mestrado-renata-ml

Carregando os pacotes exigidos

```
library(readxl)
library(tidyverse)
library(geobr)
library(skimr)
library(tidymodels)
library(ISLR)
library(modeldata)
library(vip)
library(ggpubr)
source("R/my_functions.R")
```

Listando os arquivos com os mapas de cada área separadamente

```
files_eu <- list.files("data/EU espacial/",full.names = TRUE)
files_sp <- list.files("data/SP espacial/",full.names = TRUE)
```

Carregando os mapa para Eucalipto

```
eu <- map_df(files_eu,grd_read)
```

Arquivo com os dados de emissão, temperatura e umidade (temporal)

```
temporal_eu <- eu %>%
  filter(str_detect(nome,"^[F|U|T]")) %>%
  mutate(numero = as.numeric(str_remove(nome,"F|T|U")),
         ano = numero %% 10000,
         mes = numero %% 1000000 %/% 10000,
         dia = numero %/% 1e6,
         nome = str_remove_all(nome,"[0-9]")) %>%
  pivot_wider(names_from = nome,
              values_from = vetor)
```

Arquivo com os dados dos atributos do solo, somente

geoespacializados

```
spatial_eu <- eu %>%
  filter(!str_detect(nome,"^[F|U|T]")) %>%
  mutate(nome = str_remove(nome,"_EU")) %>%
  pivot_wider(names_from = nome,
              values_from = vetor)
```

Unindo as bases de dados, ou seja, repetindo os dados do solo para cada dia de avaliação

```
data_eu <- left_join(temporal_eu, spatial_eu, by="id") %>%
  select(-numero) %>%
  mutate(data = make_date(year= ano, month=mes, day=dia)) %>%
  relocate(id,data)
```

Carregando os mapa para Sistema Silvipastoril

```
sp <- map_df(files_sp,grd_read)
```

Arquivo com os dados de emissão, temperatura e umidade (temporal)

```
temporal_sp <- sp %>%
  filter(str_detect(nome,"^[F|U|T]")) %>%
  mutate(numero = as.numeric(str_remove(nome,"F|T|U")),
         ano = numero %% 10000,
         mes = numero %% 1000000 %/% 10000,
         dia = numero %/% 1e6,
         nome = str_remove_all(nome,"[0-9]")) %>%
  pivot_wider(names_from = nome,
              values_from = vetor)
```

Arquivo com os dados dos atributos do solo, somente geoespacializados

```
spatial_sp <- sp %>%
  filter(!str_detect(nome,"^[F|U|T]")) %>%
  mutate(nome = str_remove(nome,"_SP")) %>%
  pivot_wider(names_from = nome,
              values_from = vetor)
```

Unindo as bases de dados, ou seja, repetindo os dados do solo

para cada dia de avaliação

```
data_sp <- left_join(temporal_sp, spatial_sp, by="id") %>%
  select(-numero) %>%
  mutate(data = make_date(year= ano, month=mes, day=dia)) %>%
  relocate(id,data)
```

```
tibble(xnames=names(data_eu), ynames=names(data_sp)) %>%
  mutate(logic_test = xnames == ynames)
#> # A tibble: 21 x 3
#>    xnames ynames logic_test
#>    <chr>  <chr>  <lgl>
#>  1 id     id     TRUE
#>  2 data   data   TRUE
#>  3 ano    ano    TRUE
#>  4 mes    mes    TRUE
#>  5 dia    dia    TRUE
#>  6 F      F      TRUE
#>  7 T      T      TRUE
#>  8 U      U      TRUE
#>  9 Al     Al     TRUE
#> 10 Ca     Ca     TRUE
#> # i 11 more rows
```

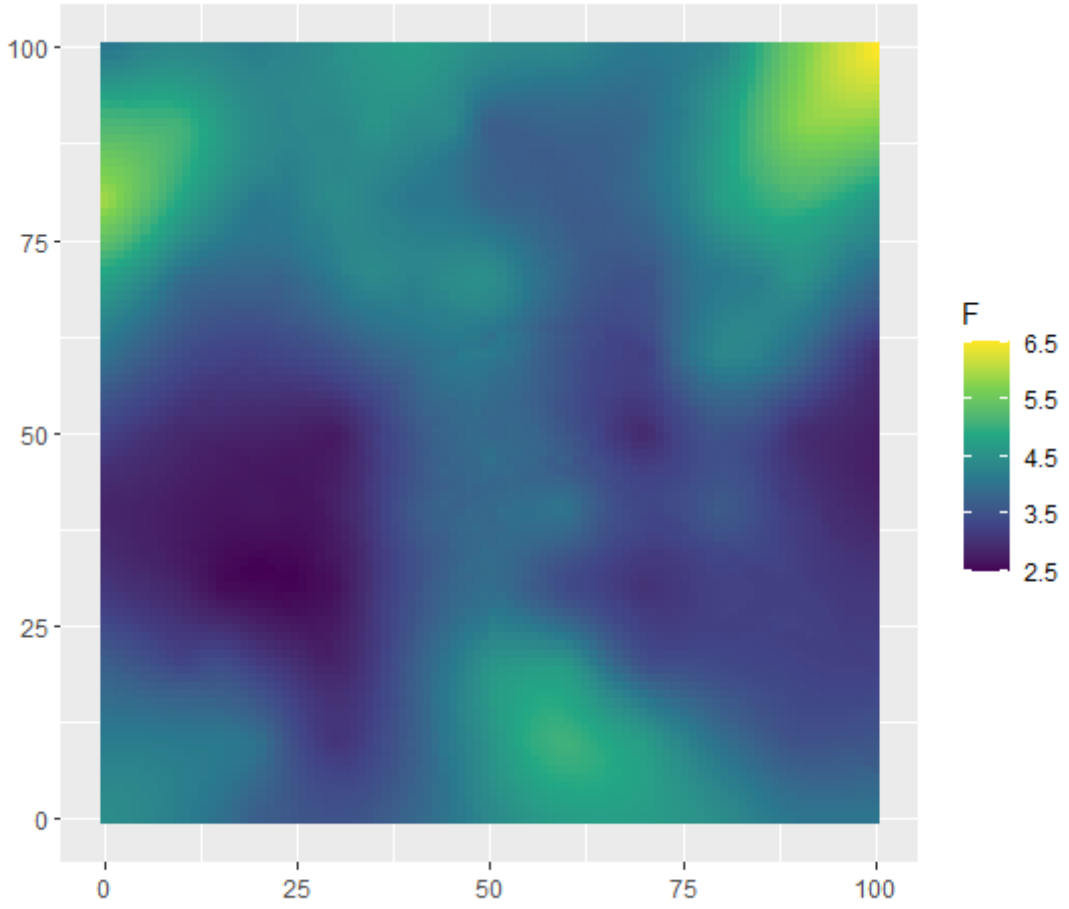
Unindo toda a base de dados

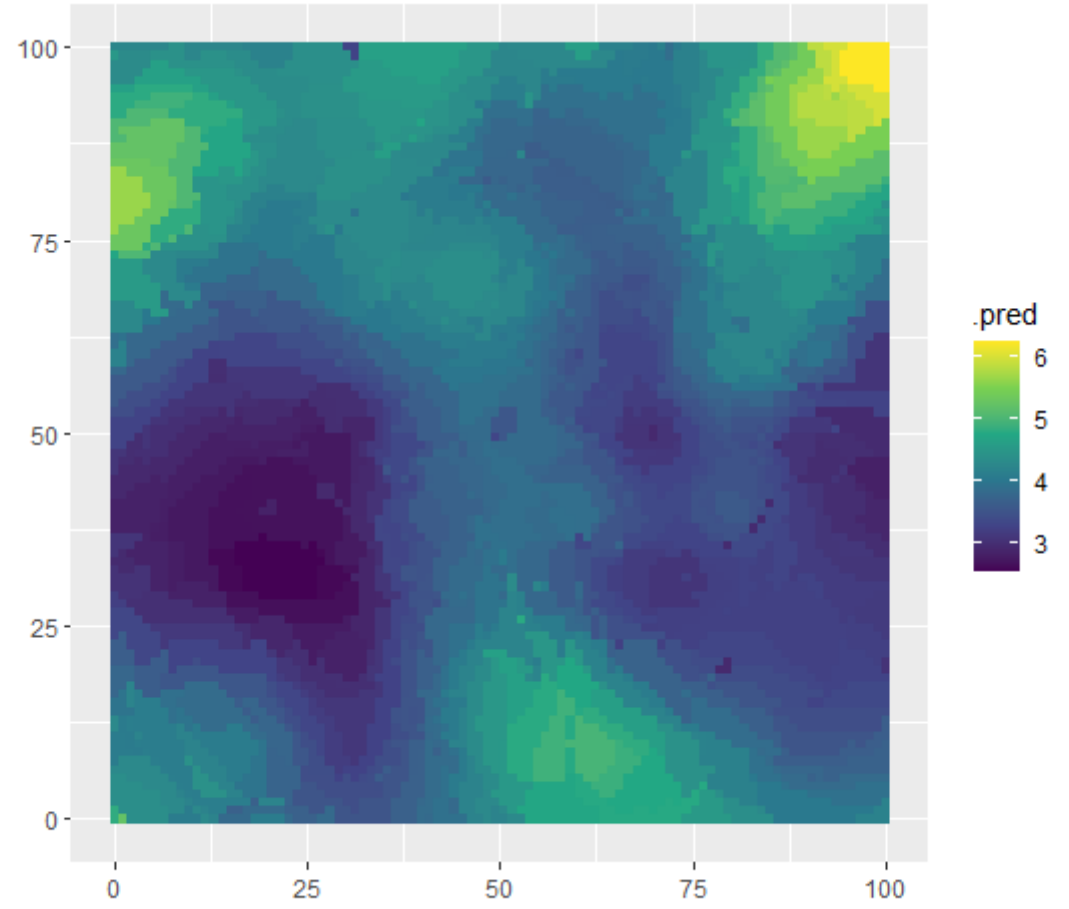
```
data_set <- rbind(data_eu %>%
                    mutate(local="EU"),
                  data_sp %>%
                    mutate(local="SP")) %>%
  relocate(local)
```

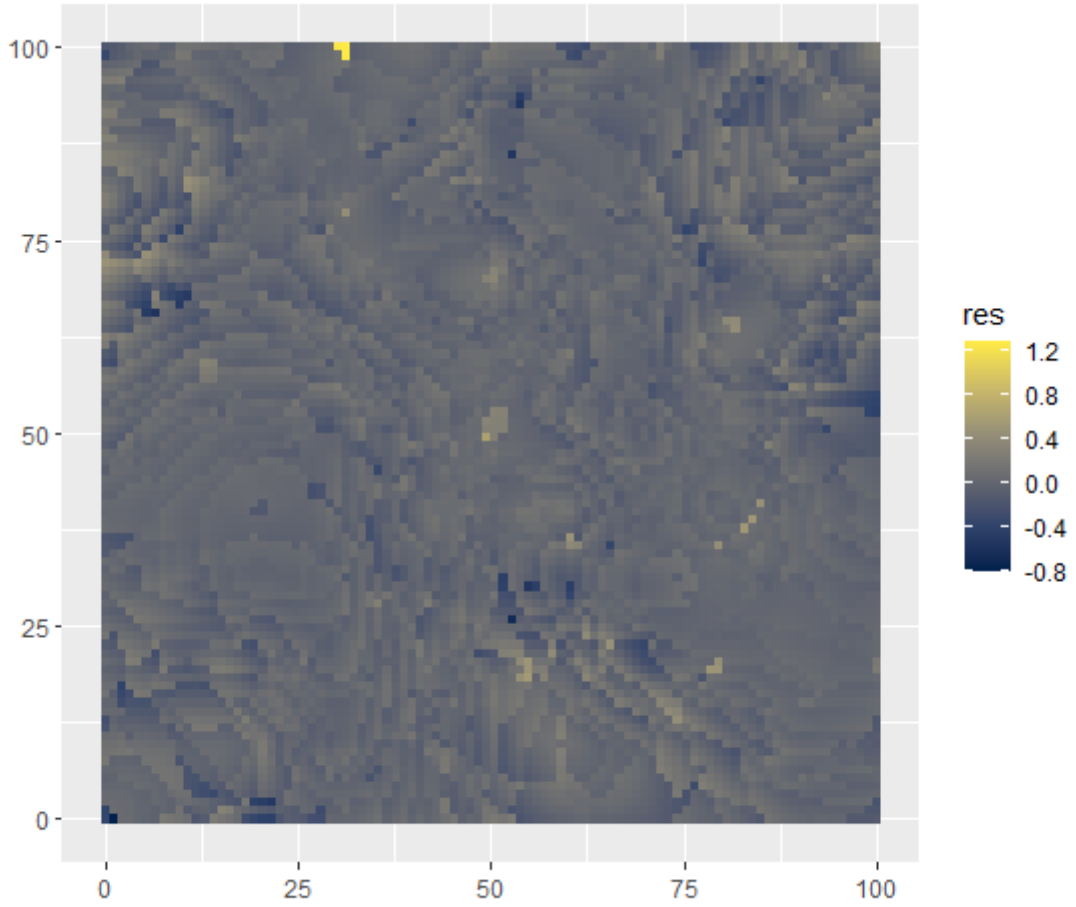
Criando os preditos para cada dia - EUCALIPTO

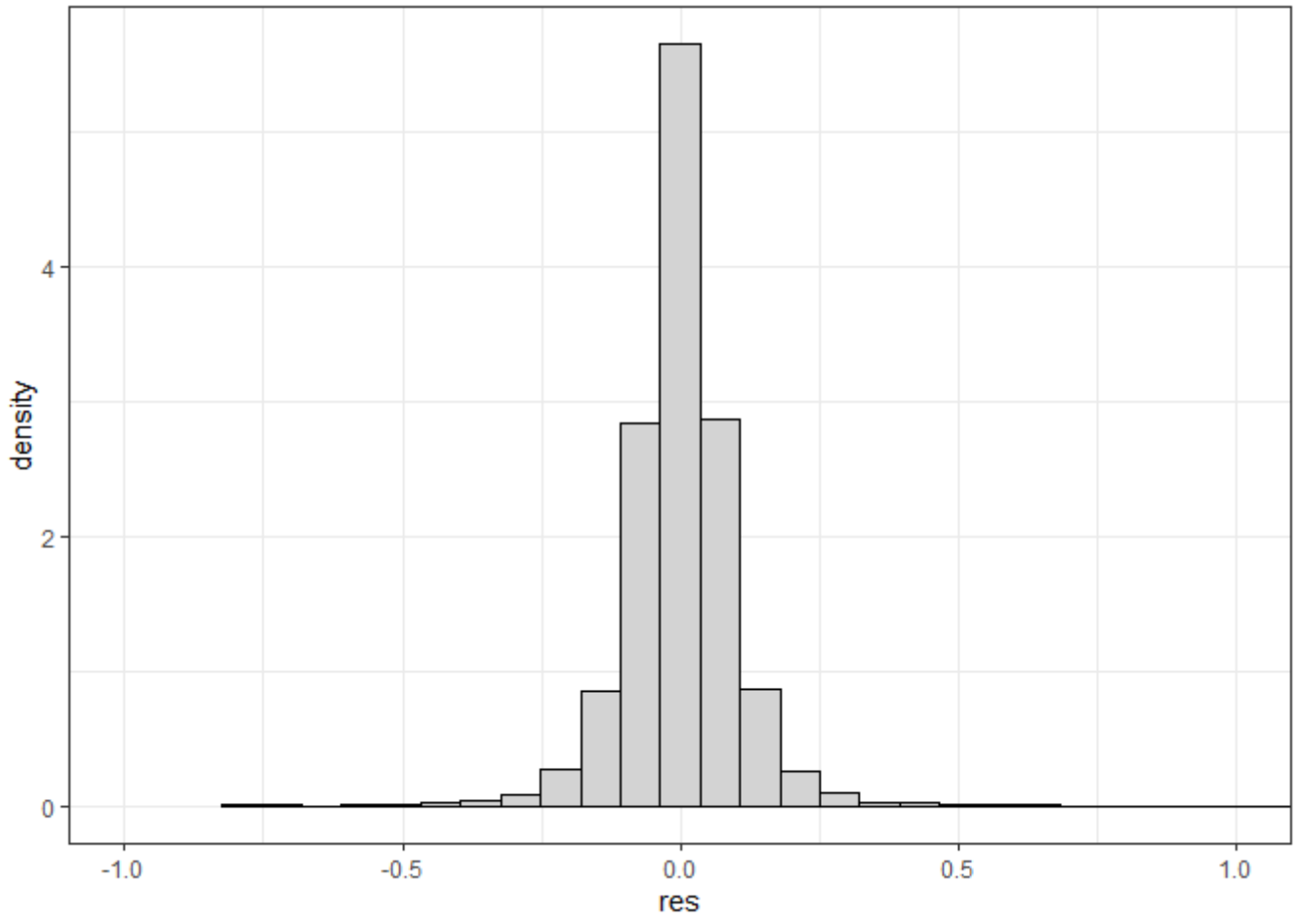
```
dias <- data_eu$data %>% unique()
dis <- 100/93
grid <- expand.grid(Y=seq(0,100,dis), X=seq(0,100,dis))
for(i in seq_along(dias)){
  di <- dias[i]
  data_eu %>% filter(data == di)
  file_models_eu <- list.files("models-3", pattern = "EU")
  file_models_eu <- grep(paste0(di), file_models_eu,value = TRUE)

  fco2_modelo_load_dt <- read_rds(
    paste0("models-3/",grep("dt",file_models_eu,value=TRUE)
    ))
  fco2_modelo_load_rf <- read_rds(
    paste0("models-3/",grep("rf",file_models_eu,value=TRUE)
    ))
  fco2_modelo_load_xgb <- read_rds(
    paste0("models-3/",grep("xgb",file_models_eu,value=TRUE)
    ))

  yp_dt <- predict(fco2_modelo_load_dt, new_data = data_eu %>%
                     filter(data == di))
  yp_rf <- predict(fco2_modelo_load_rf, new_data = data_eu %>%
                     filter(data == di))
  # yp_xgb <- predict(fco2_modelo_load_xgb, new_data = data_eu %>%
  #                 filter(data == di))


  mp_krg <- tibble(grid, data_eu %>%
                     filter(data == di)) %>%
    ggplot(aes(x=X,y=Y)) +
    geom_tile(aes(fill = F)) +
    scale_fill_viridis_c() +
    coord_equal()+labs(x="",y="")

  mp_dt <- tibble(grid, data_eu %>%
                    filter(data == di), yp_dt) %>%
    ggplot(aes(x=X,y=Y)) +
    geom_tile(aes(fill = .pred)) +
    scale_fill_viridis_c() +
    coord_equal() +labs(x="",y="")

  mp_rf <- tibble(grid, data_eu %>%
                    filter(data == di), yp_rf) %>%
    ggplot(aes(x=X,y=Y)) +
    geom_tile(aes(fill = .pred)) +
    scale_fill_viridis_c() +
    coord_equal() +labs(x="",y="")

  mp_dt_res <- tibble(grid, data_eu %>%
```

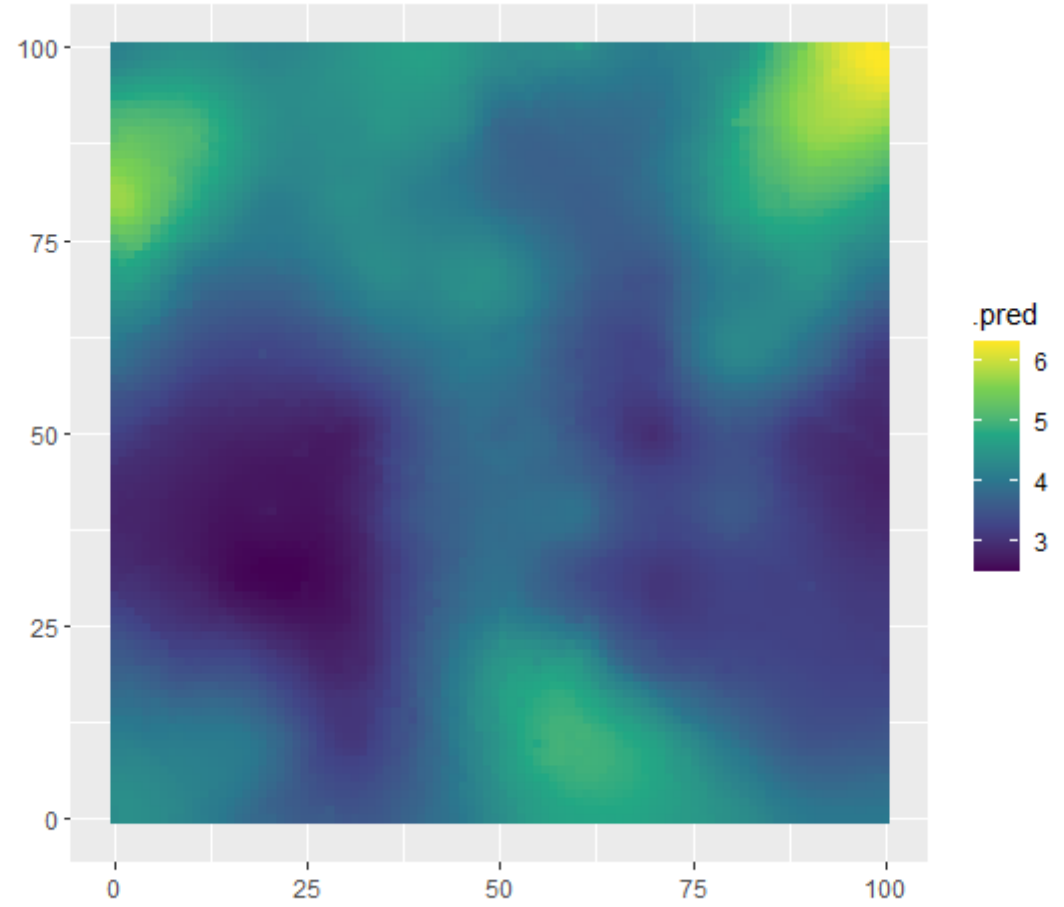
```
                    filter(data == di), yp_dt) %>%
    mutate(res = F - .pred) %>%
    ggplot(aes(x=X,y=Y)) +
    geom_tile(aes(fill = res)) +
    scale_fill_viridis_c(option = "E") +
    coord_equal() +labs(x="",y="")

  hist_dt_res <- tibble(grid, data_eu %>%
                          filter(data == di), yp_dt) %>%
    mutate(res = F - .pred) %>%
    ggplot(aes(x=res, y=..density..)) +
    geom_histogram(color="black",fill="lightgray") +theme_bw()+
    coord_cartesian(xlim=c(-1,1))

  mp_rf_res <- tibble(grid, data_eu %>%
                    filter(data == di), yp_rf) %>%
    mutate(res = F - .pred) %>%
    ggplot(aes(x=X,y=Y)) +
    geom_tile(aes(fill = res)) +
    scale_fill_viridis_c(option = "E") +
    coord_equal() +labs(x="",y="")

  hist_rf_res <- tibble(grid, data_eu %>%
                          filter(data == di), yp_rf) %>%
    mutate(res = F - .pred) %>%
    ggplot(aes(x=res, y=..density..)) +
    geom_histogram(color="black",fill="lightgray") +theme_bw()+
    coord_cartesian(xlim=c(-1,1))



  print(di)
  print(mp_krg)
  print(mp_dt)
  print(mp_dt_res)
  print(hist_dt_res)

  print(mp_rf)
  print(mp_rf_res)
  print(hist_rf_res)

}
#> [1] "2017-06-03"
```

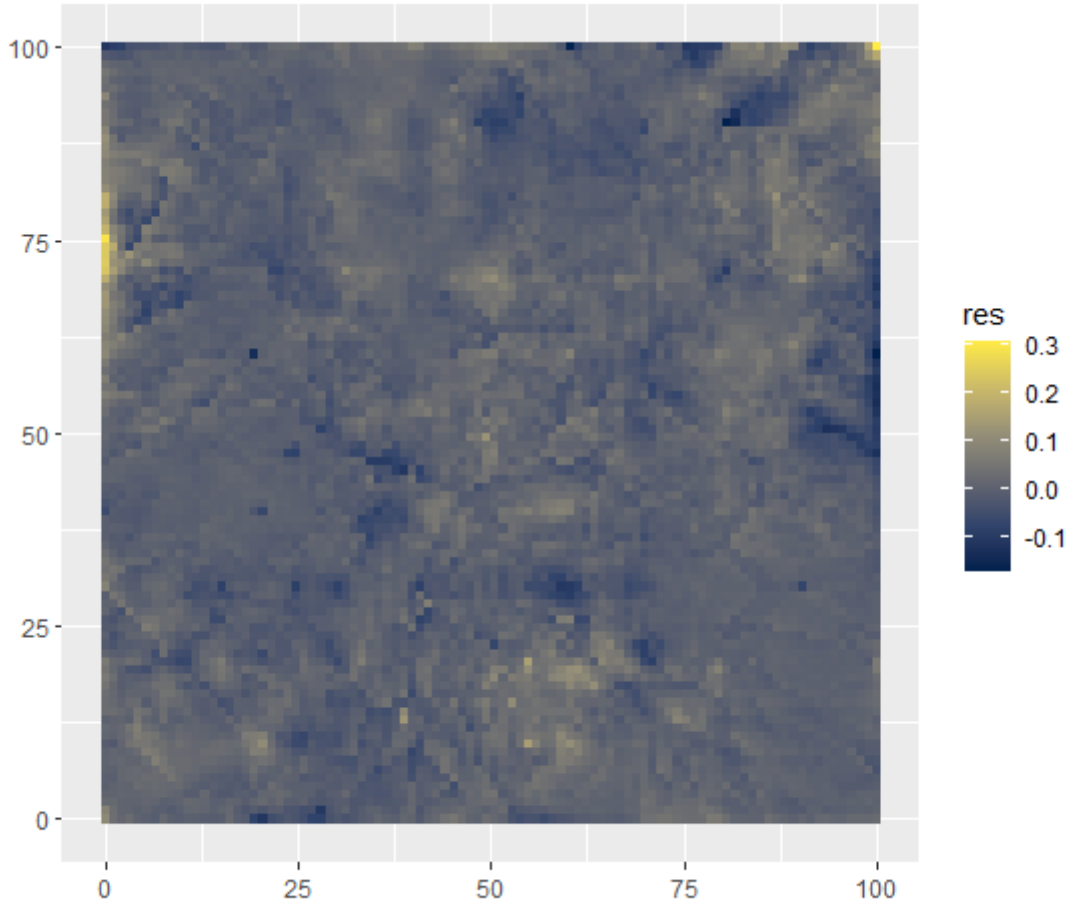

100
75
50
25
0
0
25
50
75
100
F
6.5
5.5
4.5
3.5
2.5

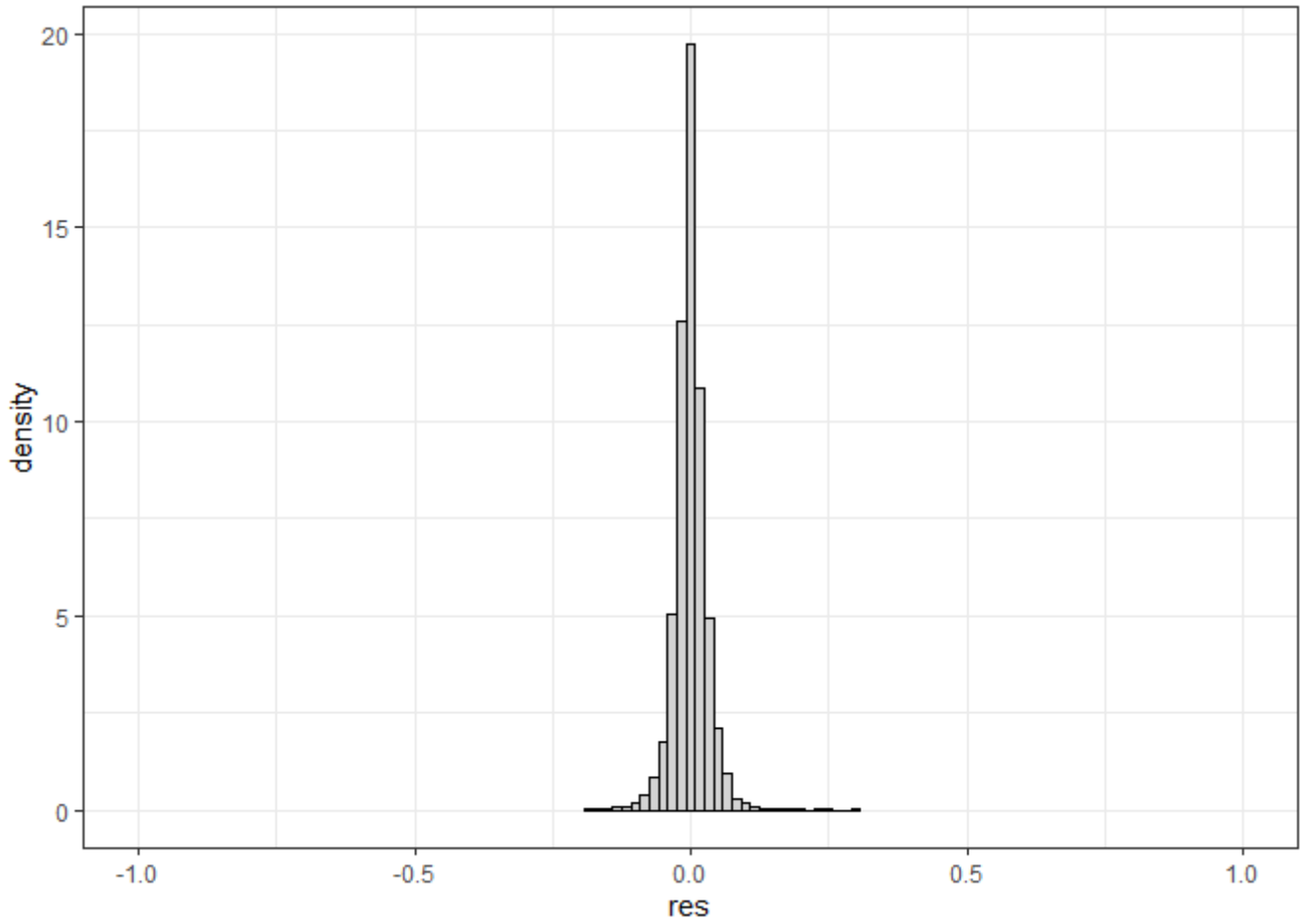

100
75
50
25
0
0
25
50
75
100
.pred
6
5
4
3


100
75
50
25
0
0
25
50
75
100
res
1.2
0.8
0.4
0.0
-0.4
-0.8

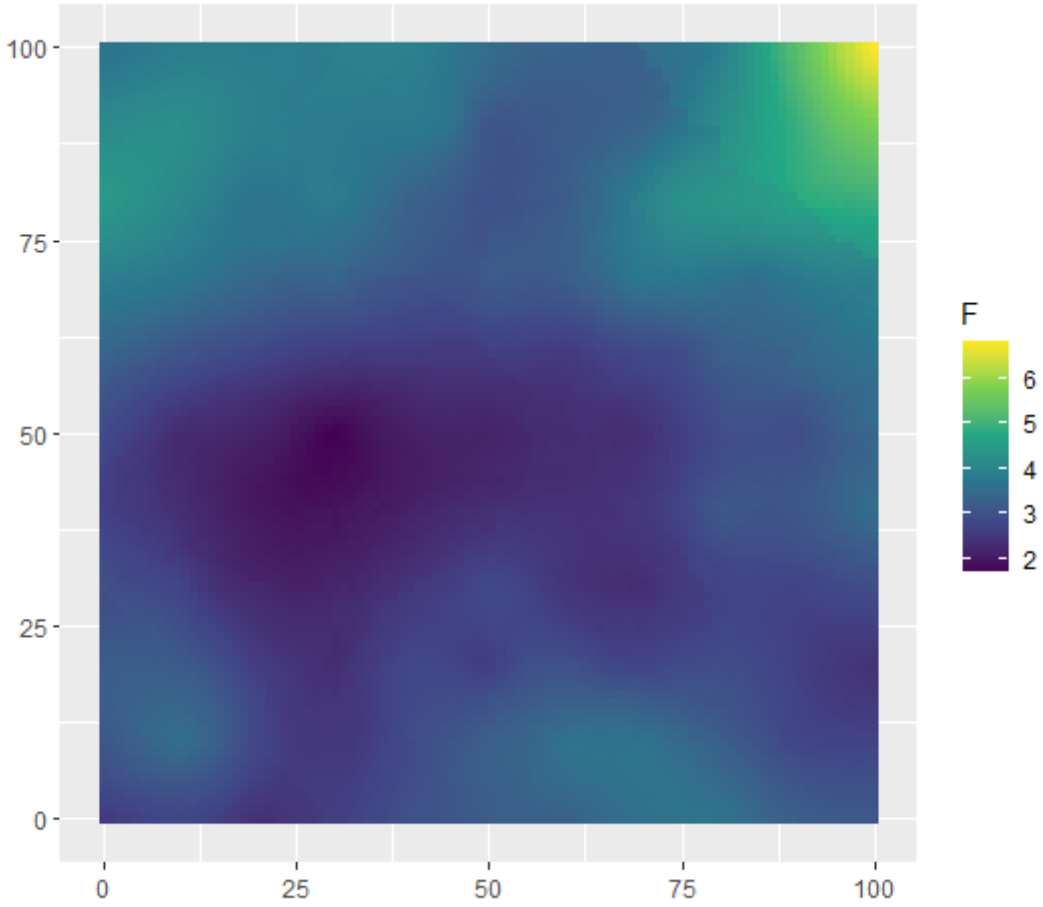

4
2
0
density
-1.0
-0.5
0.0
0.5
1.0
res

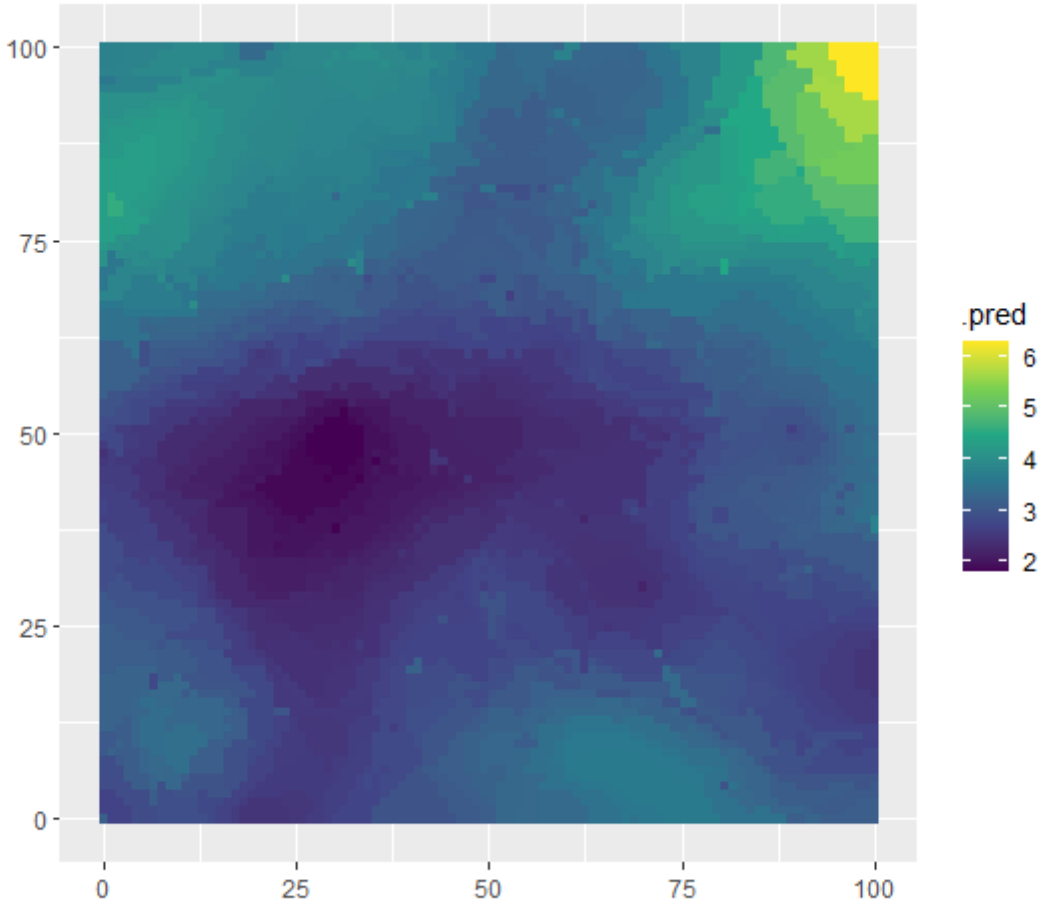

100
75
50
25
0
0
25
50
75
100
.pred
6
5
4
3

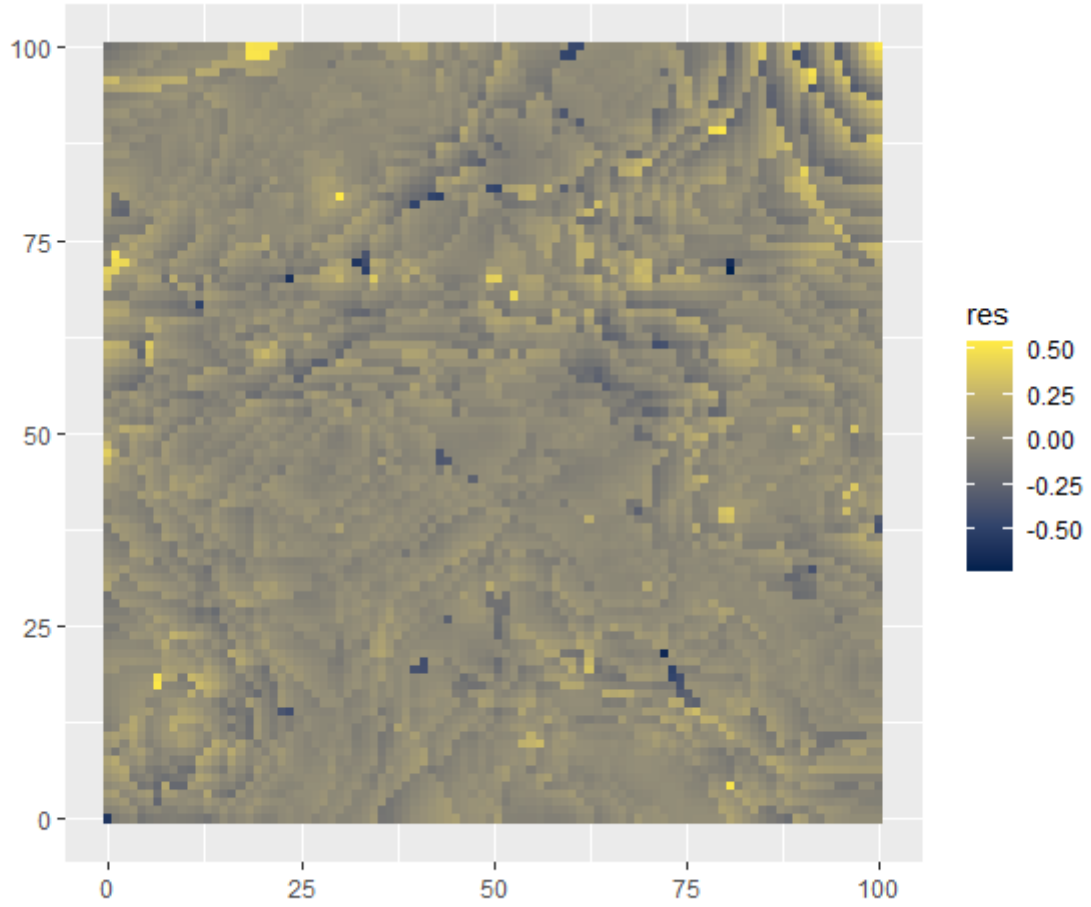

100
75
50
25
0
0
25
50
75
100
res
0.3
0.2
0.1
0.0
-0.1

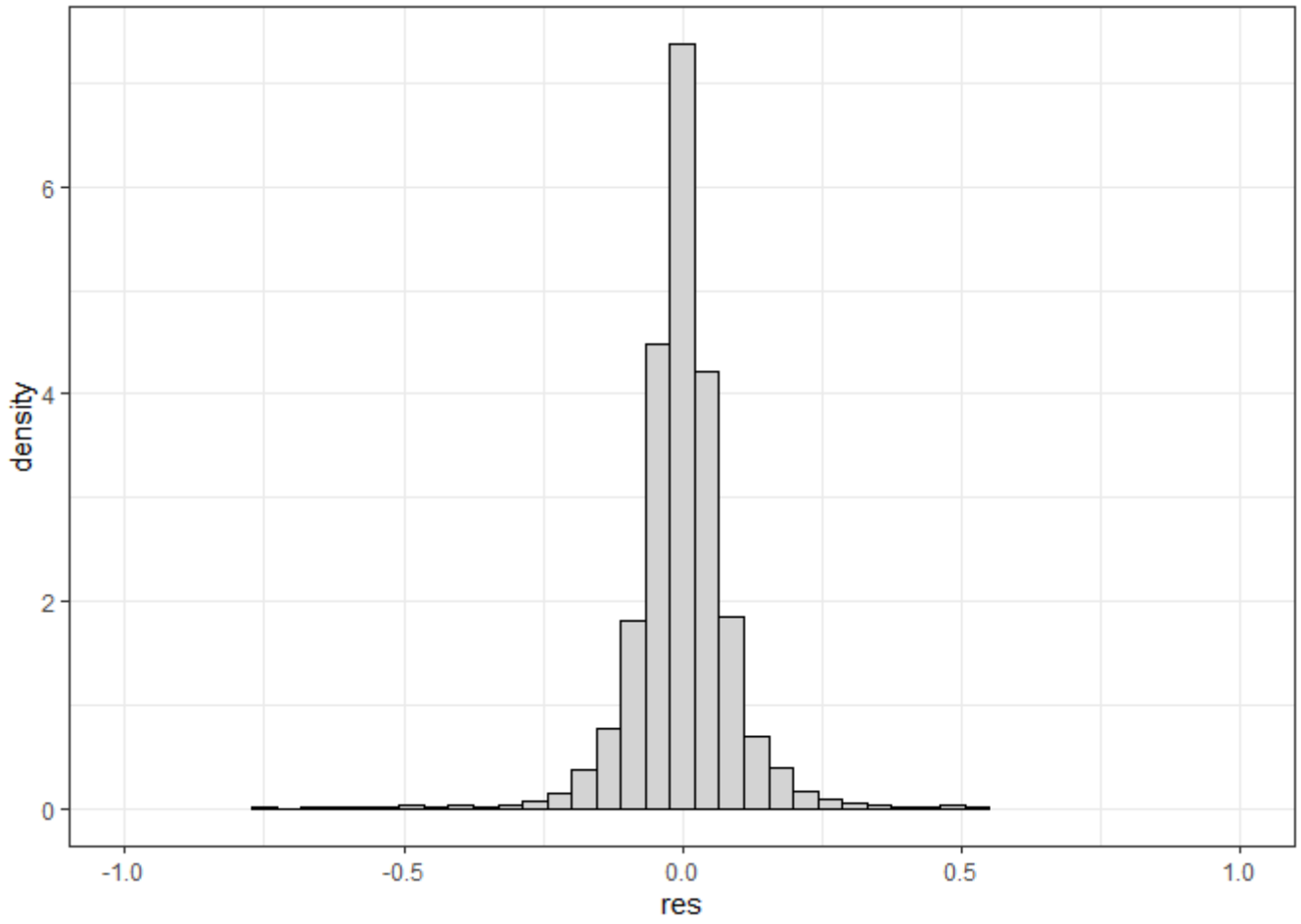


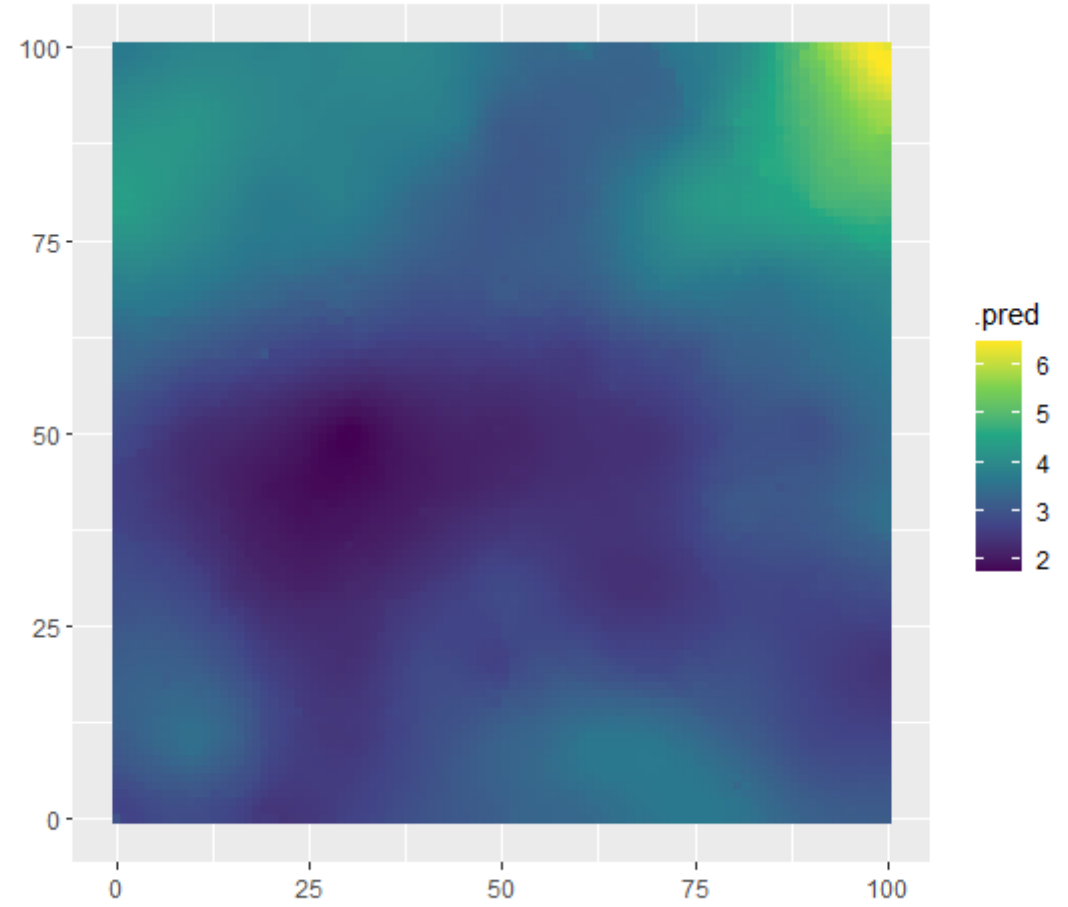

```
#> [1] "2017-06-10"
```

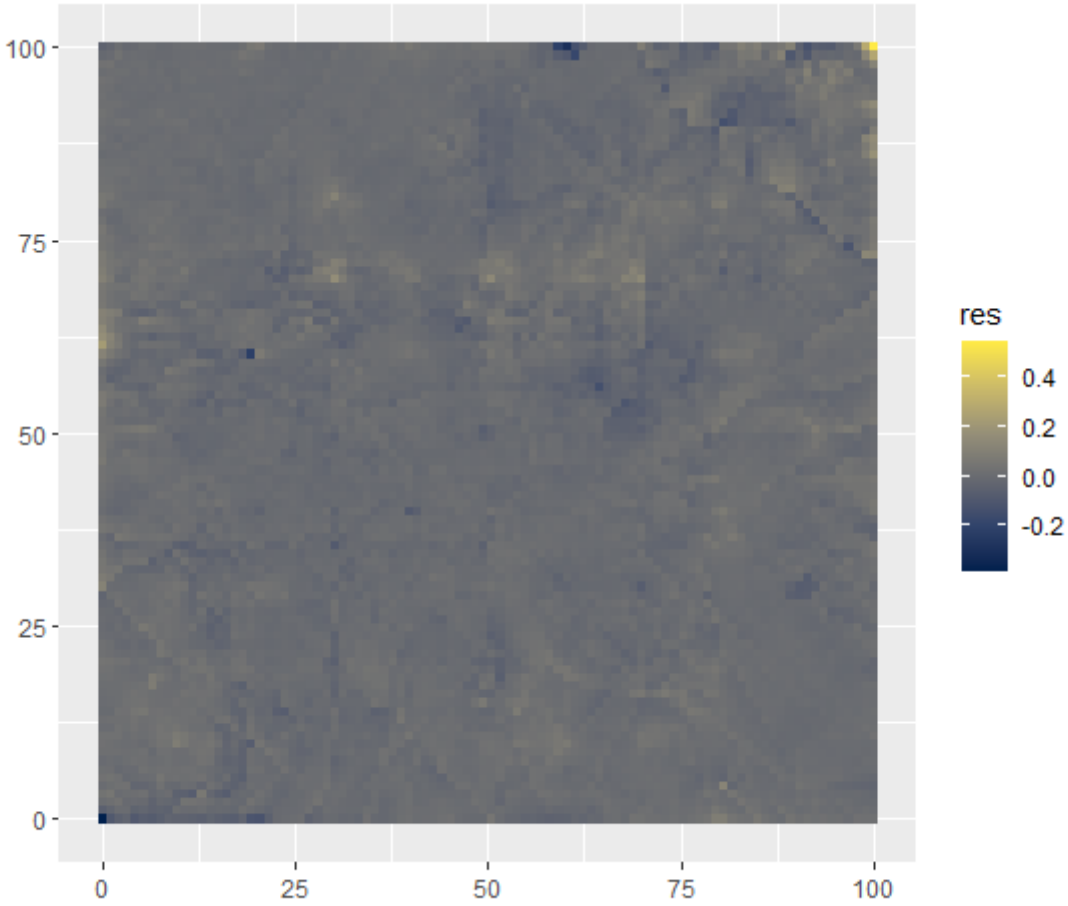

100
75
50
25
0
0
25
50
75
100
F
6
5
4
3
2

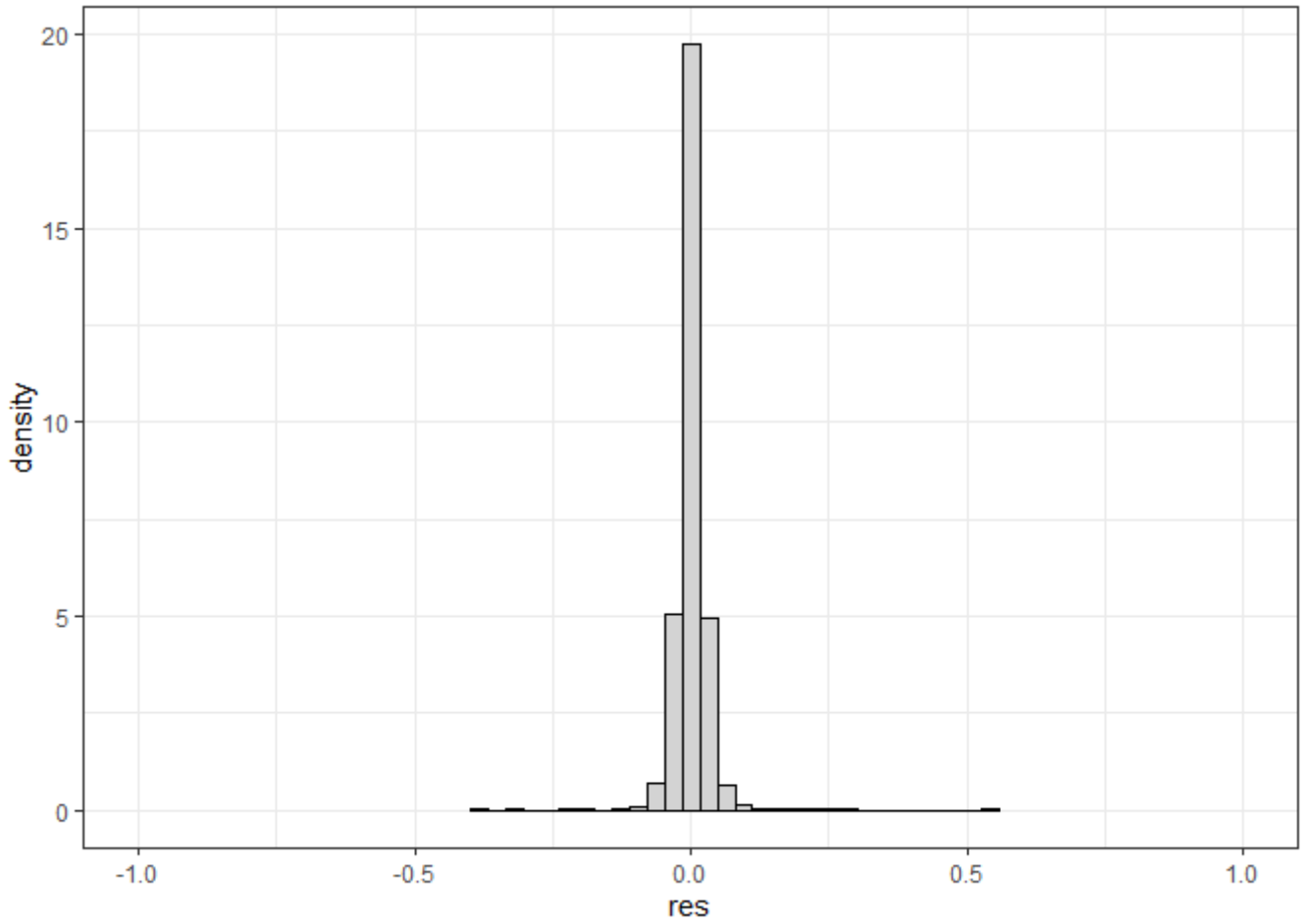

100
75
50
25
0
0
25
50
75
100
.pred
6
5
4
3
2


100
75
50
25
0
0
25
50
75
100
res
0.50
0.25
0.00
-0.25
-0.50

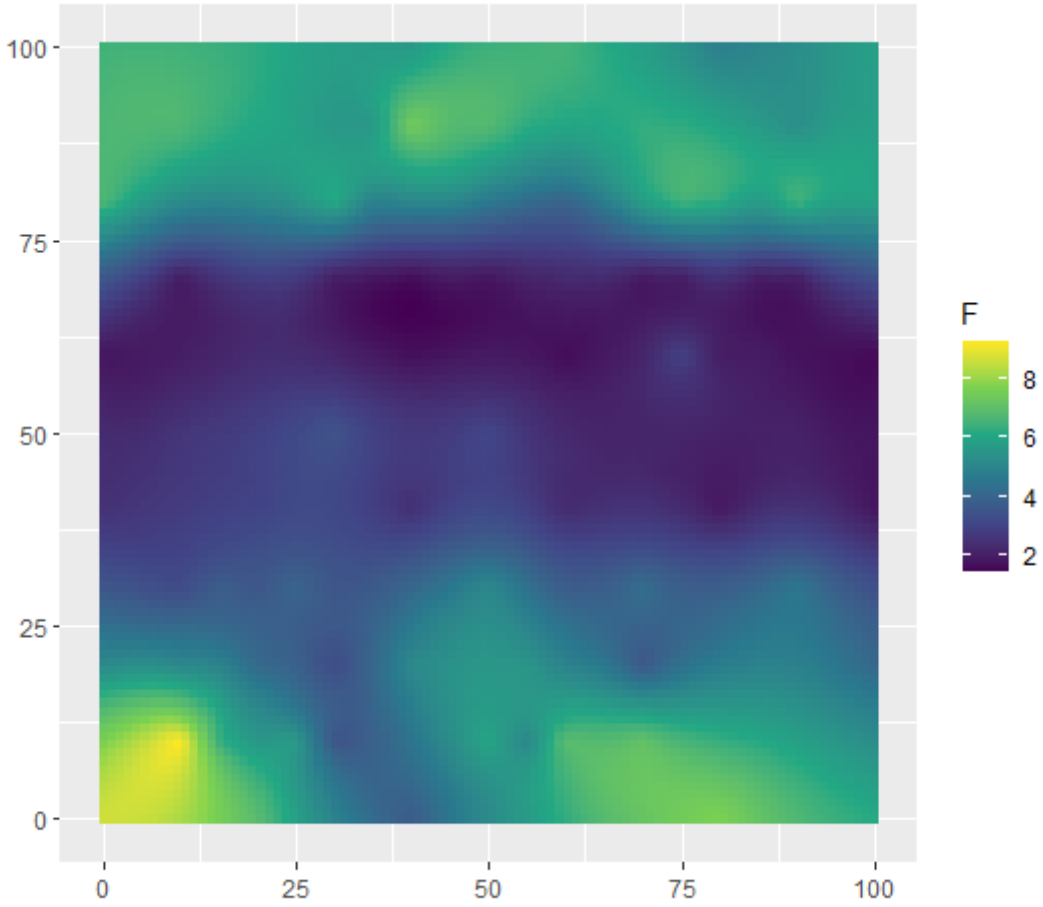

6
4
2
0
density
-1.0
-0.5
0.0
0.5
1.0
res

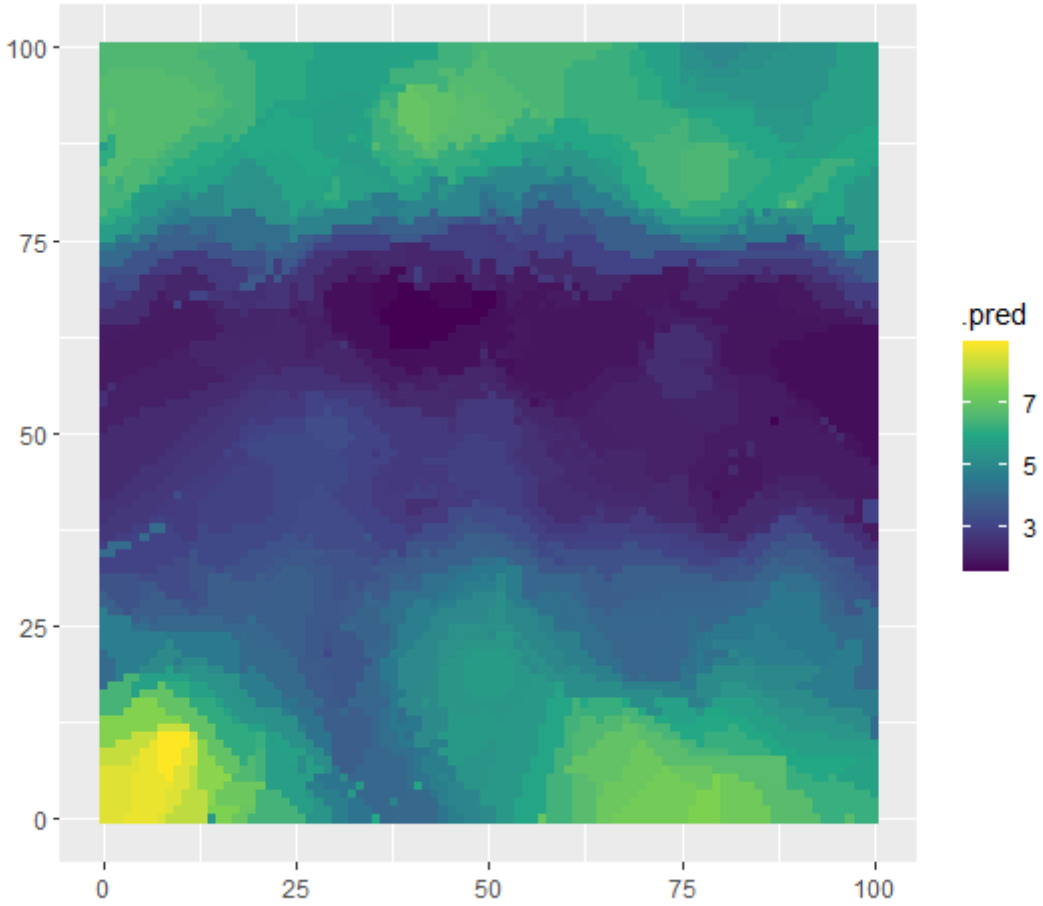

100
75
50
25
0
0
25
50
75
100
.pred
6
5
4
3
2

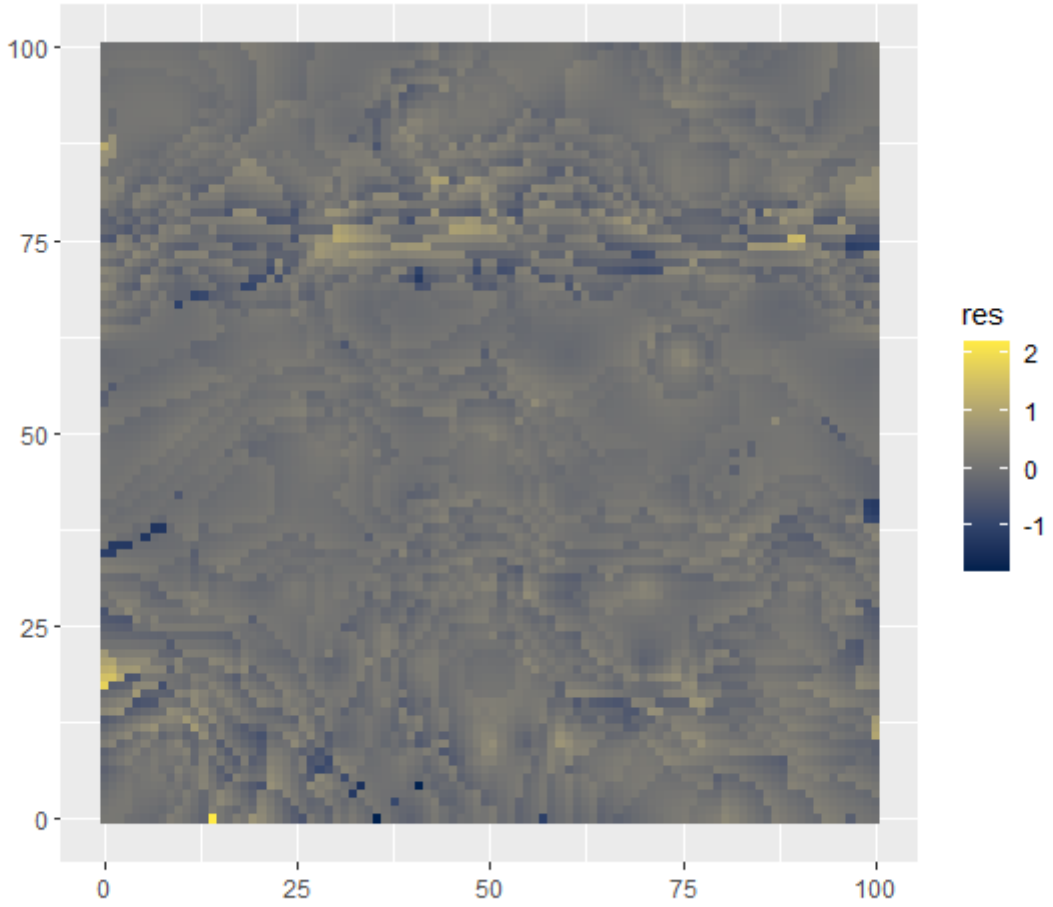

100
75
50
25
0
0
25
50
75
100
res
0.4
0.2
0.0
-0.2

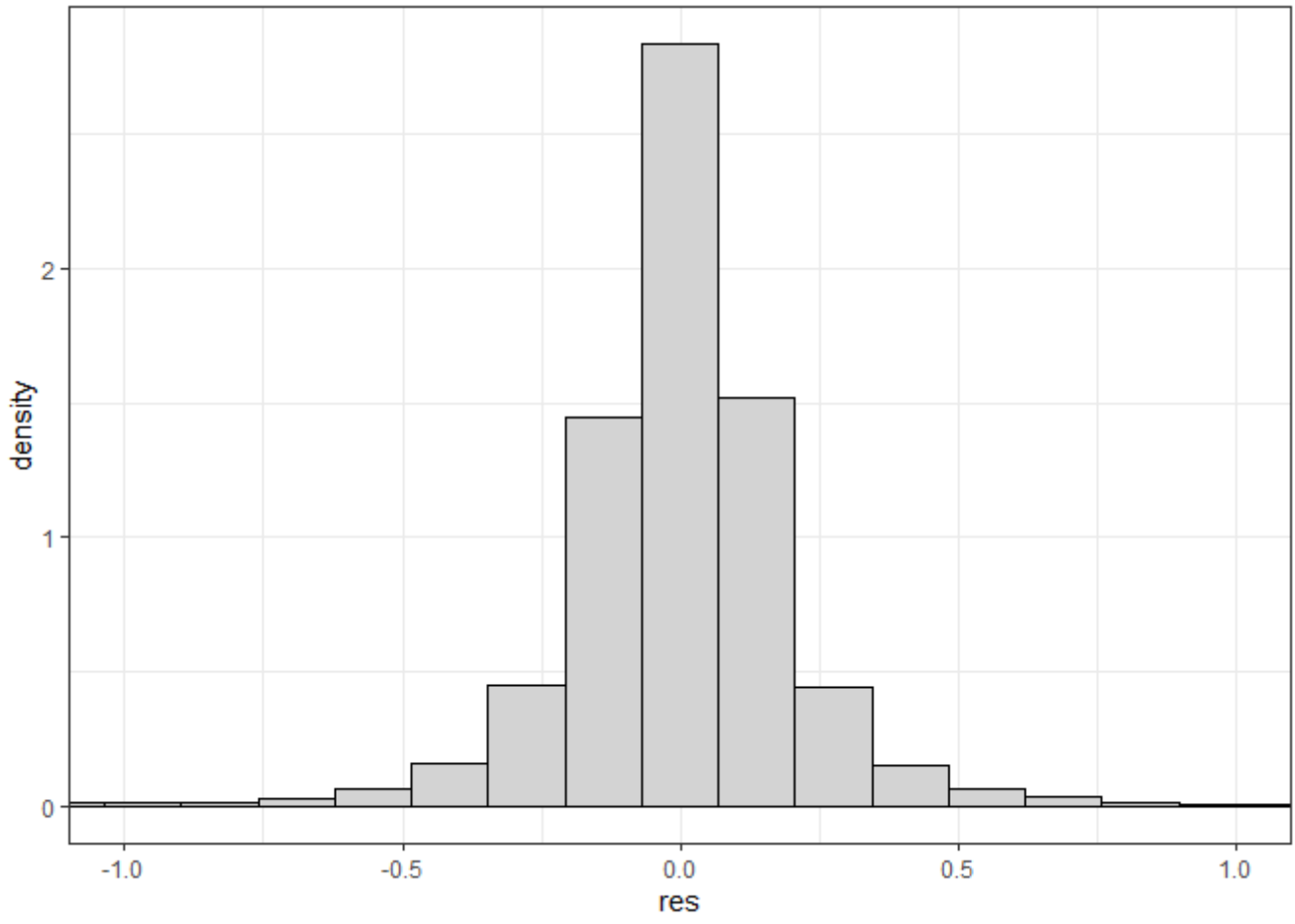


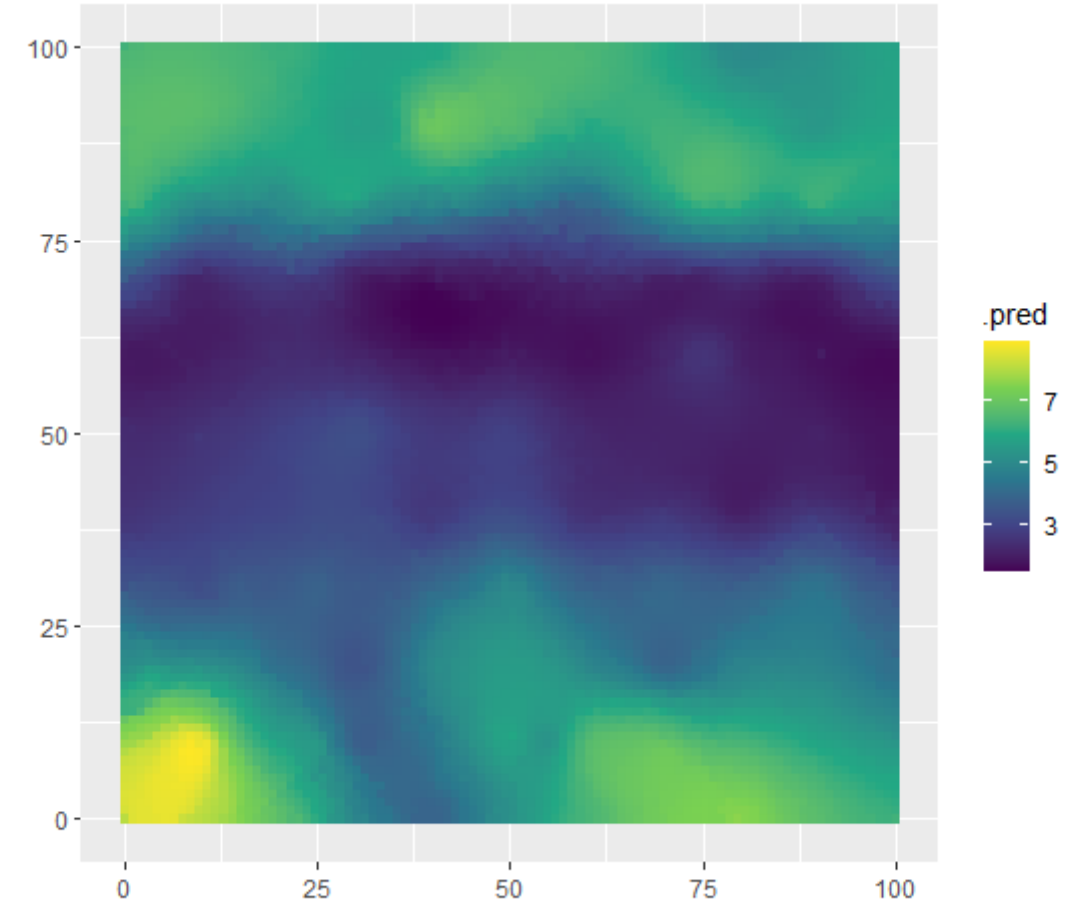

```
#> [1] "2017-03-15"
```

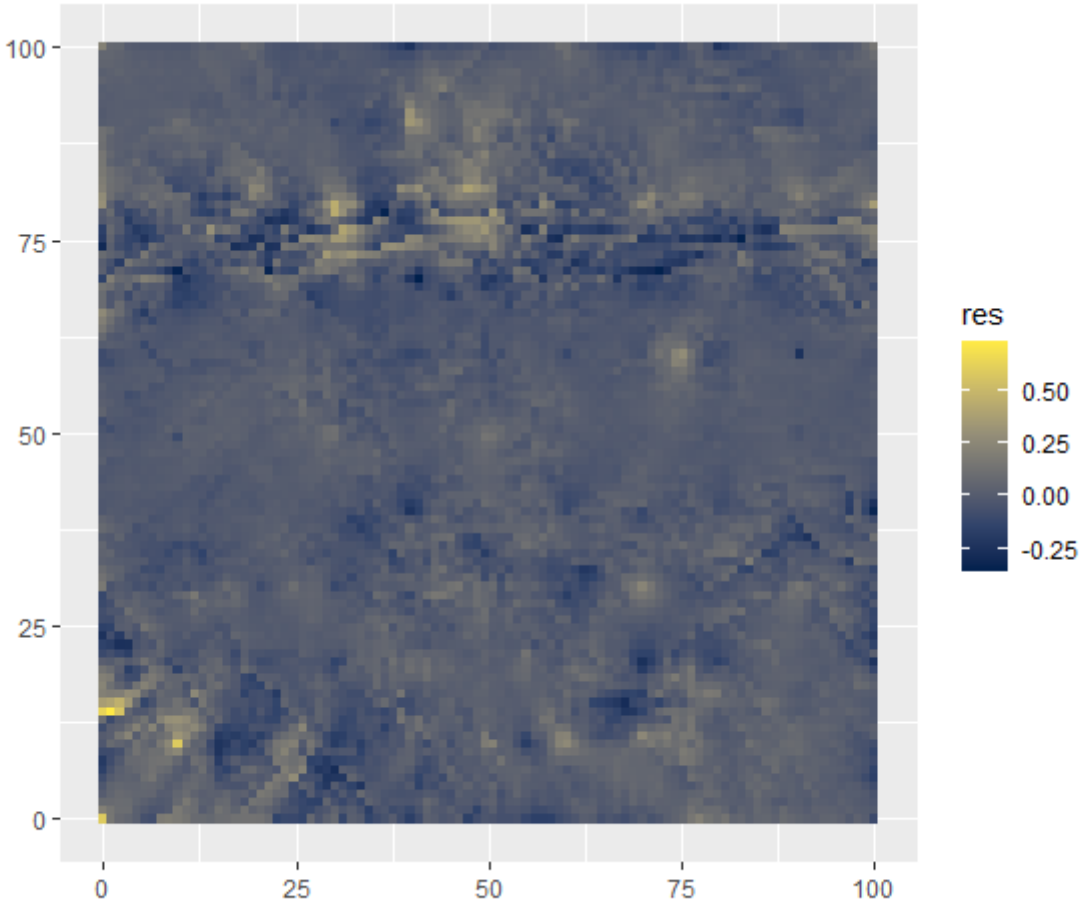

100
75
50
25
0
0
25
50
75
100
F
8
6
4
2

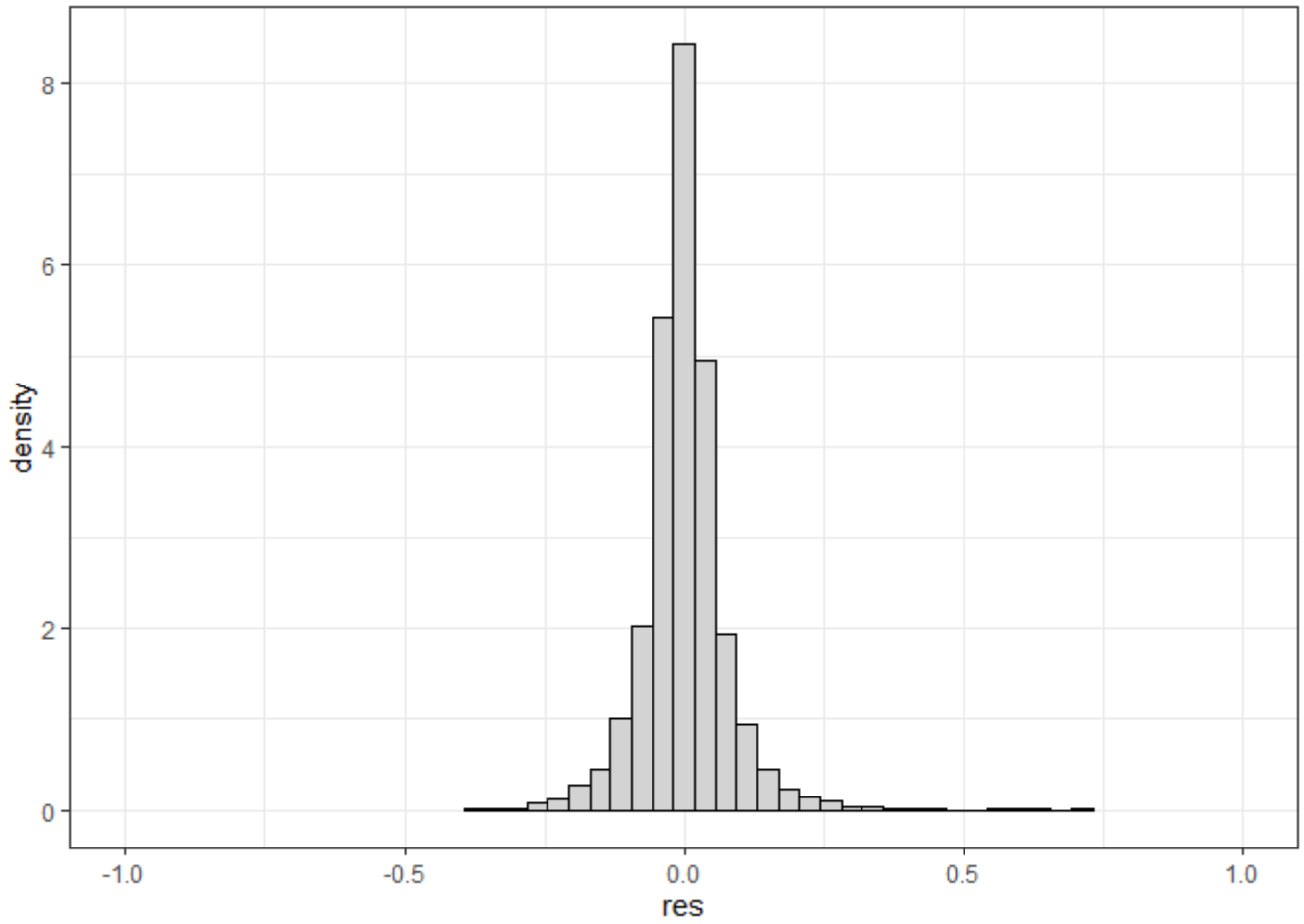

100
75
50
25
0
0
25
50
75
100
.pred
7
5
3


100
75
50
25
0
0
25
50
75
100
res
2
1
0
-1

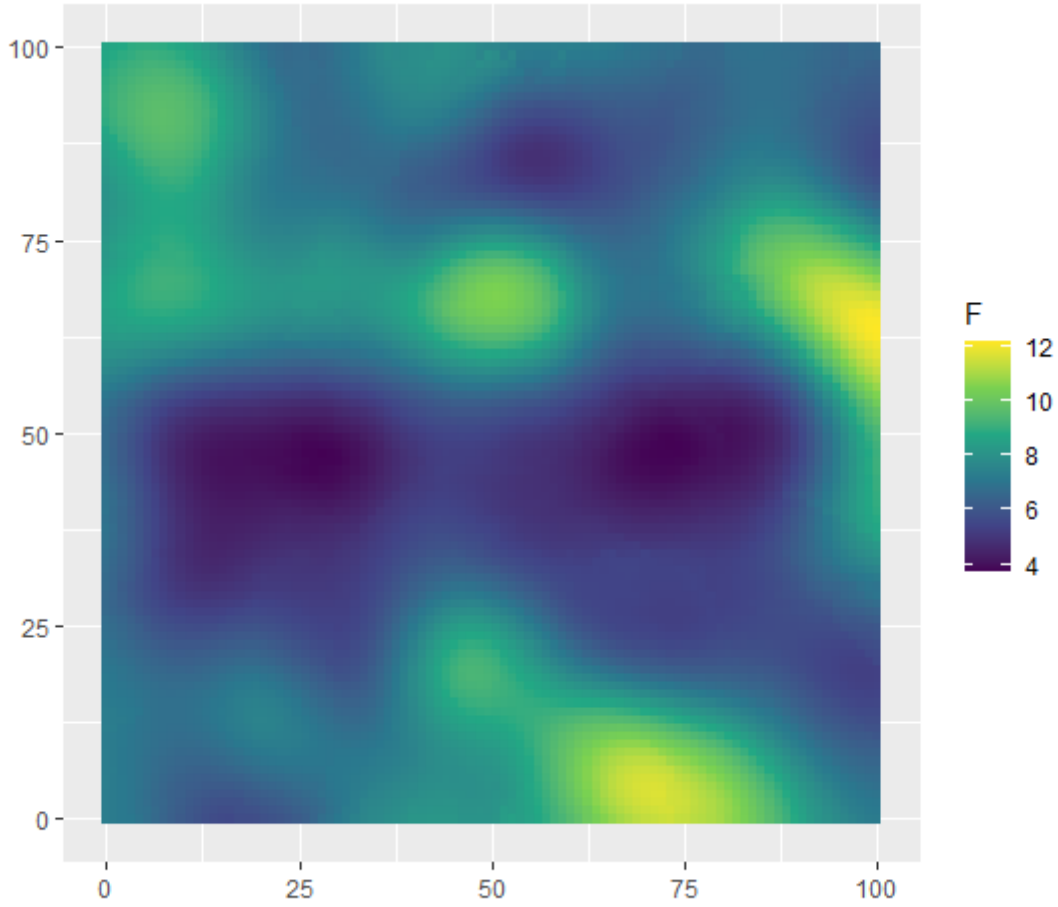

density
2
1
0
-1.0
-0.5
0.0
0.5
1.0
res

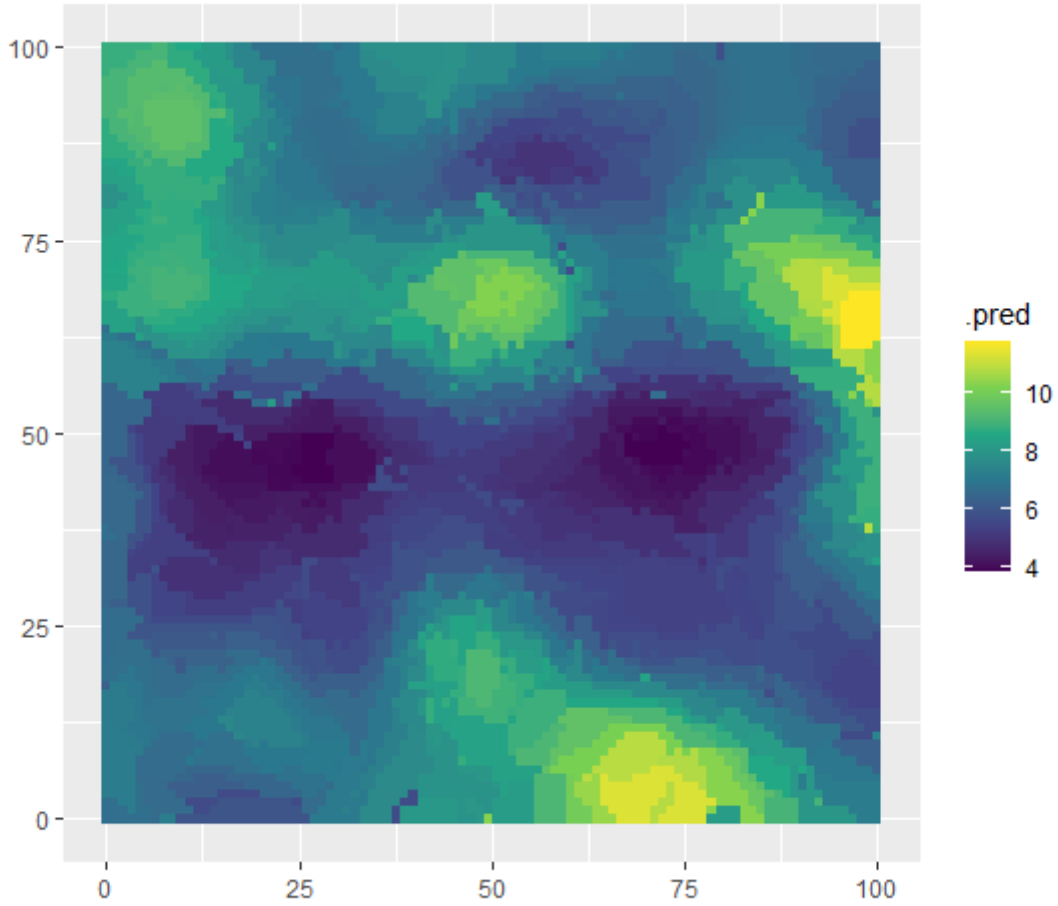

100
75
50
25
0
0
25
50
75
100
.pred
7
5
3

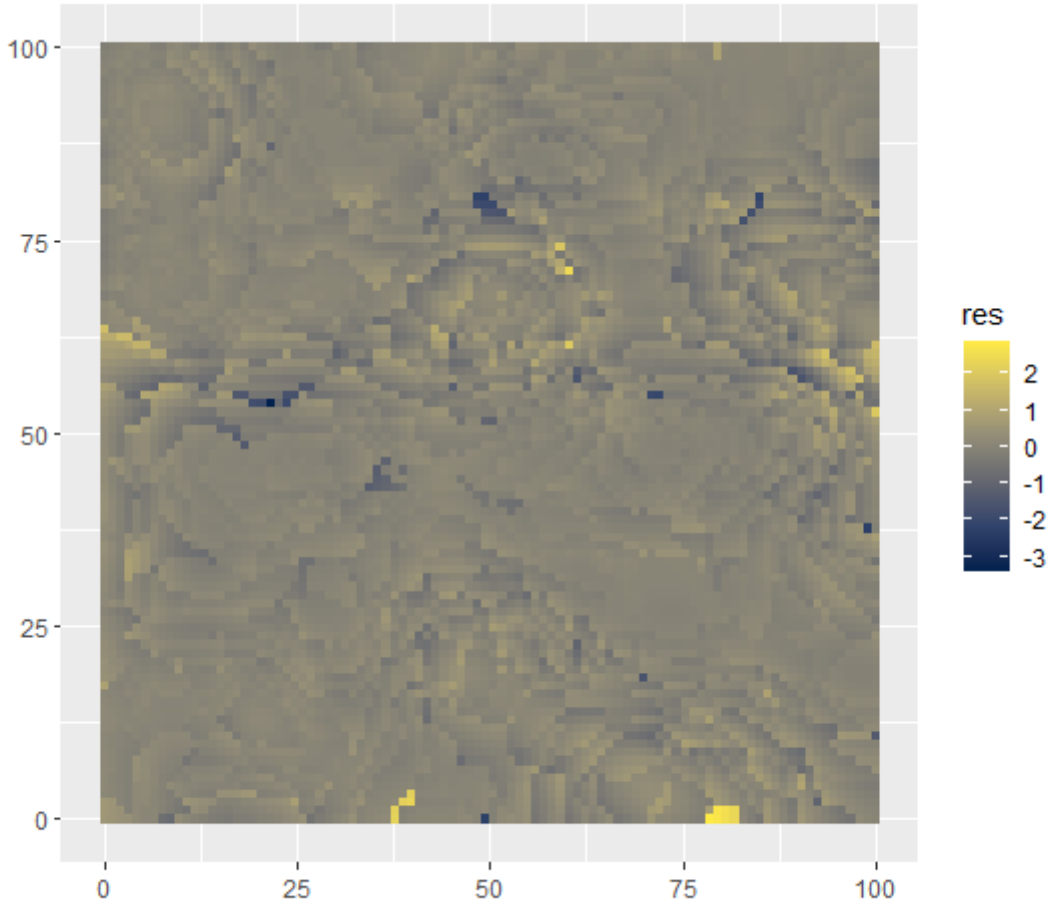

100
75
50
25
0
0
25
50
75
100
res
0.50
0.25
0.00
-0.25

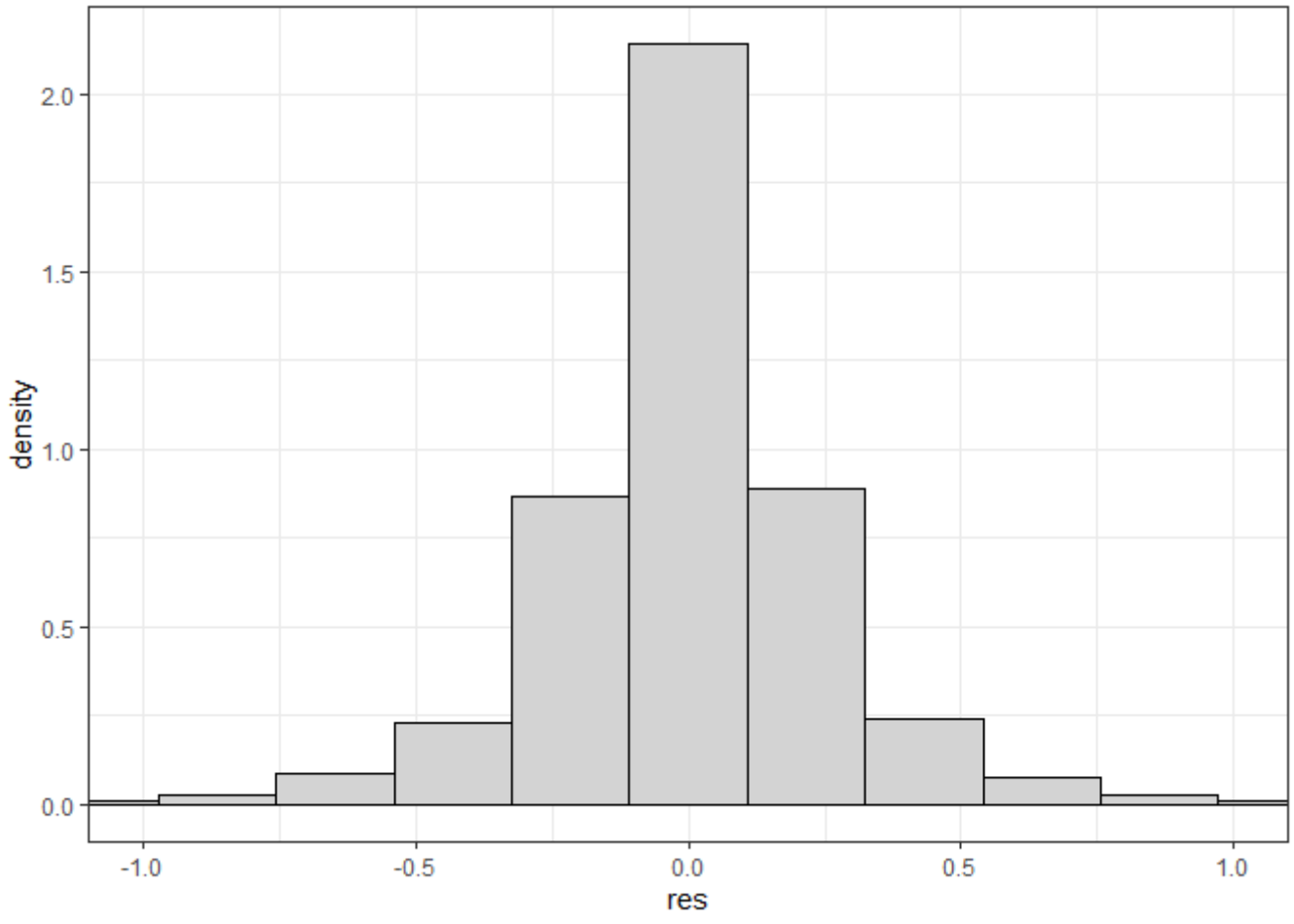


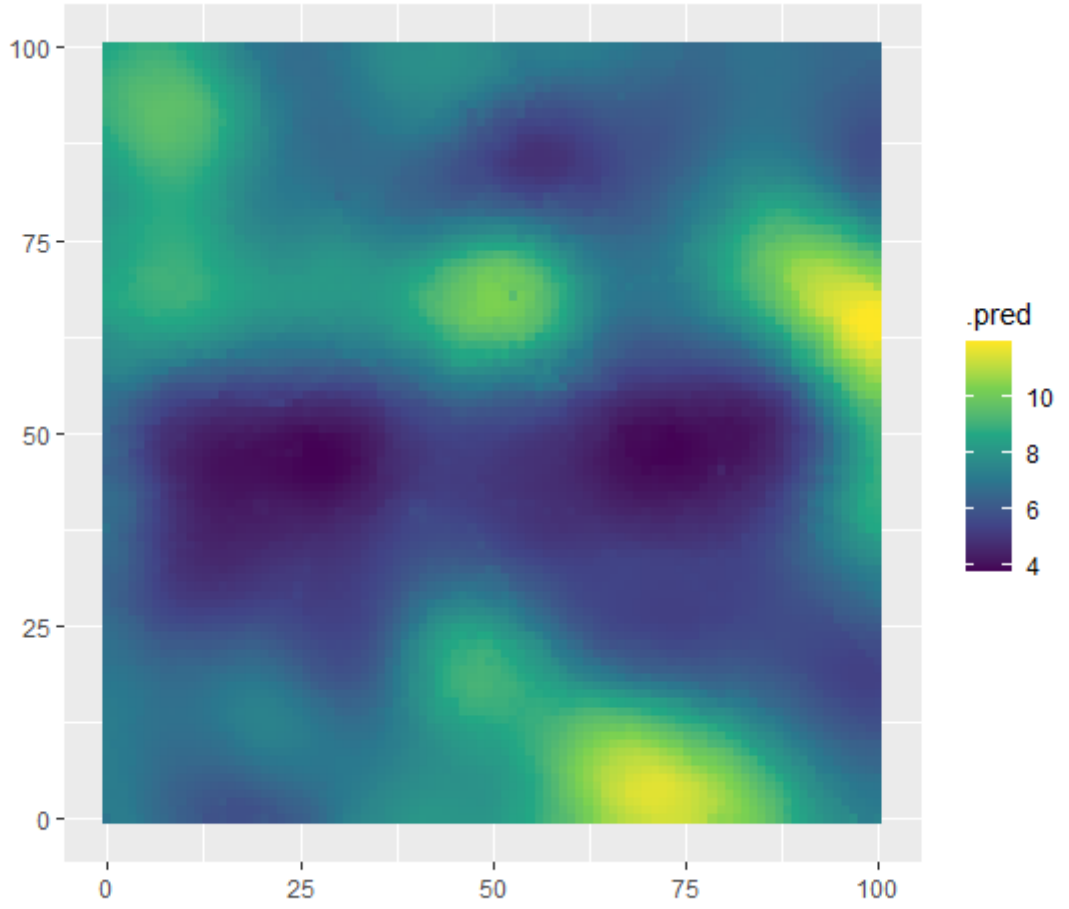

```
#> [1] "2017-02-17"
```

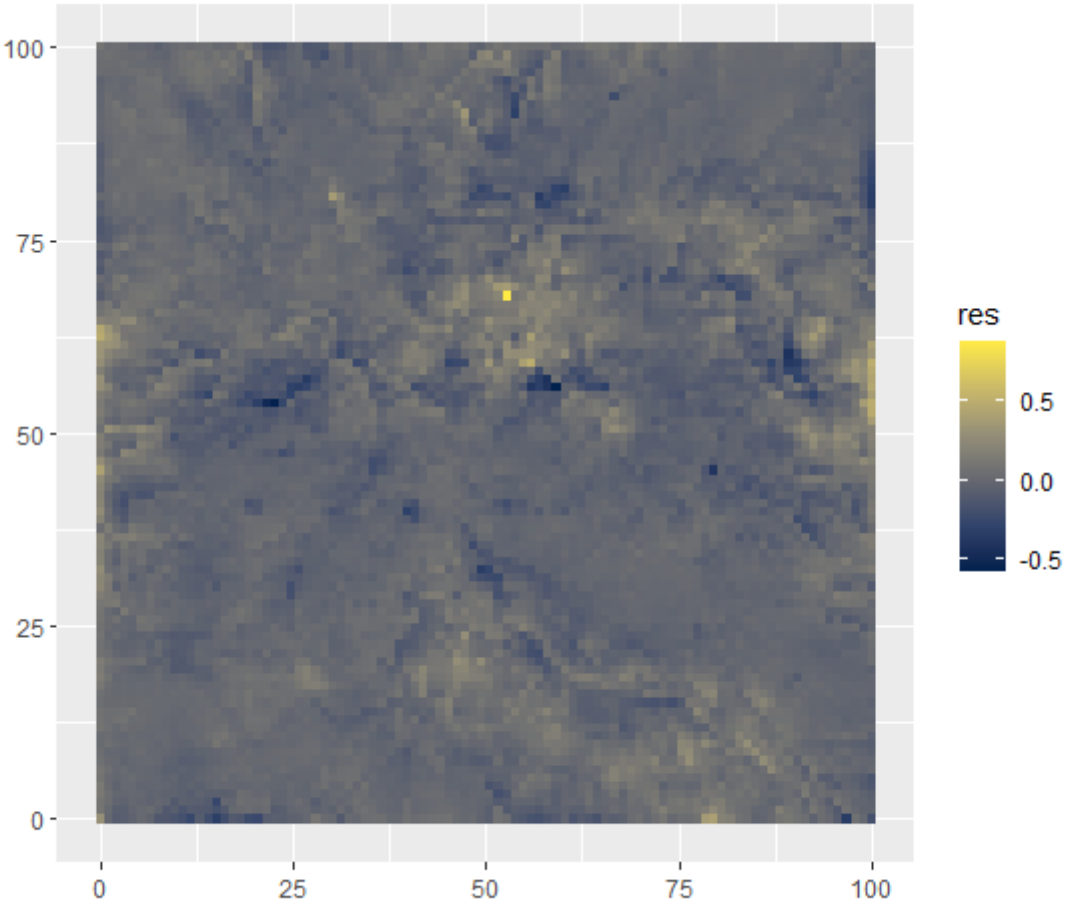

100
75
50
25
0
0
25
50
75
100
F
12
10
8
6
4

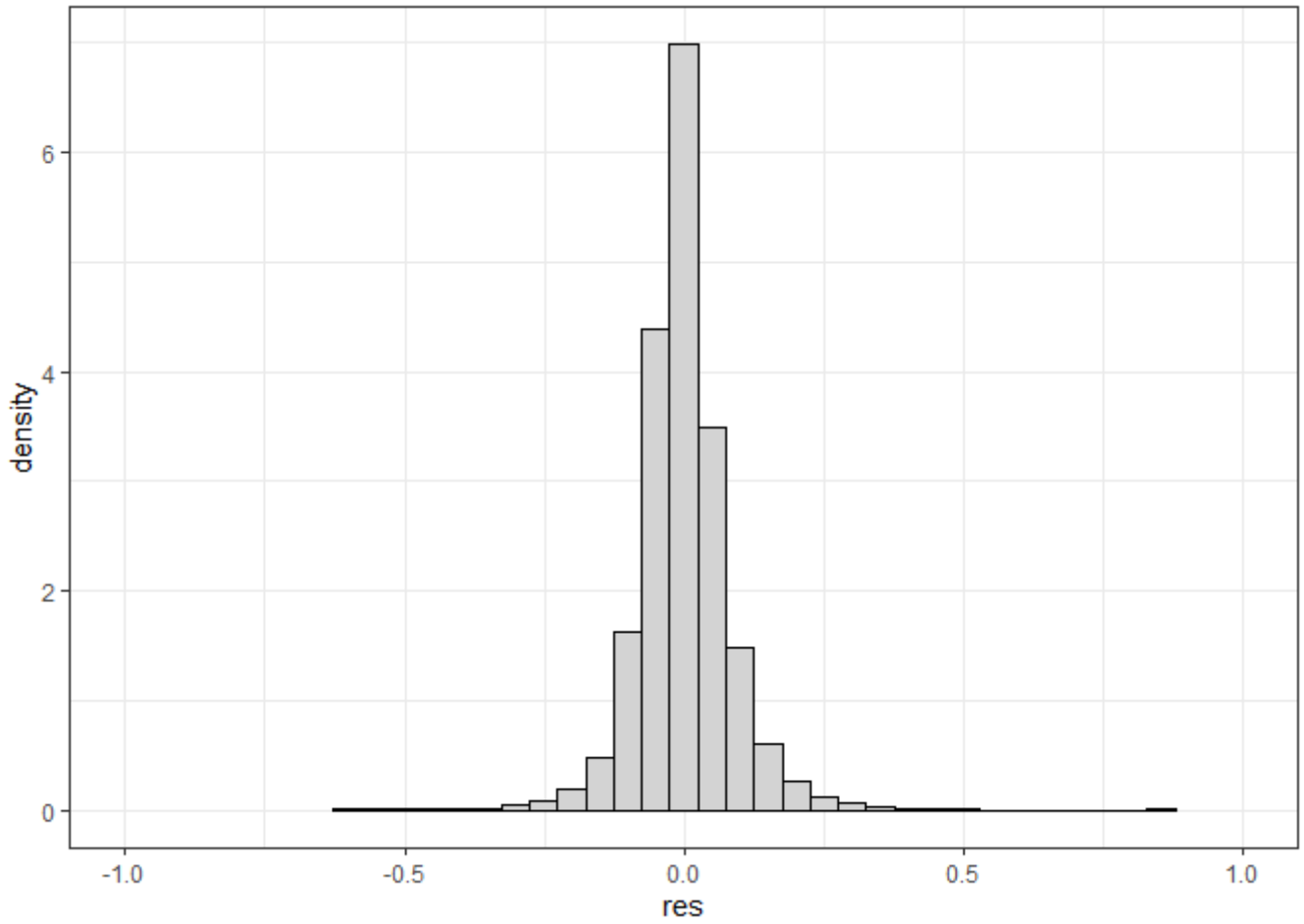

100
75
50
25
0
0
25
50
75
100
.pred
10
8
6
4


100
75
50
25
0
0
25
50
75
100
res
2
1
0
-1
-2
-3

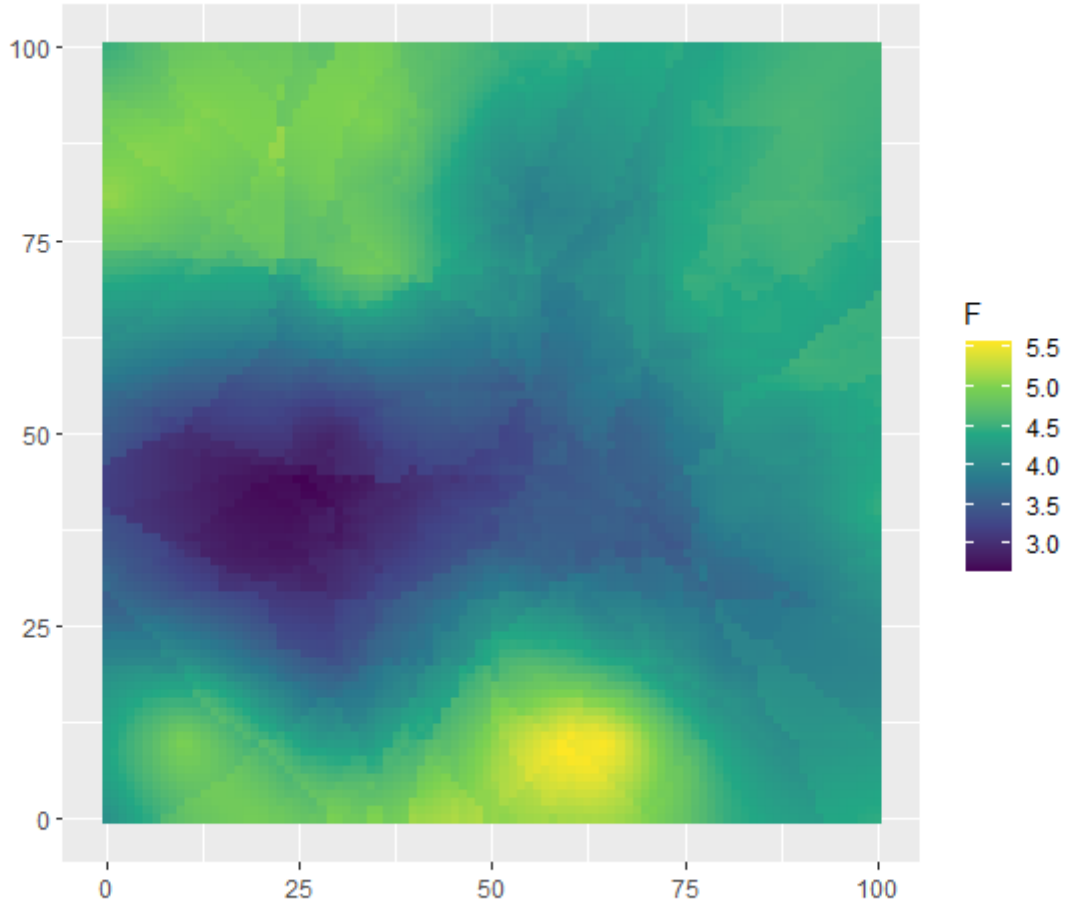

2.0
1.5
1.0
0.5
0.0
density
-1.0
-0.5
0.0
0.5
1.0
res

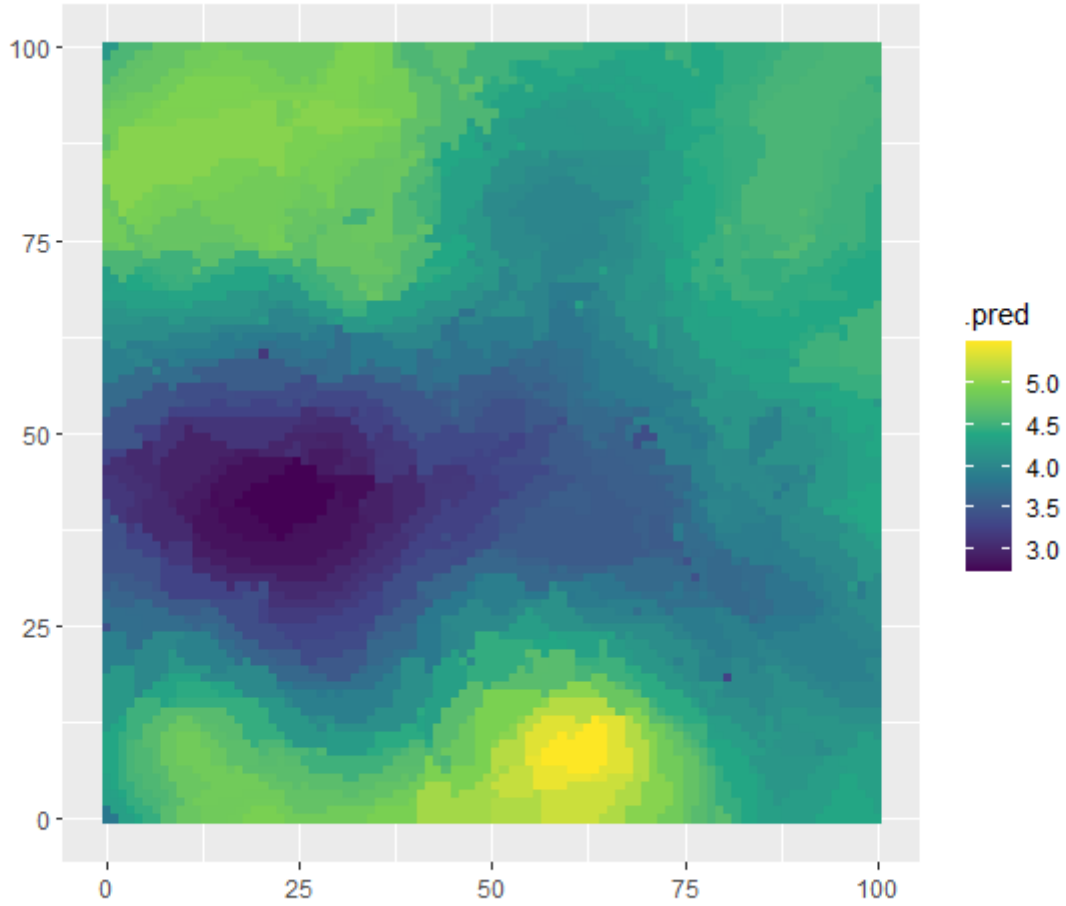

100
75
50
25
0
0
25
50
75
100
.pred
10
8
6
4

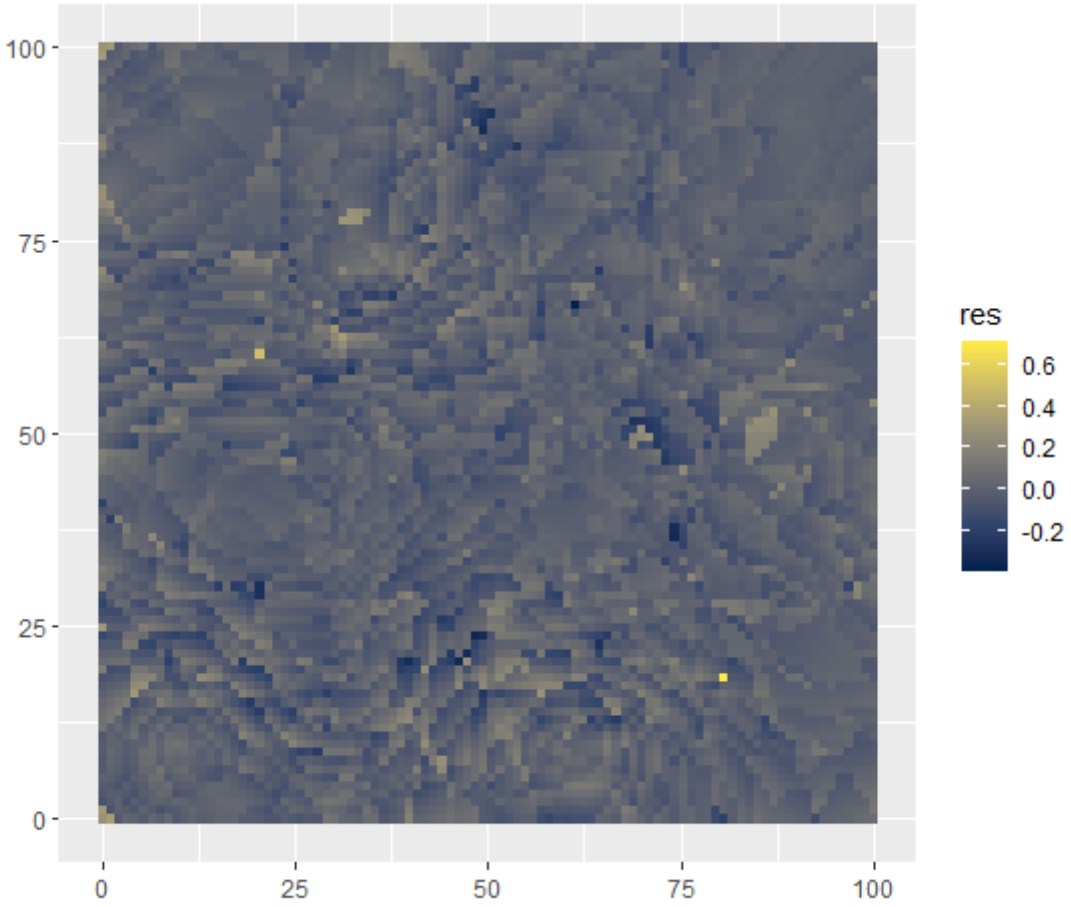

100
75
50
25
0
0
25
50
75
100
res
0.5
0.0
-0.5

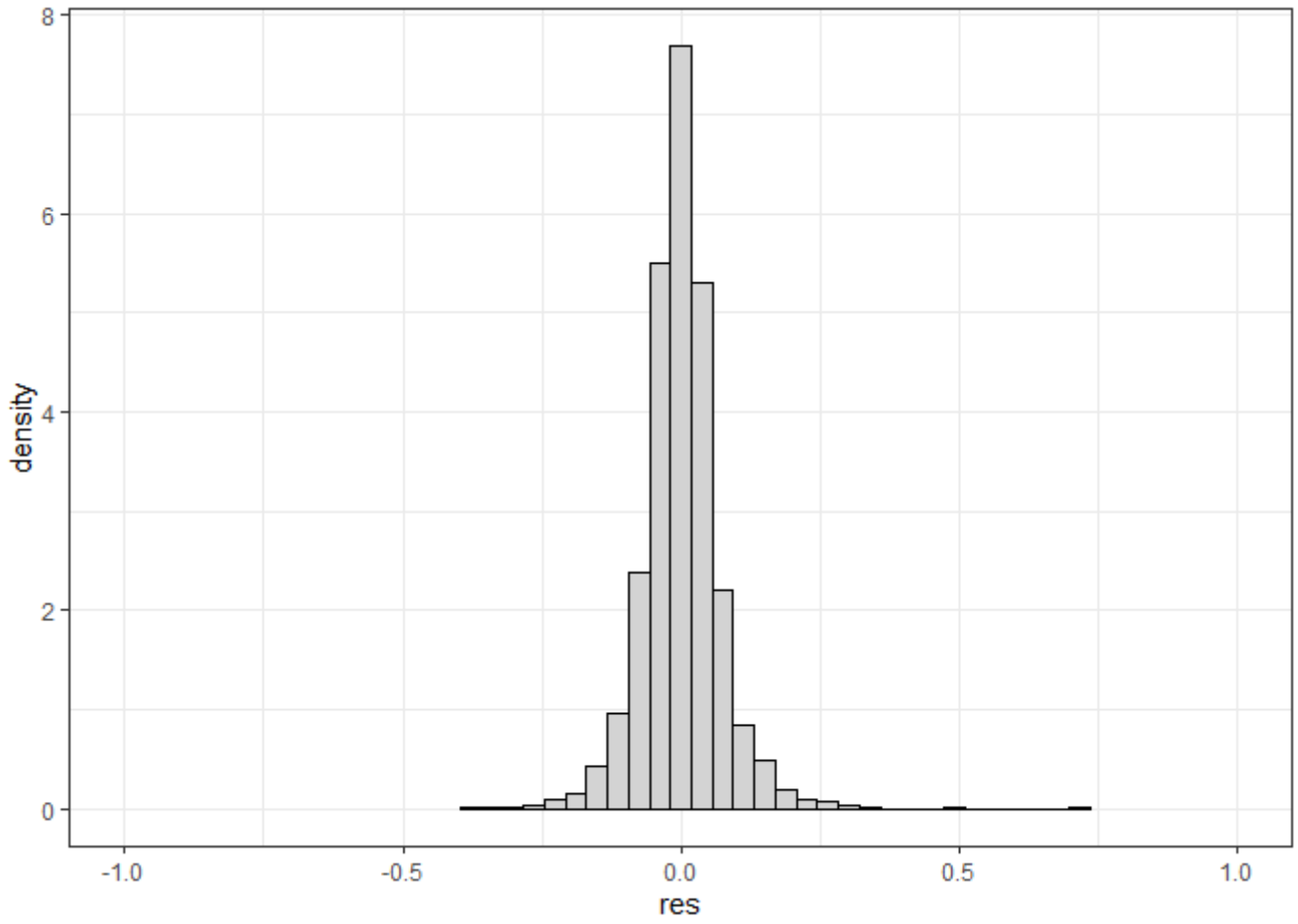


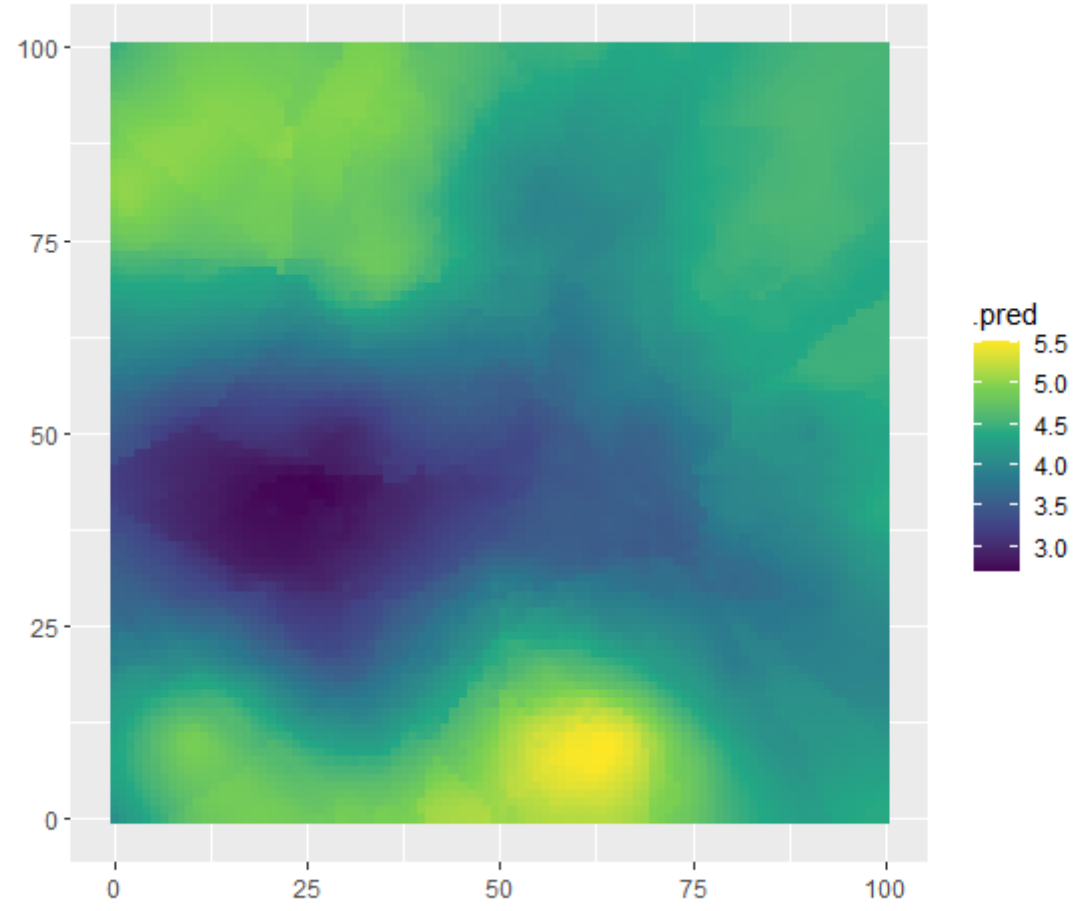

```
#> [1] "2017-06-17"
```

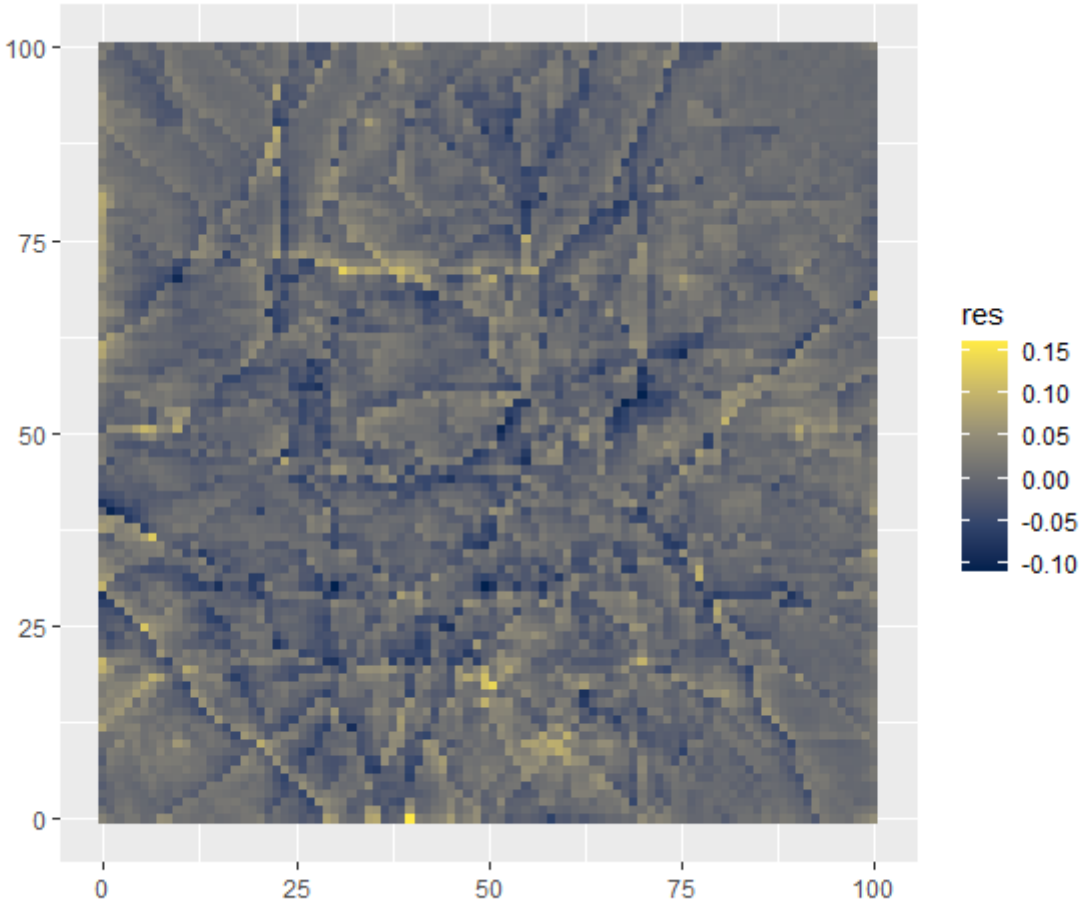

100
75
50
25
0
0
25
50
75
100
F
5.5
5.0
4.5
4.0
3.5
3.0

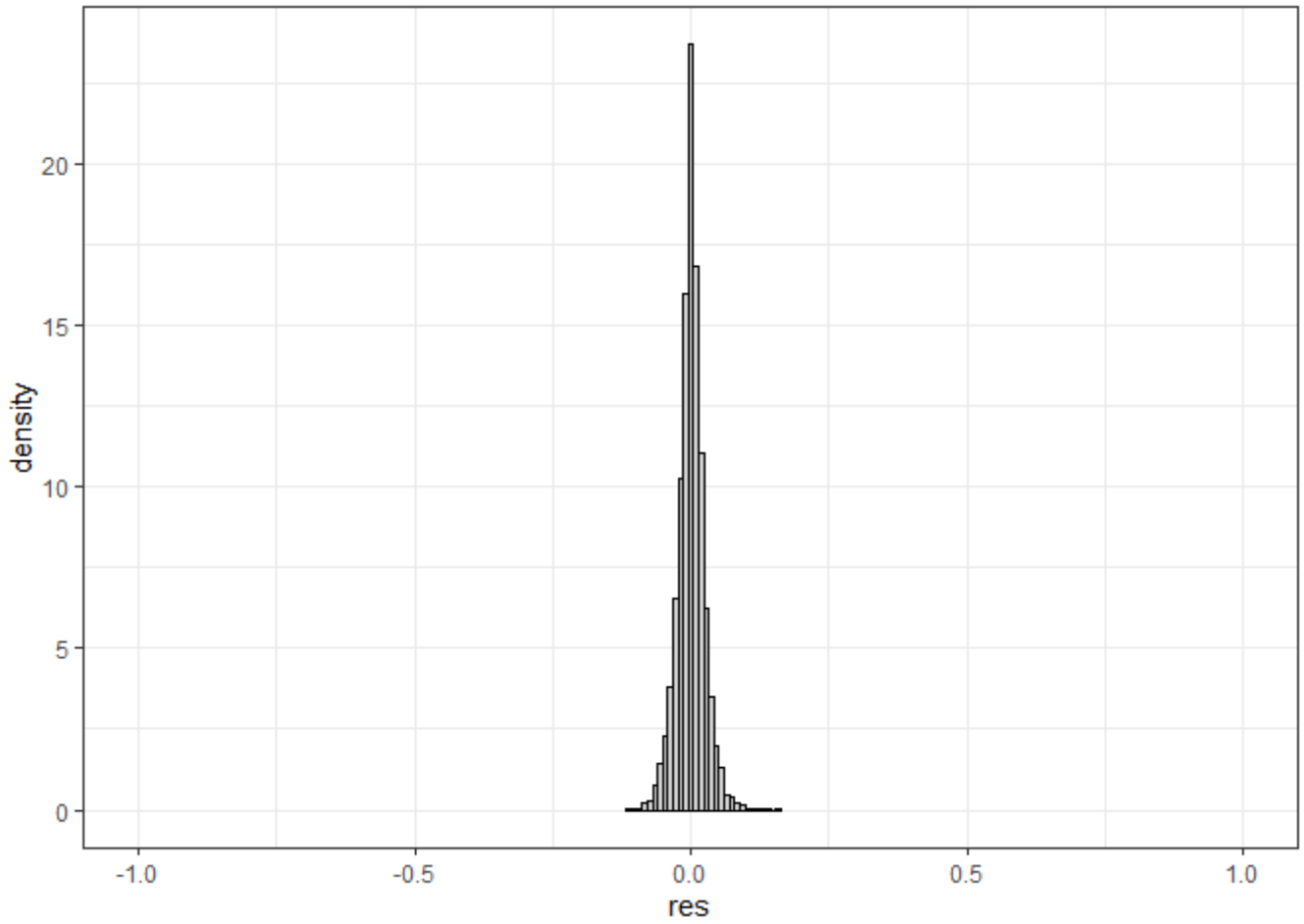

100
75
50
25
0
0
25
50
75
100
.pred
5.0
4.5
4.0
3.5
3.0


100
75
50
25
0
0
25
50
75
100
res
0.6
0.4
0.2
0.0
-0.2


8
6
4
2
0
density
-1.0
-0.5
0.0
0.5
1.0
res


100
75
50
25
0
0
25
50
75
100
.pred
5.5
5.0
4.5
4.0
3.5
3.0

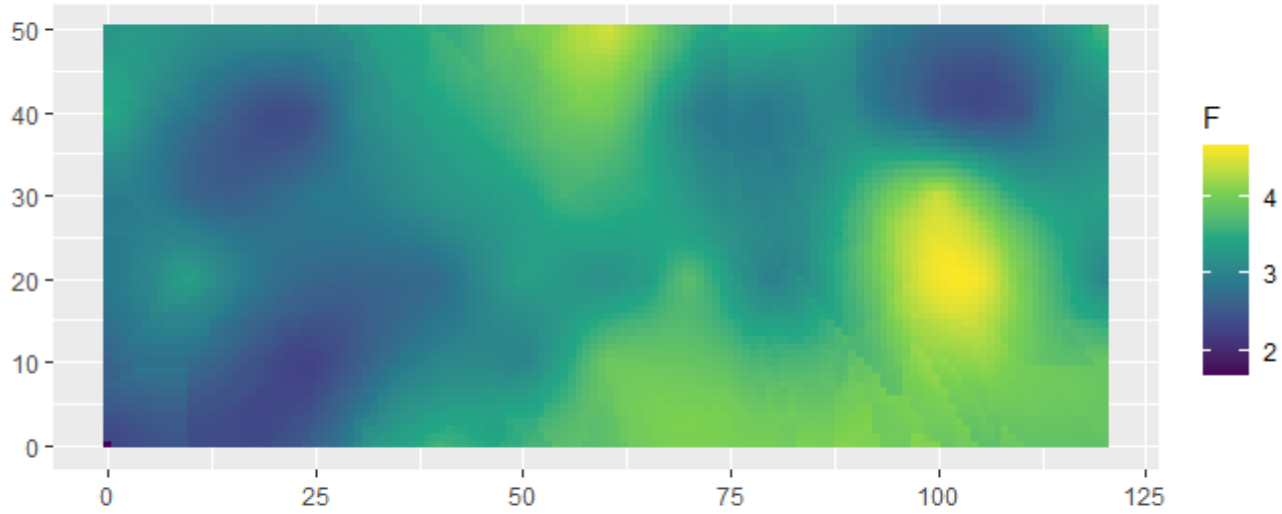

100
75
50
25
0
0
25
50
75
100
res
0.15
0.10
0.05
0.00
-0.05
-0.10

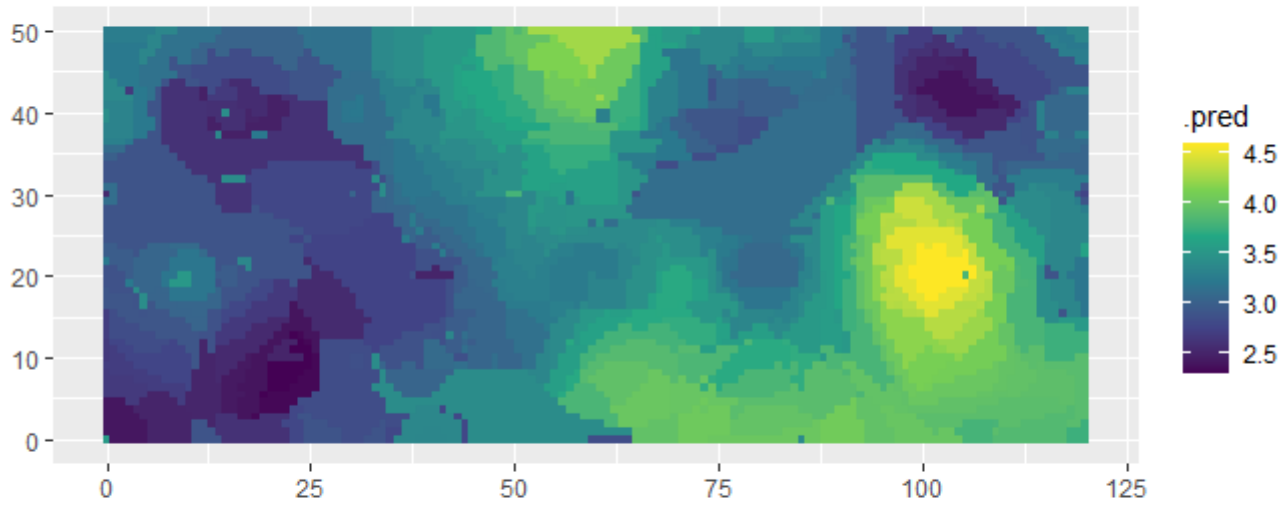

density
20
15
10
5
0
-1.0
-0.5
0.0
0.5
1.0
res

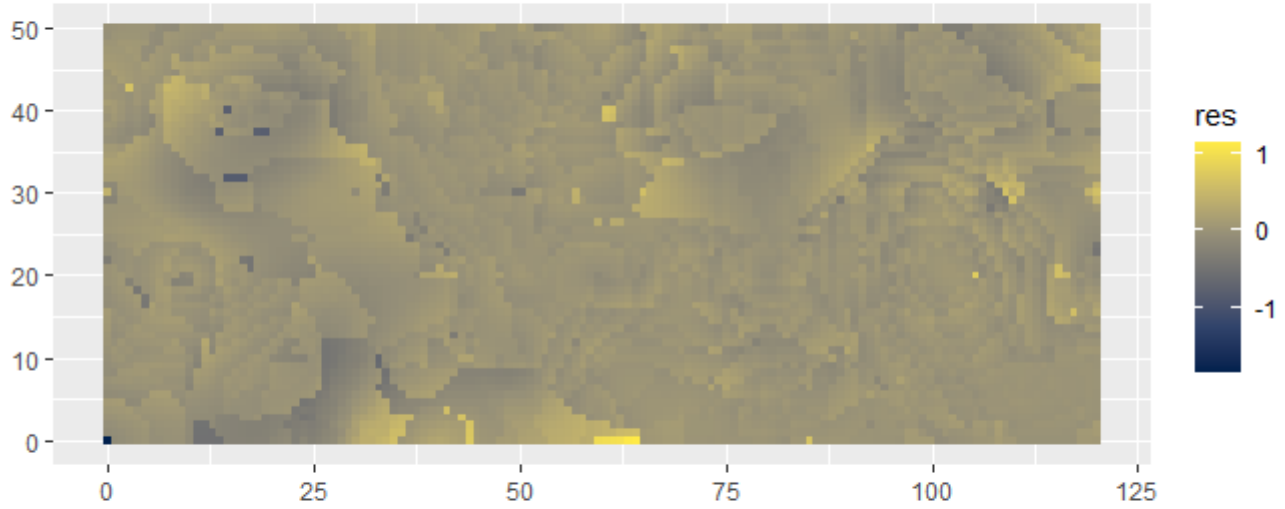
Criando os preditos para cada dia - SILVIPASTORIL

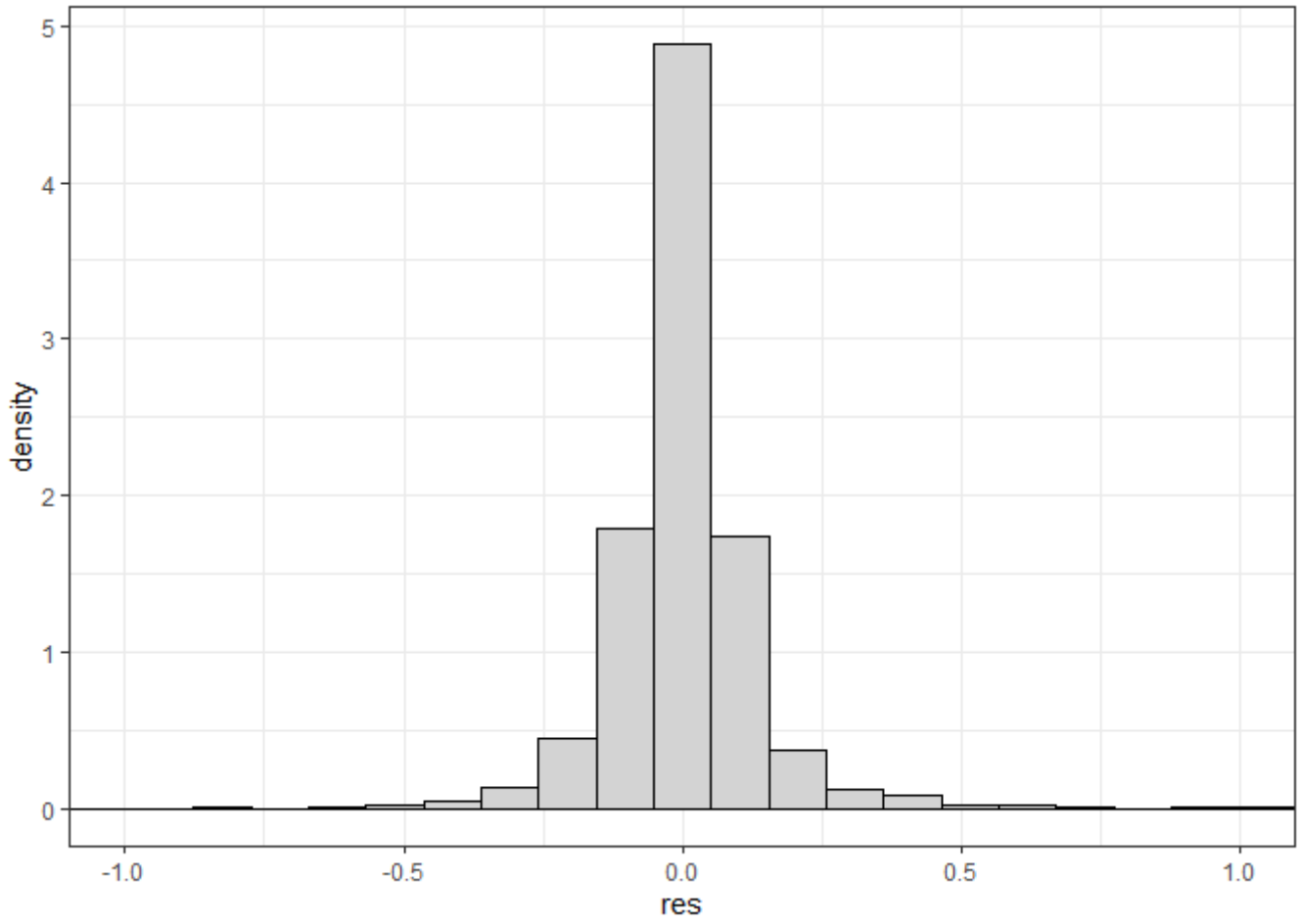
```
dias <- data_sp$data %>% unique()
for(i in seq_along(dias)){
  di <- dias[i]
  data_sp %>% filter(data == di)
  file_models_sp <- list.files("models-3", pattern = "SP")
  file_models_sp <- grep(paste0(di), file_models_sp,value = TRUE)

  fco2_modelo_load_dt <- read_rds(
    paste0("models-3/",grep("dt",file_models_sp,value=TRUE)
    ))
  fco2_modelo_load_rf <- read_rds(
    paste0("models-3/",grep("rf",file_models_sp,value=TRUE)
    ))
  # fco2_modelo_load_xgb <- read_rds(
  #   paste0("models-3/",grep("xgb",file_models_sp,value=TRUE)
  #   ))

  yp_dt <- predict(fco2_modelo_load_dt, new_data = data_sp %>%
                     filter(data == di))
  yp_rf <- predict(fco2_modelo_load_rf, new_data = data_sp %>%
                     filter(data == di))
  # yp_xgb <- predict(fco2_modelo_load_xgb, new_data = data_sp %>%
  #                filter(data == di))
dis_y <- 50/55
dis_x <- 120/131
grid <- expand.grid(Y=seq(0,50,dis_y), X=seq(0,120,dis_x))

  mp_krg <- tibble(grid, data_sp %>%
                     filter(data == di)) %>%
    ggplot(aes(x=X,y=Y)) +
    geom_tile(aes(fill = F)) +
    scale_fill_viridis_c() +
    coord_equal()+labs(x="",y="")

  mp_dt <- tibble(grid, data_sp %>%
                    filter(data == di), yp_dt) %>%
    ggplot(aes(x=X,y=Y)) +
    geom_tile(aes(fill = .pred)) +
    scale_fill_viridis_c() +
    coord_equal() +labs(x="",y="")

  mp_rf <- tibble(grid, data_sp %>%
                    filter(data == di), yp_rf) %>%
    ggplot(aes(x=X,y=Y)) +
    geom_tile(aes(fill = .pred)) +
    scale_fill_viridis_c() +
    coord_equal() +labs(x="",y="")

   mp_dt_res <- tibble(grid, data_sp %>%
```

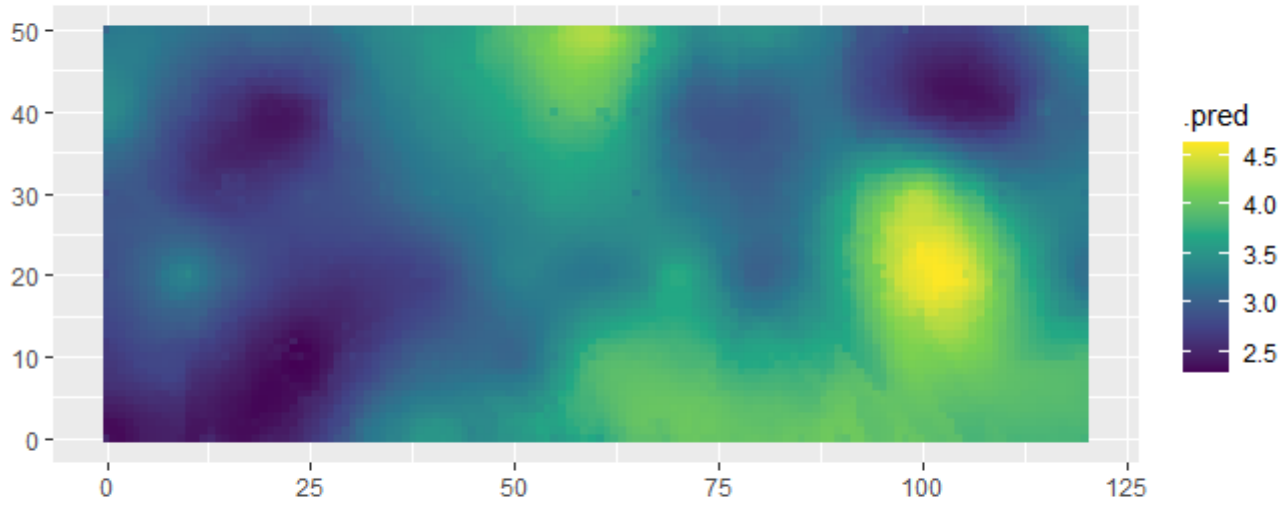
```
                    filter(data == di), yp_dt) %>%
    mutate(res = F - .pred) %>%
    ggplot(aes(x=X,y=Y)) +
    geom_tile(aes(fill = res)) +
    scale_fill_viridis_c(option = "E") +
    coord_equal() +labs(x="",y="")

   hist_dt_res <- tibble(grid, data_sp %>%
                          filter(data == di), yp_dt) %>%
    mutate(res = F - .pred) %>%
    ggplot(aes(x=res, y=..density..)) +
    geom_histogram(color="black",fill="lightgray") +theme_bw()+
    coord_cartesian(xlim=c(-1,1))

  mp_rf_res <- tibble(grid, data_sp %>%
                    filter(data == di), yp_rf) %>%
    mutate(res = F - .pred) %>%
    ggplot(aes(x=X,y=Y)) +
    geom_tile(aes(fill = res)) +
    scale_fill_viridis_c(option = "E") +
    coord_equal() +labs(x="",y="")

  hist_rf_res <- tibble(grid, data_sp %>%
                          filter(data == di), yp_rf) %>%
    mutate(res = F - .pred) %>%
    ggplot(aes(x=res, y=..density..)) +
    geom_histogram(color="black",fill="lightgray") +theme_bw()+
    coord_cartesian(xlim=c(-1,1))



  print(di)
  print(mp_krg)
  print(mp_dt)
  print(mp_dt_res)
  print(hist_dt_res)

  print(mp_rf)
  print(mp_rf_res)
  print(hist_rf_res)
}
#> [1] "2017-02-03"
```

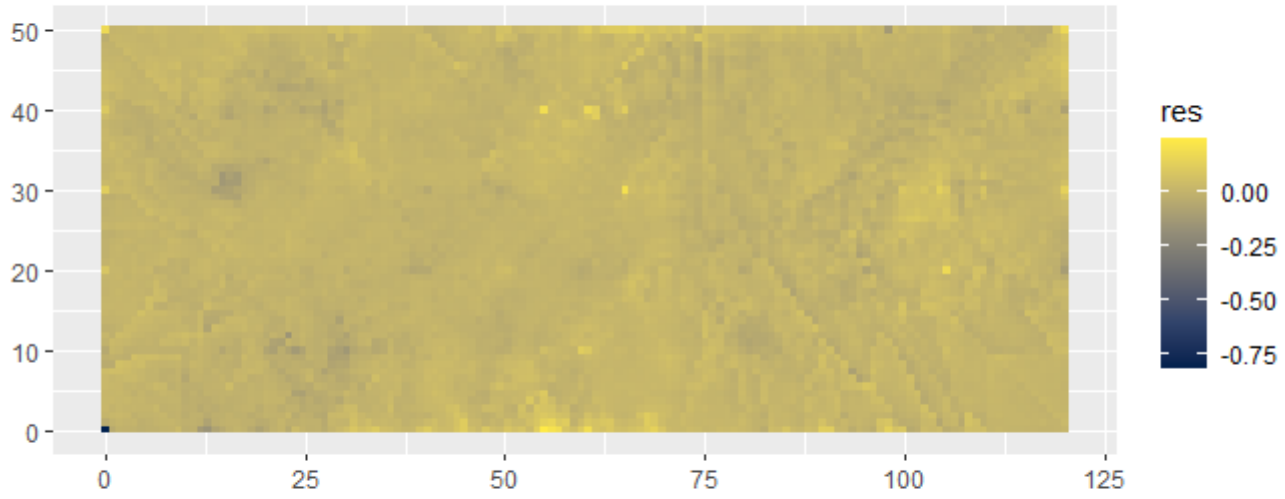

50
40
30
20
10
0
0
25
50
75
100
125
F
4
3
2

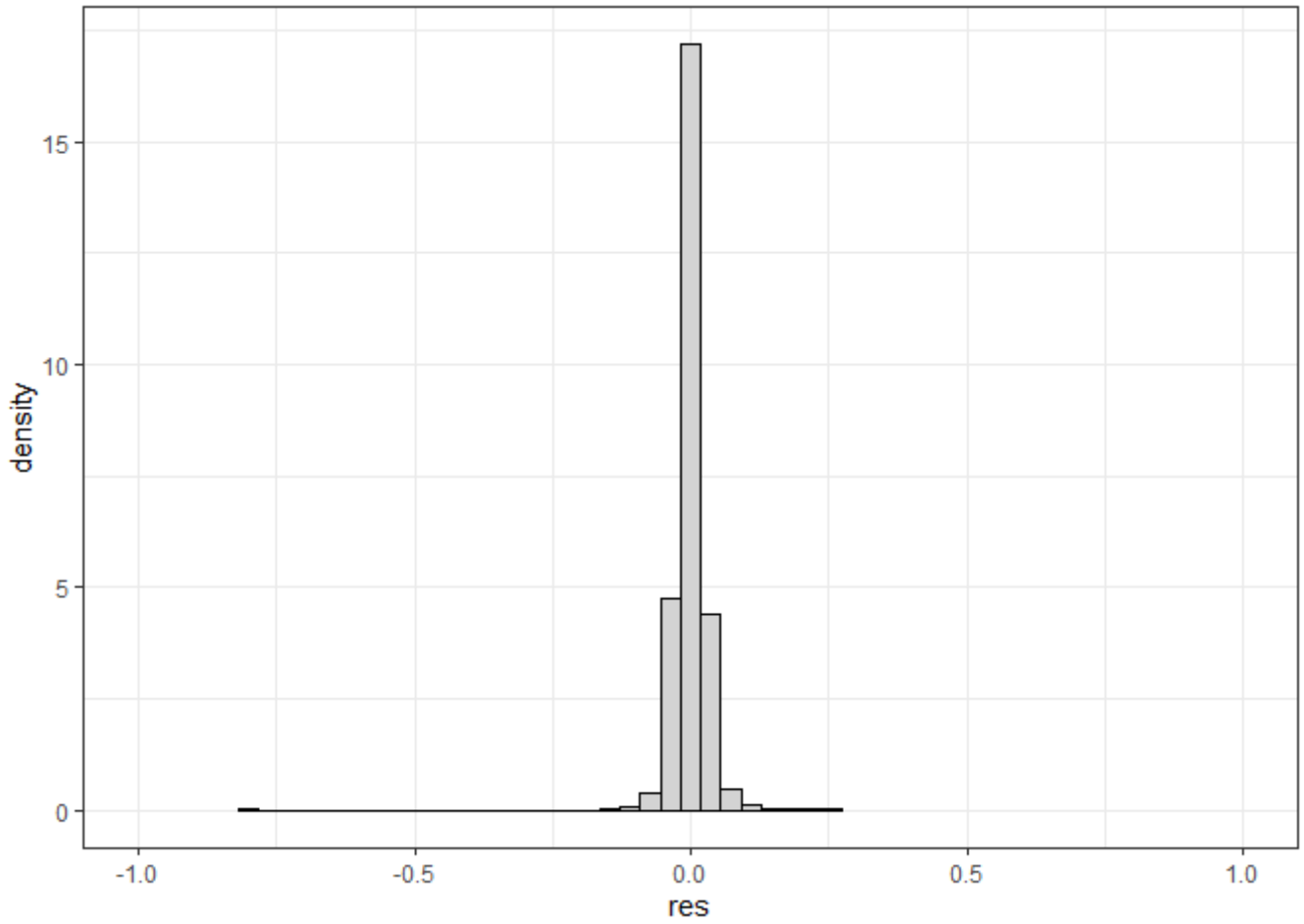

50
40
30
20
10
0
0
25
50
75
100
125
.pred
4.5
4.0
3.5
3.0
2.5


50
40
30
20
10
0
0
25
50
75
100
125
res
1
0
-1

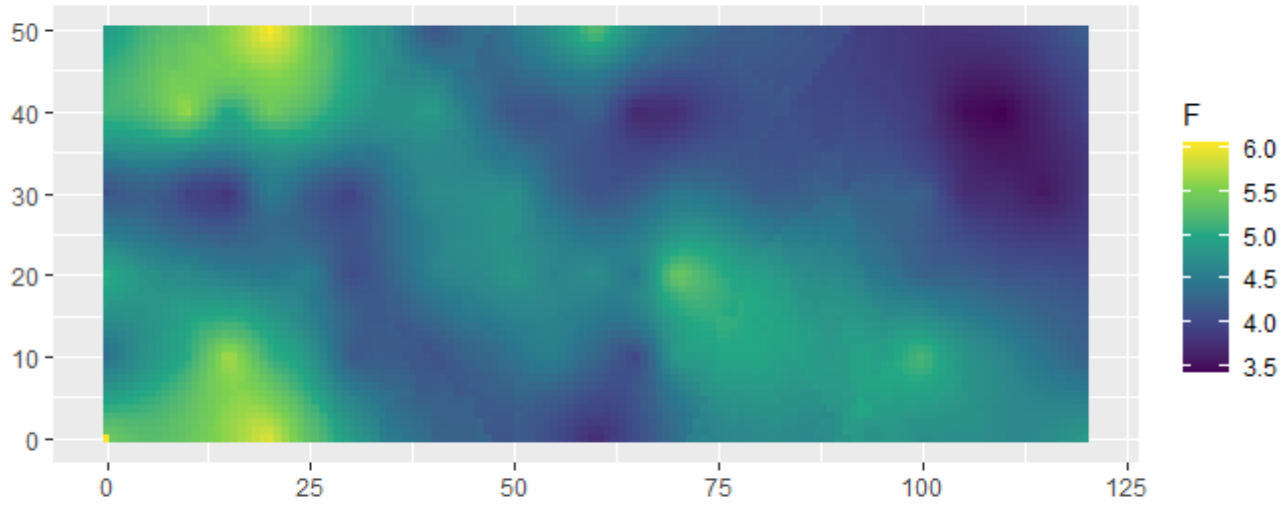

5
4
3
2
1
0
density
-1.0
-0.5
0.0
0.5
1.0
res

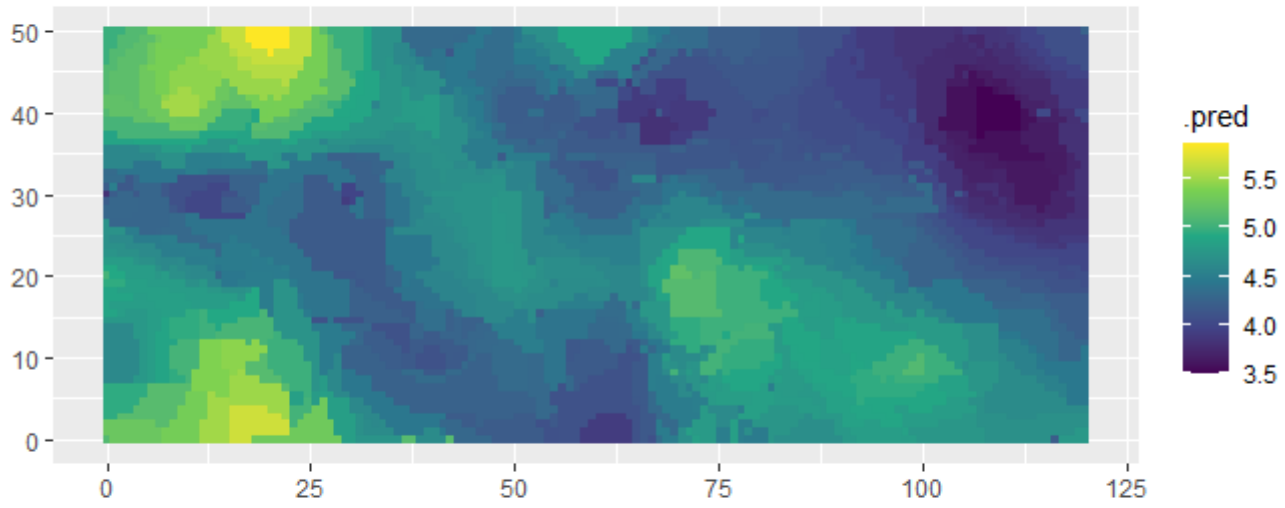

50
40
30
20
10
0
0
25
50
75
100
125
.pred
4.5
4.0
3.5
3.0
2.5

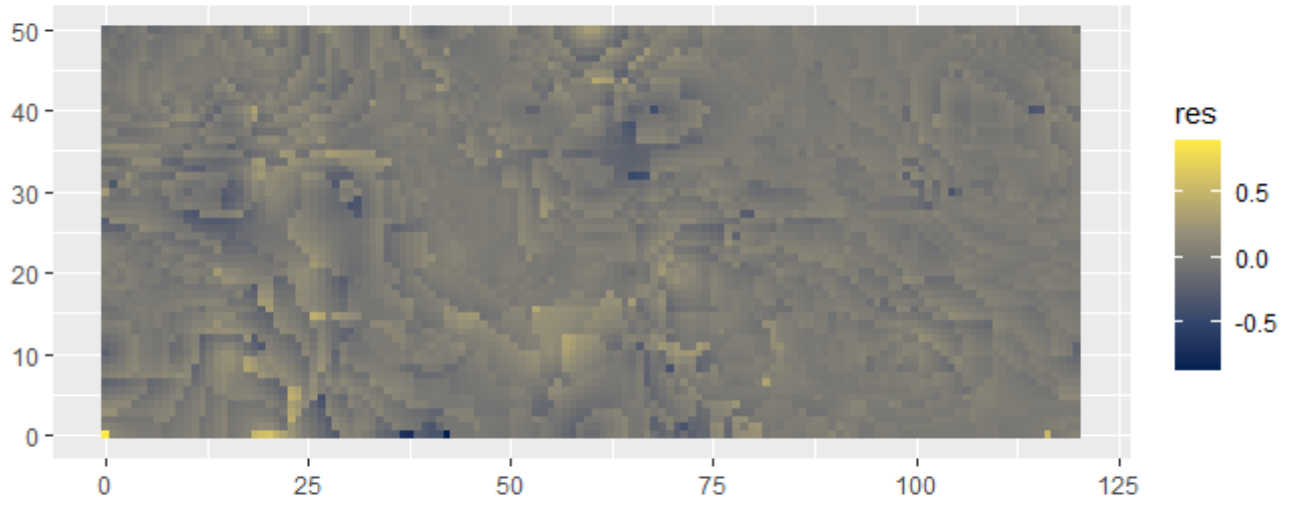

50
40
30
20
10
0
0
25
50
75
100
125
res
0.00
-0.25
-0.50
-0.75

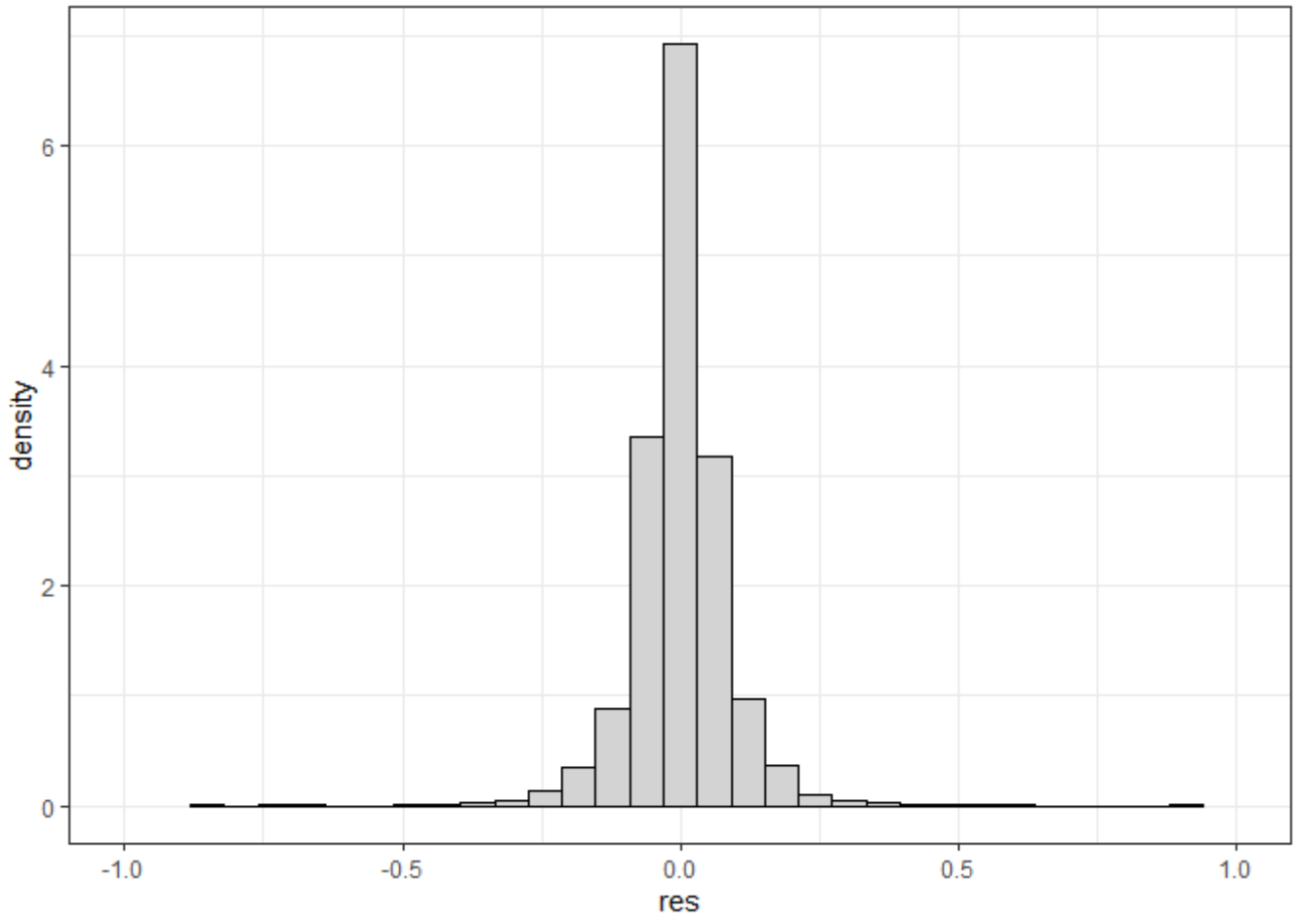


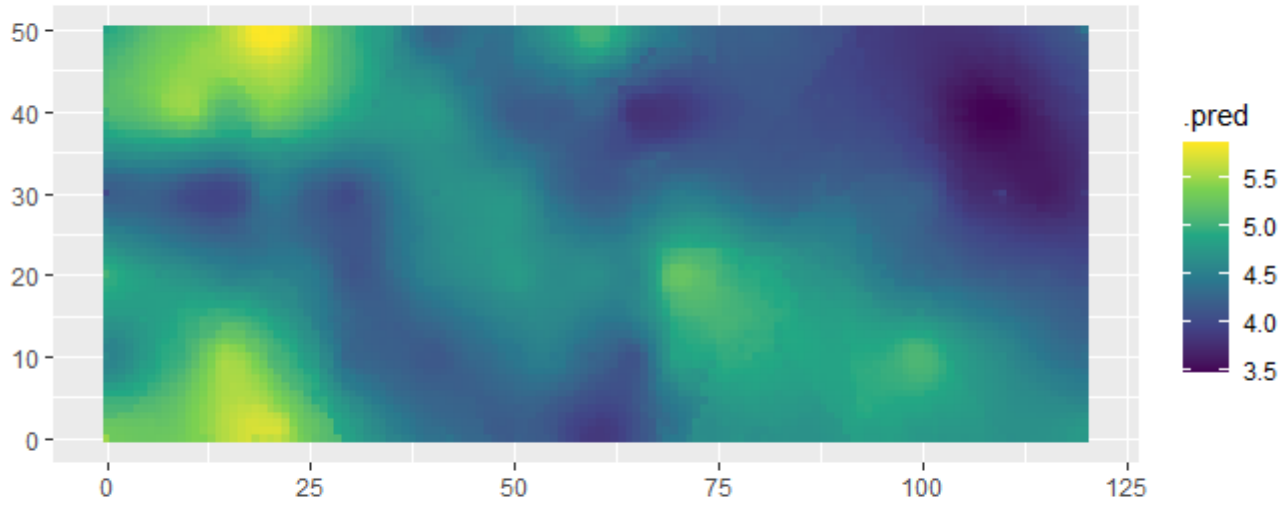

```
#> [1] "2017-03-03"
```

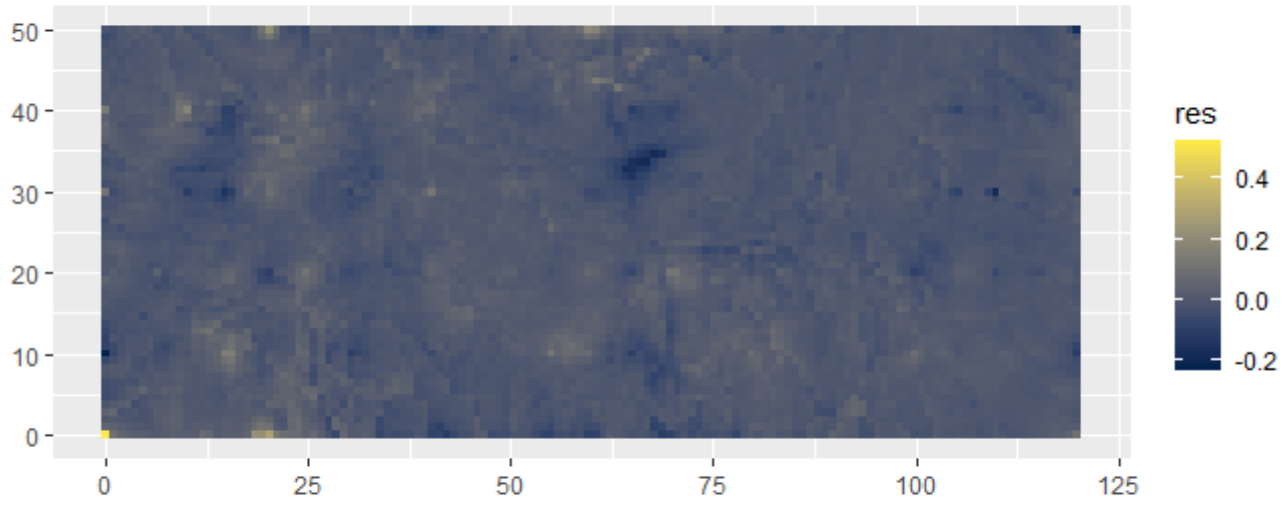

50
40
30
20
10
0
0
25
50
75
100
125
F
6.0
5.5
5.0
4.5
4.0
3.5

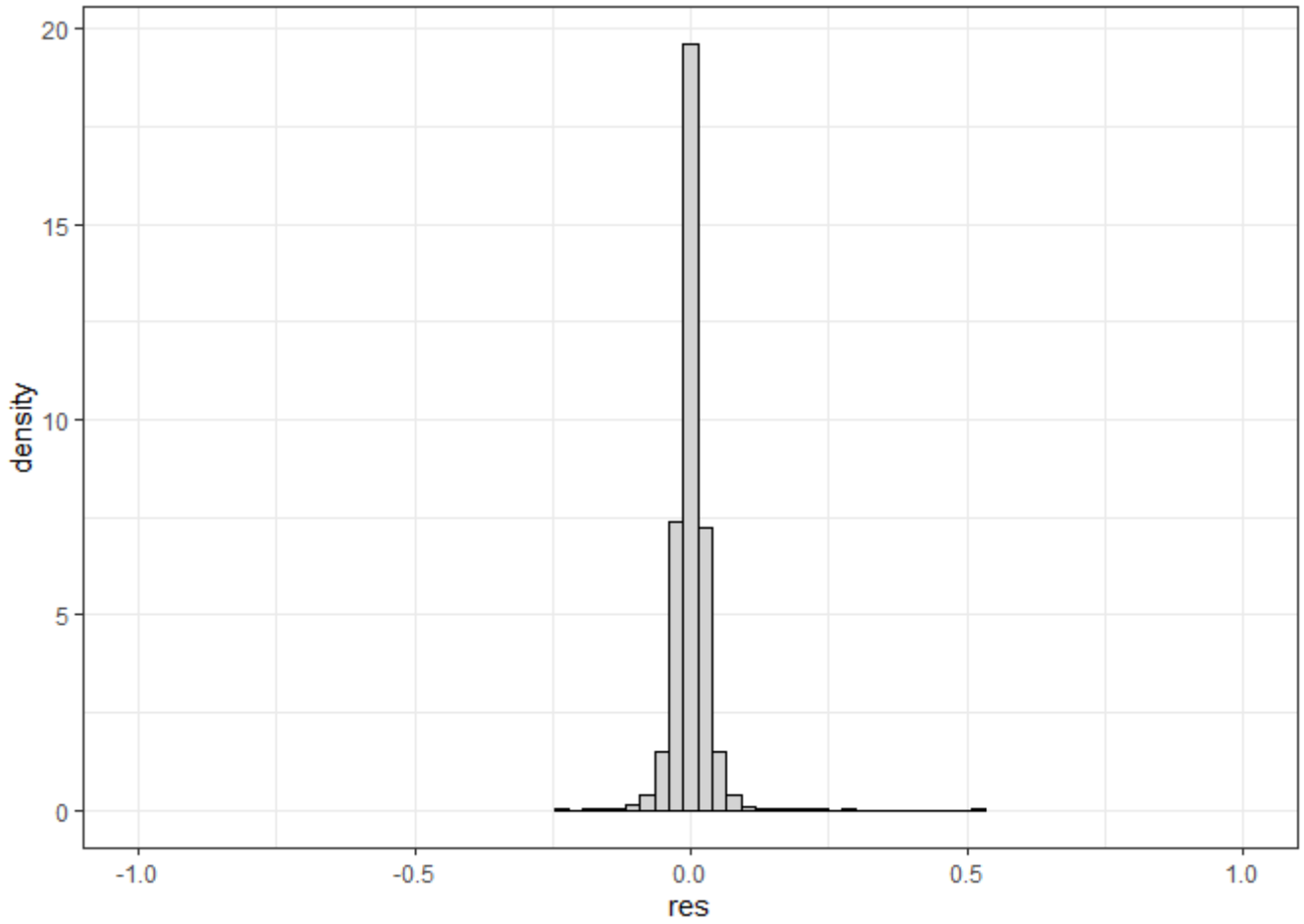

50
40
30
20
10
0
0
25
50
75
100
125
.pred
5.5
5.0
4.5
4.0
3.5


50
40
30
20
10
0
0
25
50
75
100
125
res
0.5
0.0
-0.5

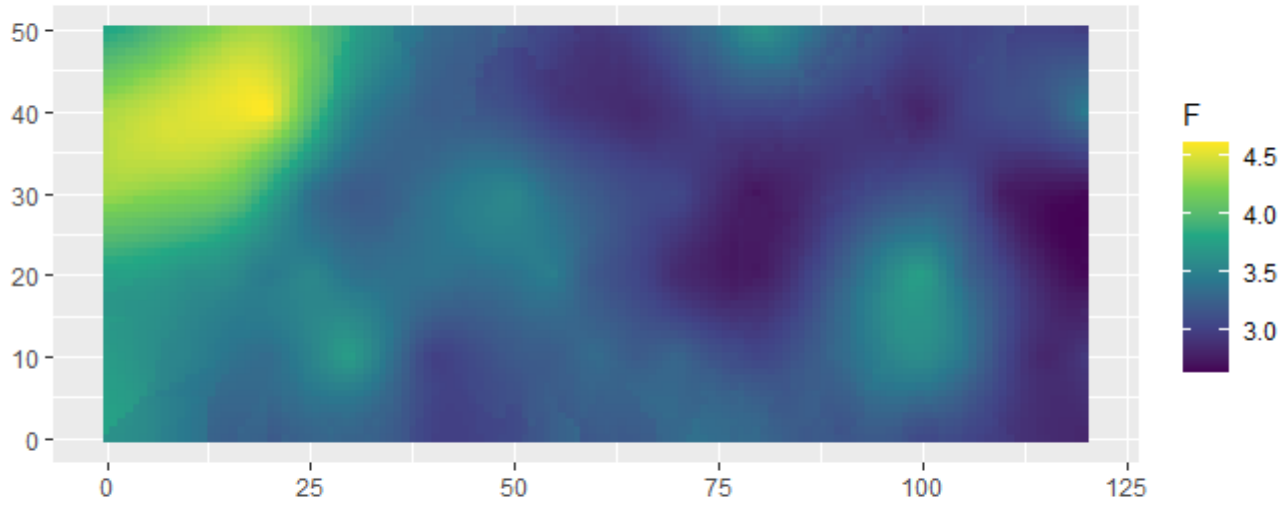

6
4
2
0
density
-1.0
-0.5
0.0
0.5
1.0
res

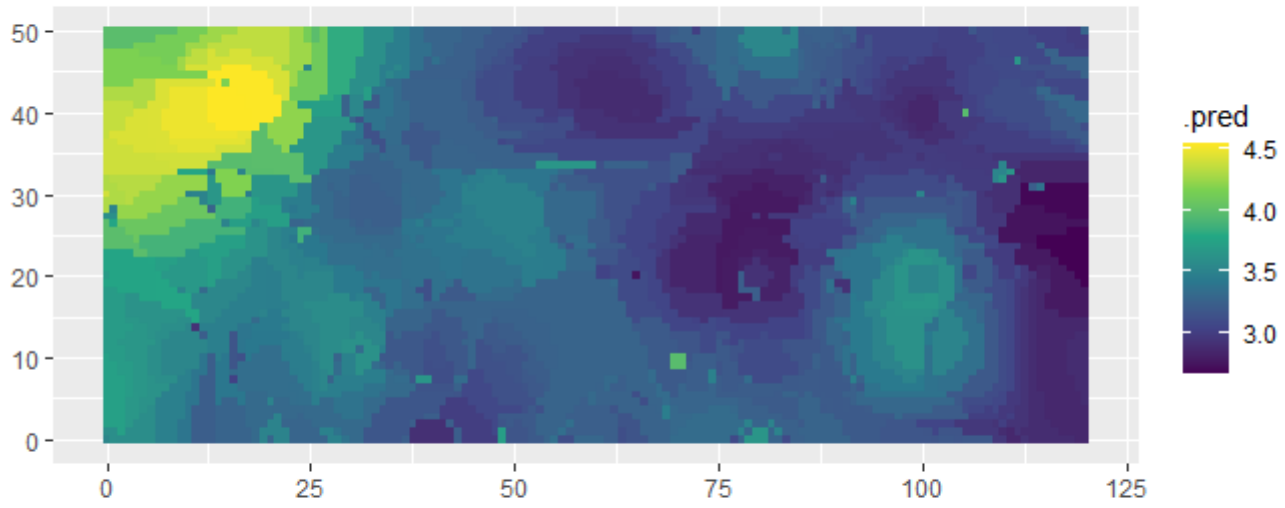

50
40
30
20
10
0
0
25
50
75
100
125
.pred
5.5
5.0
4.5
4.0
3.5

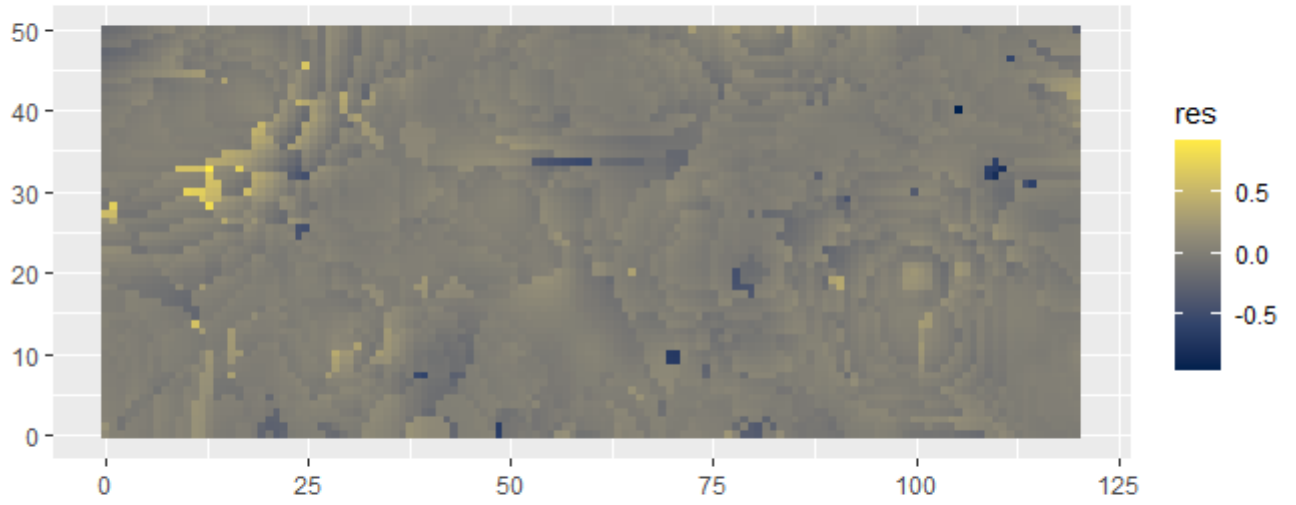

50
40
30
20
10
0
0
25
50
75
100
125
res
0.4
0.2
0.0
-0.2

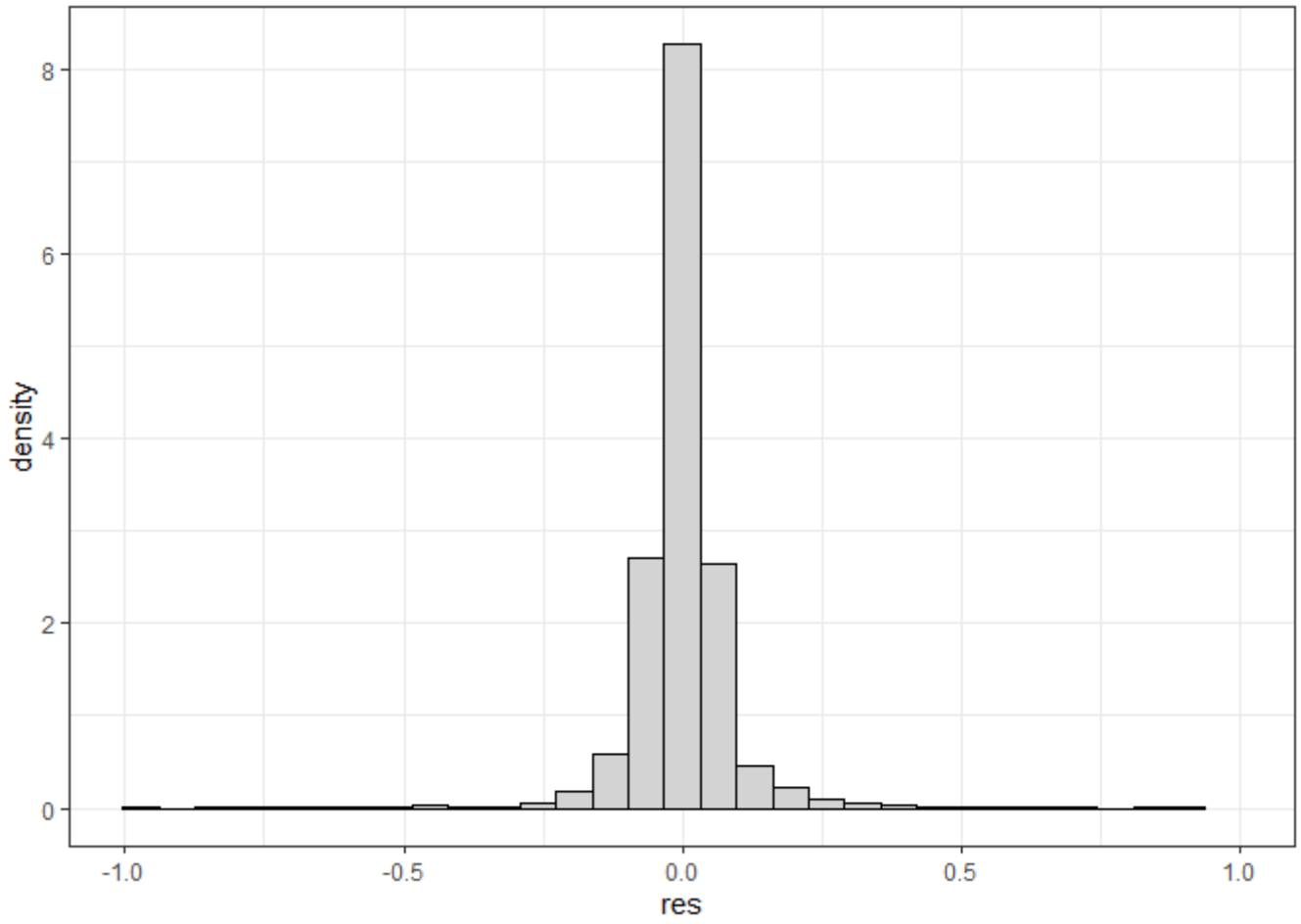


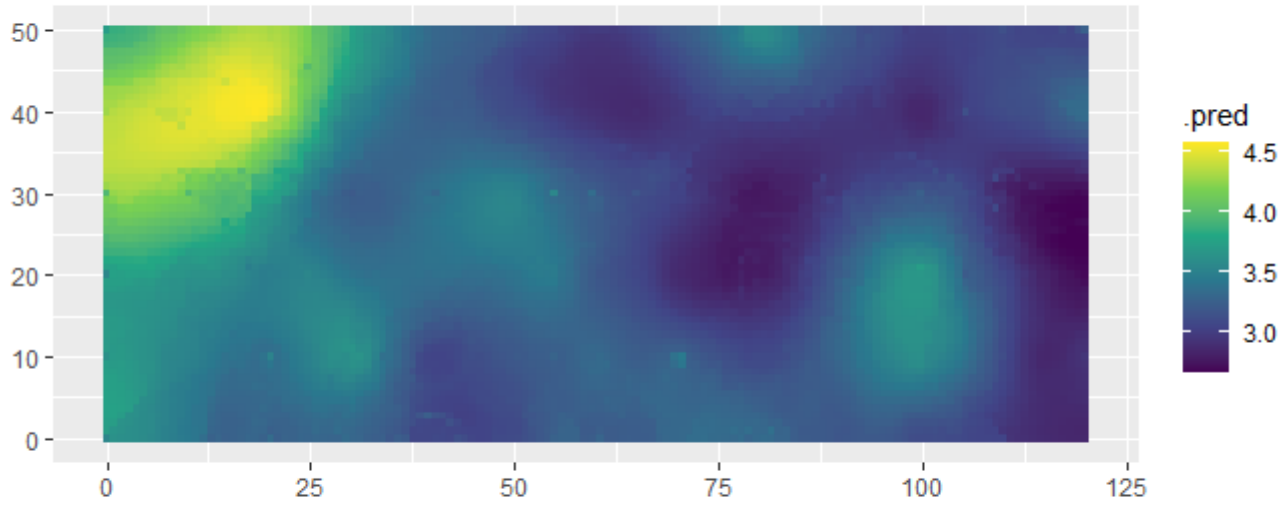

```
#> [1] "2017-06-03"
```

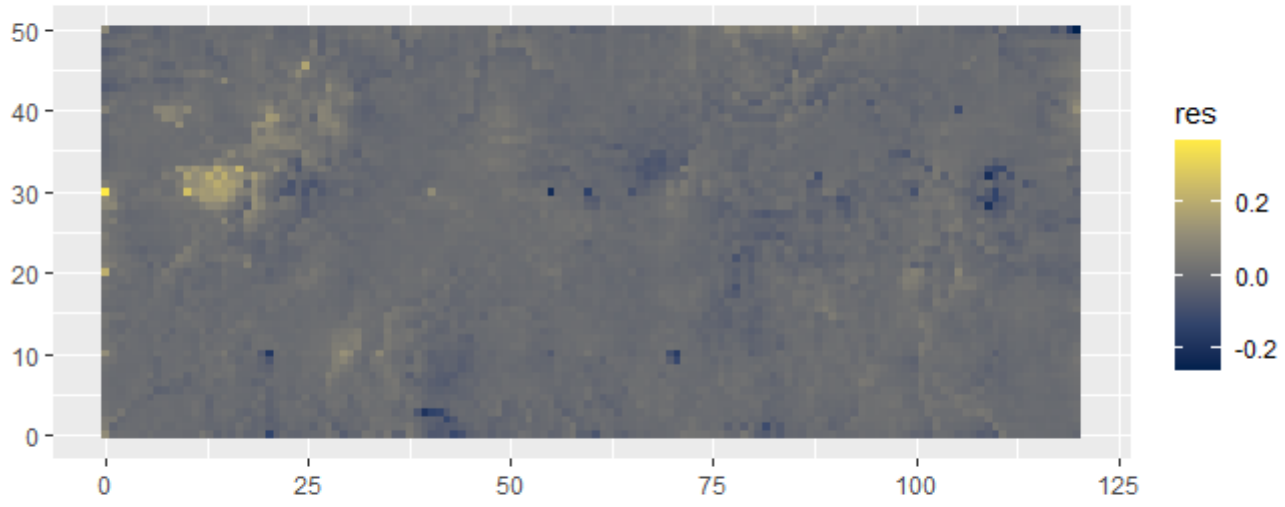

50
40
30
20
10
0
0
25
50
75
100
125
F
4.5
4.0
3.5
3.0

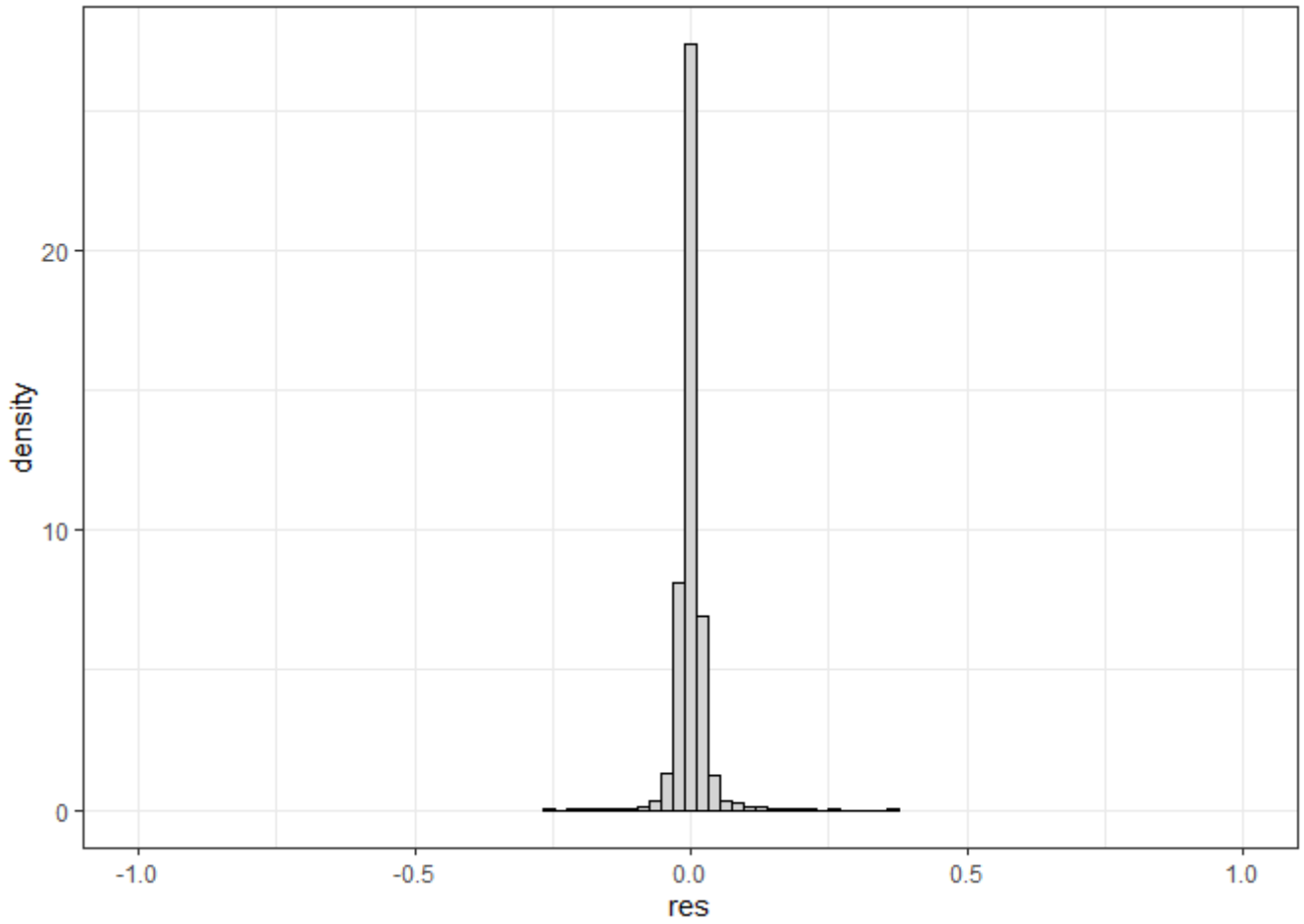

50
40
30
20
10
0
0
25
50
75
100
125
.pred
4.5
4.0
3.5
3.0


50
40
30
20
10
0
0
25
50
75
100
125
res
0.5
0.0
-0.5

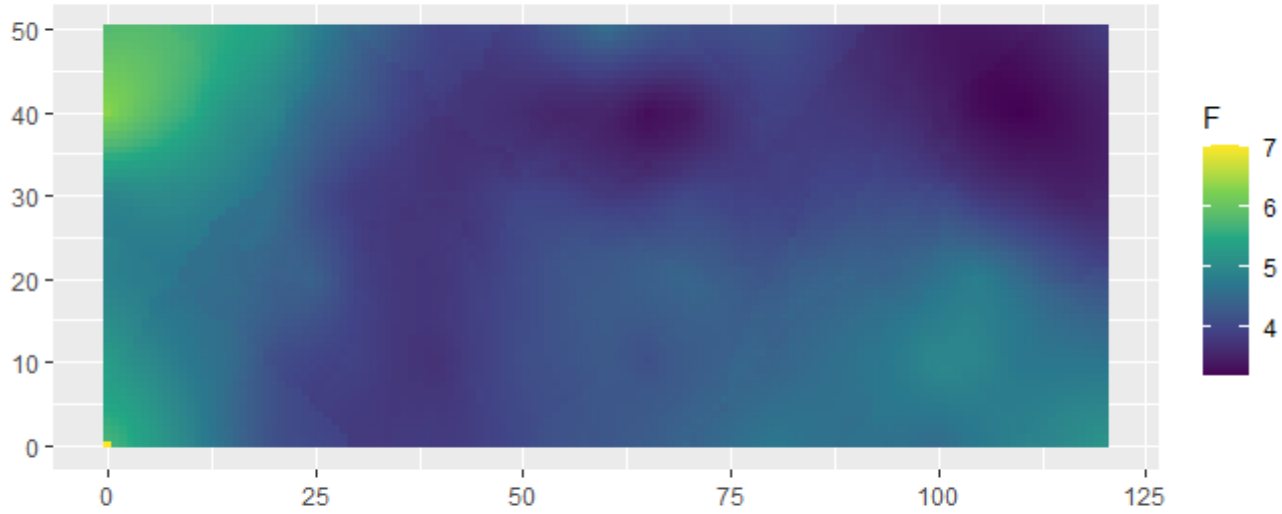

8
6
4
2
0
density
-1.0
-0.5
0.0
0.5
1.0
res

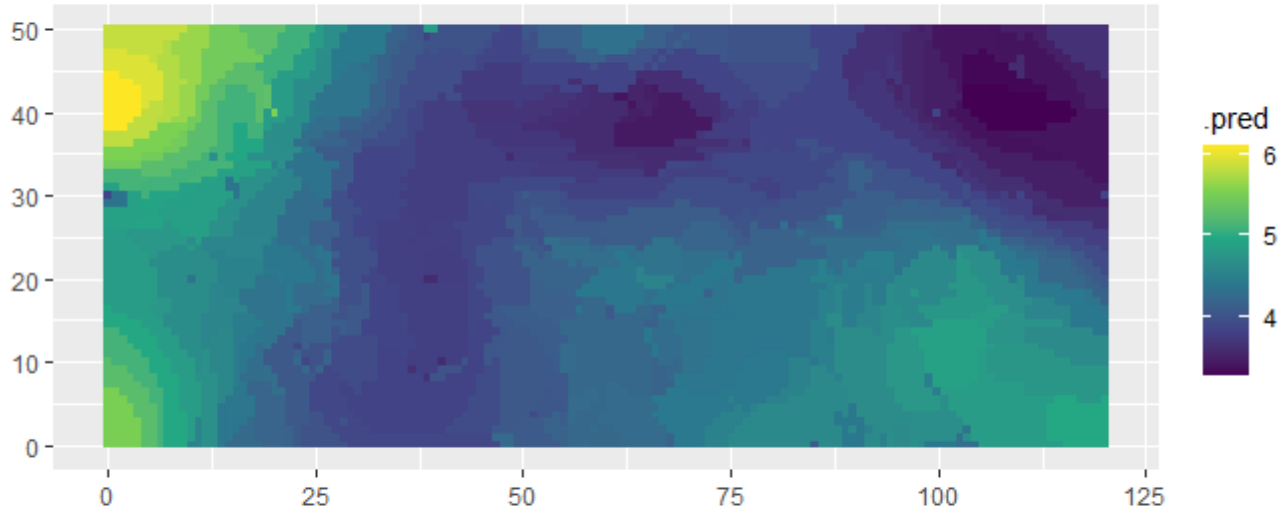

50
40
30
20
10
0
0
25
50
75
100
125
.pred
4.5
4.0
3.5
3.0

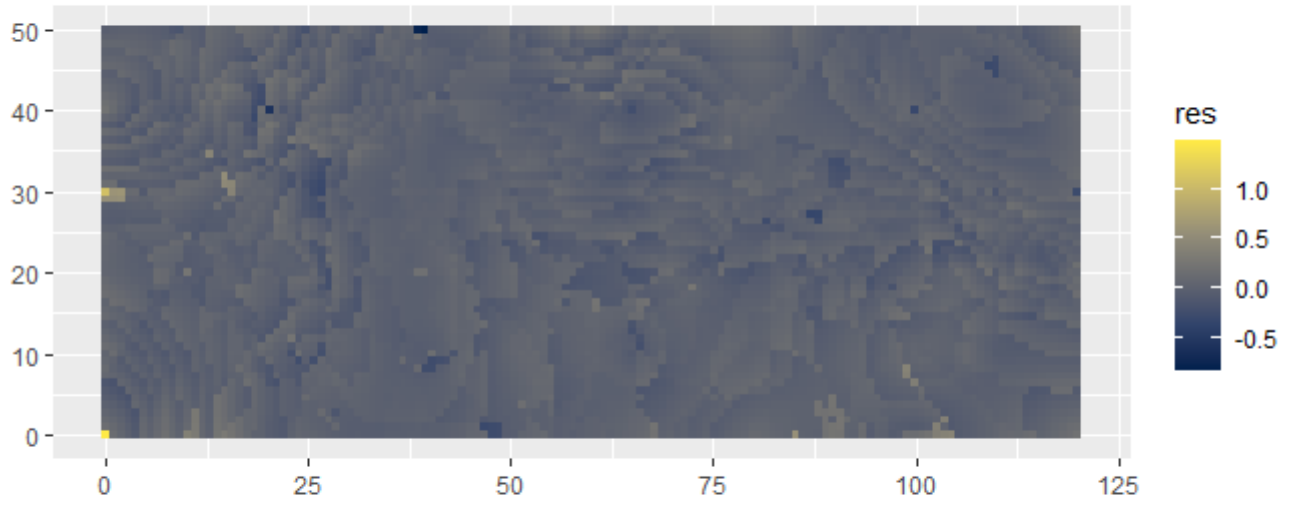

50
40
30
20
10
0
0
25
50
75
100
125
res
0.2
0.0
-0.2

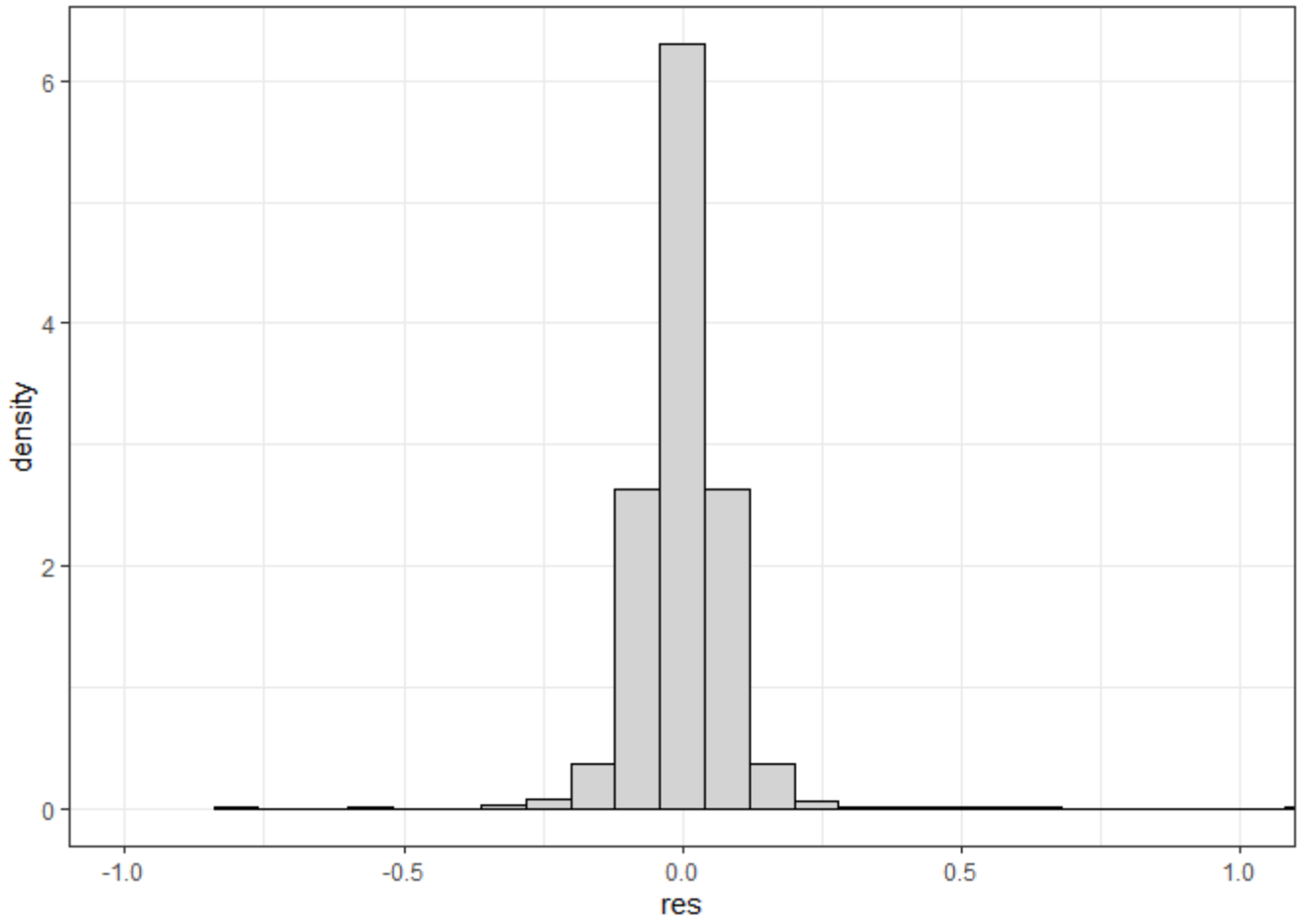


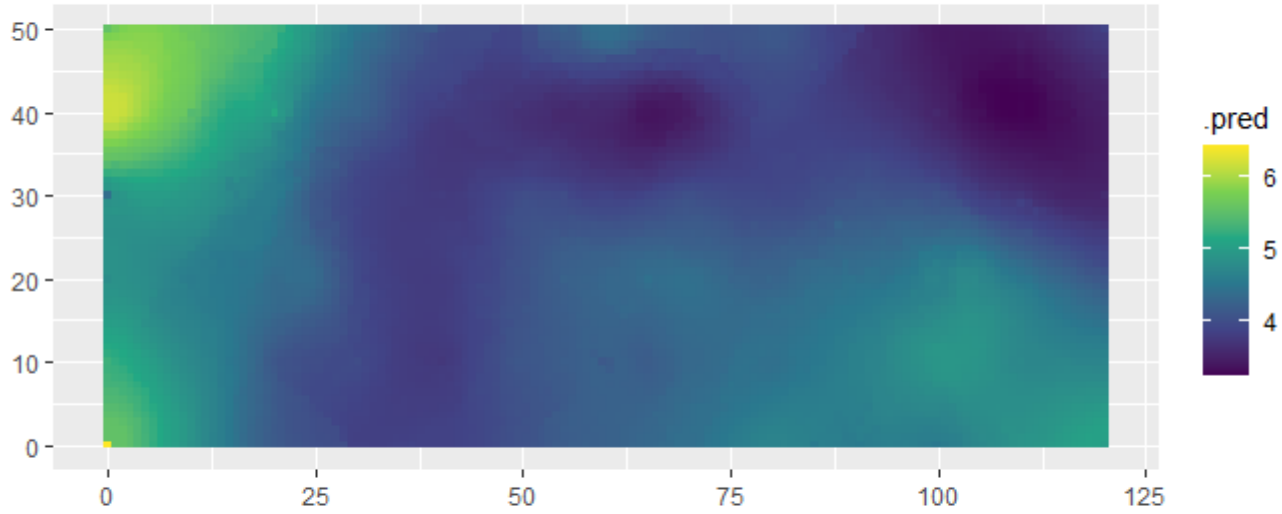

```
#> [1] "2017-03-08"
```

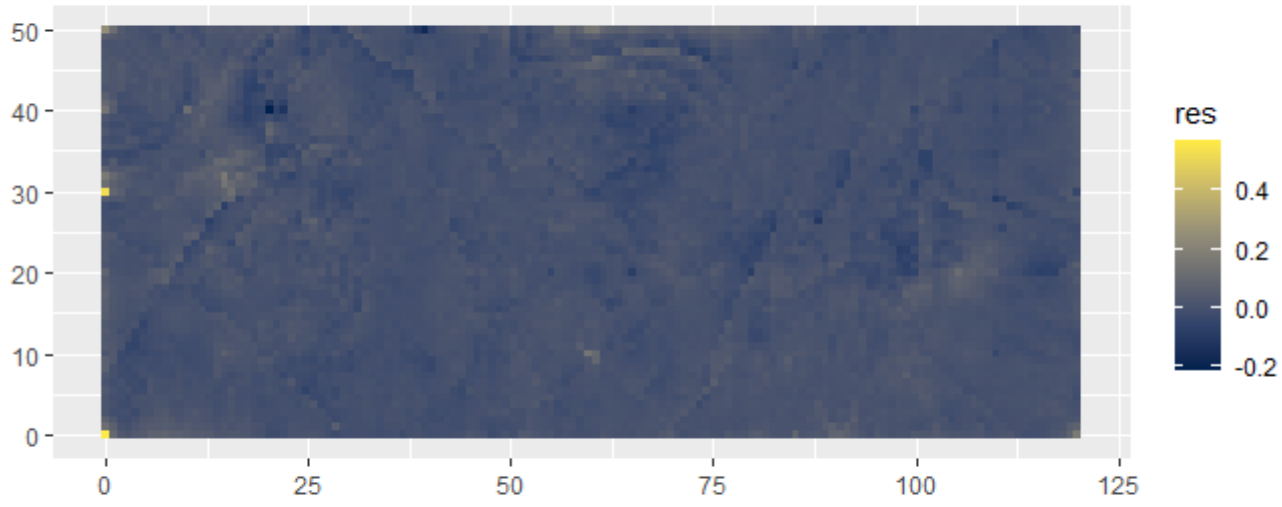

50
40
30
20
10
0
0
25
50
75
100
125
F
7
6
5
4

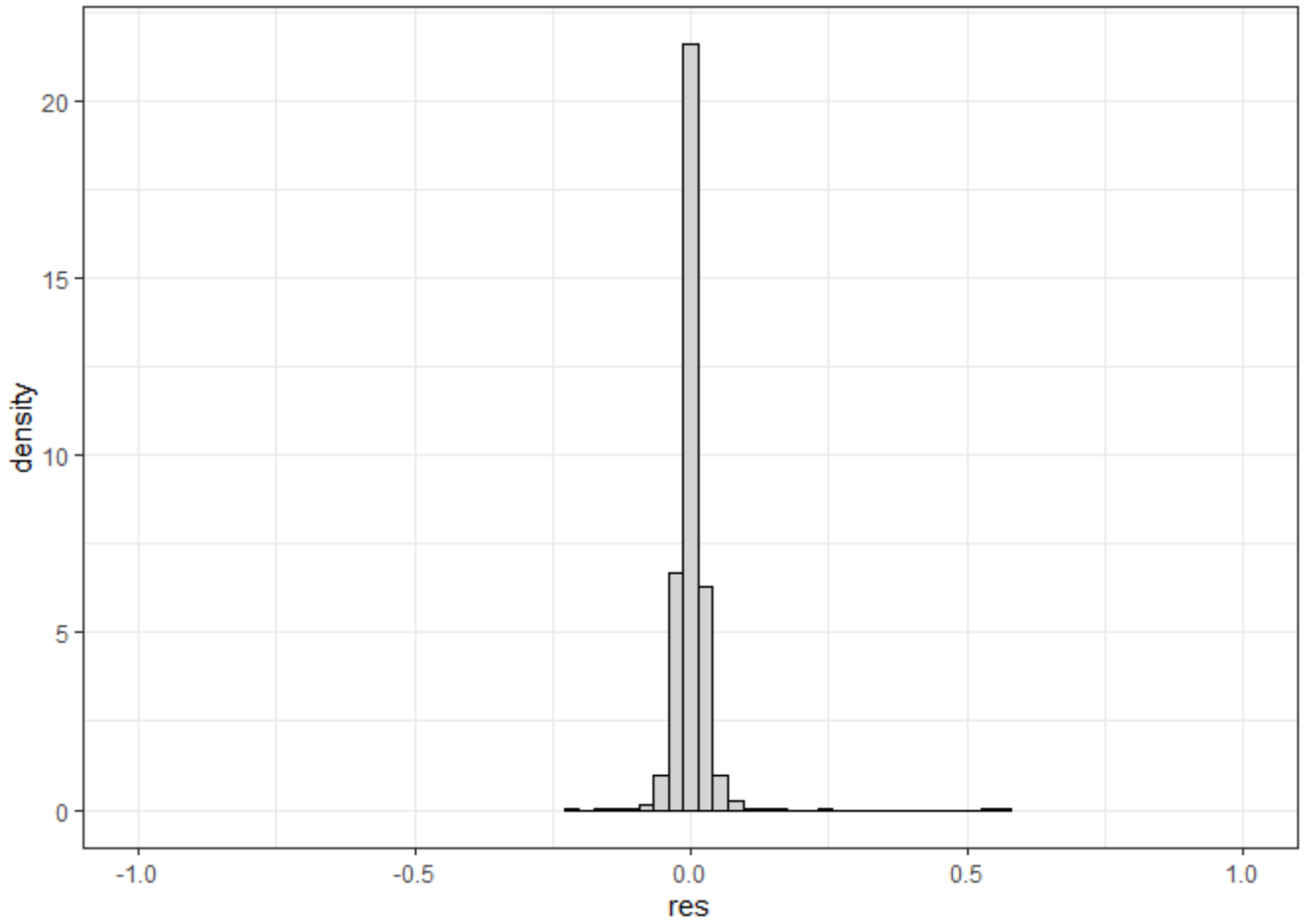

50
40
30
20
10
0
0
25
50
75
100
125
.pred
6
5
4


50
40
30
20
10
0
0
25
50
75
100
125
res
1.0
0.5
0.0
-0.5

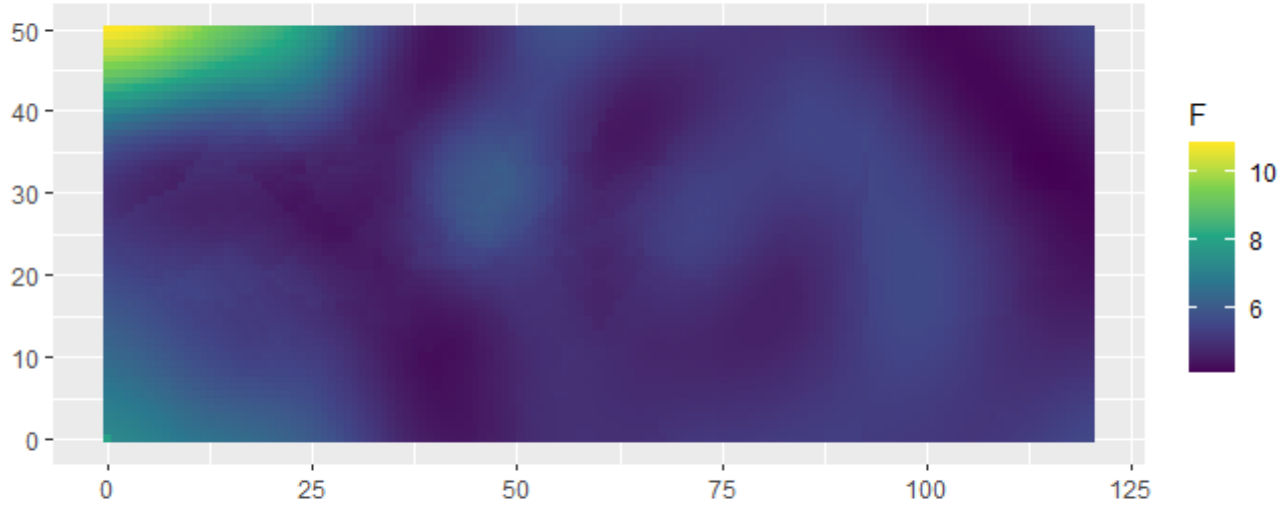

6
4
2
0
density
-1.0
-0.5
0.0
0.5
1.0
res

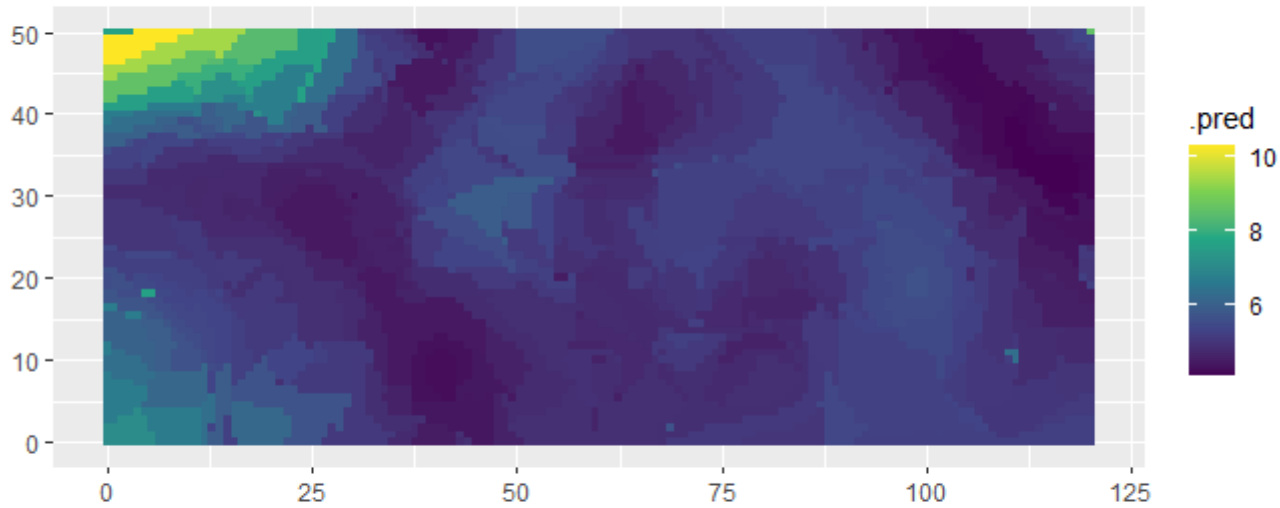

50
40
30
20
10
0
0
25
50
75
100
125
.pred
6
5
4

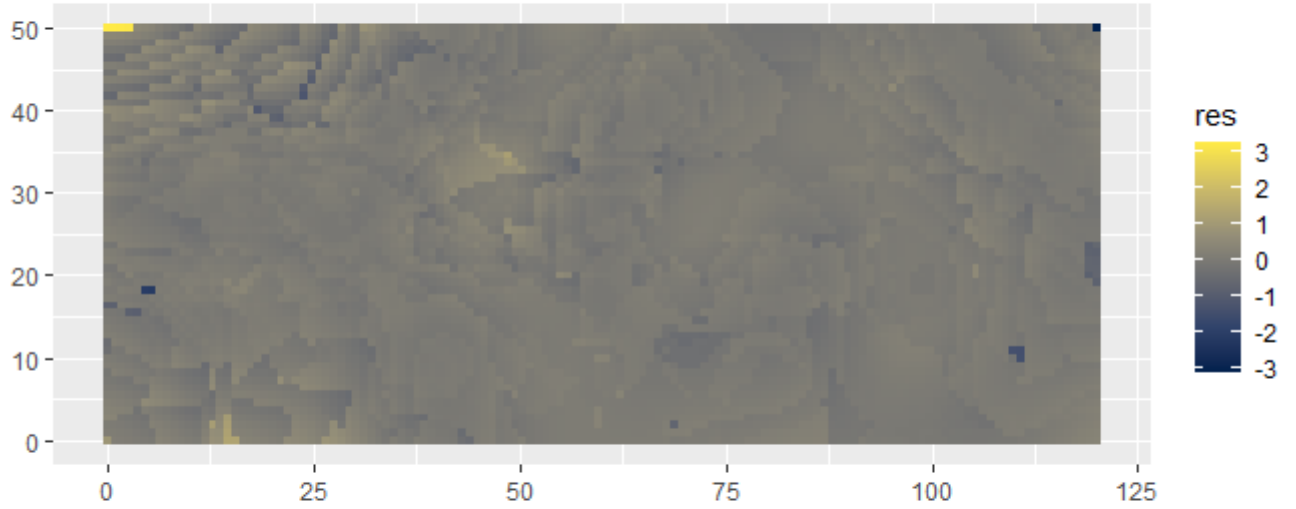

50
40
30
20
10
0
0
25
50
75
100
125
res
0.4
0.2
0.0
-0.2

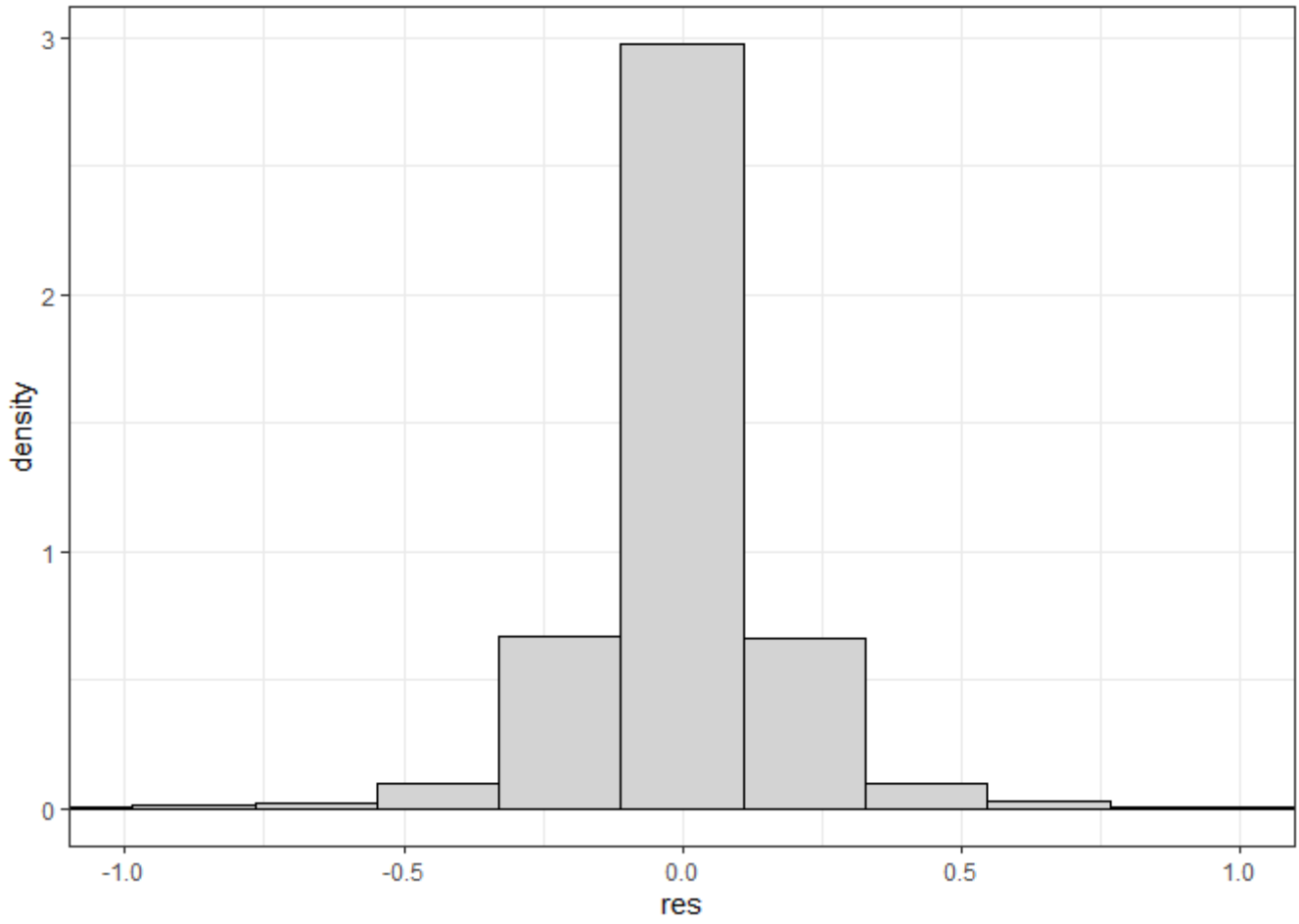


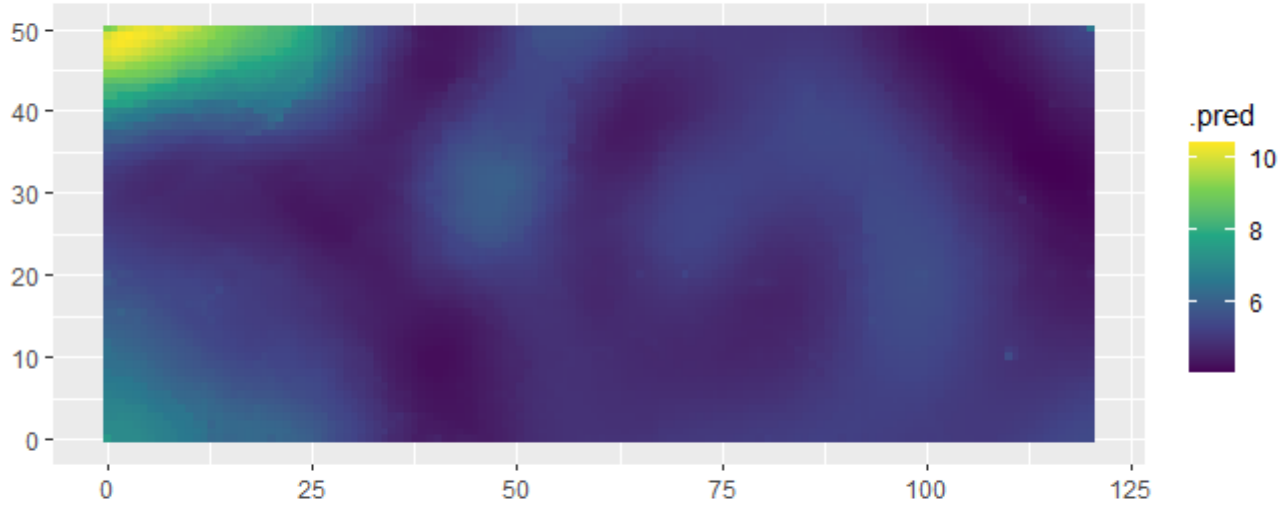

```
#> [1] "2017-02-09"
```

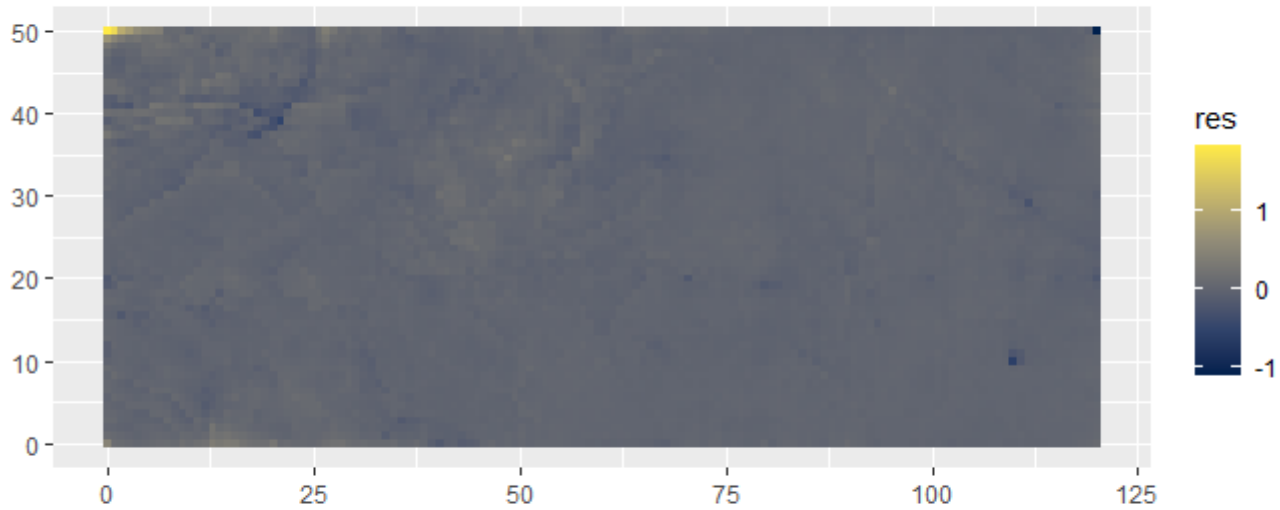

50
40
30
20
10
0
0
25
50
75
100
125
F
10
8
6

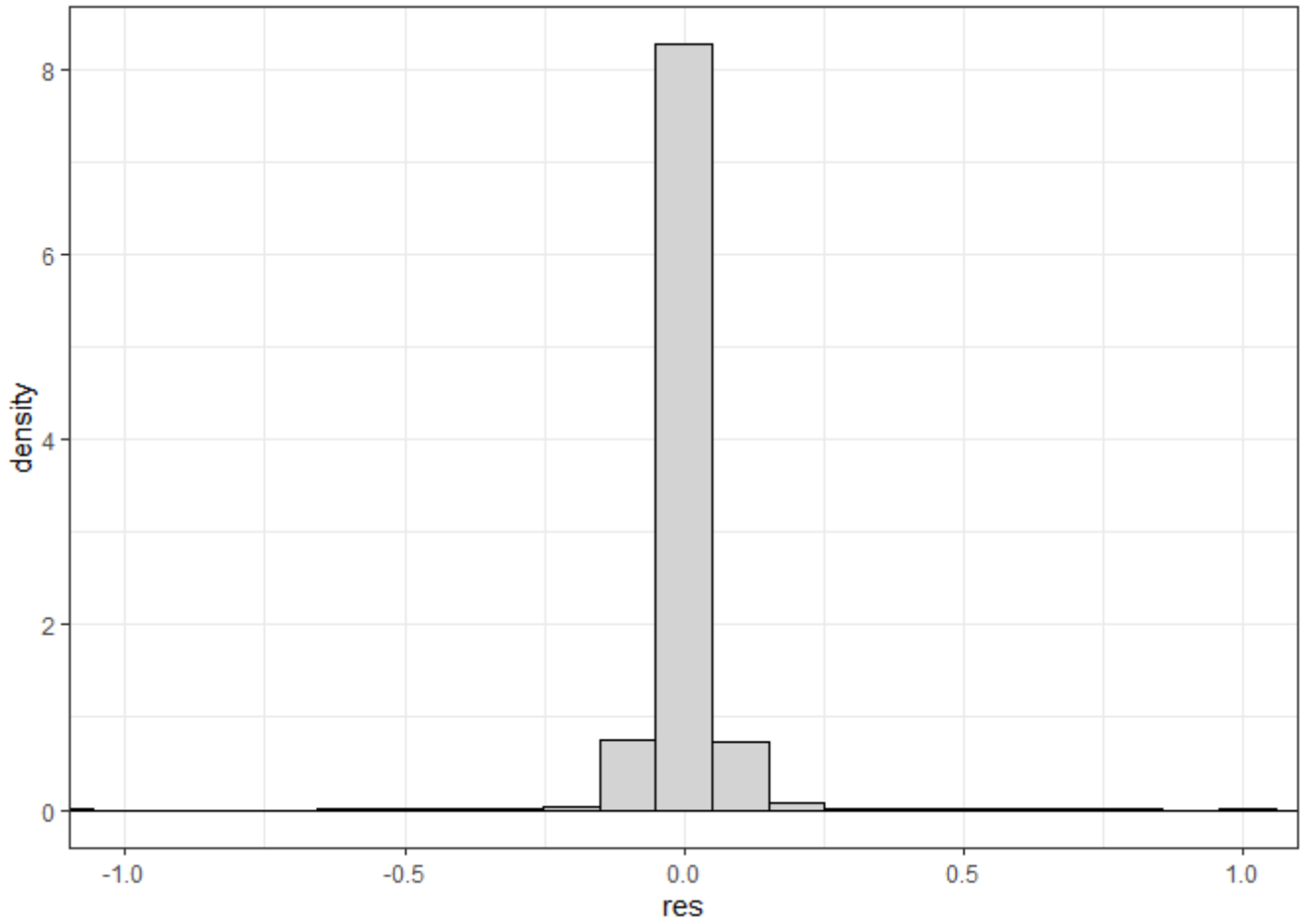

50
40
30
20
10
0
0
25
50
75
100
125
.pred
10
8
6


50
40
30
20
10
0
0
25
50
75
100
125
res
3
2
1
0
-1
-2
-3

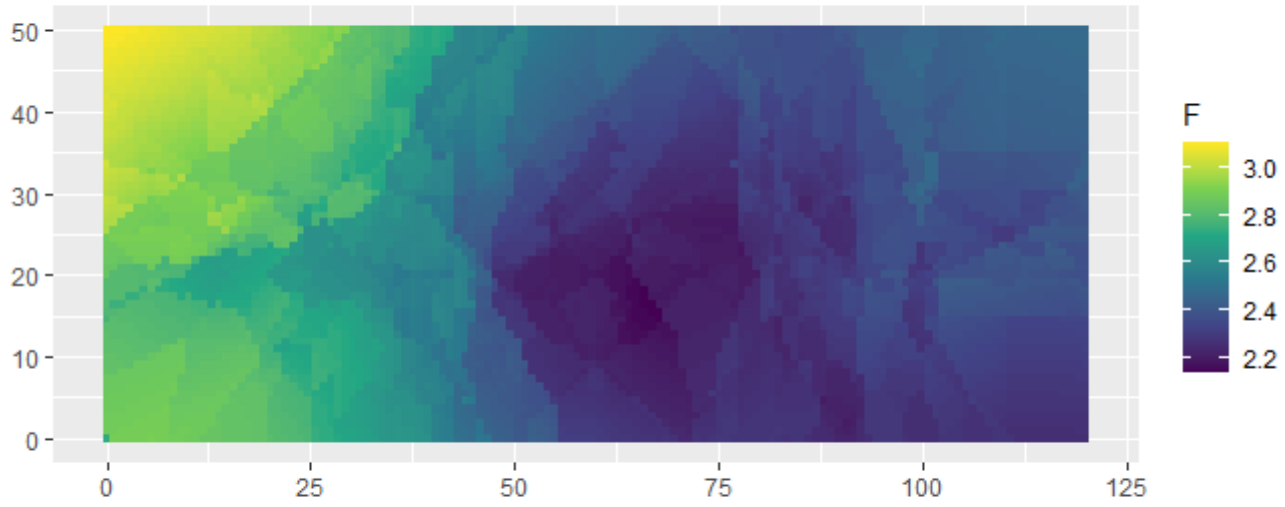

3
2
1
0
density
-1.0
-0.5
0.0
0.5
1.0
res

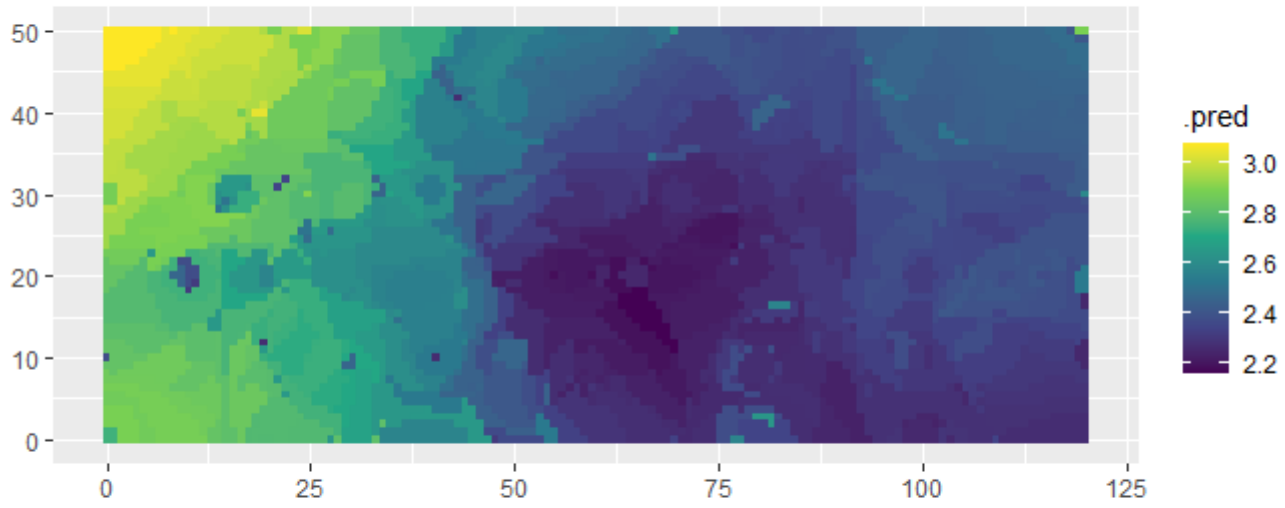

50
40
30
20
10
0
0
25
50
75
100
125
.pred
10
8
6

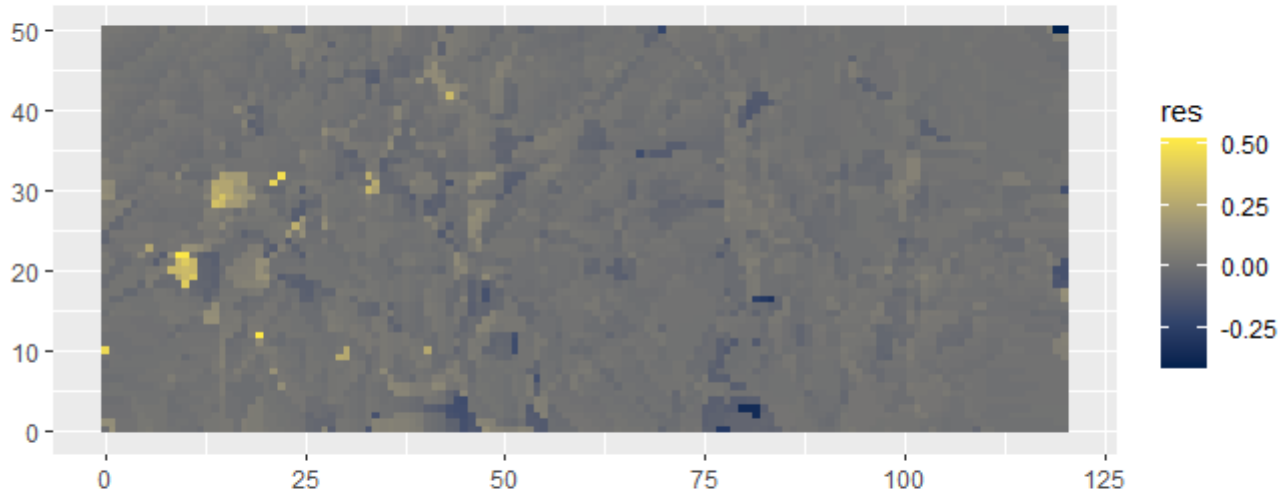

50
40
30
20
10
0
0
25
50
75
100
125
res
1
0
-1

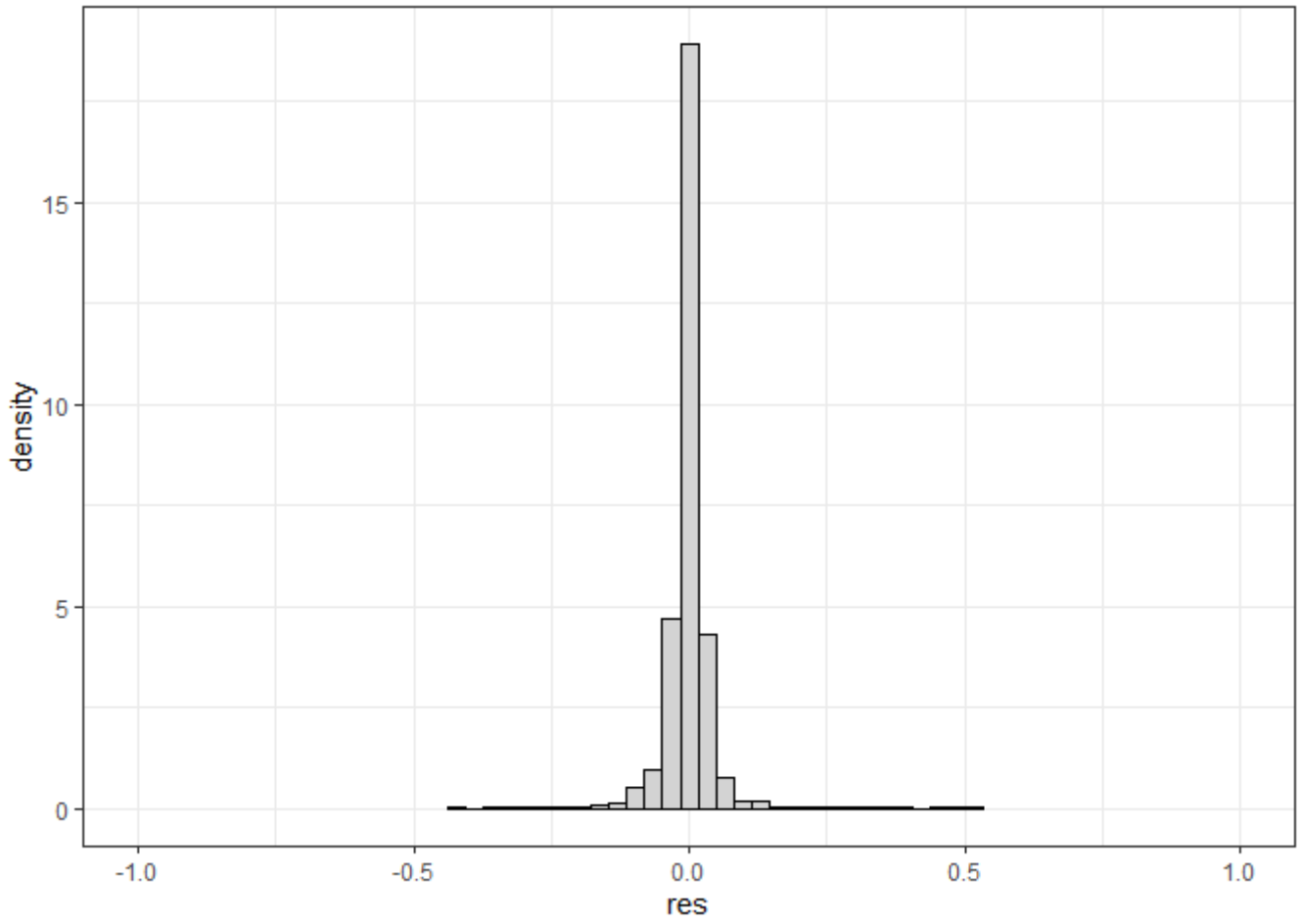


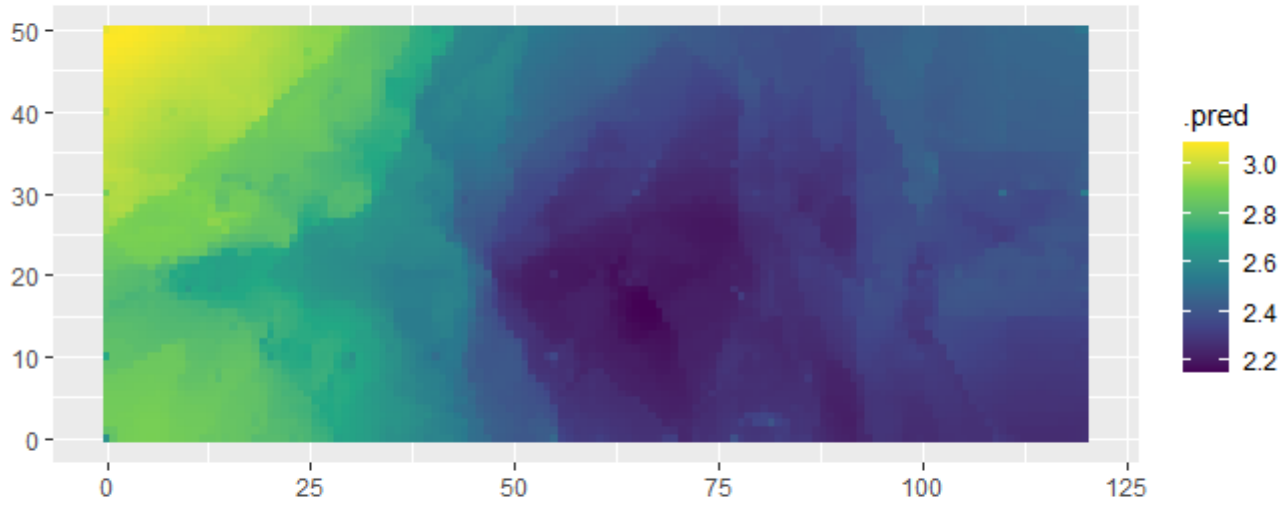

```
#> [1] "2017-06-10"
```

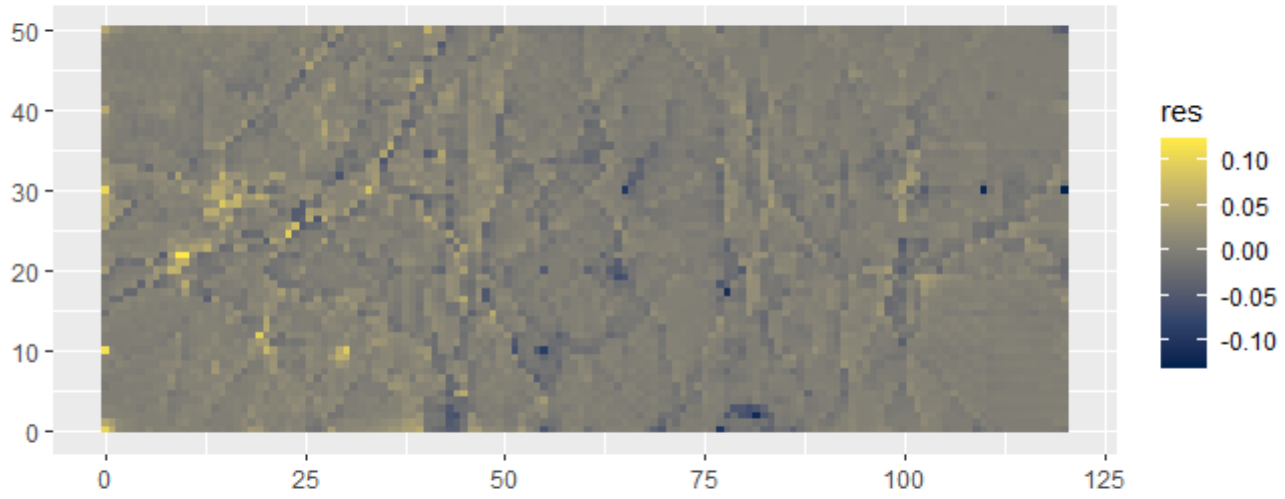

50
40
30
20
10
0
0
25
50
75
100
125
F
3.0
2.8
2.6
2.4
2.2

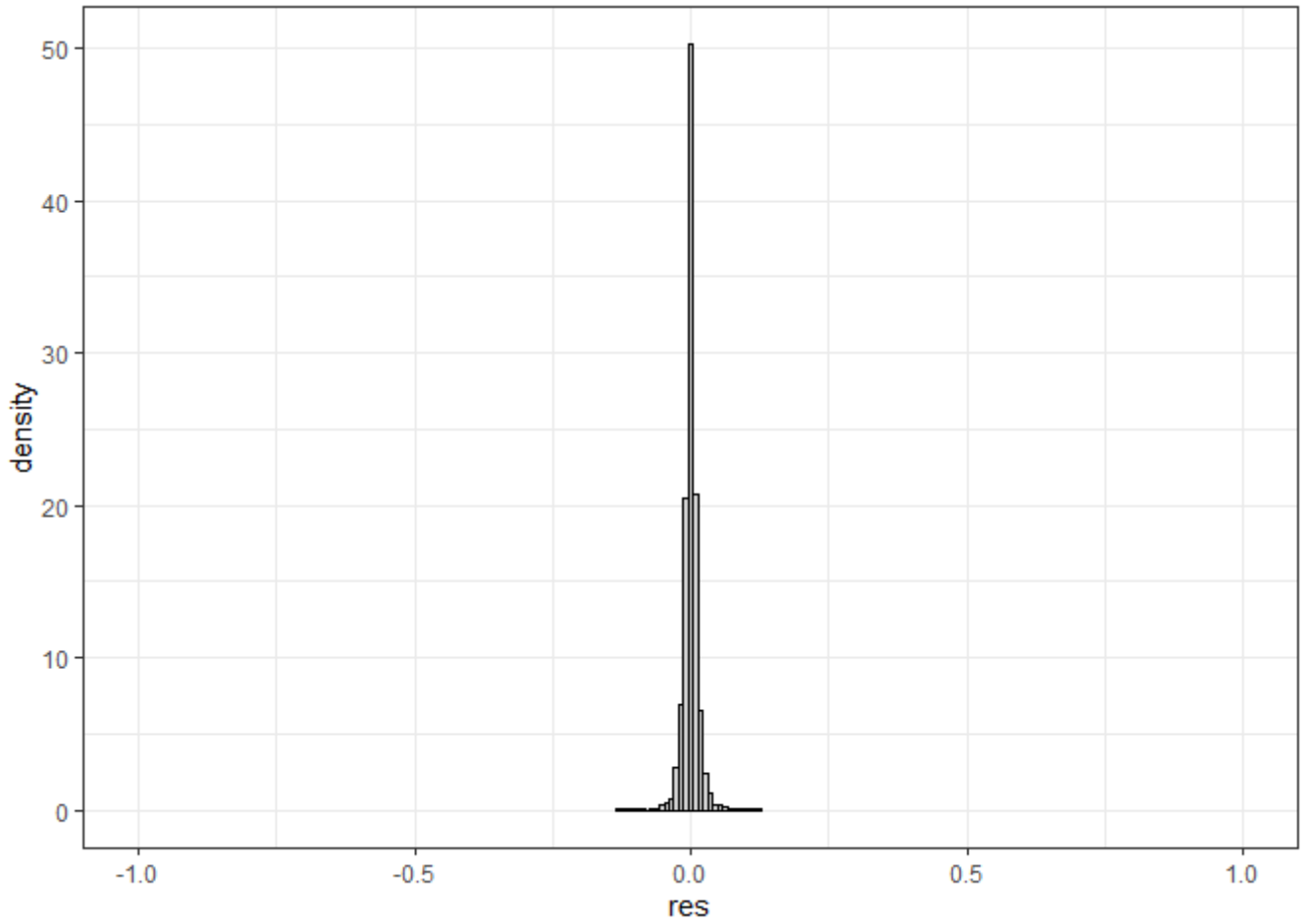

50
40
30
20
10
0
0
25
50
75
100
125
.pred
3.0
2.8
2.6
2.4
2.2


50
40
30
20
10
0
0
25
50
75
100
125
res
0.50
0.25
0.00
-0.25

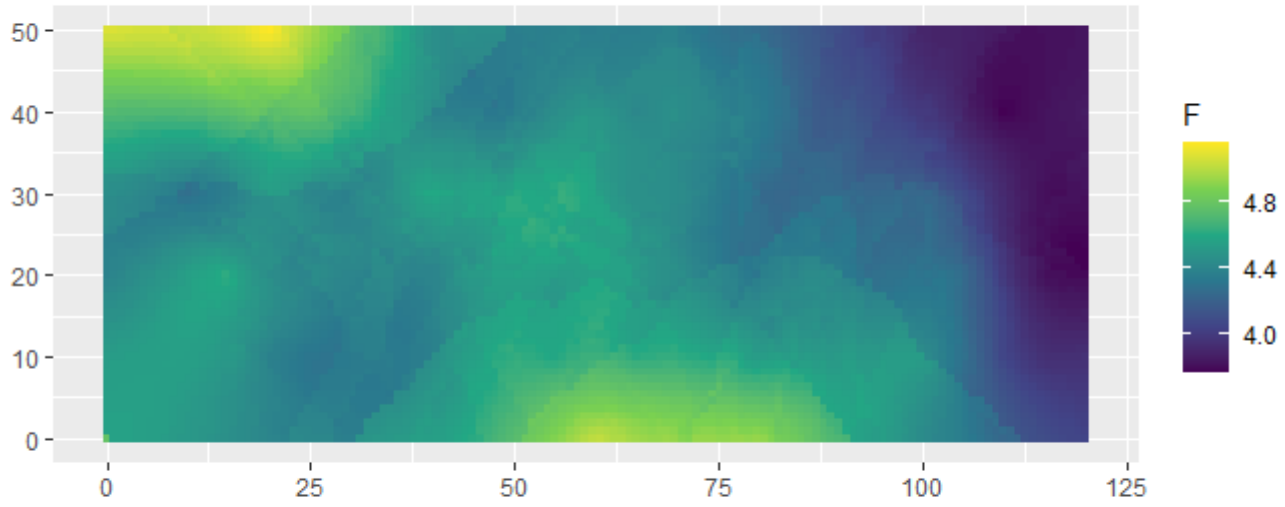

15
10
5
0
density
-1.0
-0.5
0.0
0.5
1.0
res

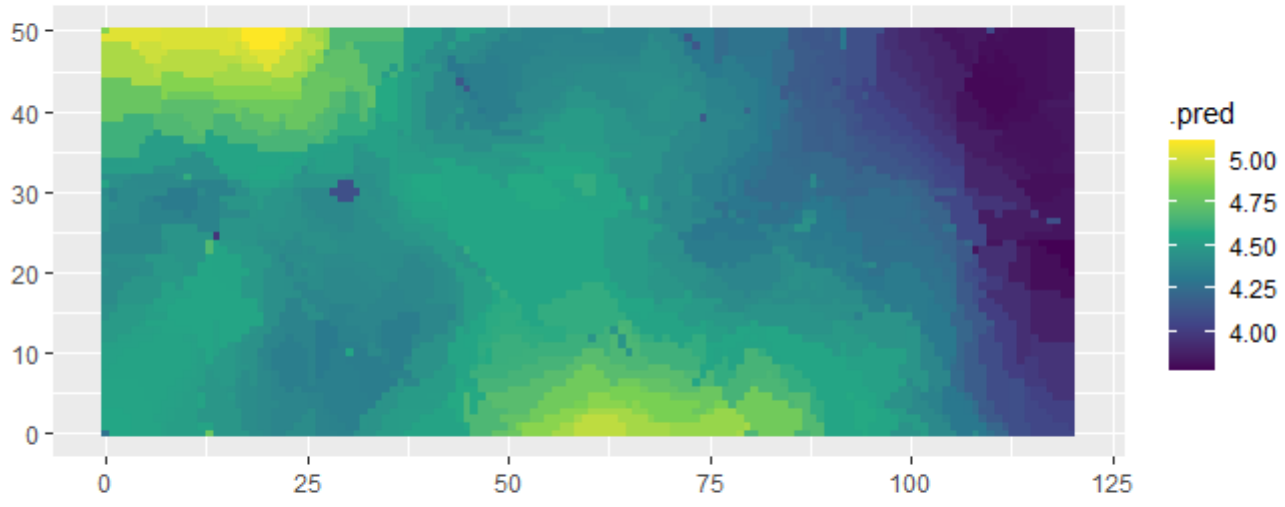

50
40
30
20
10
0
0
25
50
75
100
125
.pred
3.0
2.8
2.6
2.4
2.2

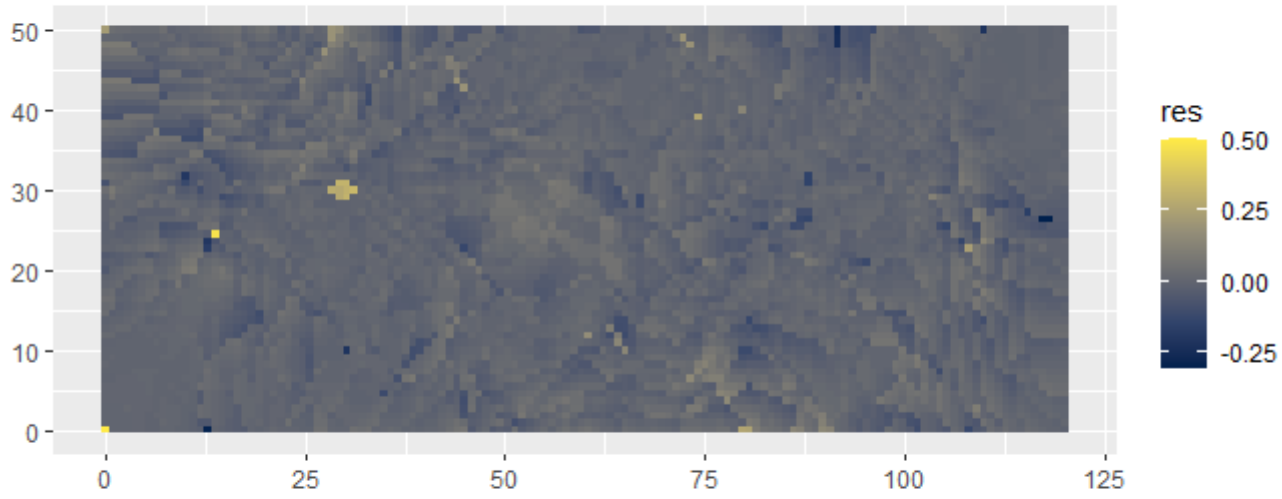

50
40
30
20
10
0
0
25
50
75
100
125
res
0.10
0.05
0.00
-0.05
-0.10

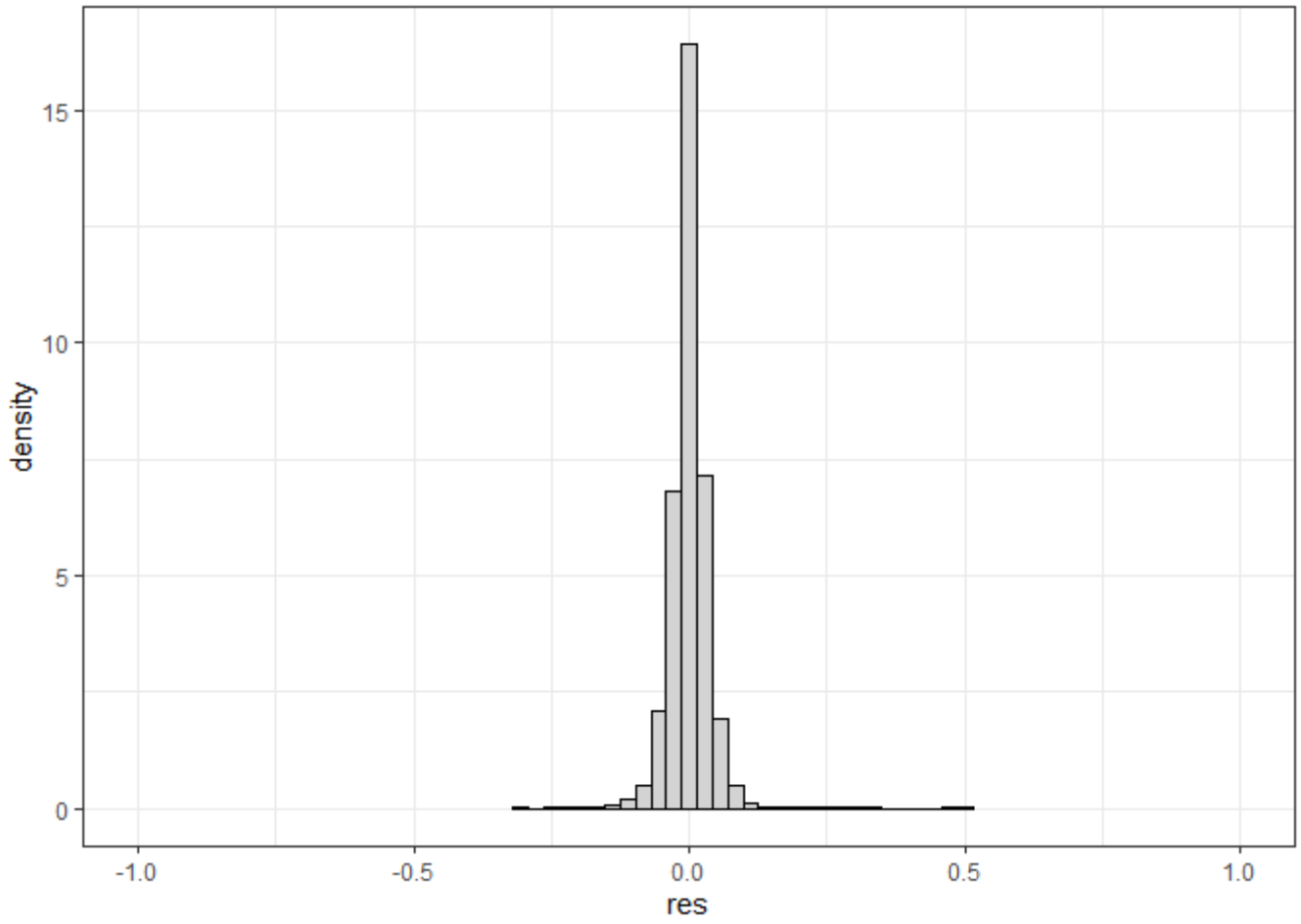


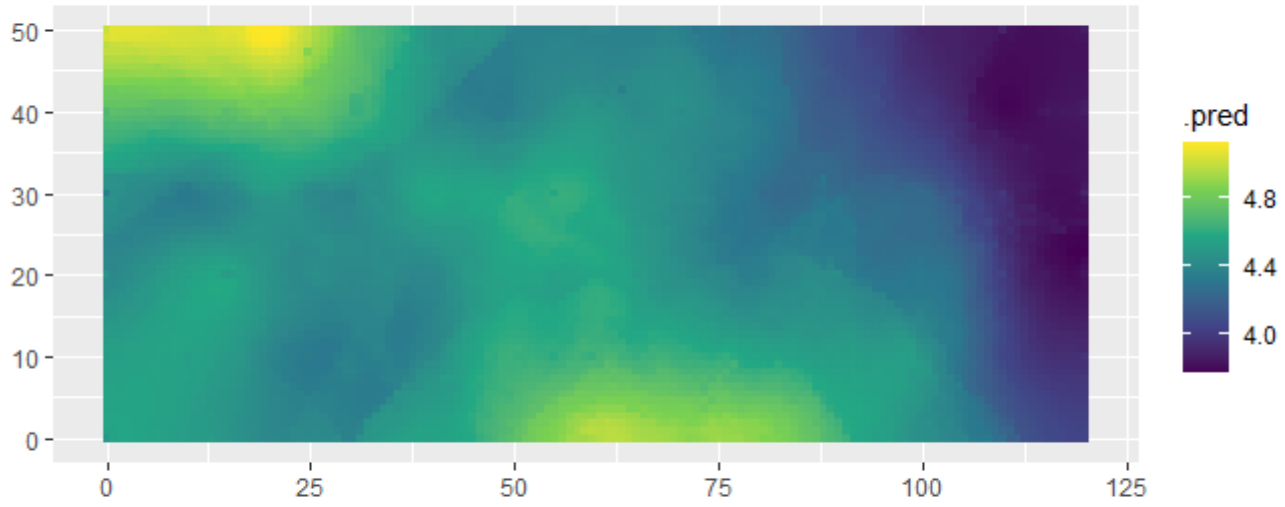

```
#> [1] "2017-03-17"
```

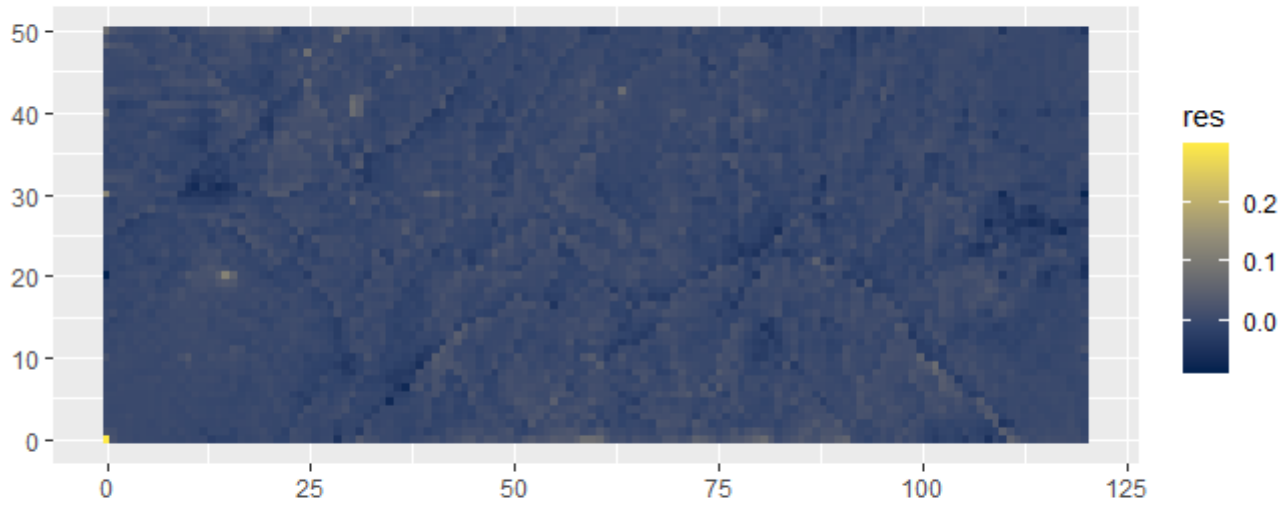

50
40
30
20
10
0
0
25
50
75
100
125
F
4.8
4.4
4.0

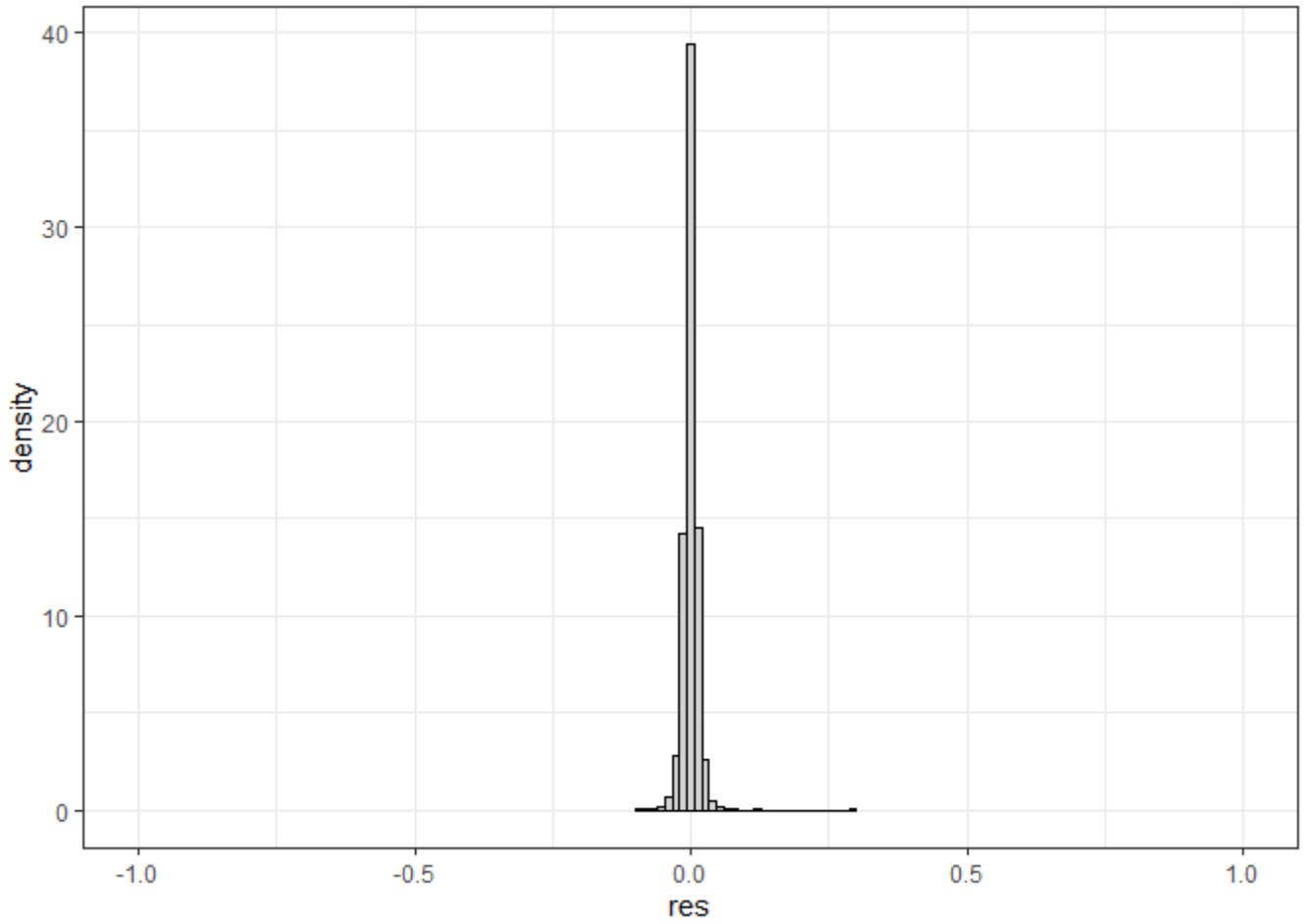

50
40
30
20
10
0
0
25
50
75
100
125
.pred
5.00
4.75
4.50
4.25
4.00


50
40
30
20
10
0
0
25
50
75
100
125
res
0.50
0.25
0.00
-0.25

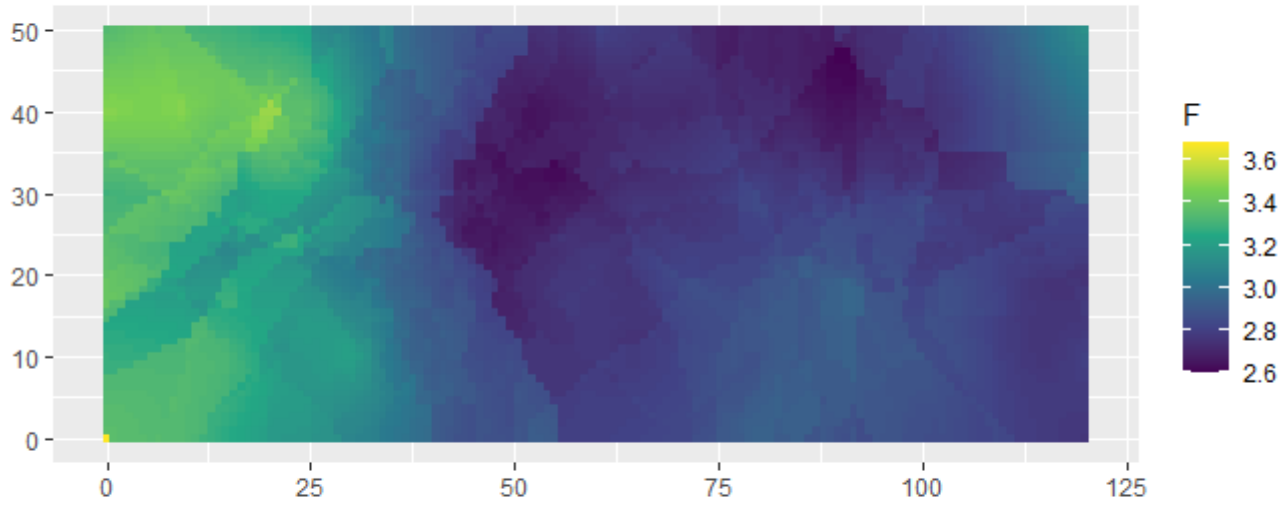

15
10
5
0
density
-1.0
-0.5
0.0
0.5
1.0
res

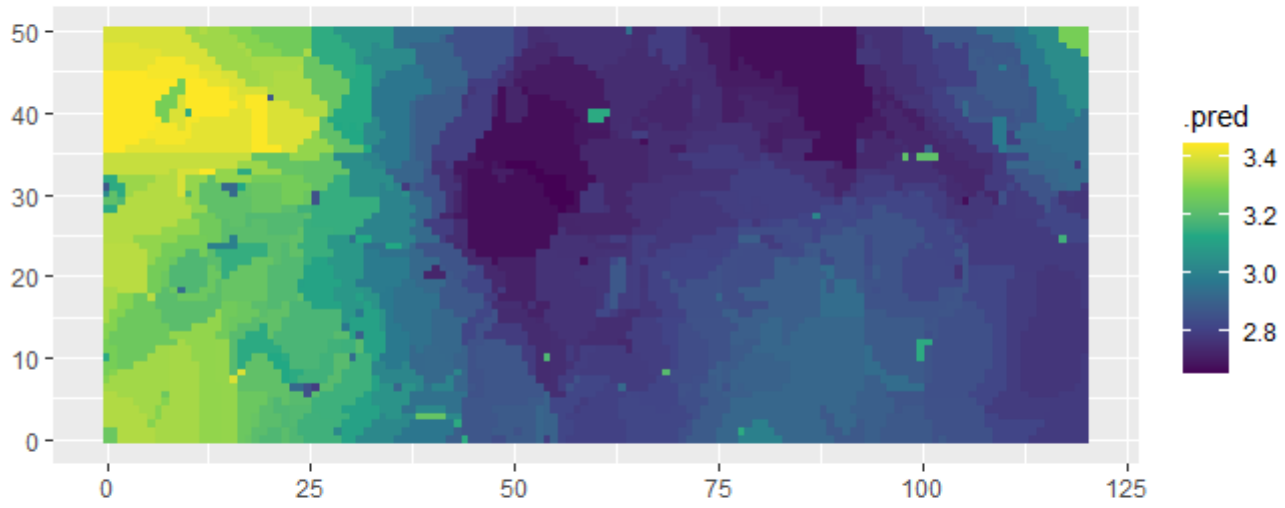

50
40
30
20
10
0
0
25
50
75
100
125
.pred
4.8
4.4
4.0

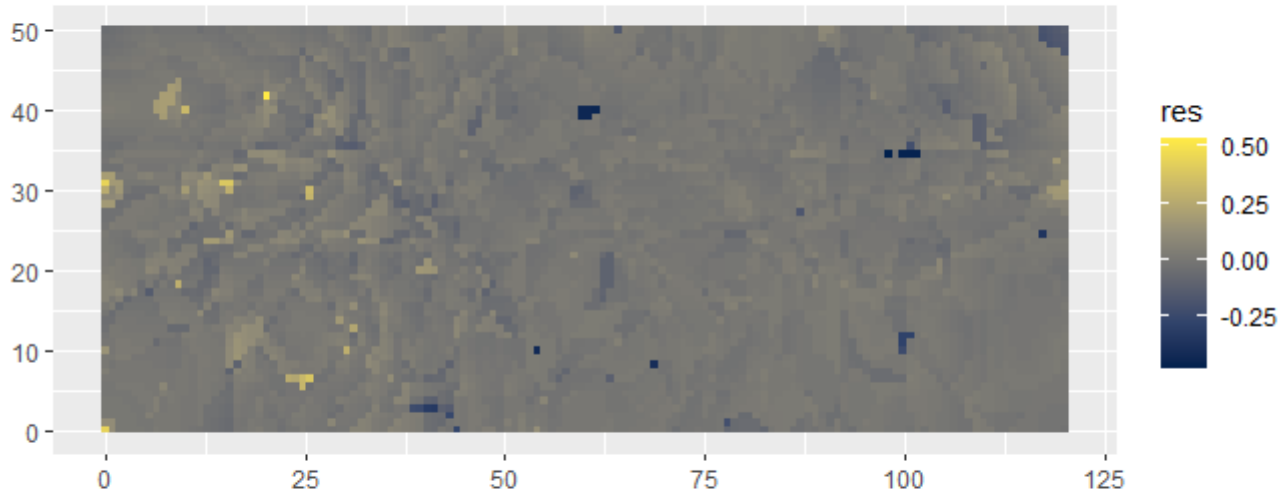

50
40
30
20
10
0
0
25
50
75
100
125
res
0.2
0.1
0.0

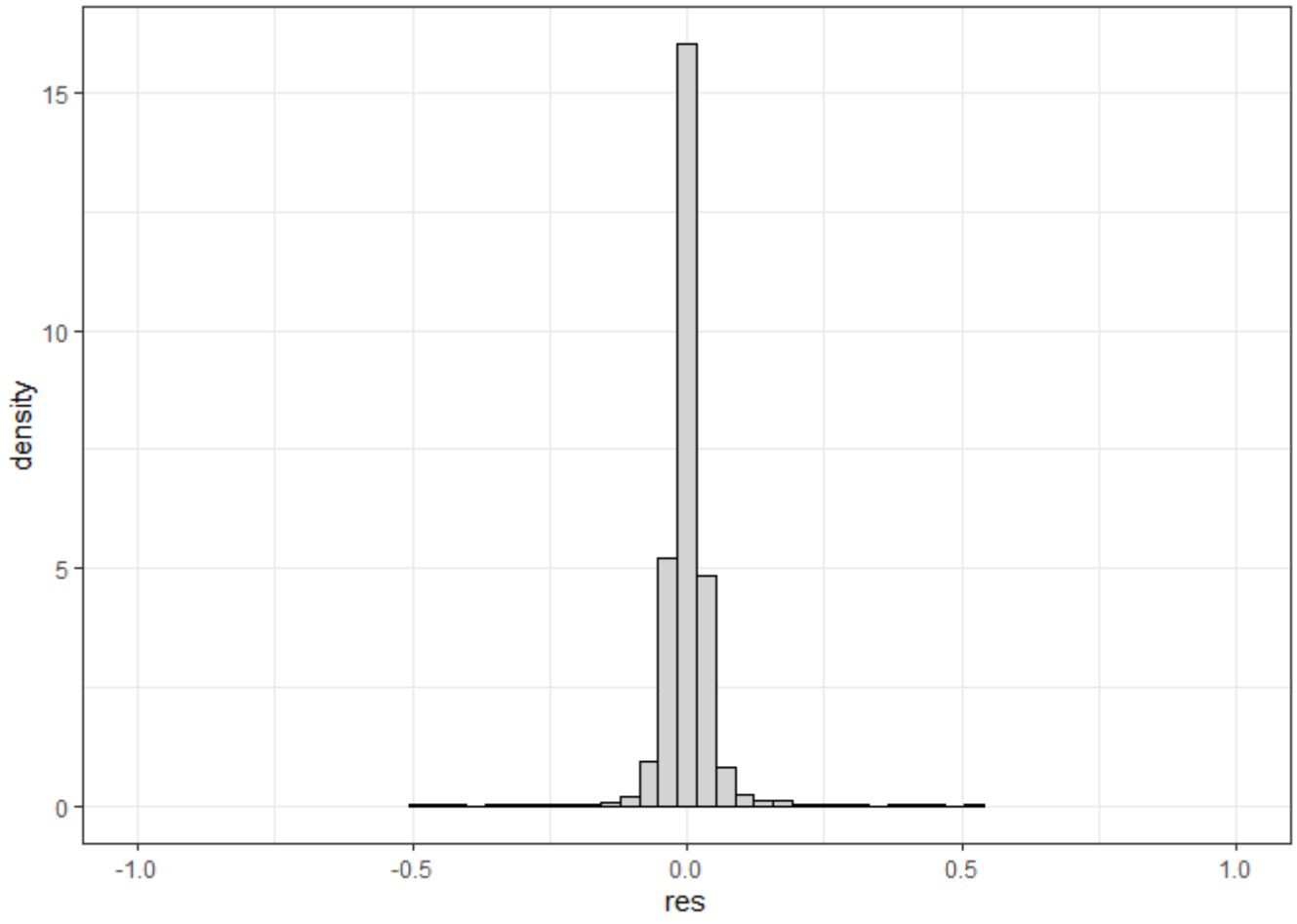


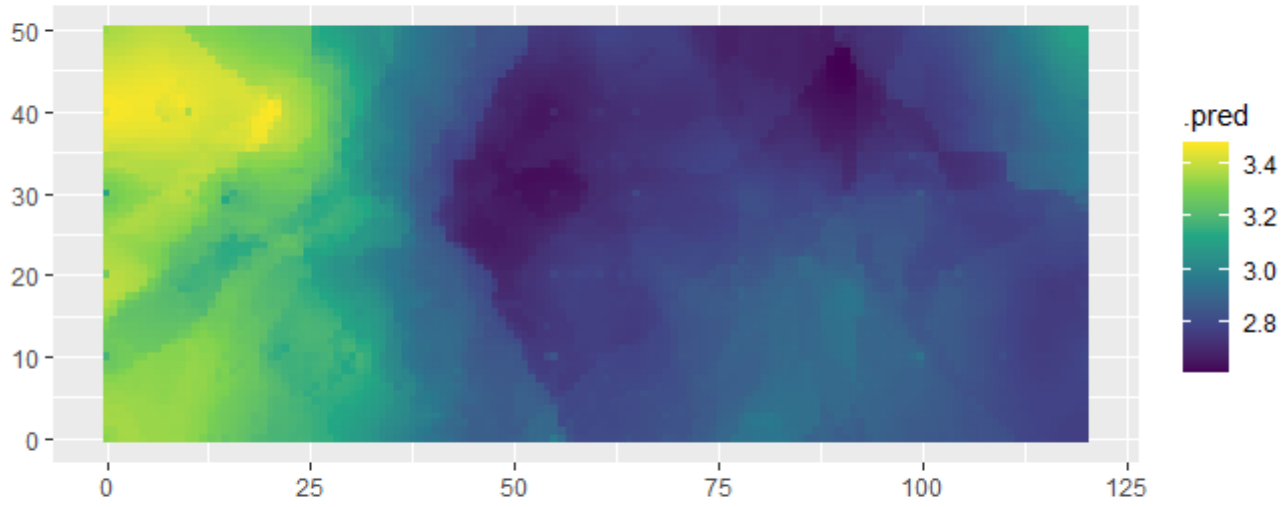

```
#> [1] "2017-06-17"
```

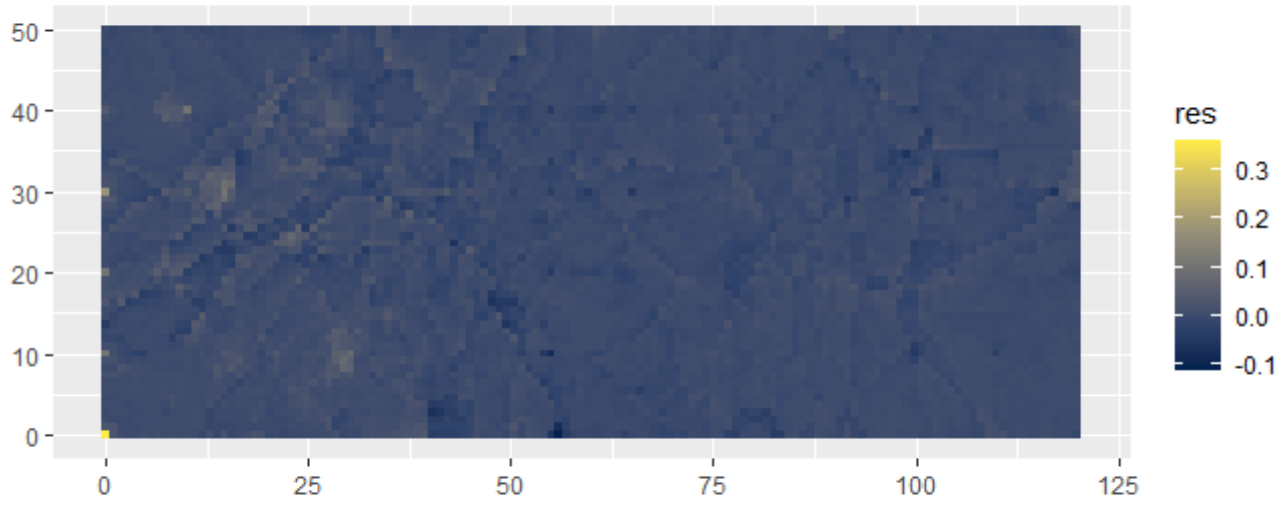

50
40
30
20
10
0
0
25
50
75
100
125
F
3.6
3.4
3.2
3.0
2.8
2.6

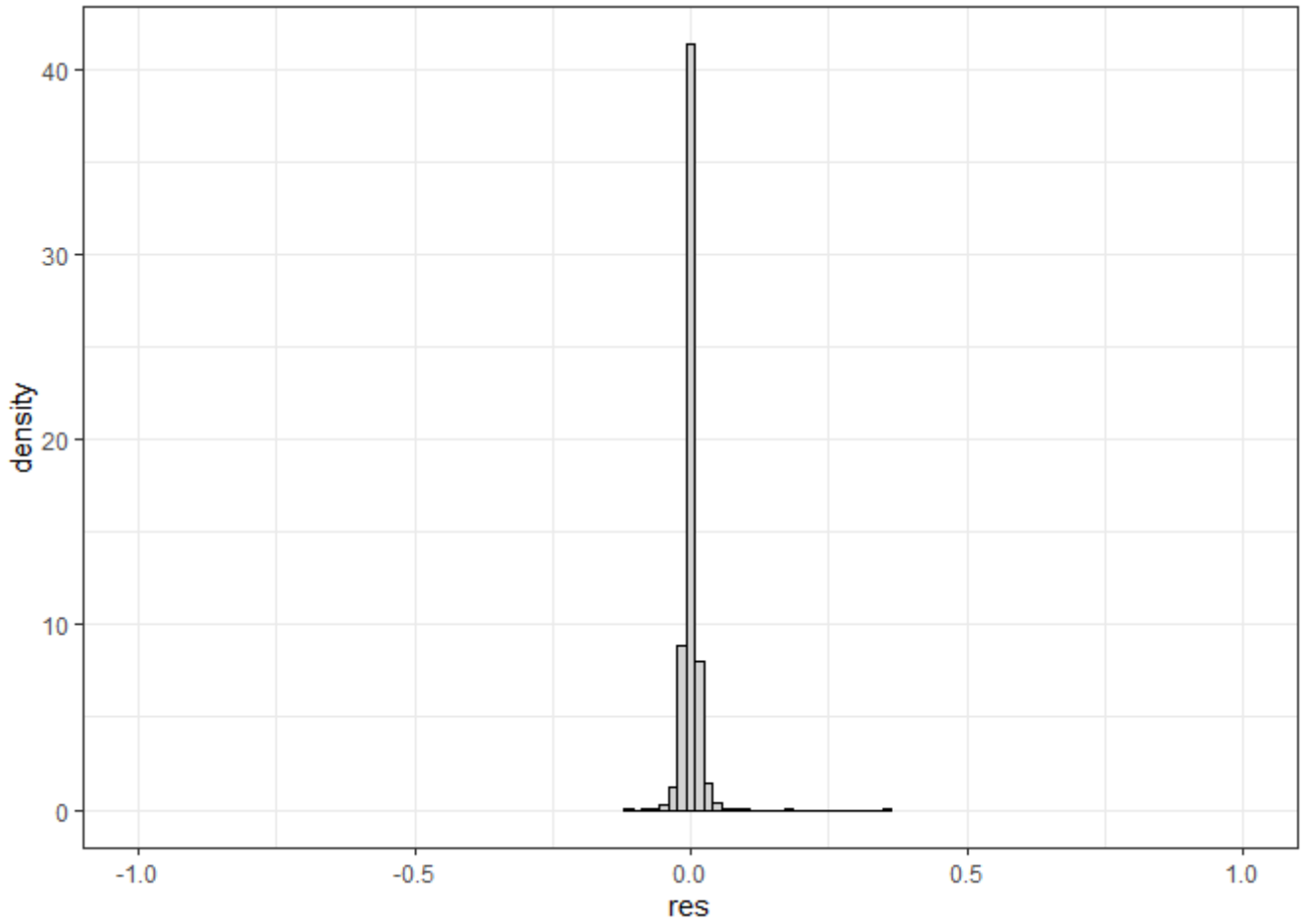

50
40
30
20
10
0
0
25
50
75
100
125
.pred
3.4
3.2
3.0
2.8


50
40
30
20
10
0
0
25
50
75
100
125
res
0.50
0.25
0.00
-0.25

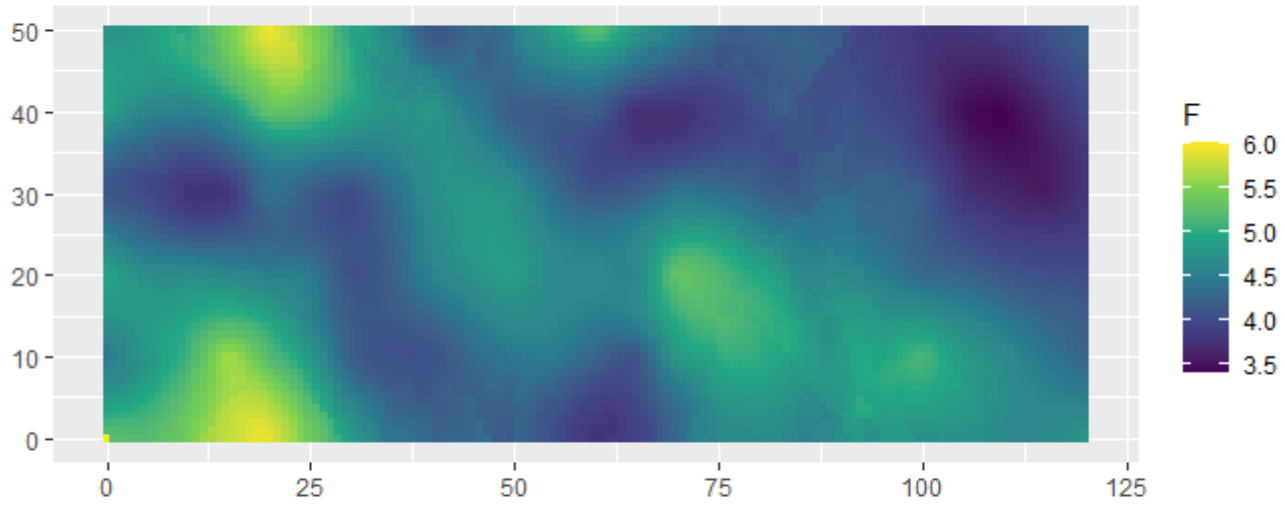

15
10
5
0
density
-1.0
-0.5
0.0
0.5
1.0
res

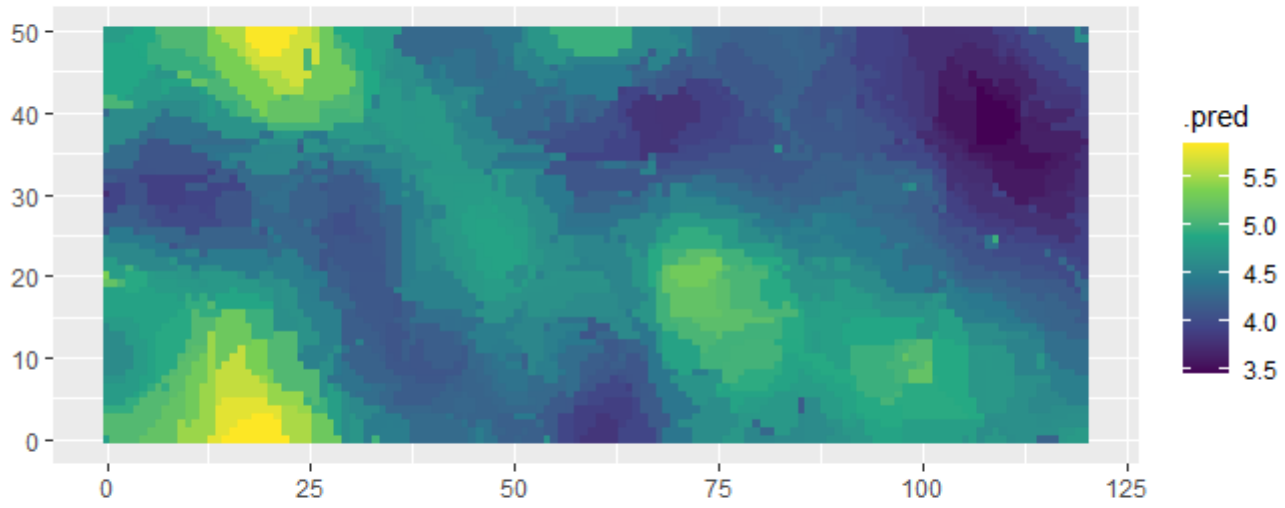

50
40
30
20
10
0
0
25
50
75
100
125
.pred
3.4
3.2
3.0
2.8

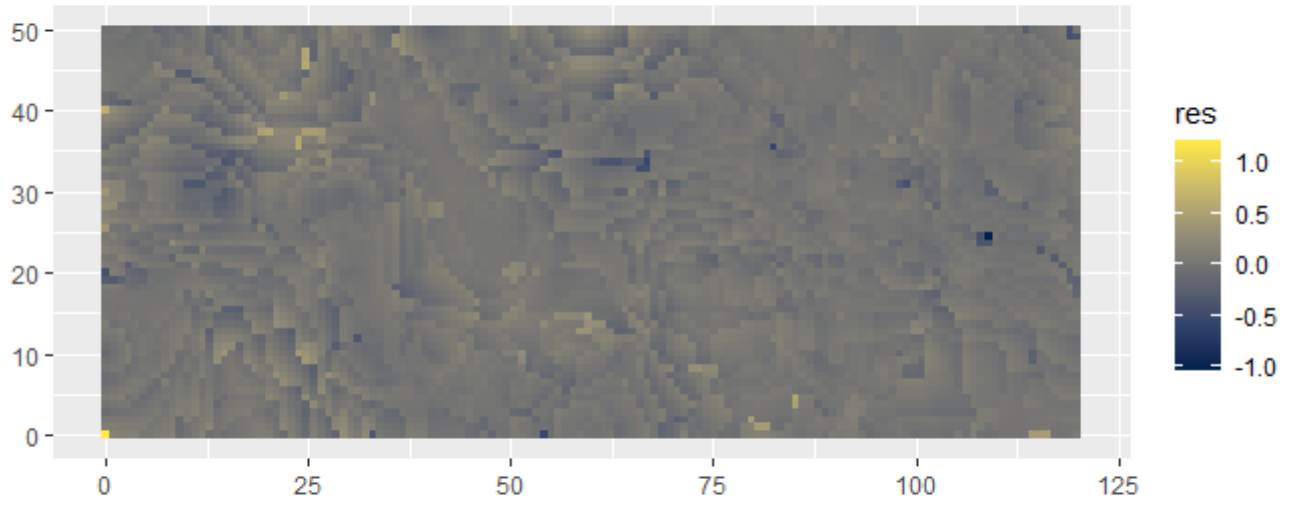

50
40
30
20
10
0
0
25
50
75
100
125
res
0.3
0.2
0.1
0.0
-0.1

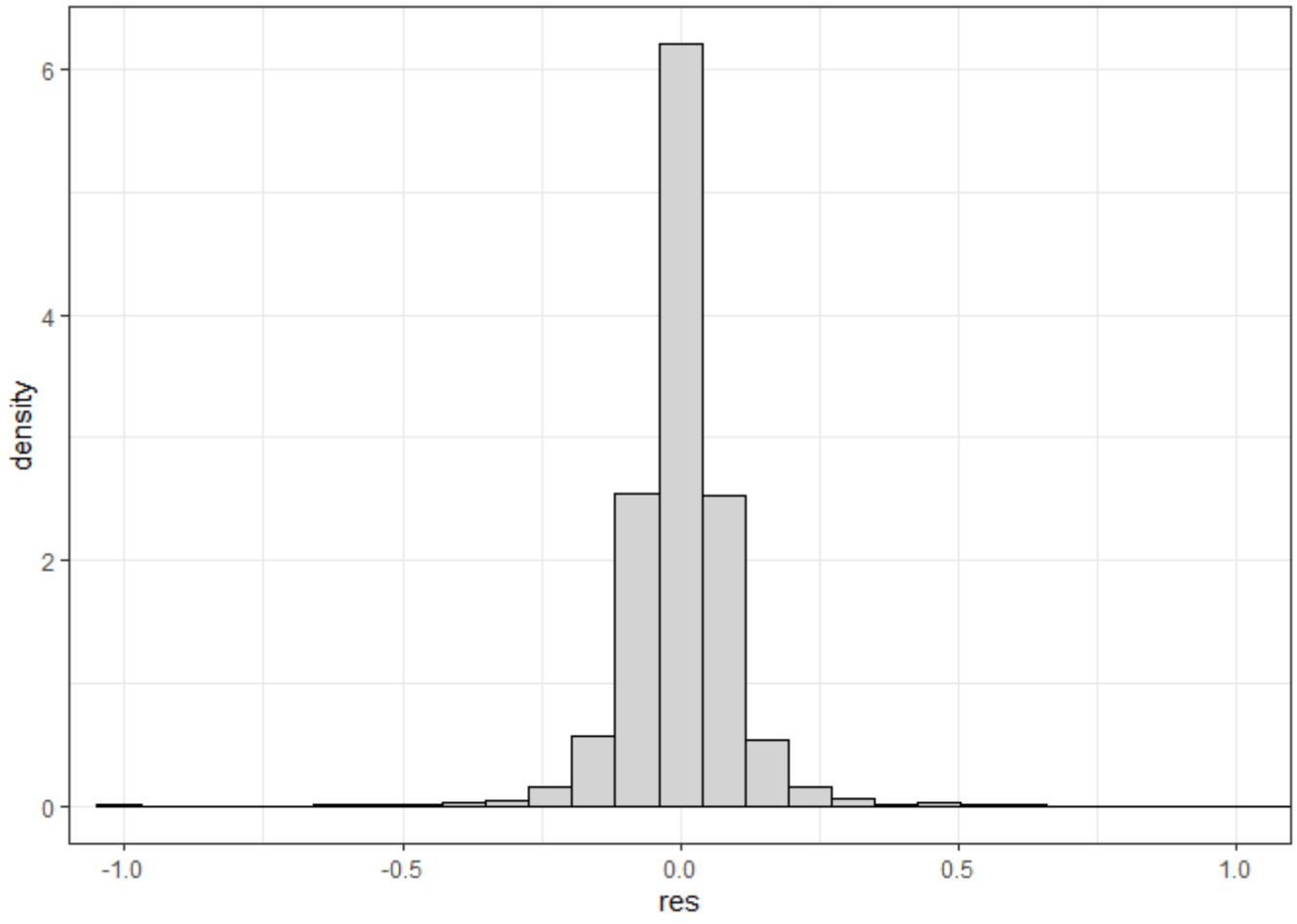


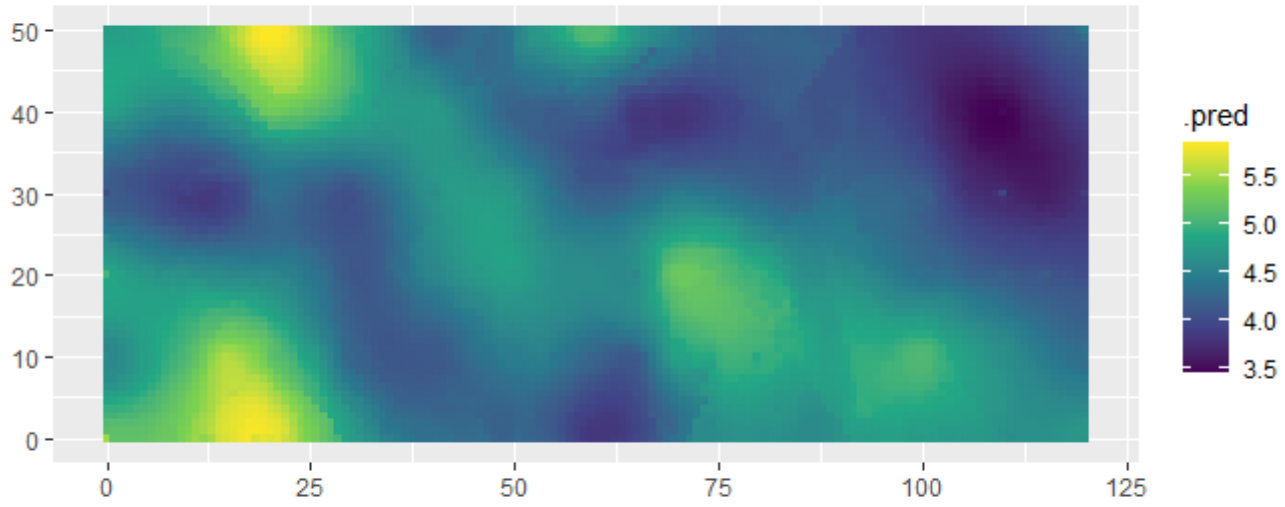

```
#> [1] "2017-02-22"
```

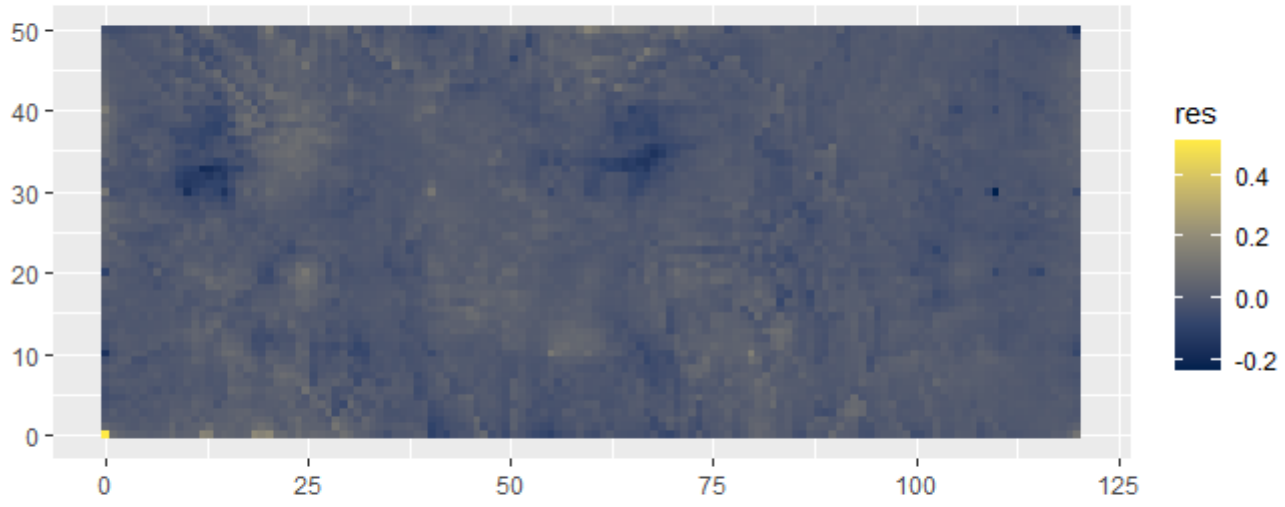

50
40
30
20
10
0
0
25
50
75
100
125
F
6.0
5.5
5.0
4.5
4.0
3.5

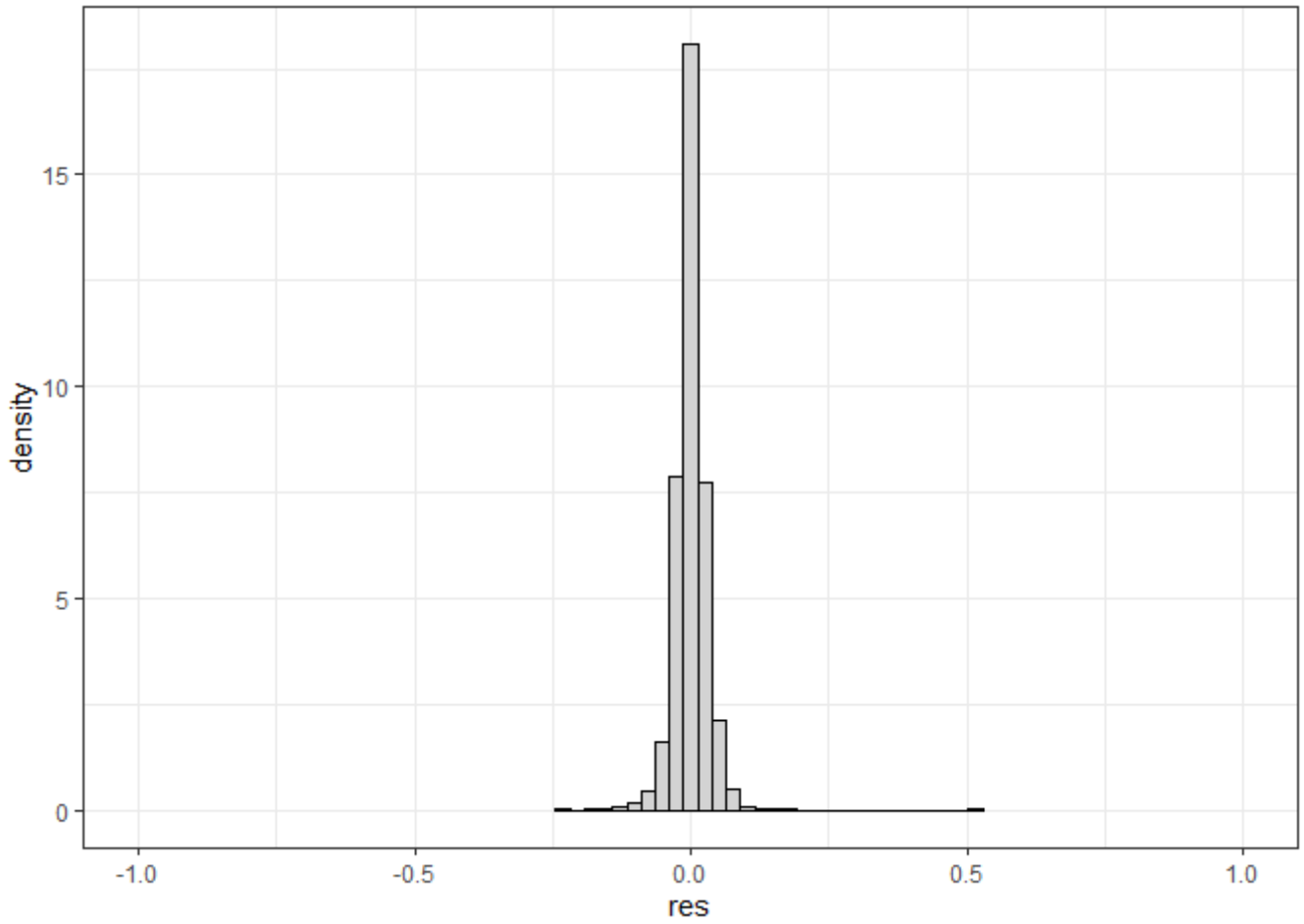

50
40
30
20
10
0
0
25
50
75
100
125
.pred
5.5
5.0
4.5
4.0
3.5


50
40
30
20
10
0
0
25
50
75
100
125
res
1.0
0.5
0.0
-0.5
-1.0


6
4
2
0
density
-1.0
-0.5
0.0
0.5
1.0
res


50
40
30
20
10
0
0
25
50
75
100
125
.pred
5.5
5.0
4.5
4.0
3.5


50
40
30
20
10
0
0
25
50
75
100
125
res
0.4
0.2
0.0
-0.2


density
15
10
5
0
-1.0
-0.5
0.0
0.5
1.0
res

Aprendizado de Máquina

```
# # Definindo a base de treino e a base de teste
# locais <- data_set %>% pull(local) %>% unique()
# for(i in seq_along(locais)){
#   lo <- locais[i]
#   data <- data_set %>% filter(local == lo)
#   dias <- data$data %>% unique()
#   for(j in seq_along(dias)){
#     di <- dias[j]
#     df <- data %>% filter(data == di)
#     fco2_initial_split <- initial_split(df  %>%
select(-c(id,ano,mes,dia,local)) %>%                                    sample_n
(trunc(nrow(df)*.51289)), prop = 0.75)
#     fco2_train <- training(fco2_initial_split)
#
#     hist_fco2 <- fco2_train  %>%
#       ggplot(aes(x=F, y=..density..))+
#       geom_histogram(bins = 30, color="black",  fill="lightgray")+
#       geom_density(alpha=.05,fill="red")+
#       theme_bw() +
#       labs(x="FCO2", y = "Densidade",title = paste(lo,di))
#     print(hist_fco2)
#
#     fco2_train   %>%
#       select(where(is.numeric)) %>%
#       drop_na() %>%
#       cor()  %>%
#       corrplot::corrplot()
#
#     fco2_recipe <- recipe(F ~ ., data = fco2_train) %>%
#       step_novel(all_nominal_predictors()) %>%
#       step_zv(all_predictors()) %>%
#       step_dummy(all_nominal_predictors())
#     bake(prep(fco2_recipe), new_data = NULL)
#     # visdat::vis_miss(bake(prep(fco2_recipe), new_data = NULL))
#
#     fco2_resamples <- vfold_cv(fco2_train, v = 5) #<-------
#
#     ### DECISION TREE
#     print("ARVORE DE DECISÃO")
#     fco2_dt_model <- decision_tree(
#       cost_complexity = tune(),
#       tree_depth = tune(),
#       min_n = tune()
#     )  %>%
#       set_mode("regression")  %>%
#       set_engine("rpart")
#
#     fco2_dt_wf <- workflow()   %>%
#       add_model(fco2_dt_model) %>%
```

```
#       add_recipe(fco2_recipe)
#
#     grid_dt <- grid_random(
#       cost_complexity(c(-6, -4)),
#       tree_depth(range = c(8, 18)),
#       min_n(range = c(42, 52)),
#       size = 2 # <-----------------------------
#     )
#
#     fco2_dt_tune_grid <- tune_grid(
#       fco2_dt_wf,
#       resamples = fco2_resamples,
#       grid = grid_dt,
#       metrics = metric_set(rmse)
#     )
#     print(autoplot(fco2_dt_tune_grid))
#
#     fco2_dt_best_params <- select_best(fco2_dt_tune_grid, "rmse")
#     fco2_dt_wf <- fco2_dt_wf %>% finalize_workflow(fco2_dt_best_params)
#     fco2_dt_last_fit <- last_fit(fco2_dt_wf, fco2_initial_split)
#
#     fco2_test_preds <- bind_rows(
#       collect_predictions(fco2_dt_last_fit)  %>%   mutate(modelo = "dt")
#     )
#     pre_obs_plot <- fco2_test_preds %>%
#       ggplot(aes(x=.pred, y=F)) +
#       geom_point()+
#       theme_bw() +
#       geom_smooth(method = "lm") +
#       stat_regline_equation(ggplot2::aes(
#         label =  paste(..eq.label.., ..rr.label.., sep = "*plain(\",\")~~")))+
#       labs(title = paste(lo,di))
#     print(pre_obs_plot)
#
#     fco2_modelo_final <- fco2_dt_wf %>% fit(df)
#     saveRDS(fco2_modelo_final,
#             paste0("models-3/fco2_modelo_dt_",lo,"_",di,".rds"))
#
#     fco2_dt_last_fit_model <- fco2_dt_last_fit$.workflow[[1]]$fit$fit
#     vip_plot <- vip(fco2_dt_last_fit_model,
#         aesthetics = list(color = "grey35", size = 0.8, fill="orange")) +
#       theme_bw()
#     print(vip_plot)
#     da <- fco2_test_preds %>%
#   filter(F > 0, .pred>0 )
#
#     my_r <- cor(da$F,da$.pred)
#     my_r2 <- my_r*my_r
#     my_mse <- Metrics::mse(da$F,da$.pred)
#     my_rmse <- Metrics::rmse(da$F,
#                              da$.pred)
#     my_mae <- Metrics::mae(da$F,da$.pred)
```

```
#     my_mape <- Metrics::mape(da$F,da$.pred)*100
#     vector_of_metrics <- c(r=my_r, R2=my_r2, MSE=my_mse,
#                            RMSE=my_rmse, MAE=my_mae, MAPE=my_mape)
#     print(data.frame(vector_of_metrics))
#
#
#     # ##RANDOM FOREST
#     print("RANDOM FOREST")
#     fco2_rf_model <- rand_forest(
#       min_n = tune(),
#       mtry = tune(),
#       trees = tune()
#     )   %>%
#       set_mode("regression")  %>%
#       set_engine("randomForest")
#
#     fco2_rf_wf <- workflow()   %>%
#       add_model(fco2_rf_model) %>%
#       add_recipe(fco2_recipe)
#
#     grid_rf <- grid_random(
#       min_n(range = c(20, 30)),
#       mtry(range = c(5, 10)),
#       trees(range = c(100, 500) ),
#       size = 2
#     )
#     fco2_rf_tune_grid <- tune_grid(
#       fco2_rf_wf,
#       resamples = fco2_resamples,
#       grid = grid_rf,
#       metrics = metric_set(rmse)
#     )
#     print(autoplot(fco2_rf_tune_grid))
#
#     fco2_rf_best_params <- select_best(fco2_rf_tune_grid, "rmse")
#     fco2_rf_wf <- fco2_rf_wf %>% finalize_workflow(fco2_rf_best_params)
#     fco2_rf_last_fit <- last_fit(fco2_rf_wf, fco2_initial_split)
#
#     fco2_test_preds <- bind_rows(
#       collect_predictions(fco2_rf_last_fit)  %>%   mutate(modelo = "rf")
#     )
#     pre_obs_plot <- fco2_test_preds %>%
#       ggplot(aes(x=.pred, y=F)) +
#       geom_point()+
#       theme_bw() +
#       geom_smooth(method = "lm") +
#       stat_regline_equation(ggplot2::aes(
#         label =  paste(..eq.label.., ..rr.label.., sep = "*plain(\",\")~~")))+
#       labs(title = paste(lo,di))
#     print(pre_obs_plot)
#
#     fco2_modelo_final <- fco2_rf_wf %>% fit(df)
```

```
#     saveRDS(fco2_modelo_final,
#             paste0("models-3/fco2_modelo_rf_",lo,"_",di,".rds"))
#
#     fco2_rf_last_fit_model <- fco2_rf_last_fit$.workflow[[1]]$fit$fit
#     vip_plot <- vip(fco2_rf_last_fit_model,
#         aesthetics = list(color = "grey35", size = 0.8, fill="orange")) +
#       theme_bw()
#     print(vip_plot)
#         da <- fco2_test_preds %>%
#   filter(F > 0, .pred>0 )
#
#     my_r <- cor(da$F,da$.pred)
#     my_r2 <- my_r*my_r
#     my_mse <- Metrics::mse(da$F,da$.pred)
#     my_rmse <- Metrics::rmse(da$F,
#                              da$.pred)
#     my_mae <- Metrics::mae(da$F,da$.pred)
#     my_mape <- Metrics::mape(da$F,da$.pred)*100
#     vector_of_metrics <- c(r=my_r, R2=my_r2, MSE=my_mse,
#                            RMSE=my_rmse, MAE=my_mae, MAPE=my_mape)
#     print(data.frame(vector_of_metrics))
#
#     ##XGBOOST
#     cores = 6
#     fco2_xgb_model <- boost_tree(
#       mtry = 0.8,
#       trees = tune(), # <---------------
#       min_n = 5,
#       tree_depth = 4,
#       loss_reduction = 0, # lambda
#       learn_rate = tune(), # epsilon
#       sample_size = 0.8
#     )  %>%
#       set_mode("regression")  %>%
#       set_engine("xgboost", nthread = cores, counts = FALSE)
#
#     fco2_xgb_wf <- workflow()  %>%
#       add_model(fco2_xgb_model) %>%
#       add_recipe(fco2_recipe)
#
#     grid_xgb <- expand.grid(
#       learn_rate =  c(0.05, 0.3),
#       trees = c(2, 250, 500)
#     )
#
#     #passo 1
#     fco2_xgb_tune_grid <- tune_grid(
#       fco2_xgb_wf,
#       resamples = fco2_resamples,
#       grid = grid_xgb,
#       metrics = metric_set(rmse)
#     )
```

```
#     # print(autoplot(fco2_xgb_tune_grid))
#     fco2_xgb_select_best_passo1 <- fco2_xgb_tune_grid %>%
#       select_best(metric = "rmse")
#
#     # passo 2
#     fco2_xgb_model <- boost_tree(
#       mtry = 0.8,
#       trees = fco2_xgb_select_best_passo1$trees,
#       min_n = tune(),
#       tree_depth = tune(),
#       loss_reduction = 0,
#       learn_rate = fco2_xgb_select_best_passo1$learn_rate,
#       sample_size = 0.8
#     ) %>%
#       set_mode("regression")  %>%
#       set_engine("xgboost", nthread = cores, counts = FALSE)
#     fco2_xgb_wf <- workflow() %>%
#       add_model(fco2_xgb_model)   %>%
#       add_recipe(fco2_recipe)
#
#     fco2_xgb_grid <- expand.grid(
#       tree_depth = c(1, 3, 4),
#       min_n = c(5, 30, 60)
#     )
#
#     fco2_xgb_tune_grid <- fco2_xgb_wf   %>%
#       tune_grid(
#         resamples =fco2_resamples,
#         grid = fco2_xgb_grid,
#         control = control_grid(save_pred = TRUE, verbose = FALSE, allow_par = TRUE),
#         metrics = metric_set(rmse)
#       )
#     fco2_xgb_select_best_passo2 <- fco2_xgb_tune_grid  %>%   select_best(metric = "rms
e")
#
#     # passo 3
#     fco2_xgb_model <- boost_tree(
#       mtry = 0.8,
#       trees = fco2_xgb_select_best_passo1$trees,
#       min_n = fco2_xgb_select_best_passo2$min_n,
#       tree_depth = fco2_xgb_select_best_passo2$tree_depth,
#       loss_reduction =tune(),
#       learn_rate = fco2_xgb_select_best_passo1$learn_rate,
#       sample_size = 0.8
#     )  %>%
#       set_mode("regression")  %>%
#       set_engine("xgboost", nthread = cores, counts = FALSE)
#
#     fco2_xgb_wf <- workflow()  %>%
#       add_model(fco2_xgb_model)  %>%
#       add_recipe(fco2_recipe)
#
```

```
#     #### Grid
#     fco2_xgb_grid <- expand.grid(
#       loss_reduction = c(0.01, 0.05, 1, 2, 4, 8)
#     )
#
#     fco2_xgb_tune_grid <- fco2_xgb_wf   %>%
#       tune_grid(
#         resamples = fco2_resamples,
#         grid = fco2_xgb_grid,
#         control = control_grid(save_pred = TRUE, verbose = FALSE, allow_par = TRUE),
#         metrics = metric_set(rmse)
#       )
#     fco2_xgb_select_best_passo3 <- fco2_xgb_tune_grid %>% select_best(metric = "rmse")
#
#     # passo 4
#     fco2_xgb_model <- boost_tree(
#       mtry = tune(),
#       trees = fco2_xgb_select_best_passo1$trees,
#       min_n = fco2_xgb_select_best_passo2$min_n,
#       tree_depth = fco2_xgb_select_best_passo2$tree_depth,
#       loss_reduction = fco2_xgb_select_best_passo3$loss_reduction,
#       learn_rate = fco2_xgb_select_best_passo1$learn_rate,
#       sample_size = tune()
#     )%>%
#       set_mode("regression")  |>
#       set_engine("xgboost", nthread = cores, counts = FALSE)
#
#       fco2_xgb_wf <- workflow()  %>%
#       add_model(fco2_xgb_model)  %>%
#       add_recipe(fco2_recipe)
#
#     fco2_xgb_grid <- expand.grid(
#       sample_size = seq(0.5, 1.0, length.out = 2), ## <---
#       mtry = seq(0.1, 1.0, length.out = 2) ## <---
#     )
#
#     fco2_xgb_tune_grid <- fco2_xgb_wf   %>%
#       tune_grid(
#         resamples = fco2_resamples,
#         grid = fco2_xgb_grid,
#         control = control_grid(save_pred = TRUE, verbose = FALSE, allow_par = TRUE),
#         metrics = metric_set(rmse)
#       )
#     fco2_xgb_select_best_passo4 <- fco2_xgb_tune_grid   %>%   select_best(metric = "rm
se")
#
#     # passo 5
#     fco2_xgb_model <- boost_tree(
#       mtry = fco2_xgb_select_best_passo4$mtry,
#       trees = tune(),
#       min_n = fco2_xgb_select_best_passo2$min_n,
#       tree_depth = fco2_xgb_select_best_passo2$tree_depth,
```

```
#       loss_reduction = fco2_xgb_select_best_passo3$loss_reduction,
#       learn_rate = tune(),
#       sample_size = fco2_xgb_select_best_passo4$sample_size
#     )  %>%
#       set_mode("regression")  %>%
#       set_engine("xgboost", nthread = cores, counts = FALSE)
#
#
#     fco2_xgb_wf <- workflow() %>%
#       add_model(fco2_xgb_model)  %>%
#       add_recipe(fco2_recipe)
#
#
#     fco2_xgb_grid <- expand.grid(
#       learn_rate = c(0.05, 0.10, 0.15, 0.25),
#       trees = c(100, 250, 500)
#     )
#
#     fco2_xgb_tune_grid <- fco2_xgb_wf   %>%
#       tune_grid(
#         resamples = fco2_resamples,
#         grid = fco2_xgb_grid,
#         control = control_grid(save_pred = TRUE, verbose = FALSE, allow_par = TRUE),
#         metrics = metric_set(rmse)
#       )
#     fco2_xgb_select_best_passo5 <- fco2_xgb_tune_grid   %>%   select_best(metric = "rm
se")
#
#     ## modelos final desempenho
#     fco2_xgb_model <- boost_tree(
#       mtry = fco2_xgb_select_best_passo4$mtry,
#       trees = fco2_xgb_select_best_passo5$trees,
#       min_n = fco2_xgb_select_best_passo2$min_n,
#       tree_depth = fco2_xgb_select_best_passo2$tree_depth,
#       loss_reduction = fco2_xgb_select_best_passo3$loss_reduction,
#       learn_rate = fco2_xgb_select_best_passo5$learn_rate,
#       sample_size = fco2_xgb_select_best_passo4$sample_size
#     ) %>%
#       set_mode("regression")  %>%
#       set_engine("xgboost", nthread = cores, counts = FALSE)
#
#     df_par <- data.frame(
#       mtry = fco2_xgb_select_best_passo4$mtry,
#       trees = fco2_xgb_select_best_passo5$trees,
#       min_n = fco2_xgb_select_best_passo2$min_n,
#       tree_depth = fco2_xgb_select_best_passo2$tree_depth,
#       loss_reduction = fco2_xgb_select_best_passo3$loss_reduction,
#       learn_rate = fco2_xgb_select_best_passo5$learn_rate,
#       sample_size = fco2_xgb_select_best_passo4$sample_size
#     )
#     fco2_xgb_wf <- fco2_xgb_wf %>% finalize_workflow(df_par) # <------
#     fco2_xgb_last_fit <- last_fit(fco2_xgb_wf, fco2_initial_split)
```

```
#
#     fco2_test_preds <- bind_rows(
#       collect_predictions(fco2_xgb_last_fit)  %>%   mutate(modelo = "xgb")
#     )
#     pre_obs_plot <- fco2_test_preds %>%
#       ggplot(aes(x=.pred, y=F)) +
#       geom_point()+
#       theme_bw() +
#       geom_smooth(method = "lm") +
#       stat_regline_equation(ggplot2::aes(
#         label =  paste(..eq.label.., ..rr.label.., sep = "*plain(\",\")~~")))+
#       labs(title = paste(lo,di))
#     print(pre_obs_plot)
#
#     fco2_modelo_final <- fco2_xgb_wf %>% fit(df)
#     saveRDS(fco2_modelo_final,
#             paste0("models-3/fco2_modelo_xgb_",lo,"_",di,".rds"))
#
#     fco2_xgb_last_fit_model <- fco2_xgb_last_fit$.workflow[[1]]$fit$fit
#     vip_plot <- vip(fco2_xgb_last_fit_model,
#         aesthetics = list(color = "grey35", size = 0.8, fill="orange")) +
#       theme_bw()
#     print(vip_plot)
#
#         da <- fco2_test_preds %>%
#   filter(F > 0, .pred>0 )
#
#     my_r <- cor(da$F,da$.pred)
#     my_r2 <- my_r*my_r
#     my_mse <- Metrics::mse(da$F,da$.pred)
#     my_rmse <- Metrics::rmse(da$F,
#                              da$.pred)
#     my_mae <- Metrics::mae(da$F,da$.pred)
#     my_mape <- Metrics::mape(da$F,da$.pred)*100
#     vector_of_metrics <- c(r=my_r, R2=my_r2, MSE=my_mse,
#                            RMSE=my_rmse, MAE=my_mae, MAPE=my_mape)
#     print(data.frame(vector_of_metrics))
#   }
# }
```

Material de Apêndice - Sinais (mapas dos atributos geoespacializados)

Mapas Eucalipto

```
# for(i in seq(files_eu)){
#   mp<-read.table(files_eu[i],skip = 5)
#   image(mp %>%  as.matrix(),xlab = files_eu[i])
# }
```

Mapas Silvipastoril

```
# for(i in seq(files_sp)){
#   mp<-read.table(files_sp[i],skip = 5)
#   image(mp %>%  as.matrix(),xlab = files_sp[i])
# }
```