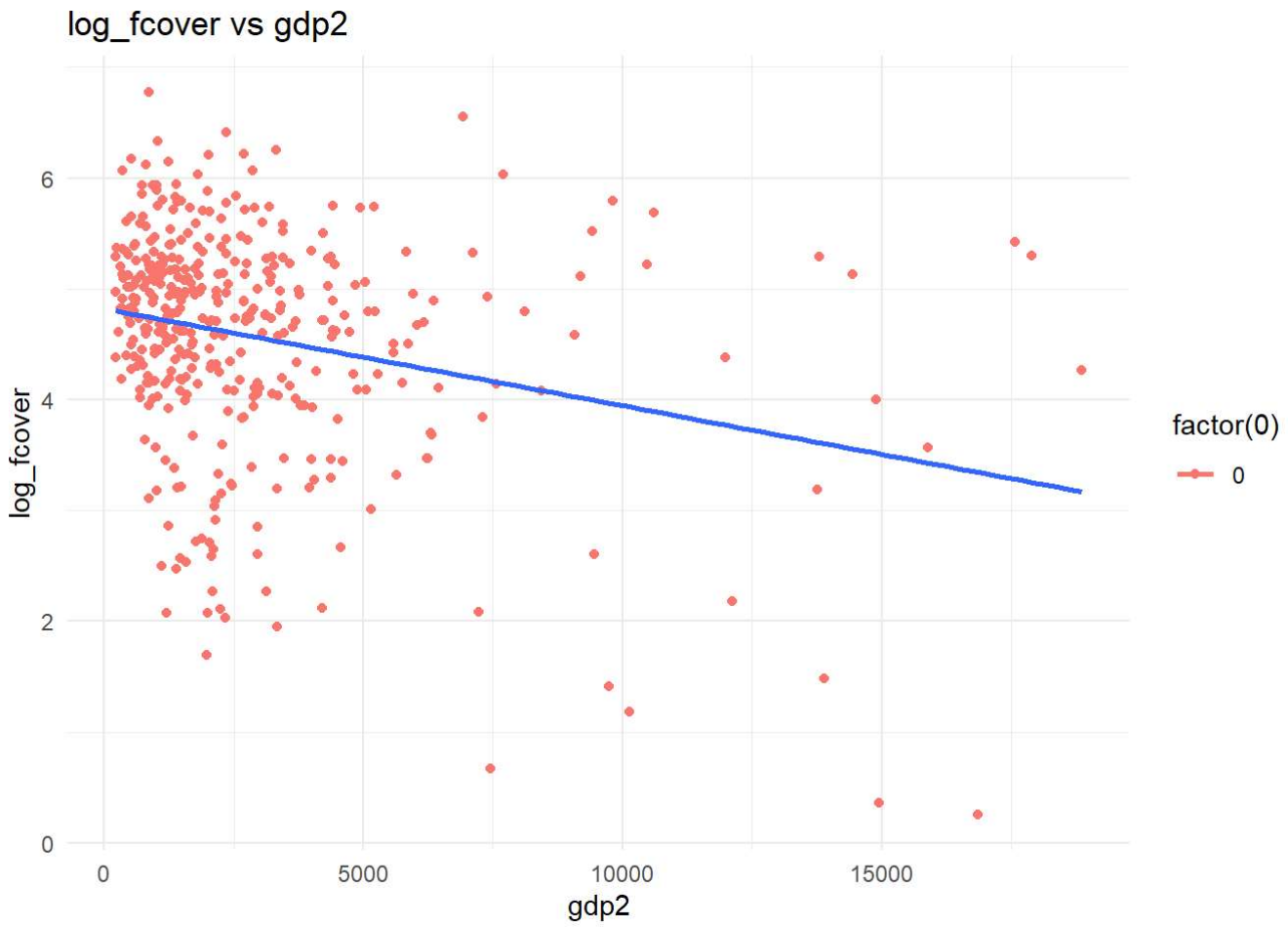

```
`geom_smooth()` using formula = 'y ~ x'
```



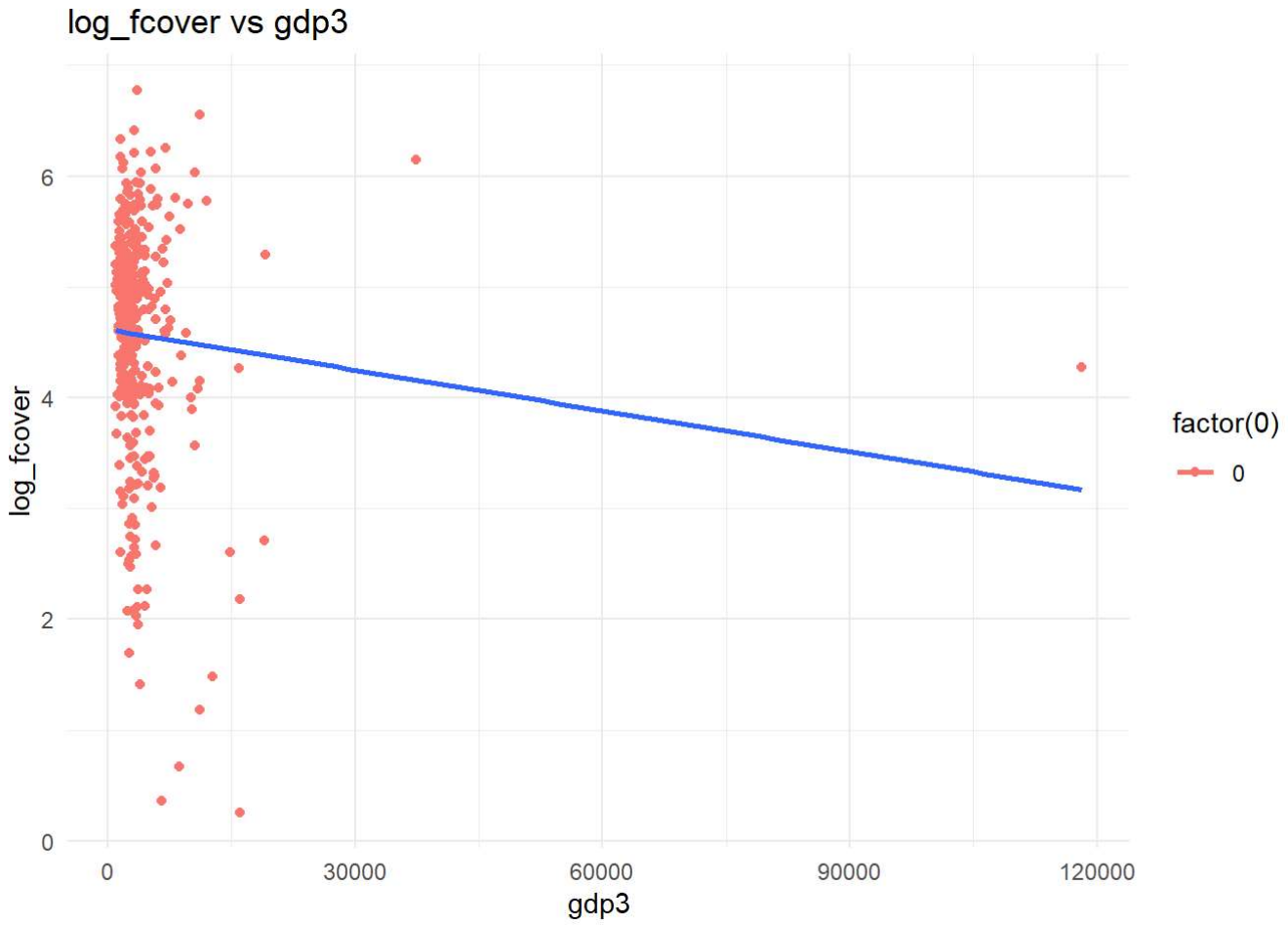

```
`geom_smooth()` using formula = 'y ~ x'
```



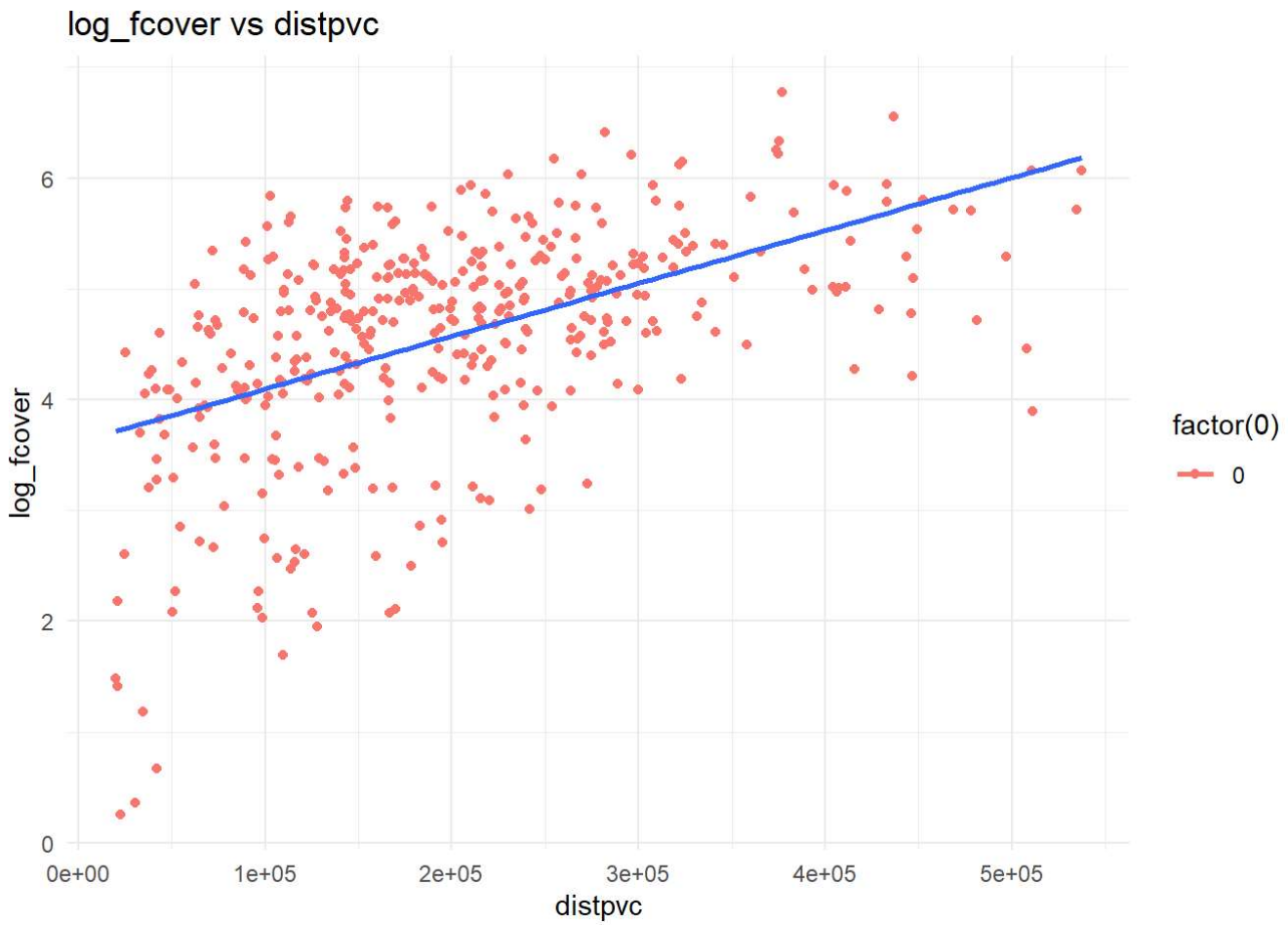

```
`geom_smooth()` using formula = 'y ~ x'
```



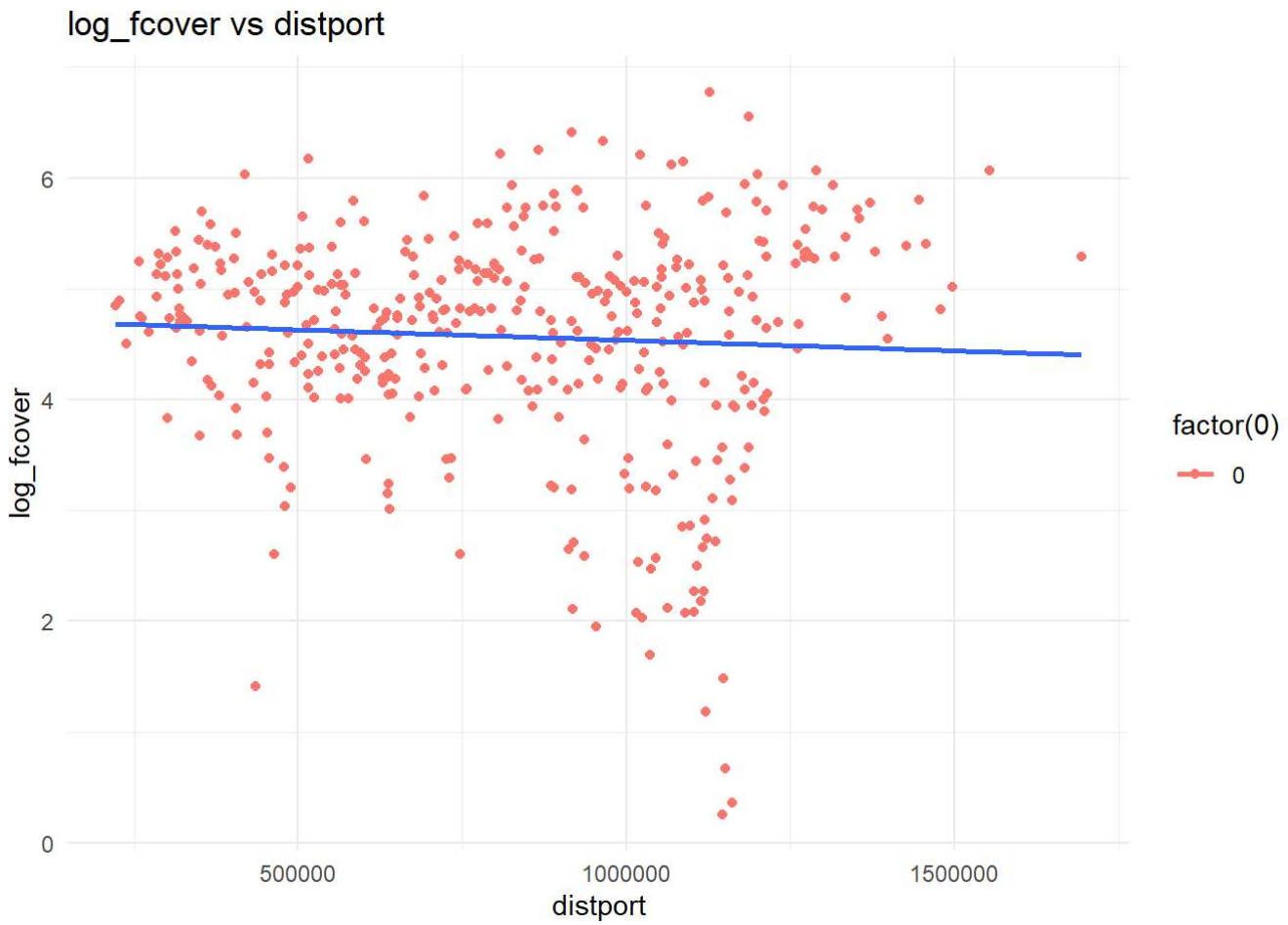

```
`geom_smooth()` using formula = 'y ~ x'
```



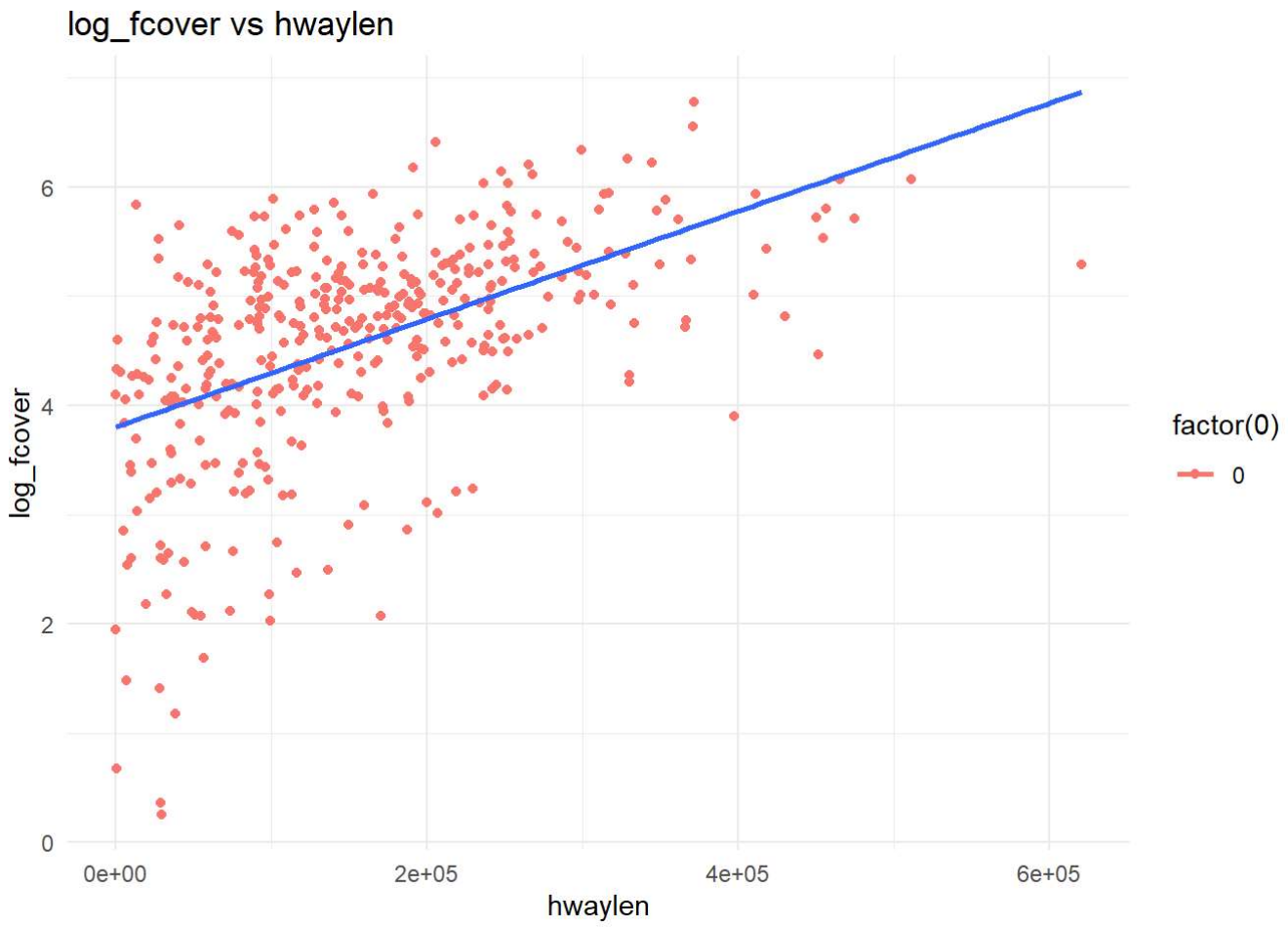

```
`geom_smooth()` using formula = 'y ~ x'
```



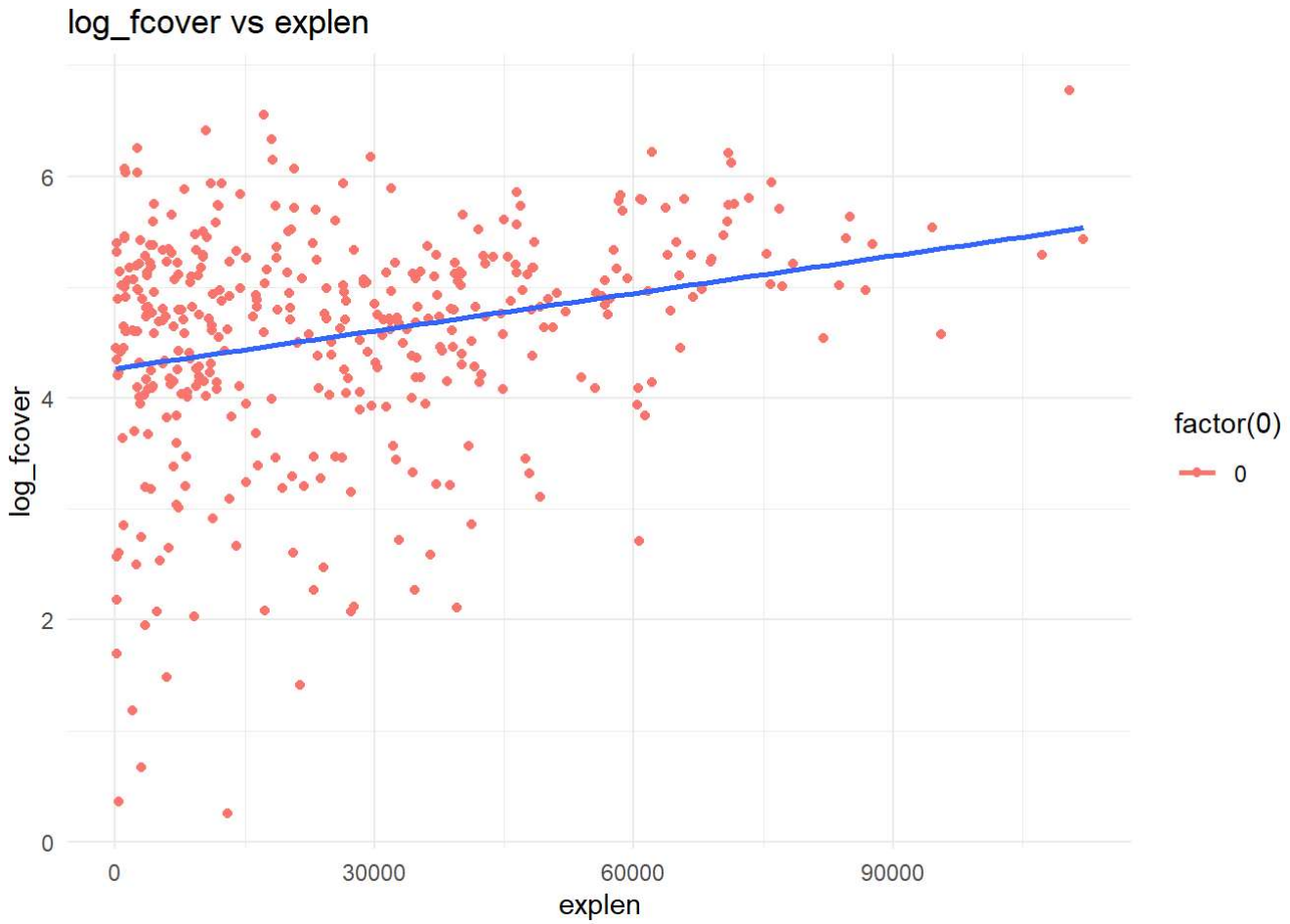

```
`geom_smooth()` using formula = 'y ~ x'
```



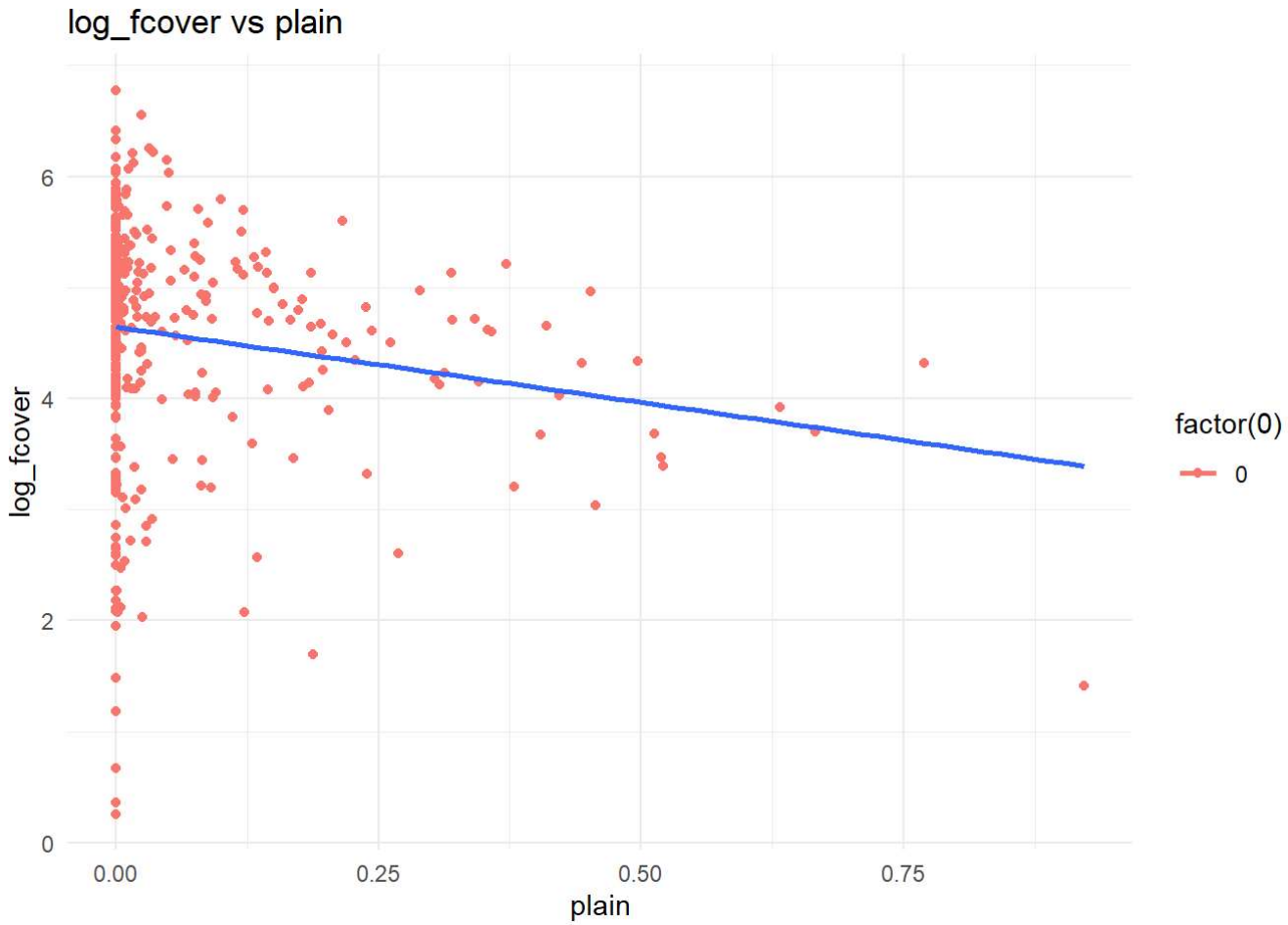

```
`geom_smooth()` using formula = 'y ~ x'
```



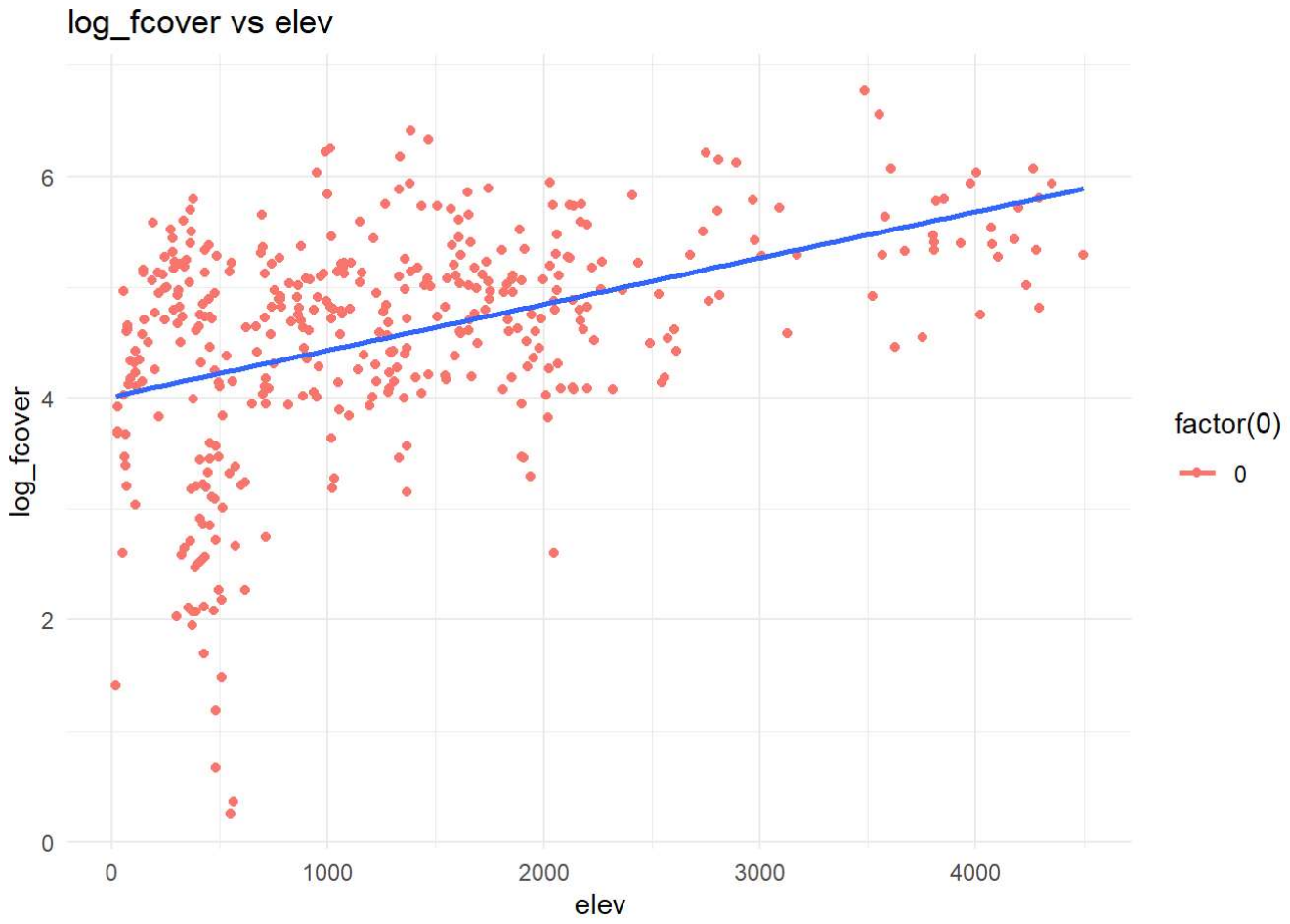

```
`geom_smooth()` using formula = 'y ~ x'
```



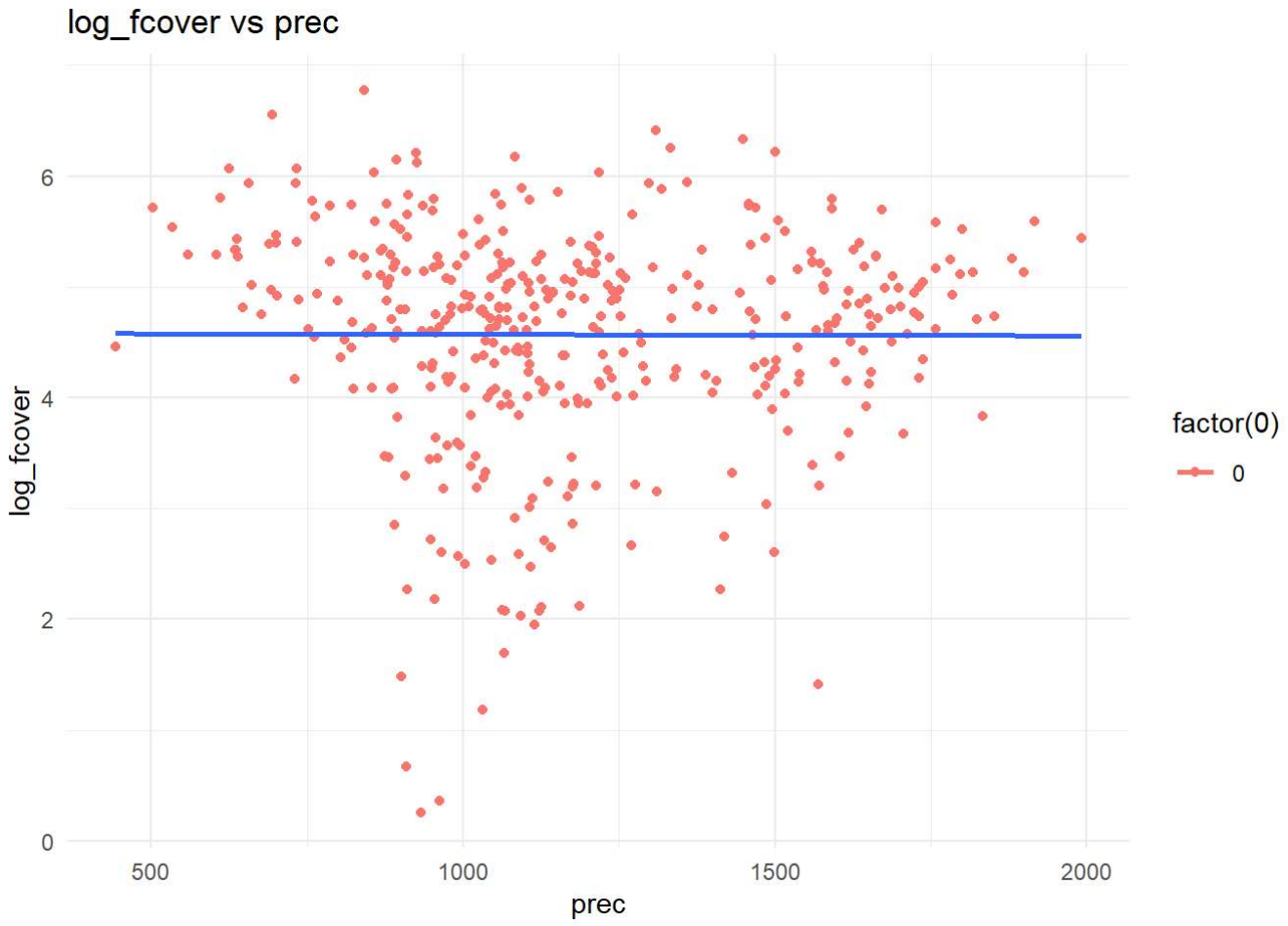

```
`geom_smooth()` using formula = 'y ~ x'
```



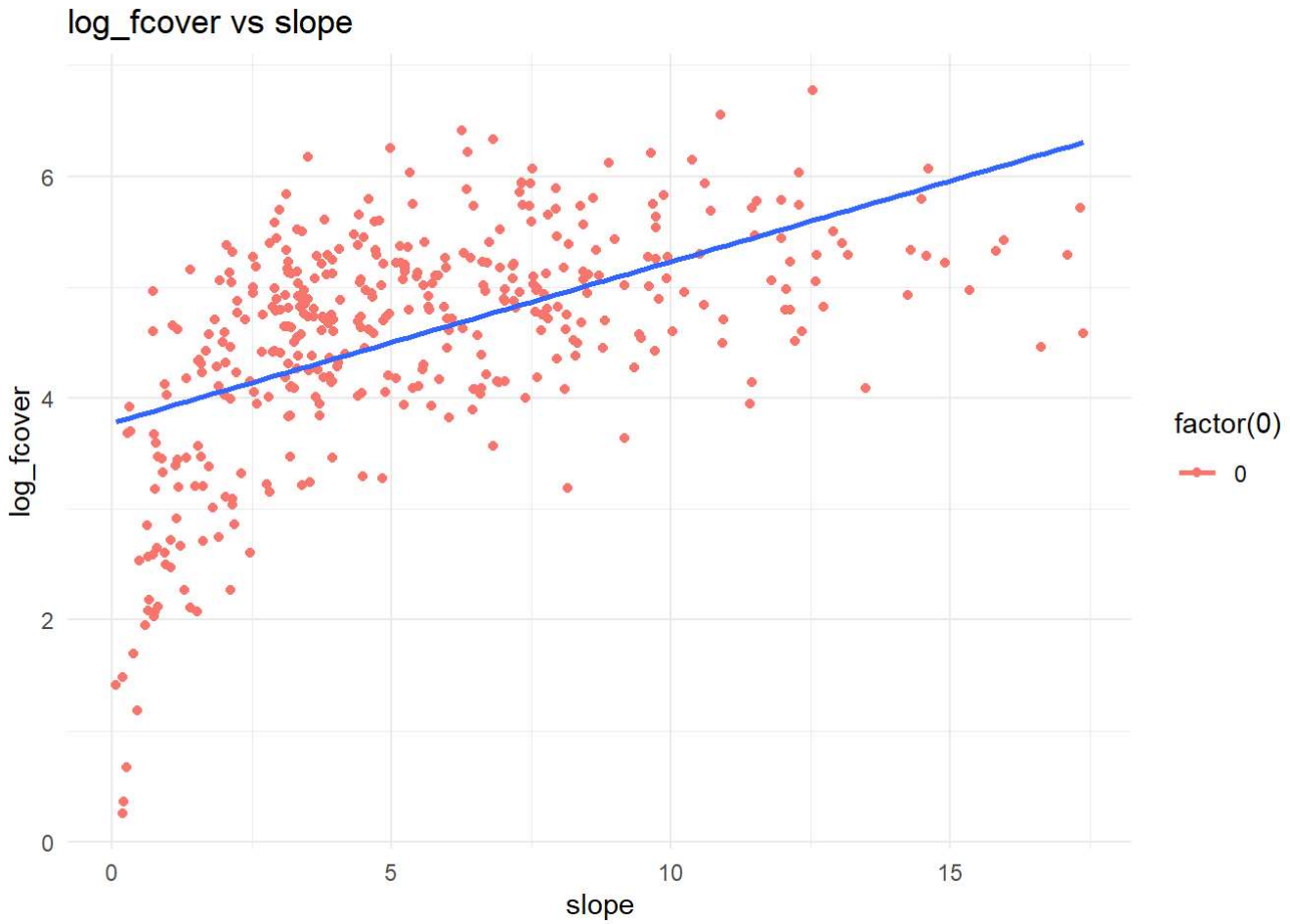

```
`geom_smooth()` using formula = 'y ~ x'
```



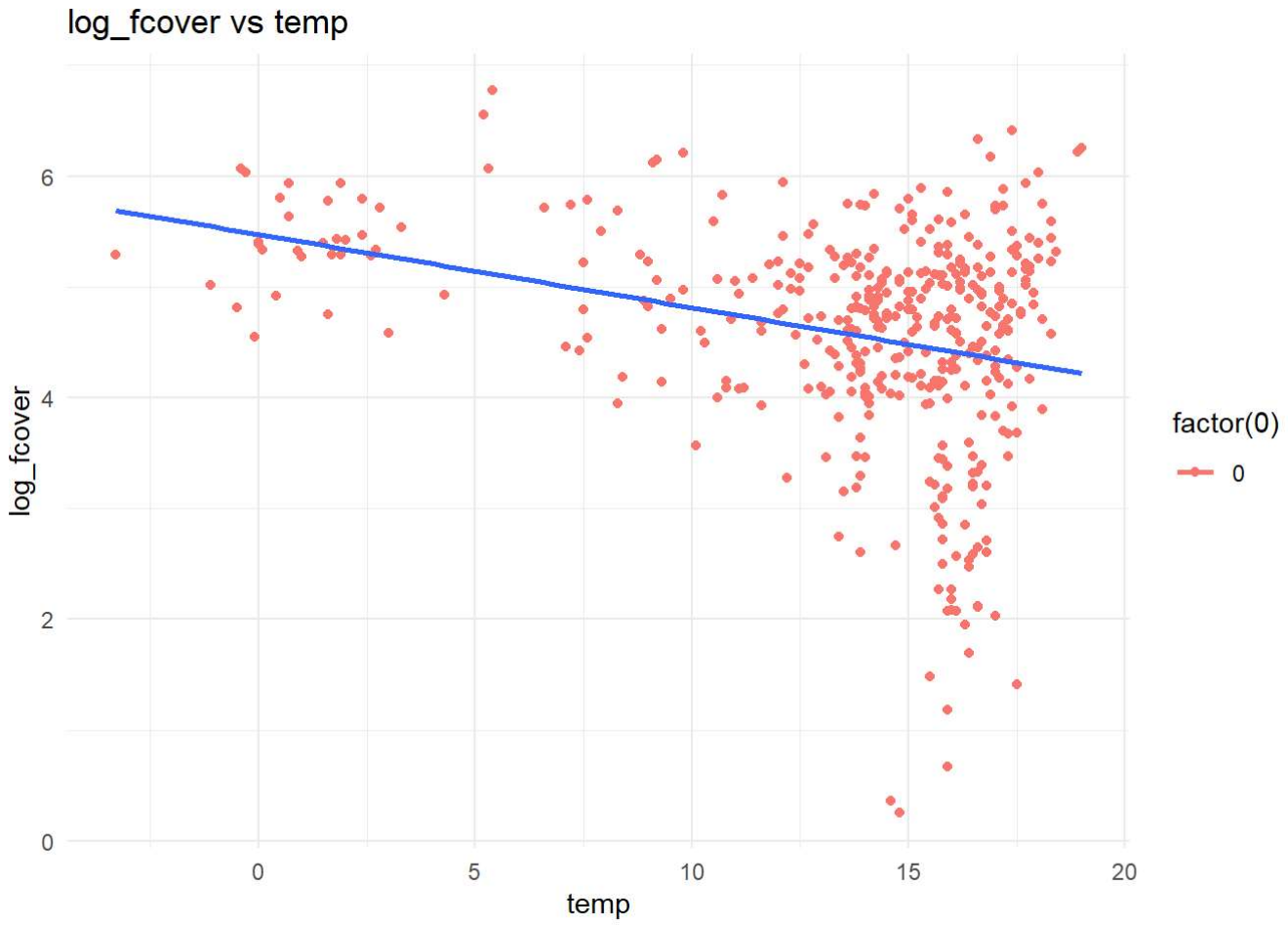

```
`geom_smooth()` using formula = 'y ~ x'
```



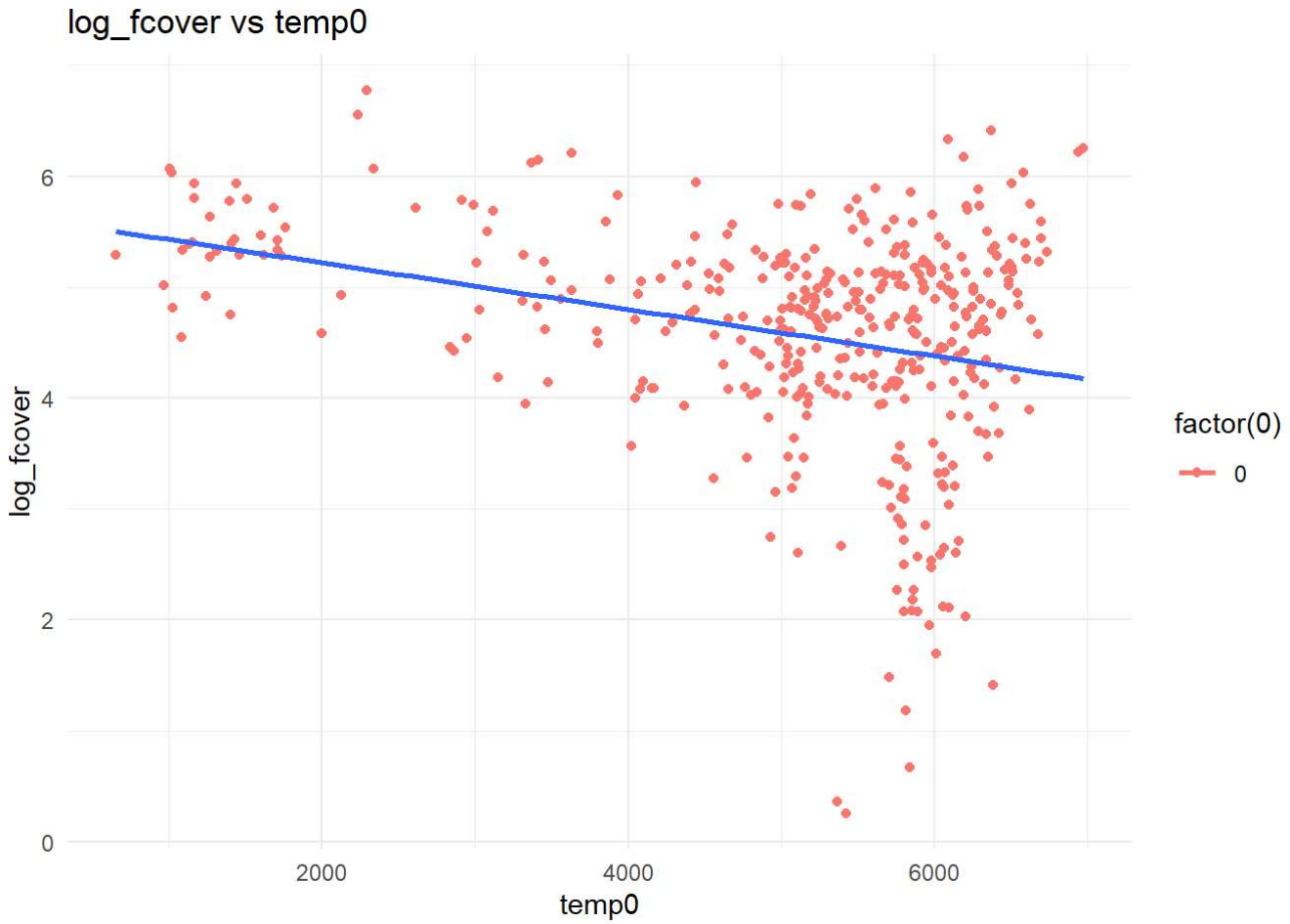

```
`geom_smooth()` using formula = 'y ~ x'
```



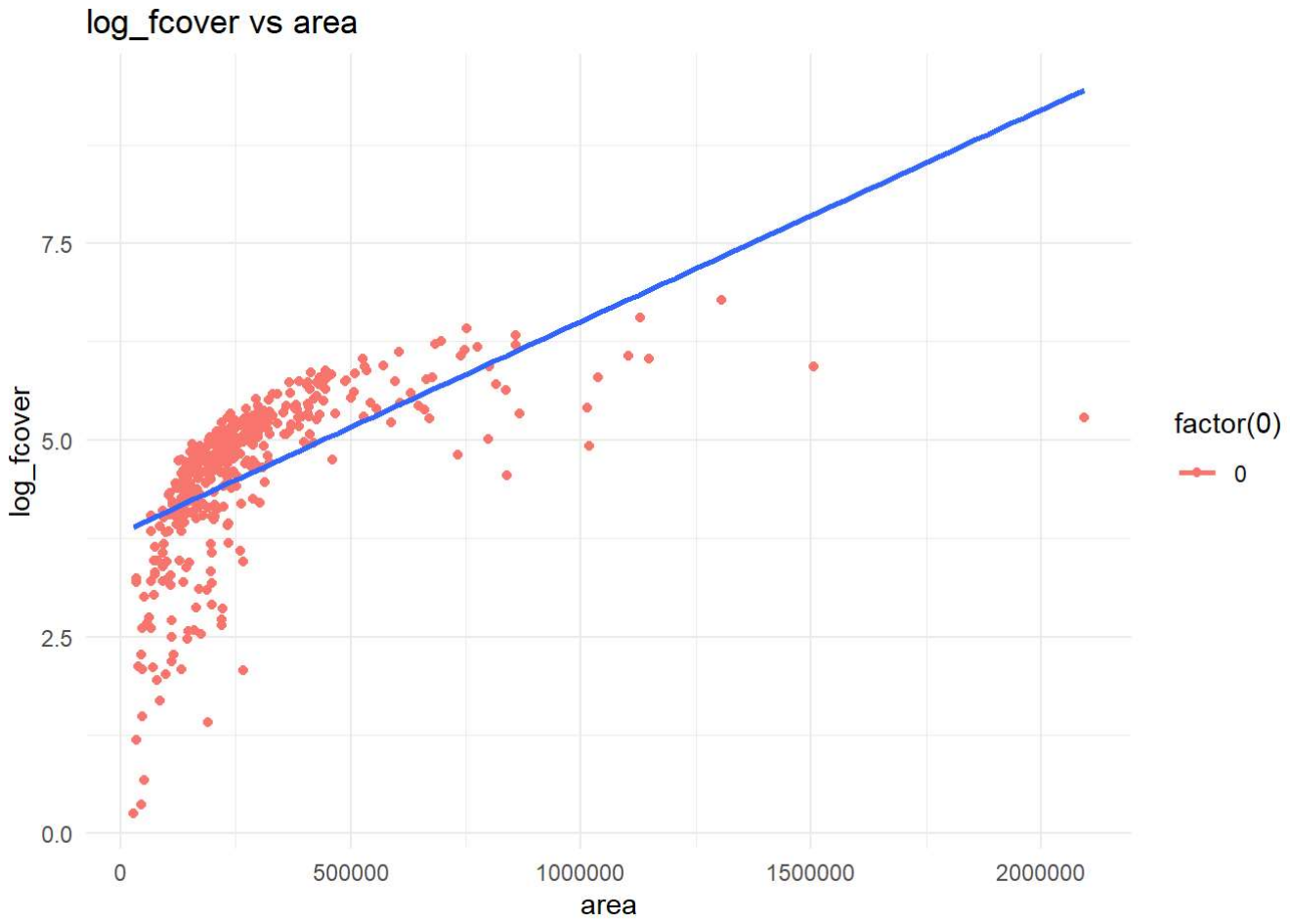

```
`geom_smooth()` using formula = 'y ~ x'
```



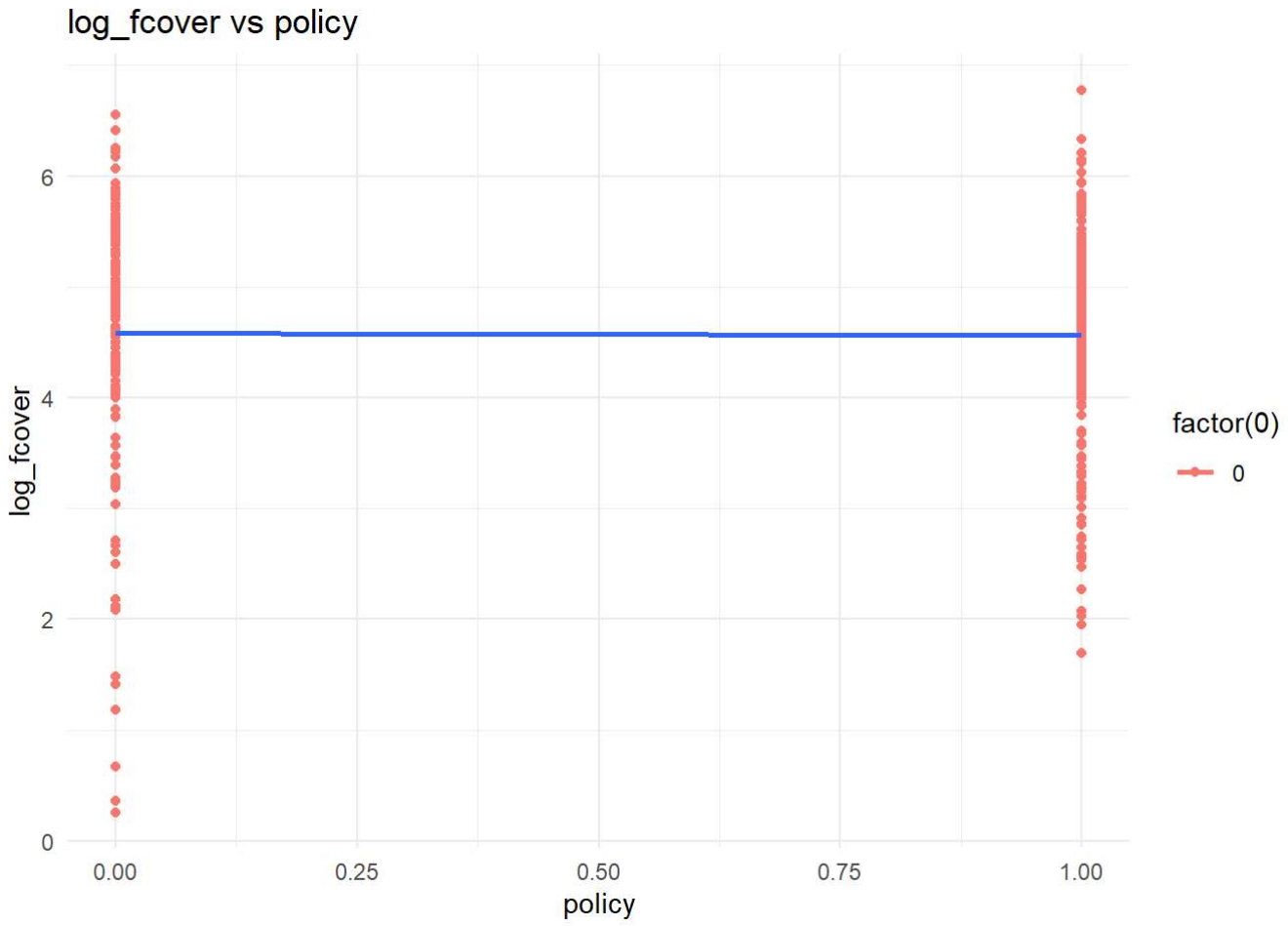

```
`geom_smooth()` using formula = 'y ~ x'
```



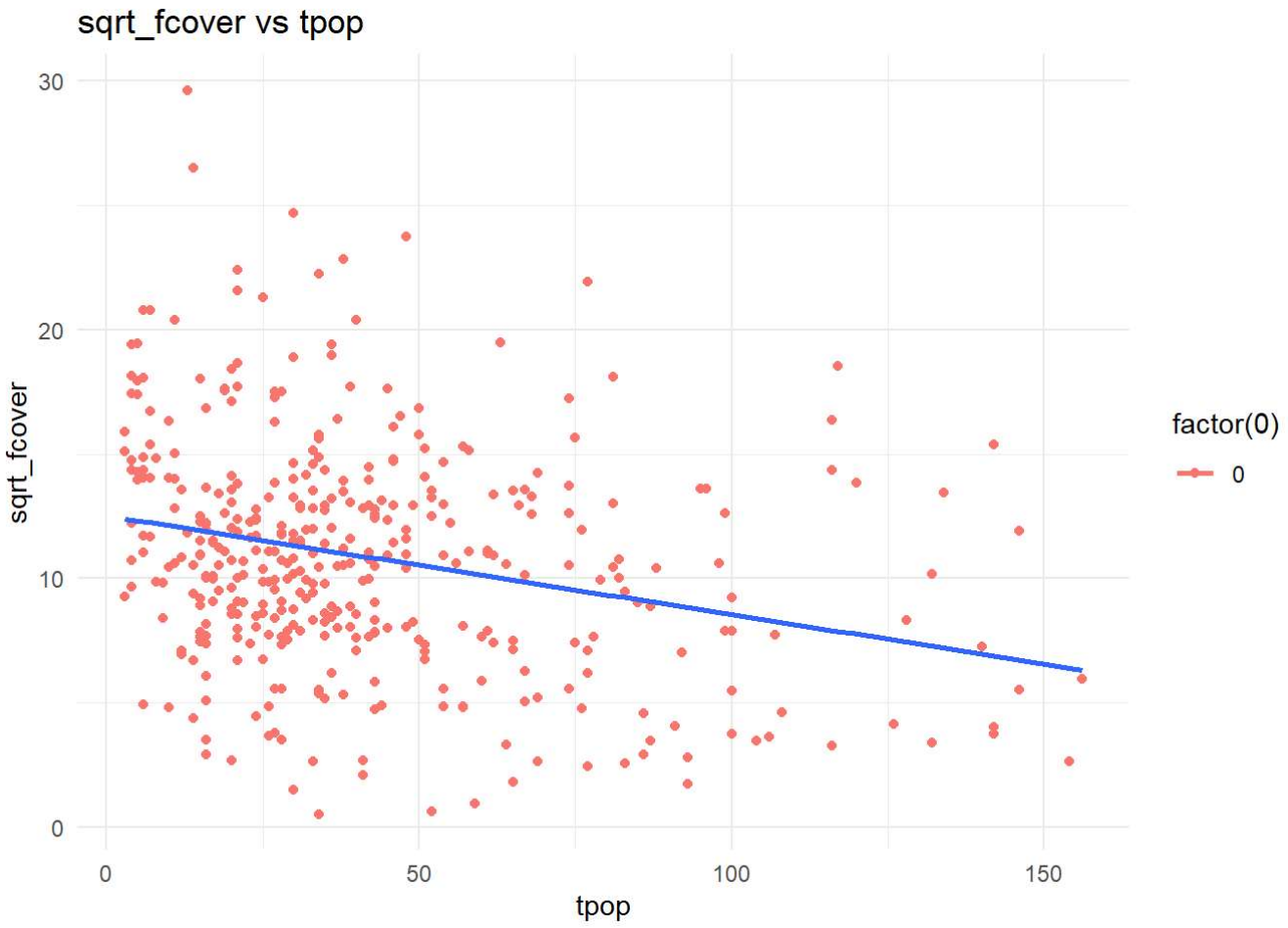

```
`geom_smooth()` using formula = 'y ~ x'
```



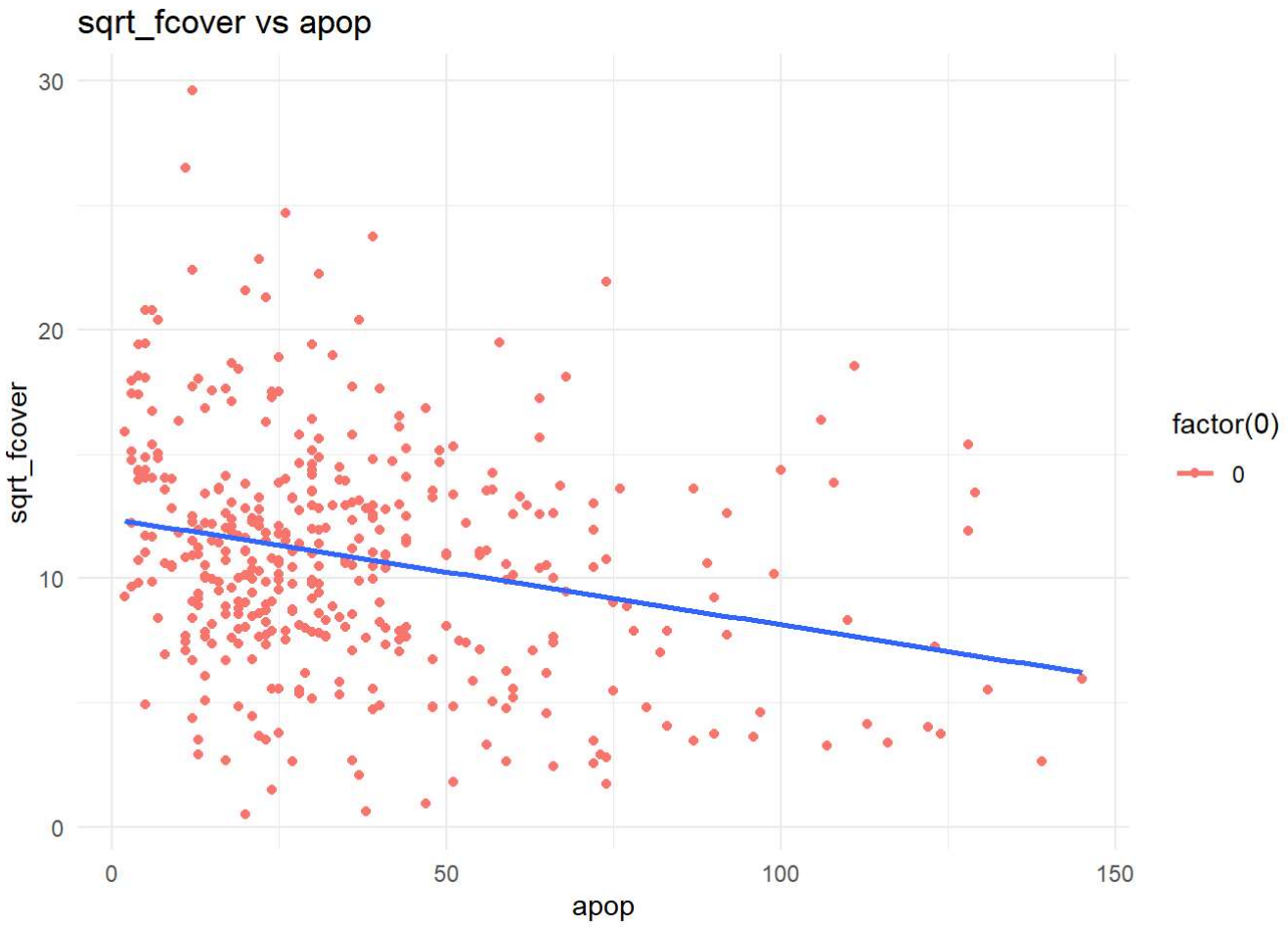

```
`geom_smooth()` using formula = 'y ~ x'
```



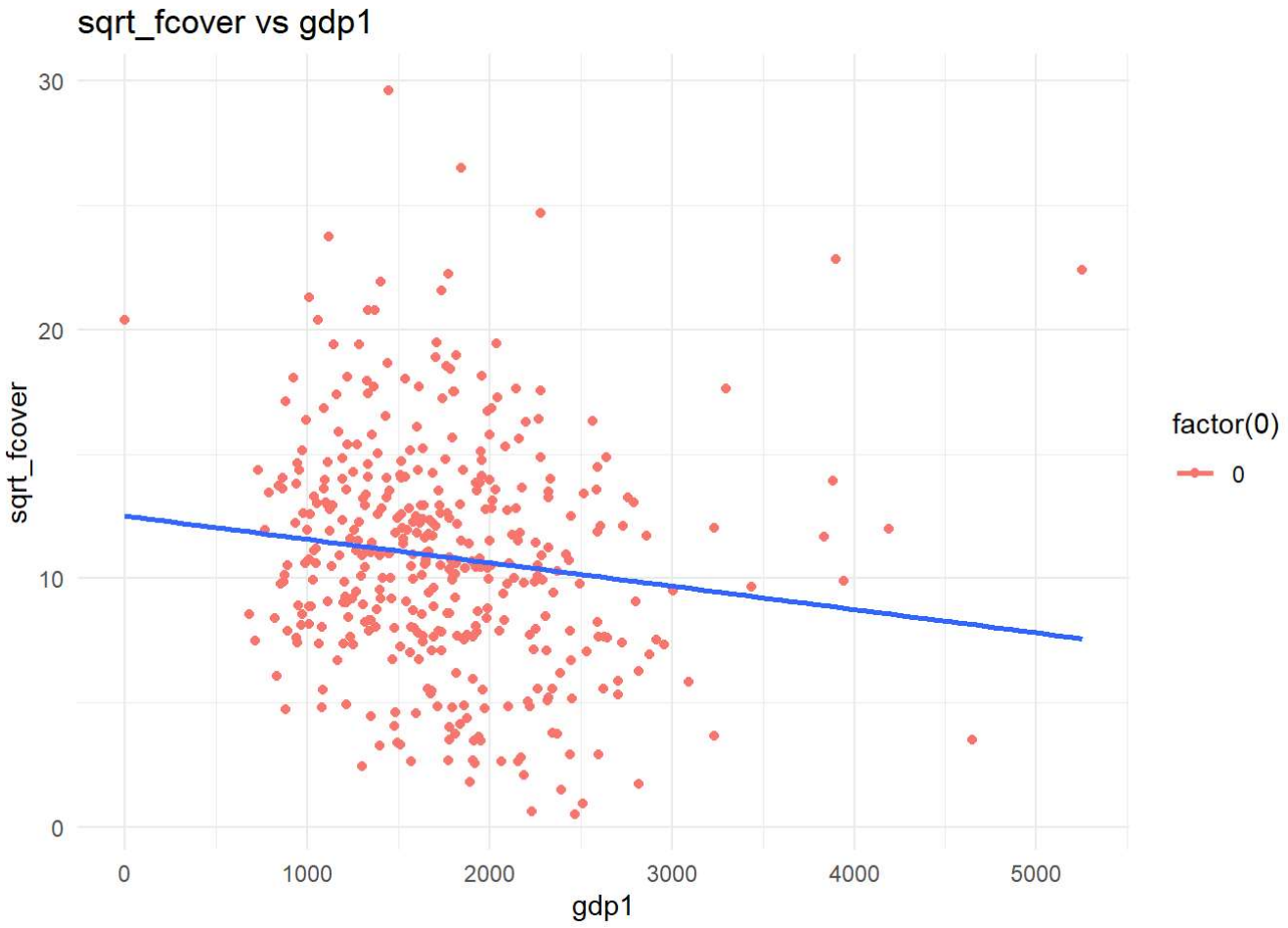

```
`geom_smooth()` using formula = 'y ~ x'
```



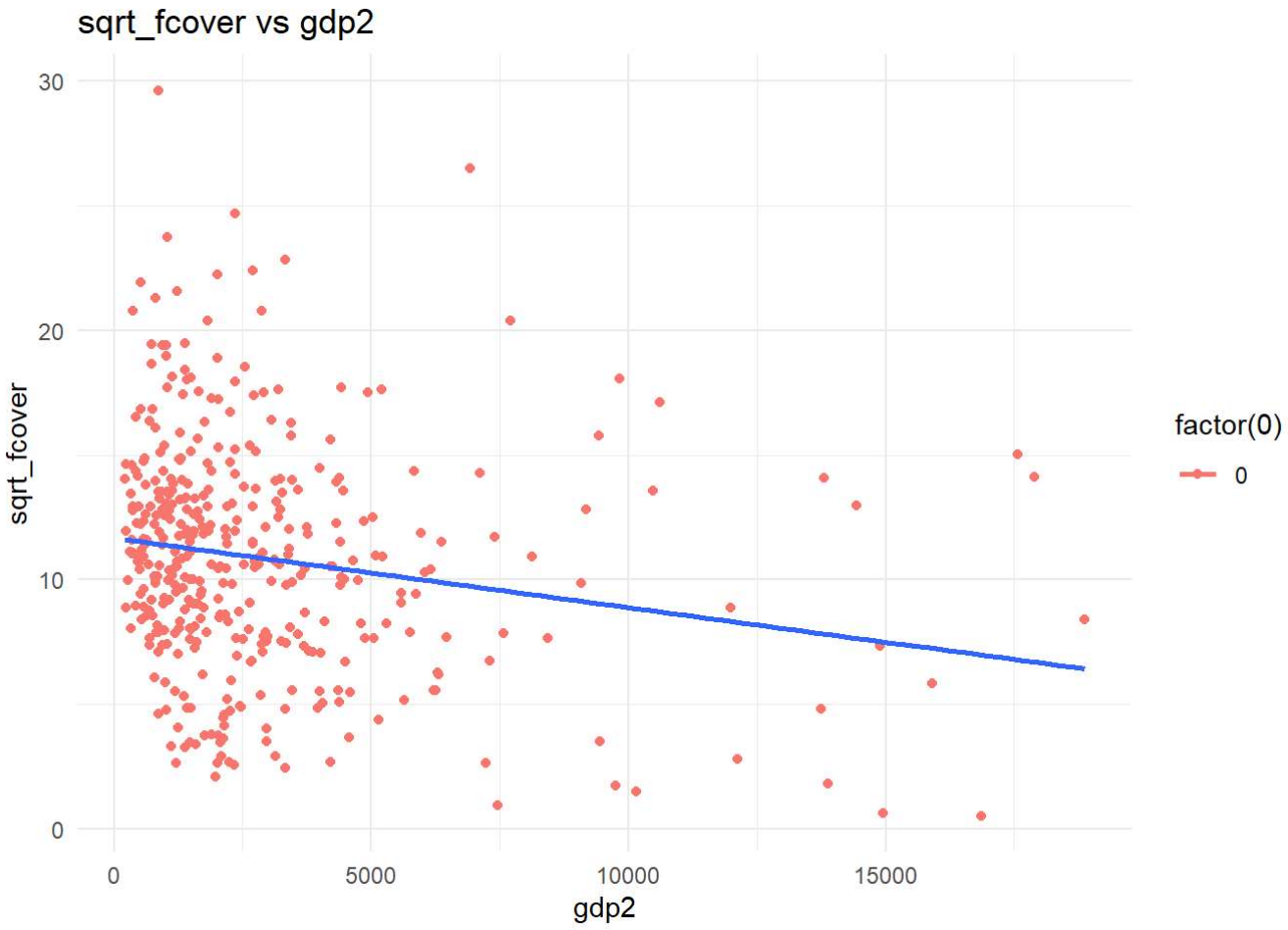

```
`geom_smooth()` using formula = 'y ~ x'
```



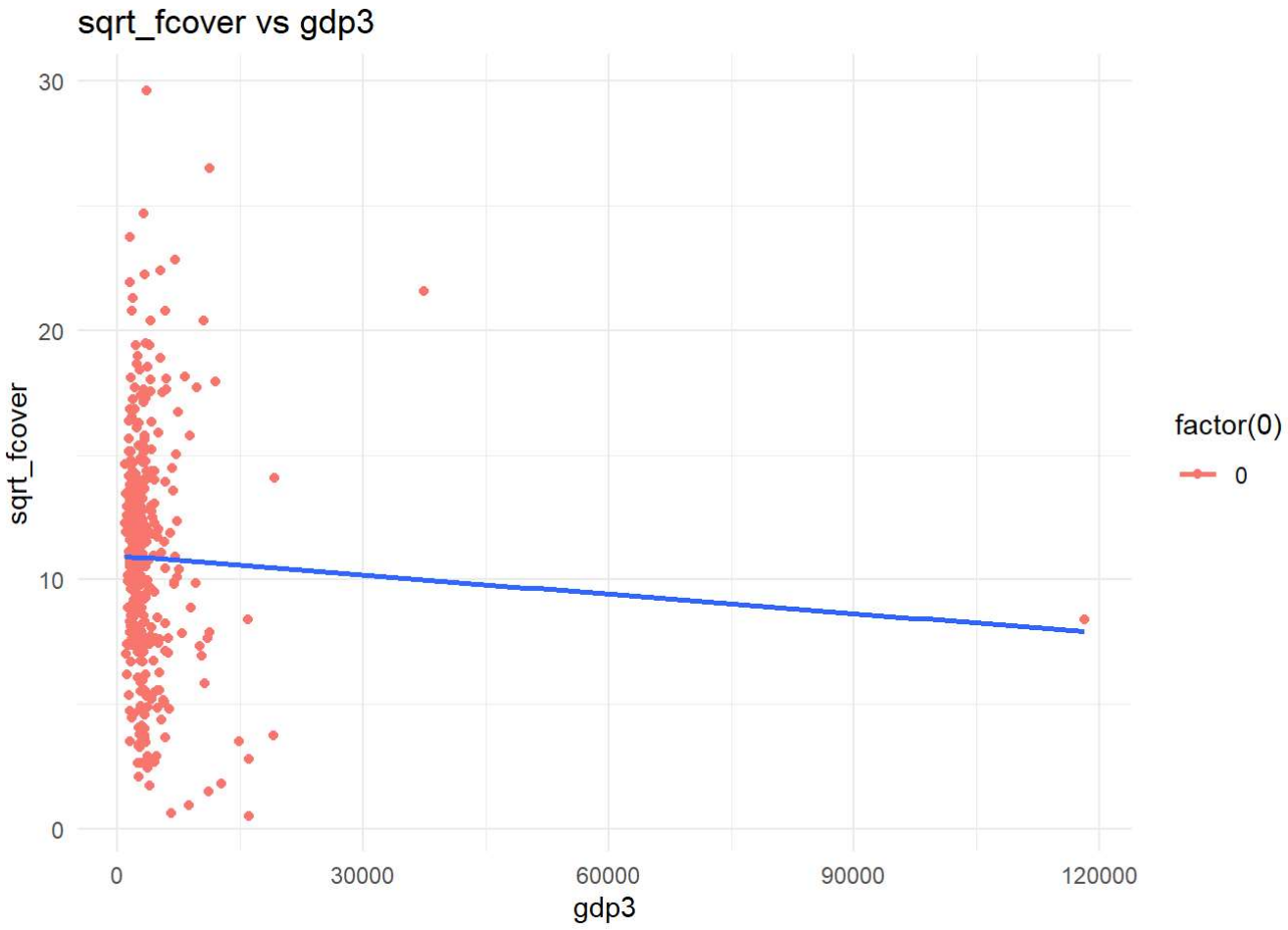

```
`geom_smooth()` using formula = 'y ~ x'
```



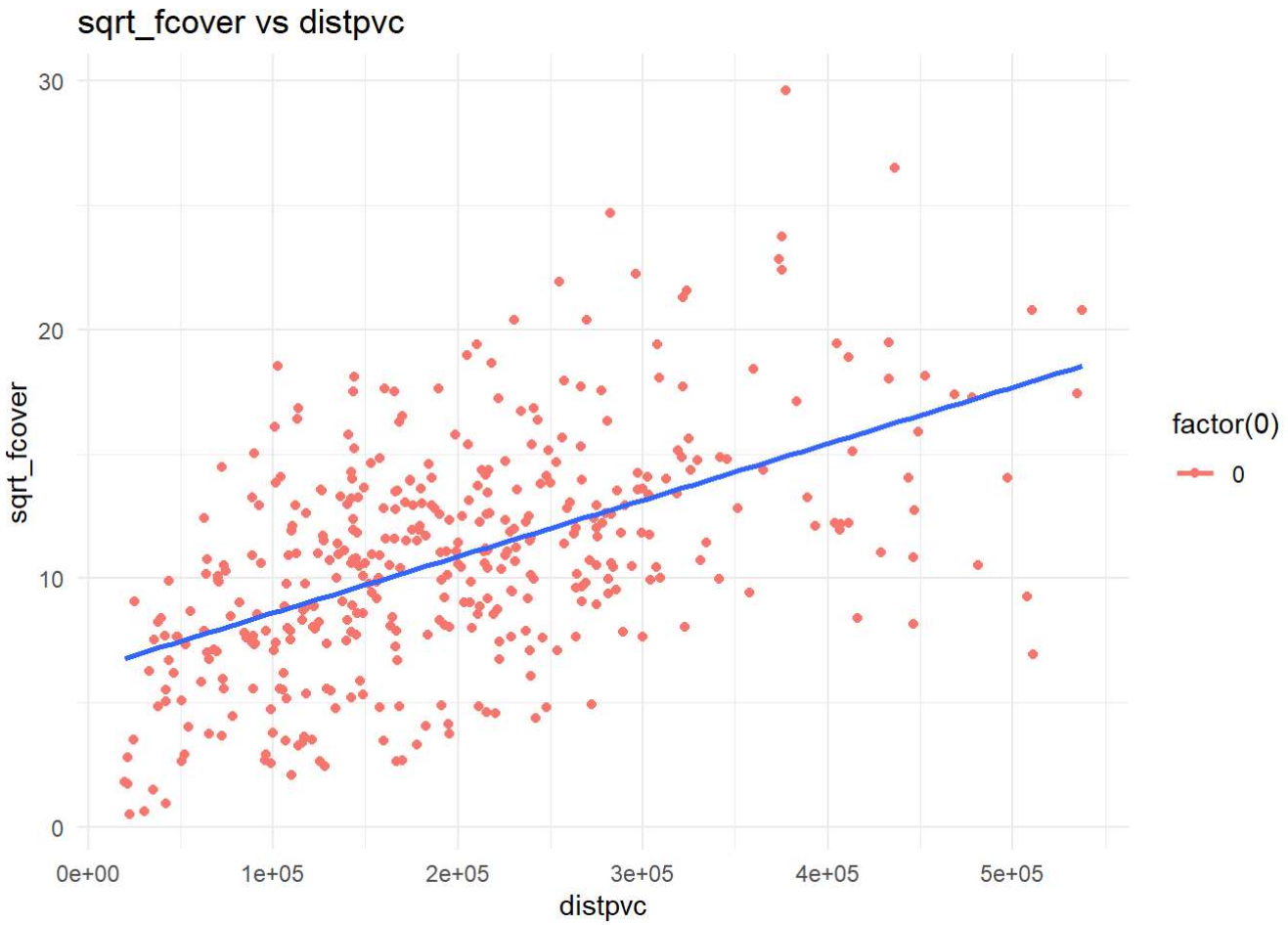

```
`geom_smooth()` using formula = 'y ~ x'
```



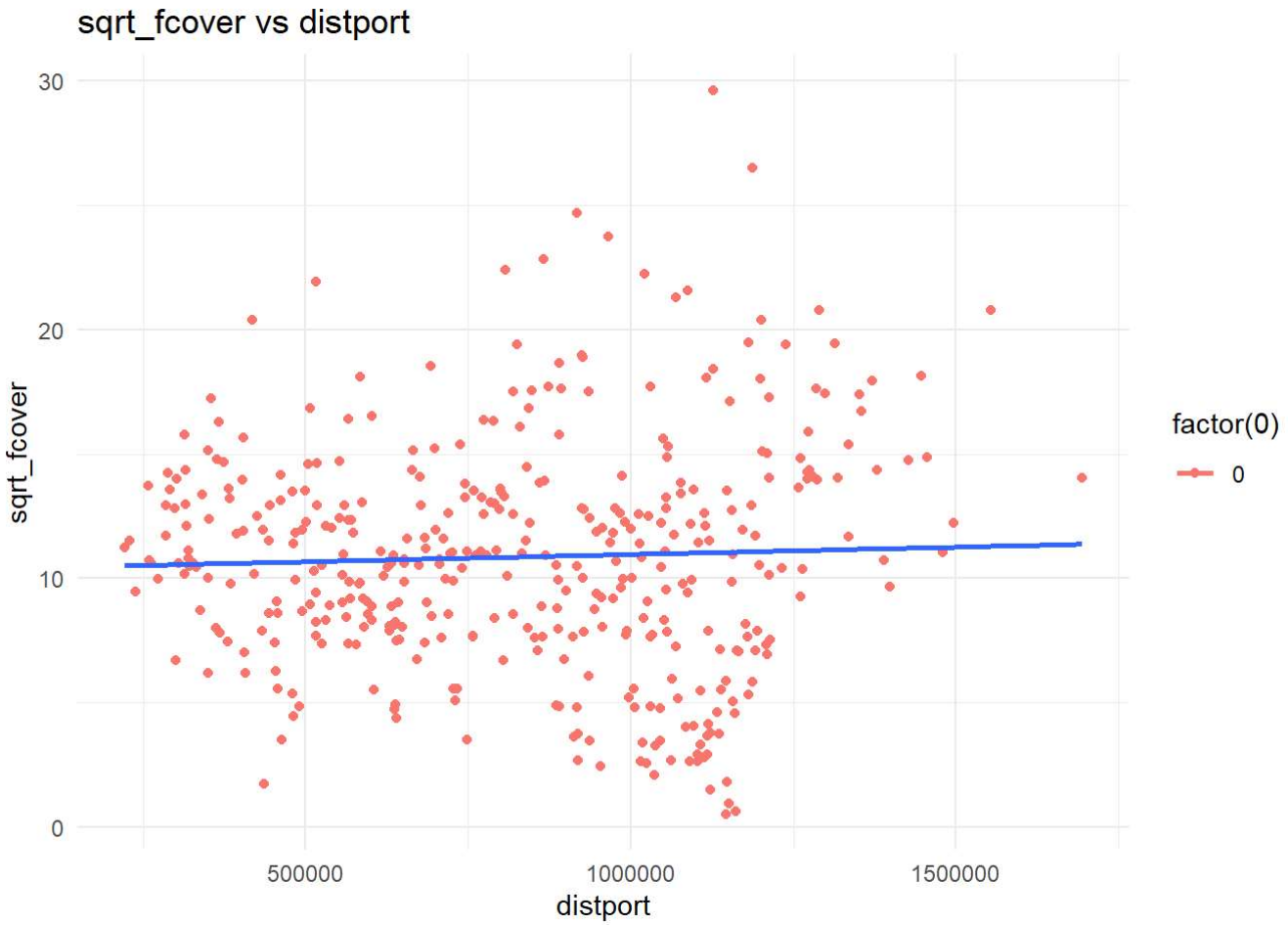

```
`geom_smooth()` using formula = 'y ~ x'
```



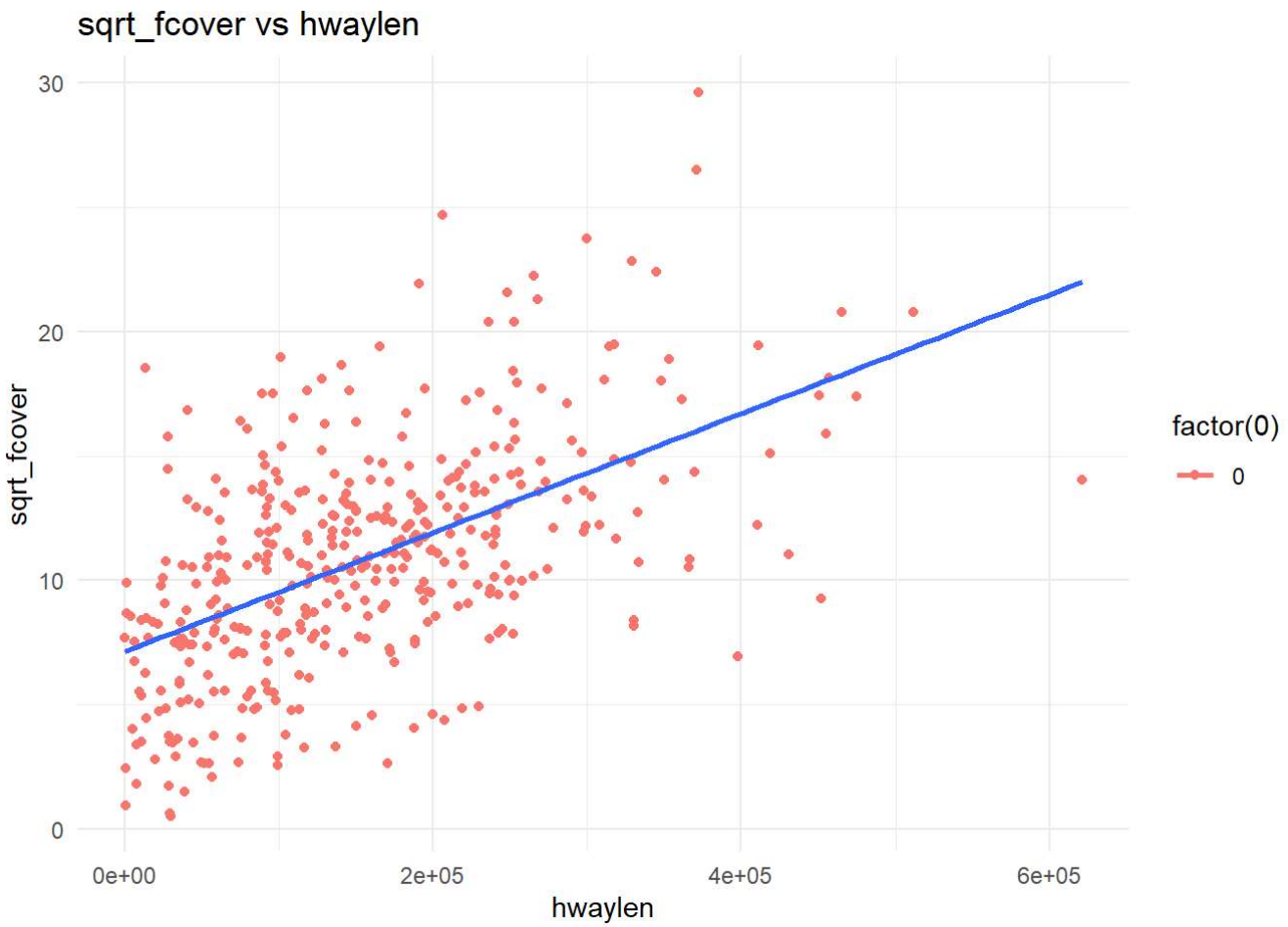

```
`geom_smooth()` using formula = 'y ~ x'
```



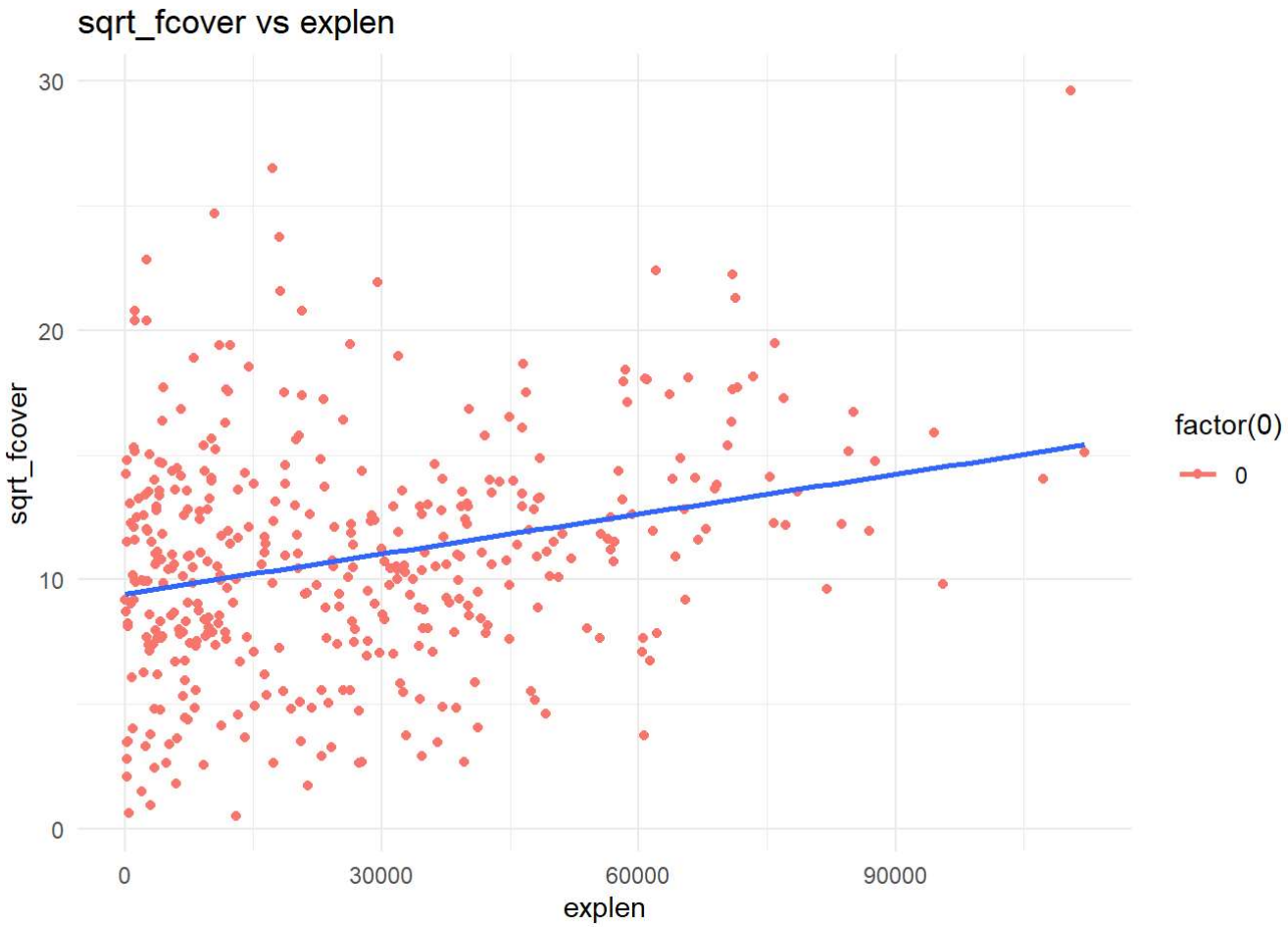

```
`geom_smooth()` using formula = 'y ~ x'
```



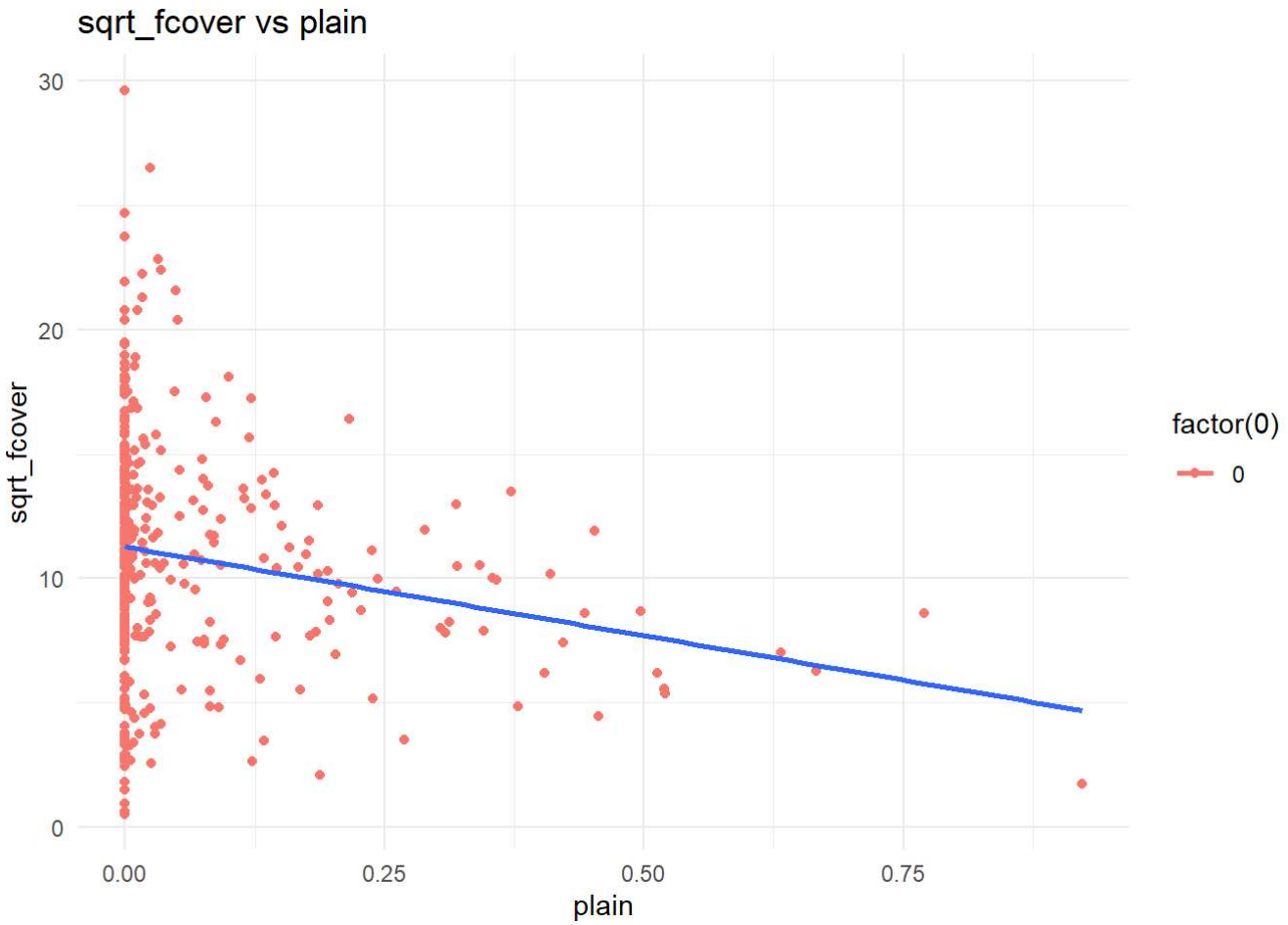

```
`geom_smooth()` using formula = 'y ~ x'
```



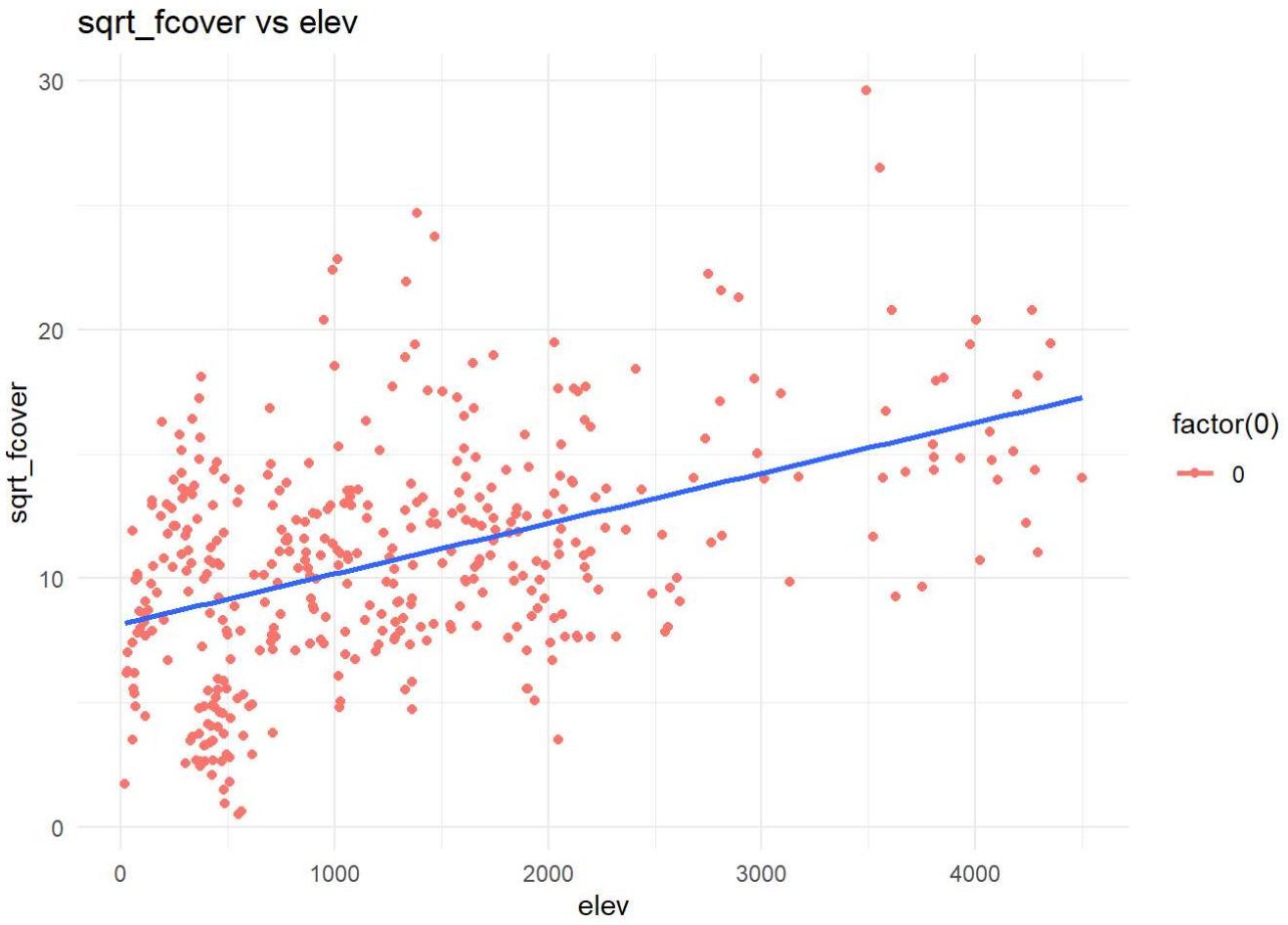

```
`geom_smooth()` using formula = 'y ~ x'
```



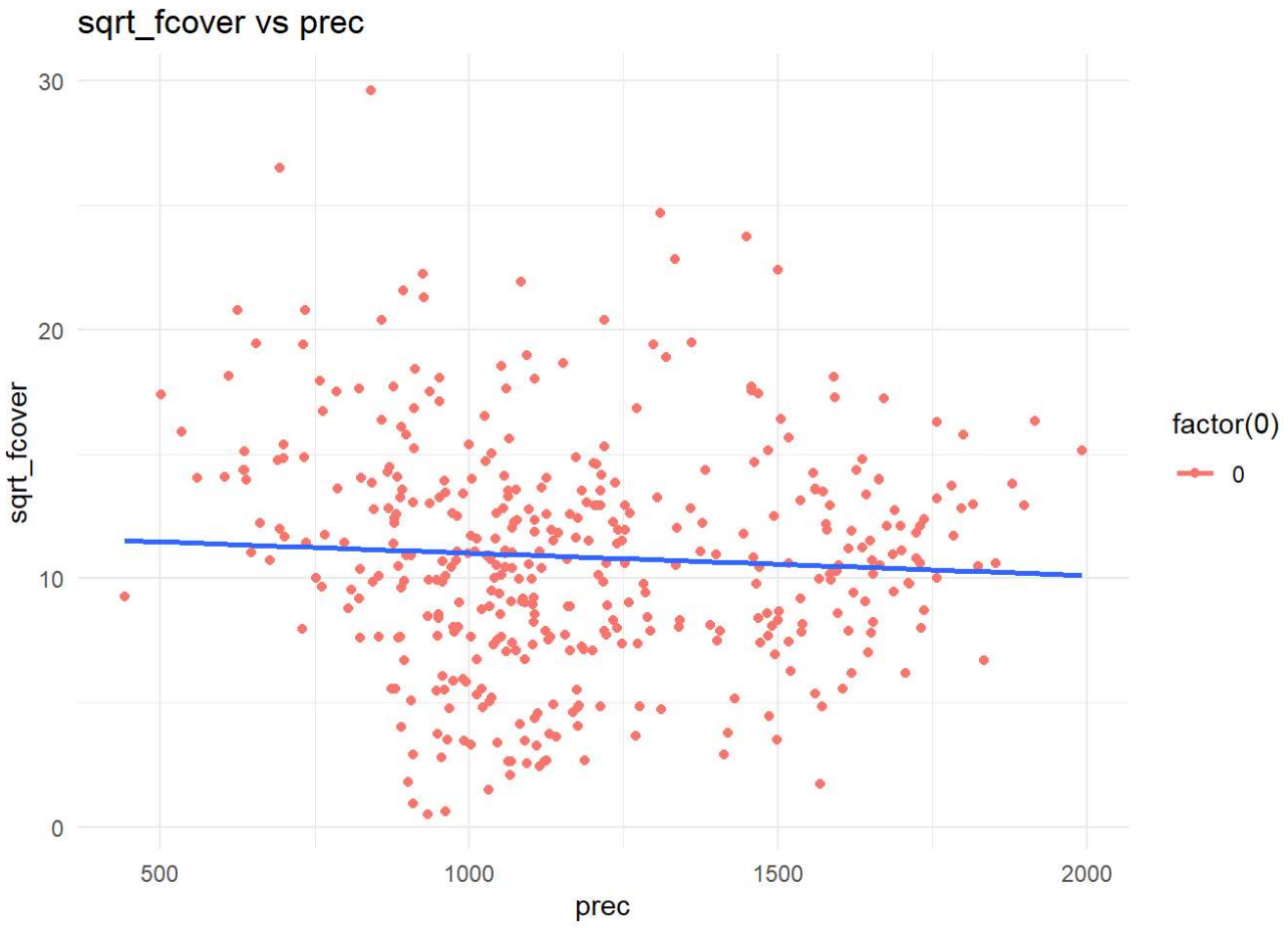

```
`geom_smooth()` using formula = 'y ~ x'
```



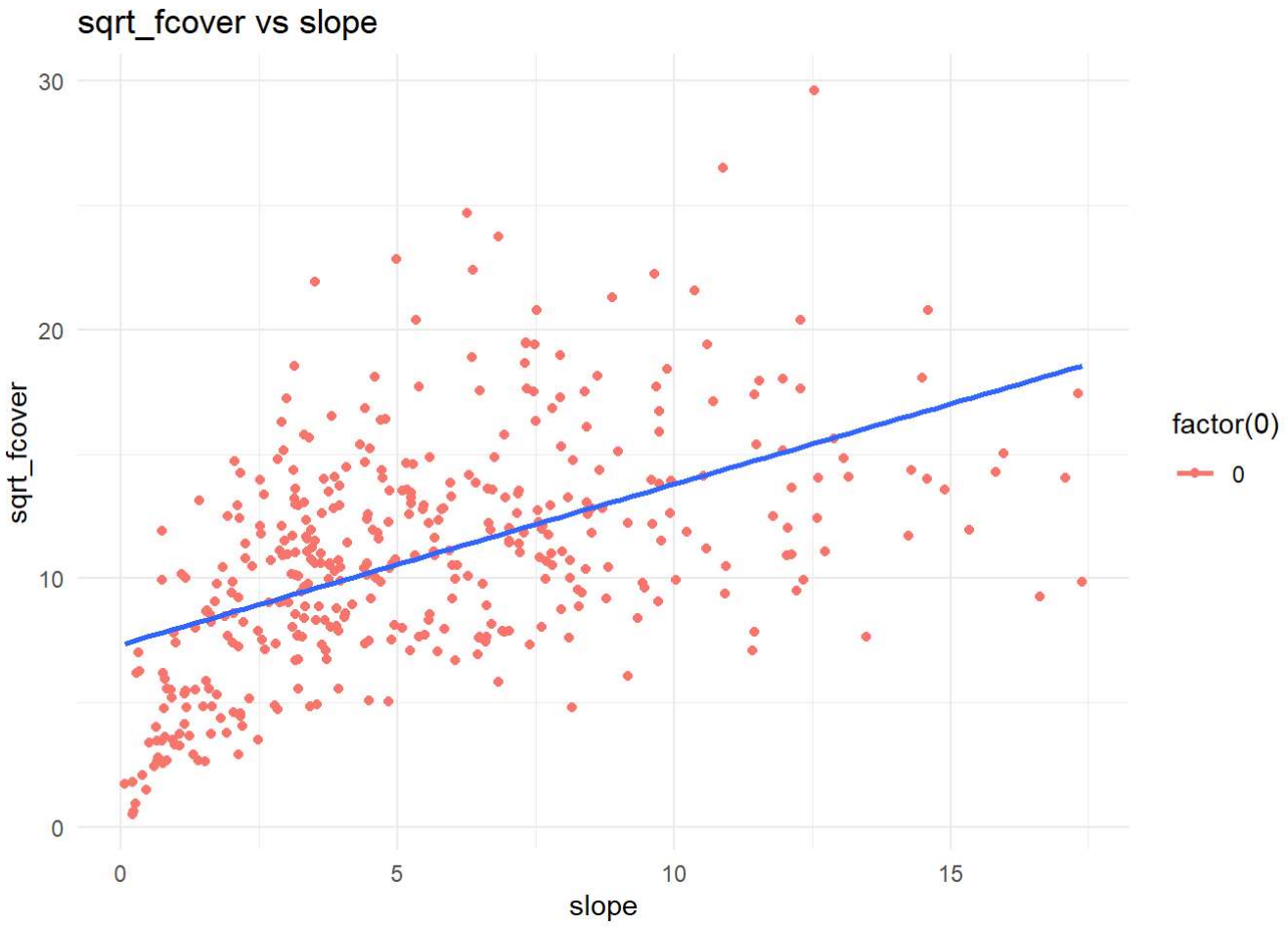

```
`geom_smooth()` using formula = 'y ~ x'
```



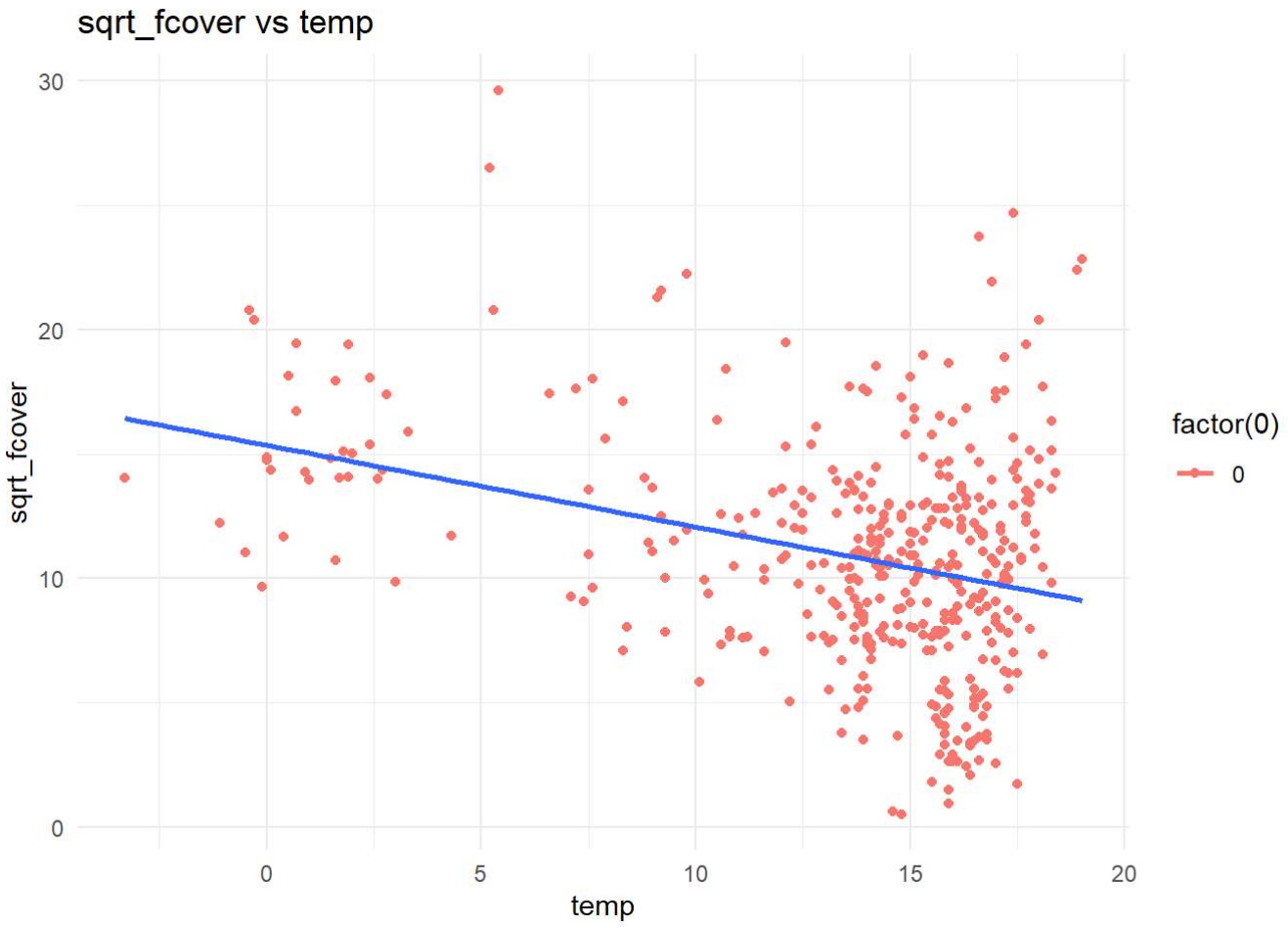

```
`geom_smooth()` using formula = 'y ~ x'
```



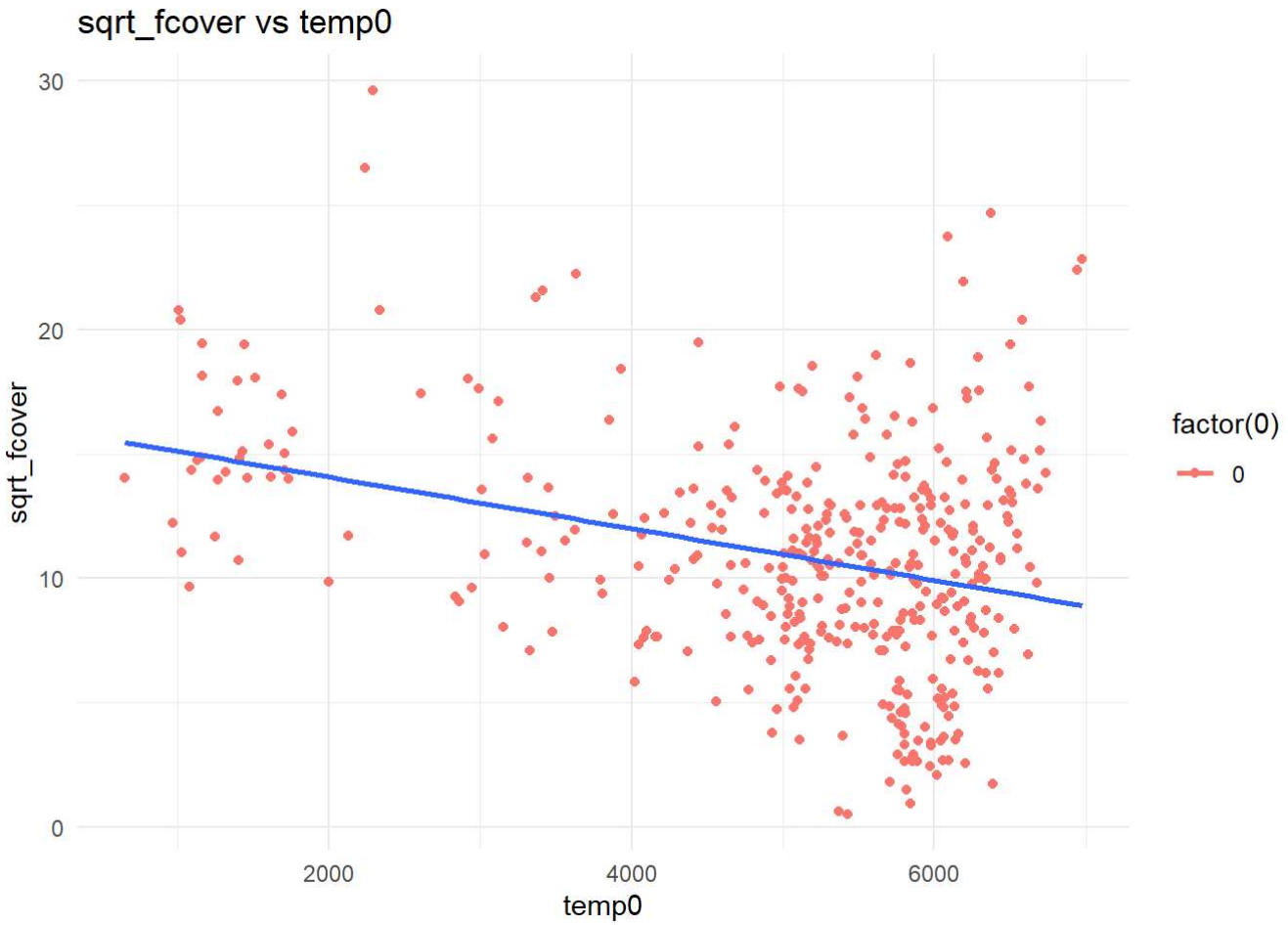

```
`geom_smooth()` using formula = 'y ~ x'
```



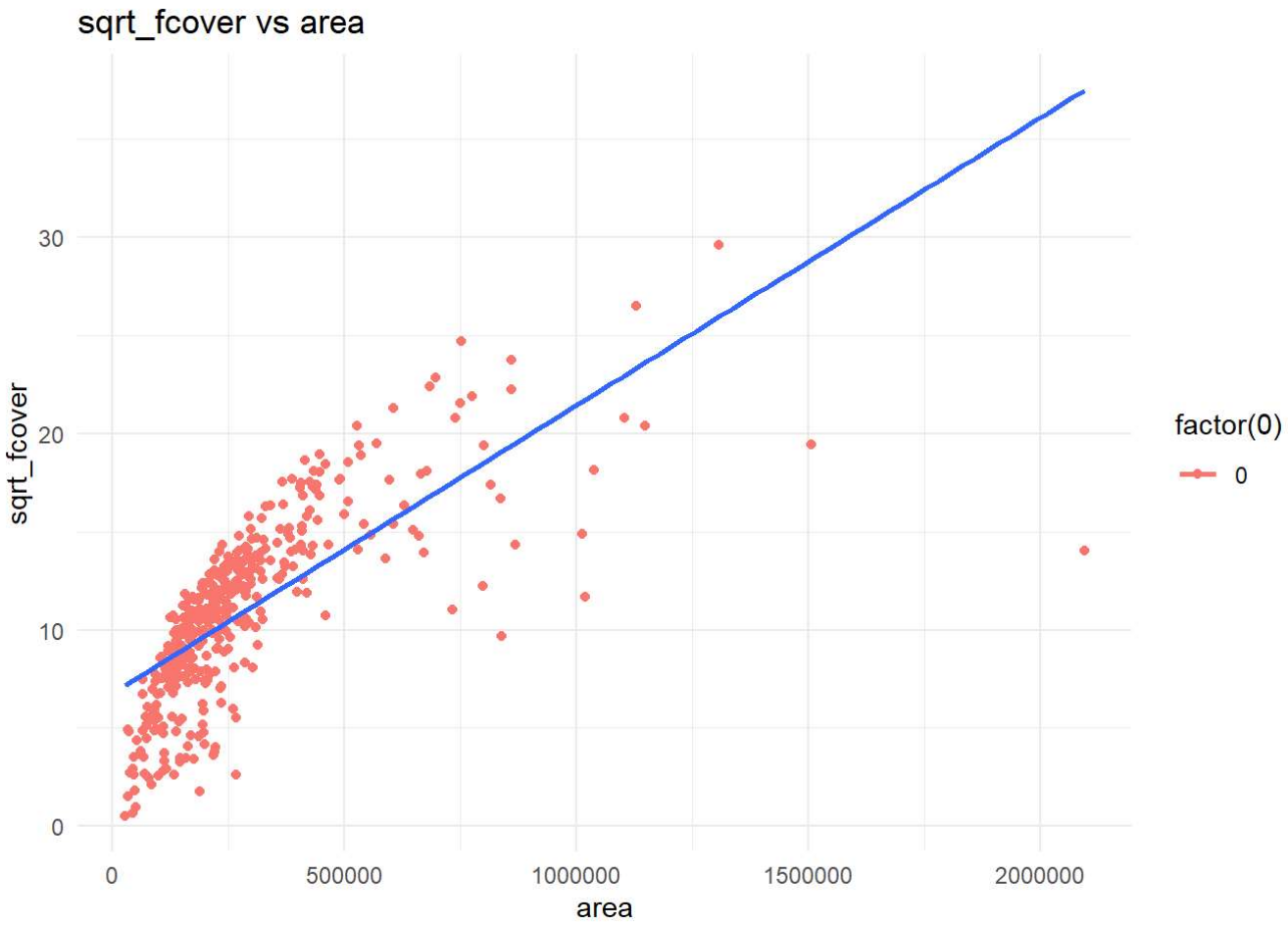

```
`geom_smooth()` using formula = 'y ~ x'
```



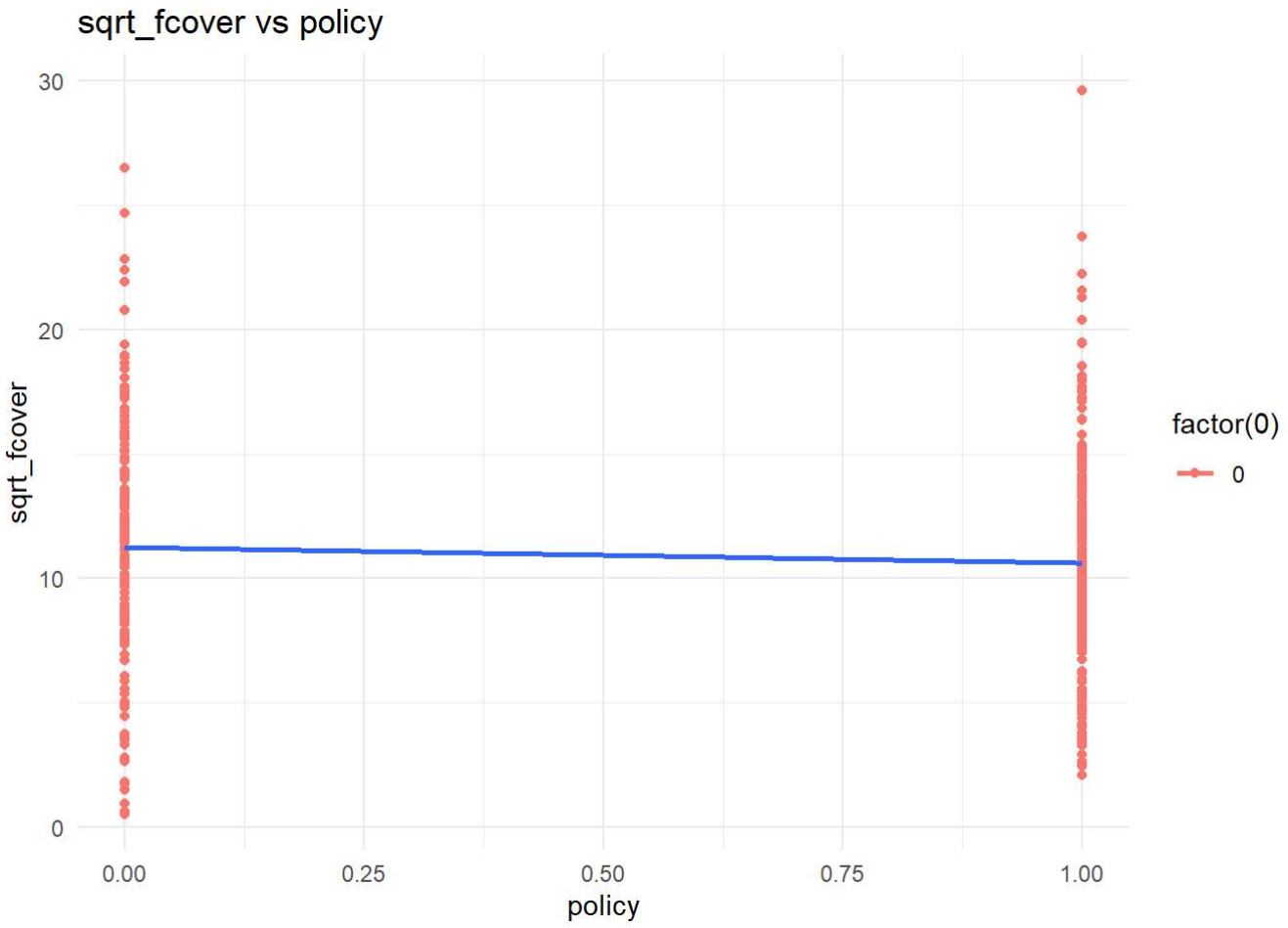

```
`geom_smooth()` using formula = 'y ~ x'
```




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




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



Fitting log $fcover$ and $\sqrt{fcover}$ Models

```
# Distribution plots for sqrt(fcover) and log(fcover)
data$sqrt_fcover <- sqrt(data$fcover)
data$log_fcover <- log1p(data$fcover)  # log1p to handle zeros


# MLR model fitting
# Using 'tpop', 'apop', 'gdp1' as predictors for demonstration

# Model with sqrt(fcover)
model_sqrt <- lm(sqrt_fcover ~ tpop + apop + gdp1 + gdp3 + distpvc +distport + hwaylen + explen +
summary(model_sqrt)
```

```
Call:
lm(formula = sqrt_fcover ~ tpop + apop + gdp1 + gdp3 + distpvc +
    distport + hwaylen + explen + plain + elev + prec + slope +
    temp + temp0 + area + policy, data = data)

Residuals:
     Min       1Q   Median       3Q      Max
```

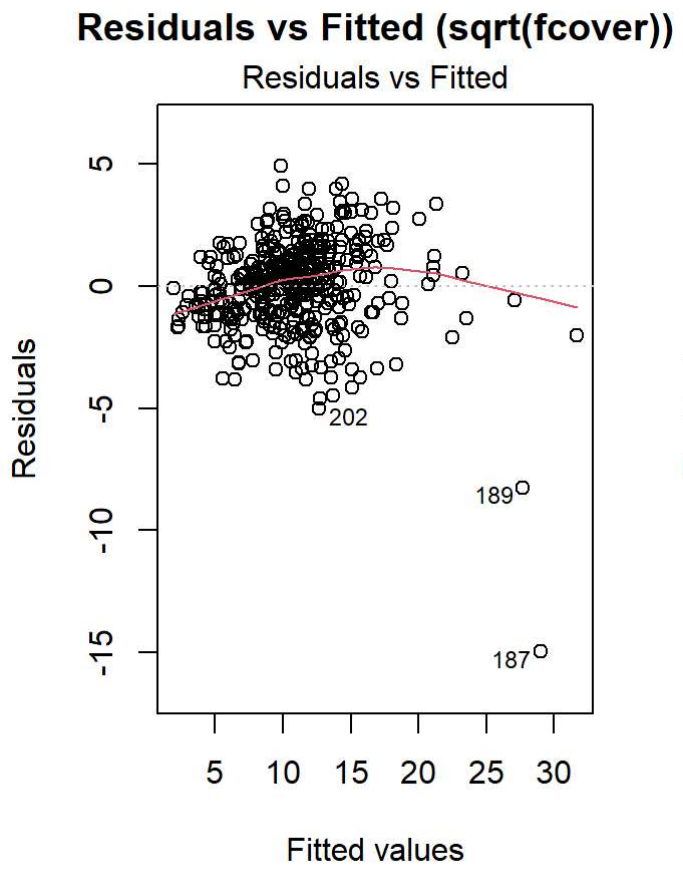
```
-14.9727  -1.0579   0.0994   1.1873   4.9110

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  6.910e+00  1.608e+00   4.297 2.19e-05 ***
tpop         3.010e-02  1.903e-02   1.582   0.1144
apop        -4.590e-02  2.124e-02  -2.161   0.0313 *
gdp1         1.555e-04  1.794e-04   0.867   0.3867
gdp3        -1.504e-05  1.529e-05  -0.984   0.3257
distpvc      3.177e-06  2.840e-06   1.119   0.2639
distport    -5.209e-06  5.755e-07  -9.051  < 2e-16 ***
hwaylen     -3.173e-06  2.918e-06  -1.087   0.2776
explen      -1.014e-06  4.645e-06  -0.218   0.8273
plain       -4.295e+00  1.011e+00  -4.249 2.69e-05 ***
elev        -7.096e-05  2.924e-04  -0.243   0.8084
prec         1.302e-03  5.970e-04   2.181   0.0298 *
slope        5.549e-01  4.768e-02  11.638  < 2e-16 ***
temp         2.484e+00  3.169e-01   7.837 4.48e-14 ***
temp0       -6.908e-03  9.909e-04  -6.972 1.36e-11 ***
area         1.954e-05  7.505e-07  26.030  < 2e-16 ***
policy      -2.050e-01  2.214e-01  -0.926   0.3550
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.919 on 388 degrees of freedom
Multiple R-squared:  0.8324,    Adjusted R-squared:  0.8255
F-statistic: 120.4 on 16 and 388 DF,  p-value: < 2.2e-16
```

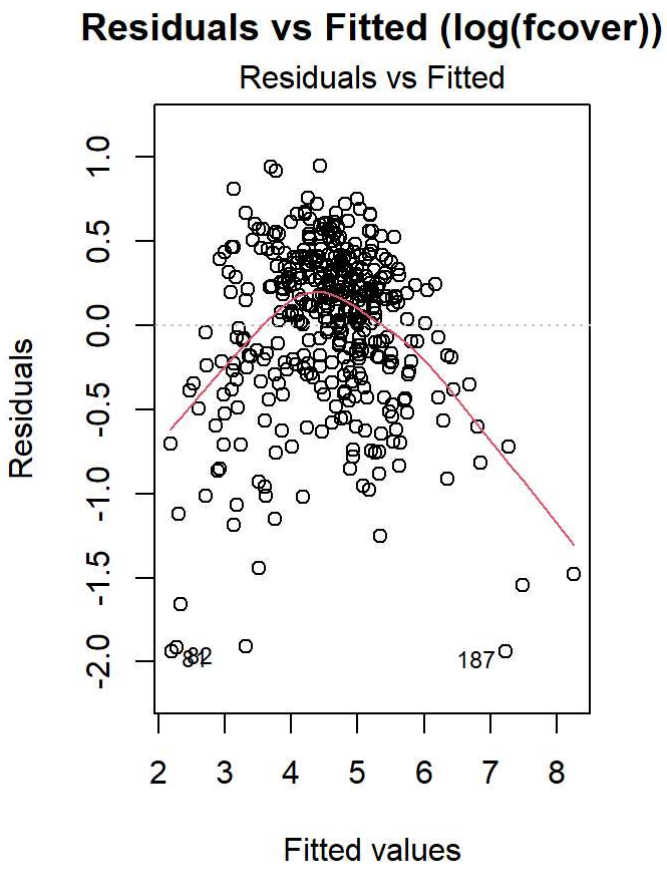
```
# Model with log(fcover)
model_log <- lm(log_fcover ~ tpop + apop + gdp1 + gdp3 + distpvc +distport + hwaylen + explen + p
summary(model_log)
```

```
Call:
lm(formula = log_fcover ~ tpop + apop + gdp1 + gdp3 + distpvc +
    distport + hwaylen + explen + plain + elev + prec + slope +
    temp + temp0 + area + policy, data = data)

Residuals:
    Min      1Q  Median      3Q     Max
-1.9409 -0.2490  0.1141  0.3329  0.9472

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.735e+00  4.197e-01   8.900  < 2e-16 ***
tpop         7.981e-03  4.965e-03   1.607 0.108803
apop        -9.392e-03  5.543e-03  -1.694 0.091016 .
gdp1         7.941e-05  4.683e-05   1.696 0.090757 .
gdp3        -8.214e-06  3.990e-06  -2.059 0.040194 *
```

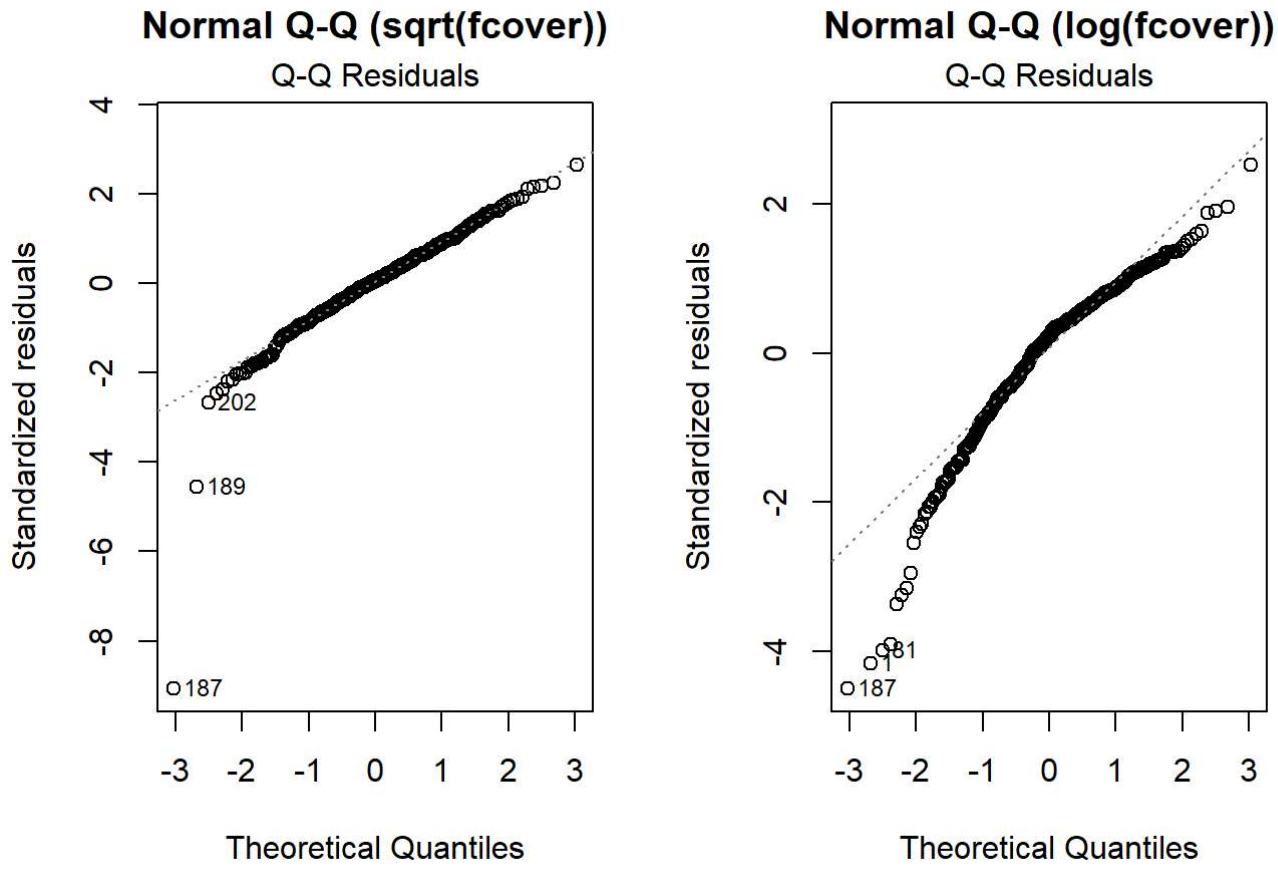
```
distpvc      2.183e-06  7.412e-07   2.946 0.003417 **
distport    -1.836e-06  1.502e-07 -12.224  < 2e-16 ***
hwaylen     -1.501e-06  7.617e-07  -1.971 0.049409 *
explen      -1.042e-08  1.212e-06  -0.009 0.993146
plain       -1.001e+00  2.638e-01  -3.793 0.000172 ***
elev         1.204e-04  7.630e-05   1.577 0.115504
prec         3.650e-04  1.558e-04   2.342 0.019667 *
slope        1.332e-01  1.244e-02  10.701  < 2e-16 ***
temp         2.775e-01  8.270e-02   3.355 0.000872 ***
temp0       -7.756e-04  2.586e-04  -2.999 0.002882 **
area         3.055e-06  1.959e-07  15.598  < 2e-16 ***
policy       1.396e-01  5.778e-02   2.415 0.016187 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.5009 on 388 degrees of freedom
Multiple R-squared:  0.7635,    Adjusted R-squared:  0.7538
F-statistic:  78.3 on 16 and 388 DF,  p-value: < 2.2e-16
```

```r
# Check Assumptions for sqrt and log
# Linearity and Homoscedasticity: Residuals vs Fitted plot
par(mfrow=c(1, 2))

plot(model_sqrt, which=1, main="Residuals vs Fitted (sqrt(fcover))")
plot(model_log, which=1, main="Residuals vs Fitted (log(fcover))")
```






```
# Normality of residuals: Q-Q plot
par(mfrow=c(1, 2))

plot(model_sqrt, which=2, main="Normal Q-Q (sqrt(fcover))")
plot(model_log, which=2, main="Normal Q-Q (log(fcover))")
```



Model Fit for $\sqrt{fcover}$ as response:

Key Metrics:

- **Multiple R-squared** R^2: 0.8324 - This means that approximately 83.24% of the variability in fcoverfcover is explained by the model.
- **Adjusted R-squared**: 0.8255 - This is the R-squared adjusted for the number of predictors in the model. It is still quite high, suggesting that most predictors are indeed useful.
- **F-statistic**: Very high and the p-value is extremely low, indicating that the predictors are statistically significant as a whole.

Significant Variables:

- apop, distport, plain, prec, slope, temp, temp0, area have low p-values (below 0.05), indicating they are statistically significant predictors for fcoverfcover.

Interpretation of Coefficients:

1. apop,: For a 1-unit increase in apop,, there is a -0.0459 decrease in $\sqrt{fcover}$.
2. distport: For a 1-unit increase in distport, there is a -0.0000052 decrease in $\sqrt{fcover}$
3. plain: For a 1-unit increase in plain, there is a -4.295 decrease in $\sqrt{fcover}$.

4. prec: For a 1-unit increase in prec, there is a 0.0013 increase in $\sqrt{fcover}$.
5. slope: For a 1-unit increase in slope, there is a 0.5549 increase in $\sqrt{fcover}$.
6. temp: For a 1-degree increase in temp, there is a 2.484 increase in $\sqrt{fcover}$
7. temp0: For a 1-unit increase in temp0, there is a -0.0069 decrease in $\sqrt{fcover}$
8. area: For a 1-unit increase in are, there is a 0.0000195 increase in $\sqrt{fcover}$

Model for $\log fcover$ as response

Key Metrics:

- **Multiple R-squared** R^2: 0.7635 - About 76.35% of the variability in log(fcover)log(fcover) is explained by this model.
- **Adjusted R-squared**: 0.7538 - This is also fairly high, suggesting that the predictors are useful.
- **F-statistic**: Again, very high and the p-value is extremely low, supporting that the predictors are statistically significant as a whole.

Significant Variables:

- gdp3, distpvc, distport, plain, prec, slope, temp, temp0, area, policy have low p-values (below 0.05), indicating they are statistically significant predictors for $\log fcover$

Interpretation of Coefficients

1. gdp3: For a 1-unit increase in gdp3, there is a -0.0000082 decrease in $\log fcover$.
2. distpvc: For a 1-unit increase in distpvc, there is a 0.0000022 increase in $\log fcover$.
3. distport: For a 1-unit increase in distport, there is a -0.0000018 decrease in $\log fcover$
4. plain For a 1-unit increase in plain, there is a -1.001 decrease in $\log fcover$
5. prec: For a 1-unit increase in prec, there is a 0.000365 increase in $\log fcover$.
6. slope: For a 1-unit increase in slope, there is a 0.1332 increase in $\log fcover$.
7. temp: For a 1-degree increase in temp, there is a 0.2775 increase in $\log fcover$.
8. temp0: For a 1-unit increase in temp0, there is a -0.0007756 decrease in $\log fcover$.
9. area: For a 1-unit increase in area, there is a 0.0000031 increase in $\log fcover$.
10. policy: For a 1-unit increase in policy, there is a 0.1396 increase in $\log fcover$.

General Observations:

- Both models have high R-squared and Adjusted R-squared values, suggesting they fit the data well.

- Some predictors are significant in both models (distport, plain, prec, slope, temp, temp0, area), suggesting they are consistently important in predicting forest cover.

PREFERENCE : SQRT TRANSFORMATION

Based on the provided R output and criteria, I would prefer to use the model with sqrt(fcover) as the dependent variable.

The primary reason is the better model fit: The Multiple R-squared value for the sqrt(fcover) model is 0.8324, compared to 0.7635 for the log(fcover) model. A higher R-squared value generally indicates that the model explains a greater proportion of the variance in the dependent variable. Therefore, if the goal is to achieve the best predictive performance, the sqrt(fcover) model seems to be the better choice.

The histogram of $\sqrt{fcover}$ model is normally distributed when compared to $\log fcover$ which is positively skewed violating the normality assumption of mulitple linear regression.

Moreover, when observed the residual vs fitted plots of the two, the residual vs fitted plot pf $\log fcover$ seems to contain a drastic curvature property thus indicating non-linearity. Although there is a slight curve in the $\sqrt{fcover}$ model, it does not seem to be as extreme as log, hence improving the linear regression model.

```
library(ggplot2)
library(gridExtra)
```

```
Attaching package: 'gridExtra'

The following object is masked from 'package:dplyr':

    combine
```

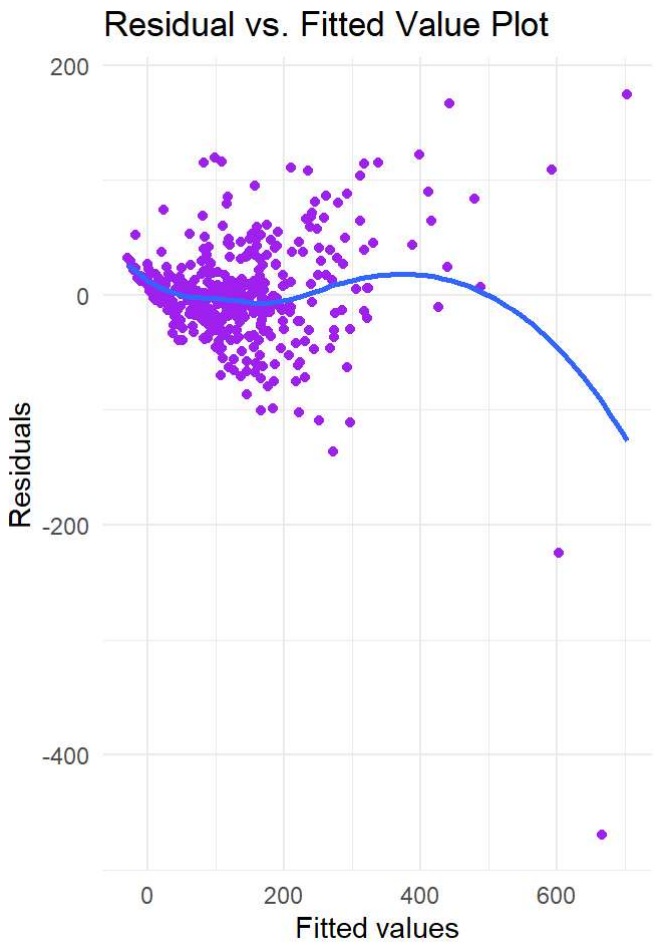
```
initial_model <- lm(fcover ~ tpop + apop + gdp1 + gdp3 + distpvc +distport + hwaylen + explen + p

summary(initial_model)
```

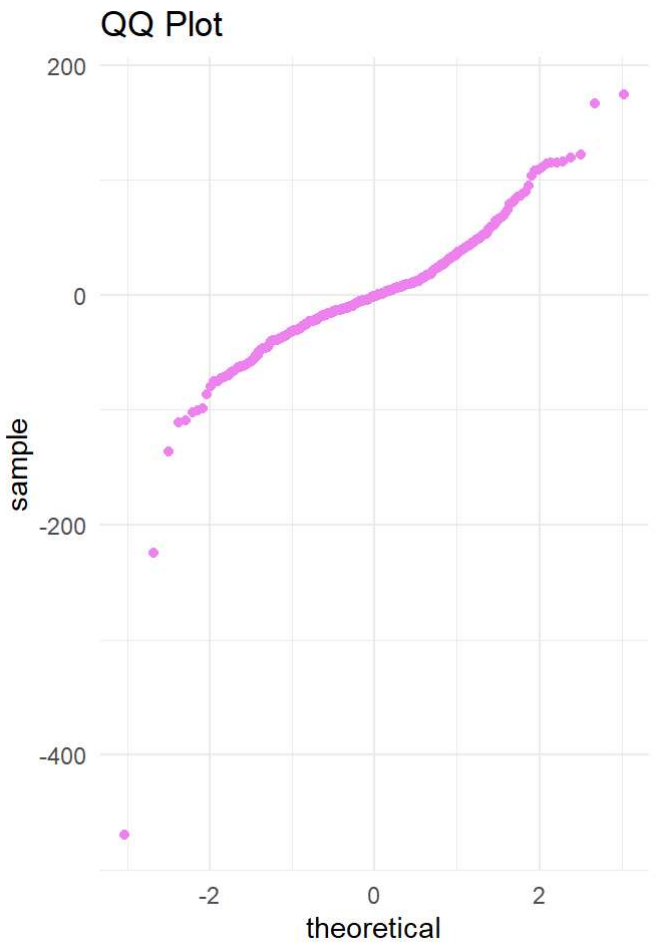
```
Call:
lm(formula = fcover ~ tpop + apop + gdp1 + gdp3 + distpvc + distport + 
    hwaylen + explen + plain + elev + prec + slope + temp + temp0 + 
    area + policy, data = data)

Residuals:
    Min      1Q  Median      3Q     Max 
-469.03  -20.30   -1.33   18.13  174.61 

Coefficients:
```

```
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  5.315e+01  4.128e+01   1.288 0.198624
tpop         4.040e-01  4.884e-01   0.827 0.408672
apop        -9.916e-01  5.452e-01  -1.819 0.069712 .
gdp1        -1.580e-04  4.606e-03  -0.034 0.972652
gdp3         1.884e-04  3.924e-04   0.480 0.631386
distpvc     -3.670e-05  7.290e-05  -0.503 0.614965
distport    -5.187e-05  1.477e-05  -3.511 0.000499 ***
hwaylen     -2.505e-05  7.491e-05  -0.334 0.738239
explen      -4.051e-05  1.192e-04  -0.340 0.734267
plain       -7.309e+01  2.594e+01  -2.817 0.005092 **
elev        -1.428e-02  7.505e-03  -1.902 0.057853 .
prec         2.326e-02  1.533e-02   1.518 0.129951
slope        1.090e+01  1.224e+00   8.909  < 2e-16 ***
temp         8.765e+01  8.134e+00  10.775  < 2e-16 ***
temp0       -2.454e-01  2.544e-02  -9.649  < 2e-16 ***
area         5.678e-04  1.927e-05  29.471  < 2e-16 ***
policy      -1.744e+01  5.683e+00  -3.069 0.002301 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 49.26 on 388 degrees of freedom
Multiple R-squared:  0.8202,    Adjusted R-squared:  0.8128
F-statistic: 110.6 on 16 and 388 DF,  p-value: < 2.2e-16
```

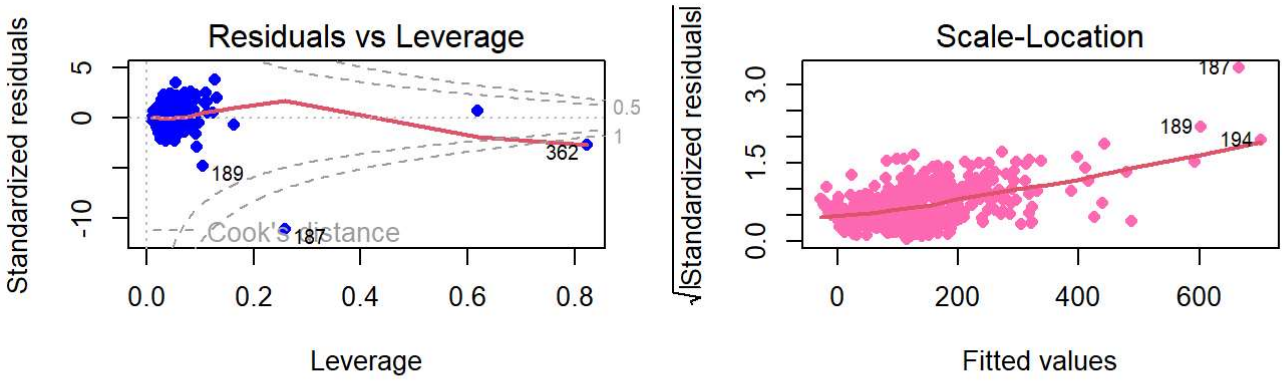
```r
# Residual vs. Fitted Value Plot
p1 <- ggplot(data, aes(x=fitted(initial_model), y=residuals(initial_model))) +
  geom_point(color='purple') +
  geom_smooth(se=FALSE) +
  theme_minimal() +
  ggtitle("Residual vs. Fitted Value Plot") +
  xlab("Fitted values") + ylab("Residuals")

# QQ plot with ggplot2
p2 <- ggplot(data, aes(sample=residuals(initial_model))) +
  stat_qq(color='violet') +
  ggtitle("QQ Plot") +
  theme_minimal()


# Combine the plots
grid.arrange(p1, p2, ncol=2)
```

```
`geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```






```
# Set up 2x2 plotting area
par(mfrow = c(2, 2))



# Residuals vs Leverage plot
p3 <- plot(initial_model, which=5, lwd=2, pch =16, col="blue")

# Scale Location Plot
p4 <- plot(initial_model, which=3, lwd=2, pch=16, col = "hotpink")

# Reset to default plotting layout
par(mfrow = c(1, 1))
```



- The curve in the plot suggests **non-linearity**.
- There may be signs of **heteroscedasticity** as the spread of residuals narrows for higher fitted values.
- In the scale-location plot, there seems to be a slight funnel shape, indicating potential heteroscedasticity.
- Normality has been met due to less deviations and residuals following along the reference line, although there are some deviations at the tails
- Residual vs leverage suggests that while there might be some outliers (like observation 189), the most concerning observations from the standpoint of influencing the regression model are 187 and 362 due to their combination of high leverage and large residuals.
- The D-W statistic is 1.604312, which is less than 2, suggesting positive autocorrelation.
- The p-value is 0.002, which is less than 0.05, meaning the result is statistically significant.
- In conclusion, based on the output, the residuals are not independent; they show significant positive autocorrelation. This means that the assumption of independent errors is violated for this model.

> ### How the model fit changed with $\log fcover$:

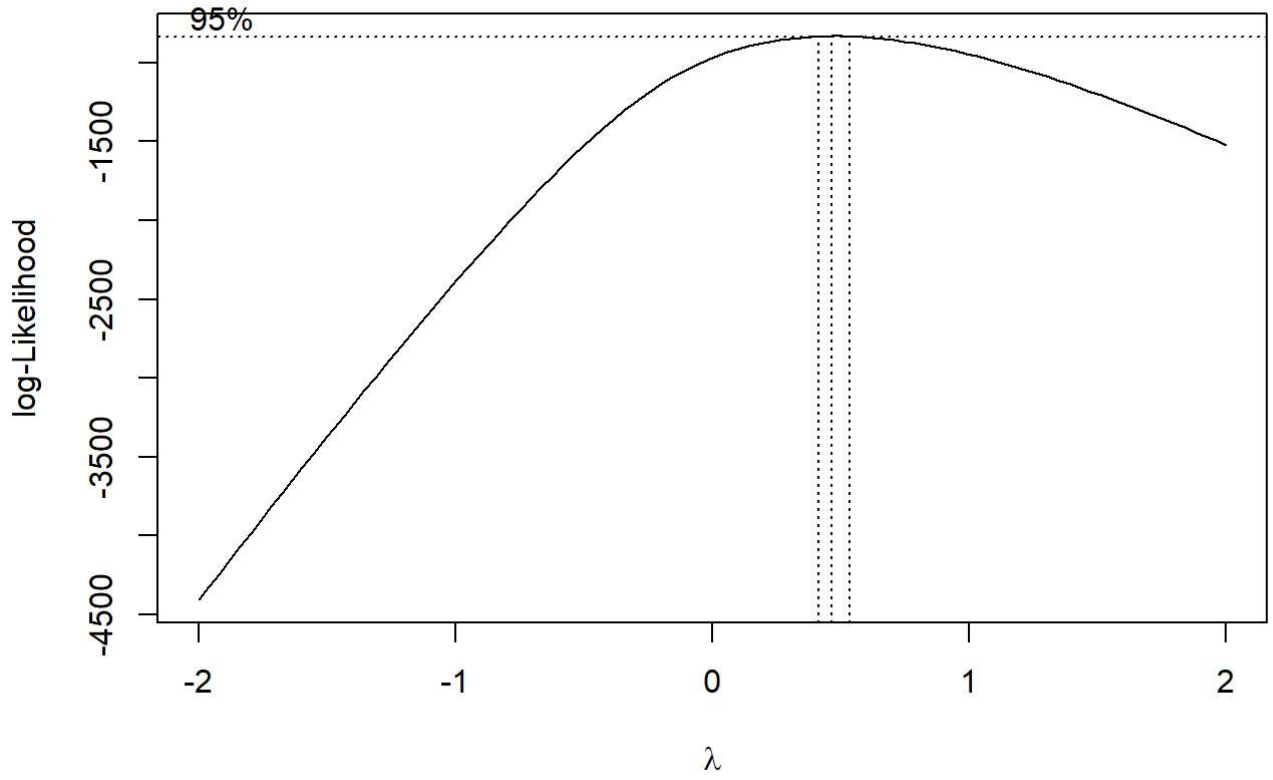
When comparing the two multiple linear regression (MLR) models—with fcover as the response variable in the first and log(fcover) in the second—the differences are notable in several respects. Firstly, the model with log(fcover) as the response variable has a slightly lower adjusted R-squared value (0.7538) compared to the model with fcover (0.8128), indicating that the original model explains a higher proportion of the variance. Secondly, the significance of individual predictors changes. For instance, variables like 'gdp3,' 'distpvc,' 'hwaylen,' and 'prec' become significant in the log-transformed model but were not in the original. Lastly, the residual standard error is substantially different between the two models; however, it's hard to directly compare this metric due to the log transformation. Overall, the log transformation appears to affect the significance of several predictors, but the original fcover model has a slightly better fit according to the adjusted R-squared.

1 (b)

```
library(MASS)
```

```
Attaching package: 'MASS'

The following object is masked from 'package:dplyr':

    select
```

```
boxcox_r <- boxcox(initial_model)
```




```
# Find the lambda that maximizes the log-likelihood
optimal_lambda <- boxcox_r$x[which.max(boxcox_r$y)]
optimal_lambda
```

```
[1] 0.4646465
```

The optimal λ value obtained from the Box-Cox transformation is approximately 0.46. This value is very close to 0.5, which corresponds to a square root transformation.

Given that the optimal λ is so close to 0.5, this strongly suggests that using the $\sqrt{fcover}$ as the response variable in the Multiple Linear Regression (MLR) model would be reasonable. It means that the square root transformation would make the distribution of fcover more normal, thereby satisfying one of the key assumptions of linear regression.

So, in summary, based on the Box-Cox transformation result, using $\sqrt{fcover}$ as the response variable in the MLR model appears to be a good choice.

2

```
library(RobStatTM)
```

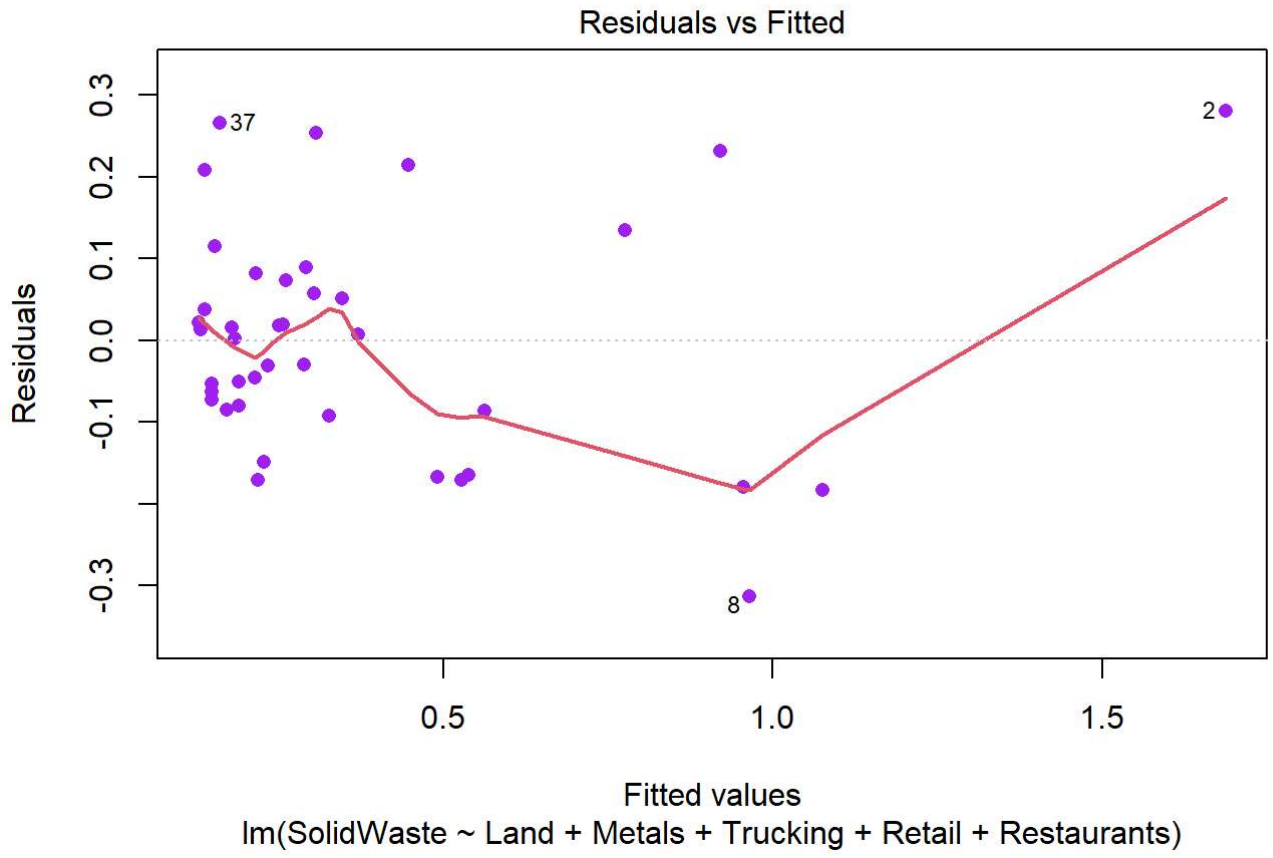
```
Attaching package: 'RobStatTM'
```

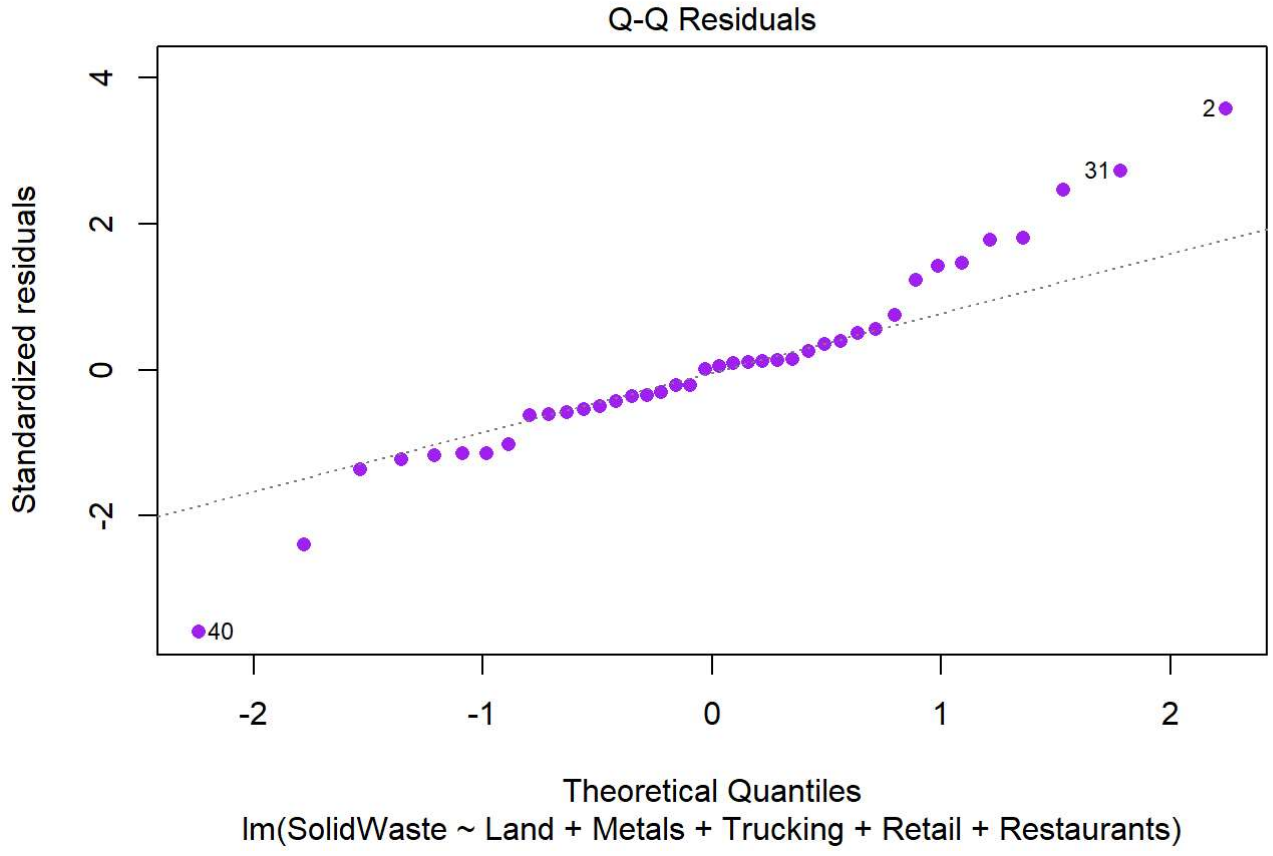
```
The following objects are masked from 'package:MASS':

    huber, oats

The following object is masked from 'package:datasets':

    stackloss
```

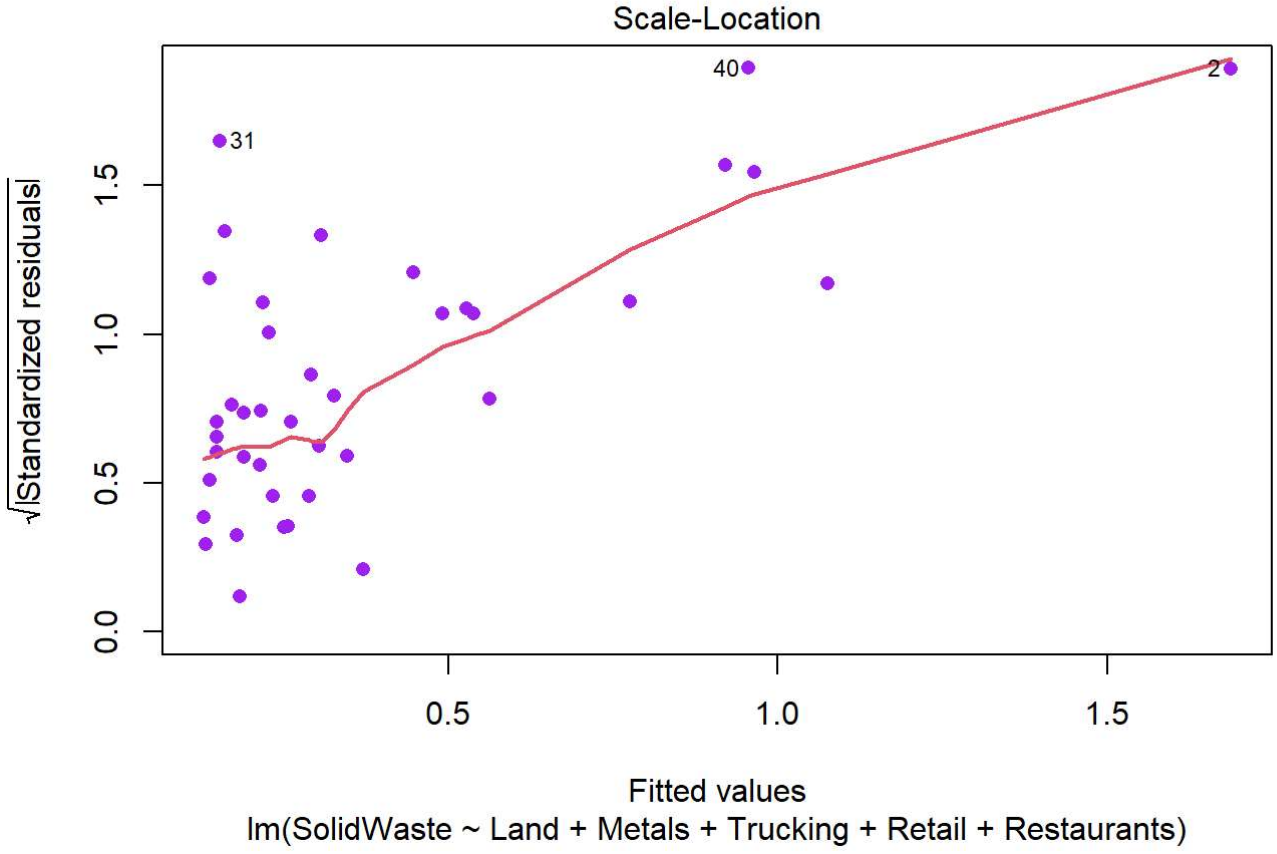
```
model <- lm(SolidWaste ~ Land + Metals + Trucking + Retail + Restaurants, data=waste)
summary(model)
```

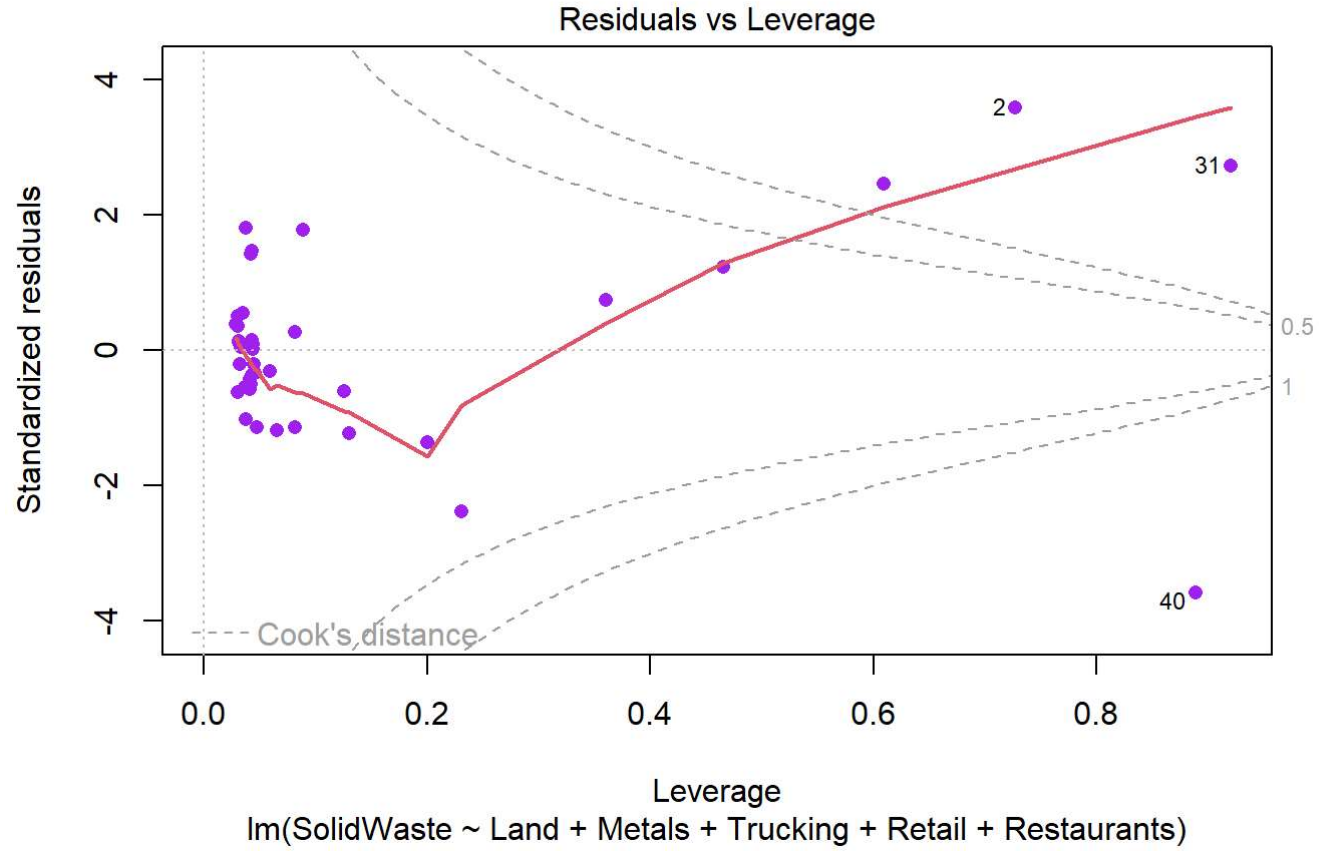
```
Call:
lm(formula = SolidWaste ~ Land + Metals + Trucking + Retail + 
    Restaurants, data = waste)

Residuals:
      Min        1Q    Median        3Q       Max 
-0.313964 -0.085317  0.004264  0.075333  0.280372 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  1.243e-01  3.160e-02   3.934 0.000392 ***
Land        -5.249e-05  1.786e-05  -2.938 0.005895 ** 
Metals       4.146e-05  1.533e-04   0.271 0.788409    
Trucking     2.504e-04  8.831e-05   2.835 0.007663 ** 
Retail      -8.616e-04  3.753e-04  -2.295 0.027999 *  
Restaurants  1.335e-02  2.276e-03   5.868 1.28e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1501 on 34 degrees of freedom
Multiple R-squared:  0.8491,    Adjusted R-squared:  0.8269 
F-statistic: 38.25 on 5 and 34 DF,  p-value: 5.094e-13
```

```
plot(model, lwd=2, pch =16, col="purple")
```


Residuals vs Fitted
Residuals
0.3
0.2
0.1
0.0
-0.1
-0.3
37
2
8
0.5
1.0
1.5
Fitted values
lm(SolidWaste ~ Land + Metals + Trucking + Retail + Restaurants)


Q-Q Residuals
Standardized residuals
4
2
0
-2
2
31
40
-2
-1
0
1
2
Theoretical Quantiles
lm(SolidWaste ~ Land + Metals + Trucking + Retail + Restaurants)


Scale-Location
40
2
31
√|Standardized residuals|
0.0
0.5
1.0
1.5
0.5
1.0
1.5
Fitted values
lm(SolidWaste ~ Land + Metals + Trucking + Retail + Restaurants)


Residuals vs Leverage
2
31
40
0.5
1
Cook's distance
Standardized residuals
4
2
0
-2
-4
0.0
0.2
0.4
0.6
0.8
Leverage
lm(SolidWaste ~ Land + Metals + Trucking + Retail + Restaurants)

Potential issues are displayed in the diagnostic plots from the regression analyses. Due to its observable pattern, the "Residuals vs Fitted" plot suggests non-linearity and potential heteroscedasticity. The "Q-Q Residuals" figure implies that, particularly at the tails, the residuals may not be perfectly normally distributed. The "Scale-Location" plot, which demonstrates variance inconsistency across the levels of the independent variable, further supports the hypothesis of heteroscedasticity. Furthermore, labeled data points with values of "37", "40", and "31" may be significant data points or outliers.

2 (b)

```
studentized_residuals <- rstudent(model)
potential_outliers <- which(abs(studentized_residuals) > 2)
potential_outliers
```

```
2  8 15 31 40
2  8 15 31 40
```

In the context of the dataframe, rows 2, 8, 15, 31, and 40 in the dataframe are potential outliers. These observations have residuals that significantly deviate from the model's prediction. They could be due to data errors, represent unique cases, or be influential points that might skew the model's results.

2 (c)

```
wd <- waste
# Calculate the leverage values
n <- nrow(wd)
p <- ncol(wd) - 1  # Assuming the last column is the response variable
hat_values <- influence(model)$hat

# Identify high leverage points
high_leverage_points <- which(hat_values > max(2*(p+1)/n, 0.5))
high_leverage_points
```

```
2 15 31 40
2 15 31 40
```

High leverage points are observations that have extreme or unusual values for the predictor variables.

In the output "2 15 31 40", rows 2, 15, 31, and 40 in the dataframe waste are identified as high leverage points.

With respect to the dataframe:

1. Rows 2, 15, 31, and 40 have extreme or unusual predictor values compared to the rest of the data.
2. These observations can disproportionately influence the shape and position of the regression line.

```
# Calculate Cook's distance
cook_d <- cooks.distance(model)
```

```
# Identify influential points based on Cook's distance
influential_points_cook <- which(cook_d > 4/n)  # Using a common threshold
cook_d
```

```
          1           2           3           4           5           6
1.632934e-02 5.688788e+00 1.448359e-06 5.226139e-02 1.660260e-04 5.737489e-05
          7           8           9          10          11          12
9.012042e-03 2.855211e-01 1.601346e-02 1.473064e-02 1.313659e-03 6.375655e-04
         13          14          15          16          17          18
2.461565e-04 1.859004e-03 1.575934e+00 5.090478e-02 1.872281e-03 1.320309e-03
         19          20          21          22          23          24
8.053439e-05 1.032624e-03 6.747223e-03 1.000786e-03 3.732108e-02 2.404187e-03
         25          26          27          28          29          30
9.451605e-04 1.133465e-05 1.817378e-03 7.790166e-02 7.463276e-04 1.014863e-03
         31          32          33          34          35          36
1.418269e+01 8.748663e-05 1.096434e-02 1.956959e-02 2.189350e-01 3.383743e-04
         37          38          39          40
2.155252e-02 8.152537e-05 2.043863e-03 1.710696e+01
```

```
influential_points_cook
```

```
2  8 15 31 35 40
2  8 15 31 35 40
```

Cook's distance measures the influence of each observation on all the fitted values. It considers the overall impact of deleting a given observation and is used to detect outliers in the X-space and/or Y-space.

Rows 2, 8, 15, 31, 35, and 40 have been flagged by Cook's distance as potentially influential.

```
# Calculate DFFITS
dffits_values <- dffits(model)
dffits_values
```

```
          1           2           3           4           5
-0.314855430  7.289186051  0.002904240  0.556255251  0.031104330
          6           7           8           9          10
 0.018281078 -0.230363563 -1.413249299  0.315424912  0.301894606
         11          12          13          14          15
 0.087783957  0.061042910 -0.037885908  0.104516393  3.342200519
         16          17          18          19          20
 0.571514827 -0.104871779 -0.087922301  0.021659840 -0.077658813
         21          22          23          24          25
-0.201289484 -0.076490728 -0.476820757 -0.118913514 -0.074318878
         26          27          28          29          30
 0.008124742 -0.103251697 -0.692833084  0.066074347  0.076954303
         31          32          33          34          35
10.274201878  0.022577030 -0.257711994 -0.344296393  1.155068705
         36          37          38          39          40
-0.044418767  0.372681238  0.021793941 -0.109735347 -12.660215457
```

```
# Identify influential points based on DFFITS
influential_points_dffits <- which(abs(dffits_values) > 2*sqrt((p+1)/n))
influential_points_dffits
```

```
2  8 15 31 35 40
2  8 15 31 35 40
```

DFFITS measures the change in the predicted value for a specific observation when that observation is excluded from the analysis. It gives the difference between the fits with and without each observation, standardized by the estimated standard error.

Observations with a large absolute value of DFFITS are influential concerning the predicted value.

Impact on Regression Fit:

Both Cook's distance and DFFITS measure the influence of an observation on the fitted values of the regression model. However:

- Cook's distance gauges the overall influence on the entire set of fitted values.
- DFFITS assesses the influence on the predicted value for the specific observation in question.

Rows 2, 8, 15, 31, 35, and 40 in data have been identified by both Cook's distance and DFFITS as potentially influential points. These observations may exert disproportionate leverage over regression model.

Cook's Distance: This shows how much a single data point affects the overall model. When Cook's distance is high for a point, taking it out would make a big difference to the model's predictions. In simple terms, it lets us see how predictions might shift if we didn't include a certain data point.

DFFITS: This tells us how the model's guess for a specific data point would change if we didn't use that point in making the model. Instead of looking at the entire model like Cook's distance, DFFITS focuses just on one data point's effect on its own guessed value.

Influence on Model's Fit:

Cook's Distance: It looks at data points that can really change how the model predicts all the other points.
DFFITS: It identifies data points that change the model's guess for that specific point a lot, just by being there.

3

```
library(car)

# Calculate VIF
vif_values <- vif(model)
print(vif_values)
```

```
      Land      Metals    Trucking      Retail Restaurants
  1.415870    5.625816    6.786948    8.310634    7.712421
```

```
# Check for high VIF values
high_vif_vars <- names(vif_values[vif_values > 5])
```

- **Land (1.415870)**: This VIF is above 1 but well below 5, indicating that 'Land' is not highly collinear with the other variables.
- **Metals (5.625816)**: This VIF is above 5, suggesting that 'Metals' could be collinear with one or more of the other predictors.
- **Trucking (6.786948)**: This VIF is also above 5, indicating potential multicollinearity with other variables.
- **Retail (8.310634)**: With a VIF above 8, 'Retail' is showing signs of high multicollinearity.
- **Restaurants (7.712421)**: This VIF is also above 5, suggesting high multicollinearity.

Since most of the VIF values are significantly above 5, this is a strong indication that multicollinearity is present.

```
library(glmnet)
```

```
Loading required package: Matrix

Loaded glmnet 4.1-8
```

```
# Prepare the data

X <- as.matrix(waste[, c("Land", "Metals", "Trucking", "Retail", "Restaurants")])
y <- waste$SolidWaste

# Fit the Ridge Regression model
ridge_model <- glmnet(X, y, alpha = 0, lambda = 1)  # lambda is the penalty term, you may want to

# View the model
print(ridge_model)
```

```
Call:  glmnet(x = X, y = y, alpha = 0, lambda = 1)

  Df %Dev Lambda
1  5 54.7      1
```

```
# Perform cross-validation to find the optimal lambda
cv_ridge <- cv.glmnet(X, y, alpha = 0)
optimal_lambda <- cv_ridge$lambda.min  # or cv_ridge$lambda.1se for a more regularized model
```

```
# Fit the Ridge Regression model with the optimal lambda
ridge_model_optimal <- glmnet(X, y, alpha = 0, lambda = optimal_lambda)
ridge_model_optimal
```

```
Call:  glmnet(x = X, y = y, alpha = 0, lambda = optimal_lambda)

  Df  %Dev Lambda
1  5 77.62 0.1184
```

```
coef(ridge_model_optimal)
```

```
6 x 1 sparse Matrix of class "dgCMatrix"
                       s0
(Intercept)  1.740644e-01
Land        -1.434427e-05
Metals       1.168928e-04
Trucking     1.203099e-04
Retail       3.715687e-04
Restaurants  5.062606e-03
```

MODEL 1

- **Df**: The degrees of freedom are 5, indicating that all 5 predictors are used in the model.
- **%Dev**: The percentage of deviance explained by the model is 54.7%. This is a measure of how well the model fits the data.
- **Lambda**: The regularization parameter λ is 1, as specified.

MODEL 2

- **Df**: The degrees of freedom are also 5, indicating that all 5 predictors are used in this model as well.
- **%Dev**: The model explains 77.62% of the deviance, which is a significant improvement over the model with λ.
- **Lambda**: The optimal regularization parameter λ is approximately 0.1184.

Coefficients

The coefficients represent the change in the response variable for a one-unit change in the predictor, keeping all other predictors constant.

- **Land**: A one-unit increase in 'Land' would result in a decrease of $-1.43\times10-5-1.43\times10-5$ in the response variable.
- **Metals**: A one-unit increase in 'Metals' would result in an increase of $1.17\times10-41.17\times10-4$ in the response variable.

- **Trucking**: A one-unit increase in 'Trucking' would result in an increase of $1.20\times10-41.20\times10-4$ in the response variable.
- **Retail**: A one-unit increase in 'Retail' would result in an increase of $3.72\times10-43.72\times10-4$ in the response variable.
- **Restaurants**: A one-unit increase in 'Restaurants' would result in an increase of $5.06\times10-35.06\times10-3$ in the response variable.

Summary

The Ridge Regression model with the optimal λ of 0.1184 explains a higher percentage of the deviance (77.62%) compared to the model with λ = 1(54.7%). This suggests that the model with the optimal λ is a better fit for the data.

Also, none of the coefficients are exactly zero, meaning that all predictors are contributing to the model, albeit with different magnitudes. This is consistent with Ridge Regression's tendency to shrink coefficients towards zero without making them exactly zero.

In summary, the Ridge Regression model with the optimal λ =0.09826 λ =0.09826 seems to be a better fit for the data, explaining more variance and likely handling multicollinearity more effectively.