

# Manuscript results: Forecasting influenza using ARIMA and ARIMAX

Victor Felix

January 14, 2026

This document has the results that we present in our manuscript draft.

Loading libraries.

```
library("tidyverse")
library("MMWRweek")
library("data.table")
library("caret")

## Loading required package: ggplot2

## Loading required package: lattice

library("purrr")

##
## Attaching package: 'purrr'

## The following object is masked from 'package:caret':
##     lift

## The following object is masked from 'package:data.table':
##     transpose

library("dplyr")

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:data.table':
##     between, first, last

## The following objects are masked from 'package:stats':
##     filter, lag
```

```

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library("tseries")

## Registered S3 method overwritten by 'quantmod':
##   method           from
##   as.zoo.data.frame zoo

library("gtools")
library("forecast")
library("scoringutils")

## Note: scoringutils is currently undergoing major development changes (with an update planned for the
##       next few weeks). See https://github.com/ropensci/scoringutils/pull/167 for more details.

library("covidHubUtils")
library("parallel")
library("future")#https://cran.r-project.org/web/packages/future/vignettes/future-4-issues.html

## 
## Attaching package: 'future'

## The following object is masked from 'package:tseries':
##
##     value

## The following object is masked from 'package:caret':
##
##     cluster

library("listenv")

## 
## Attaching package: 'listenv'

## The following object is masked from 'package:purrr':
##
##     map

library("epitools")
library("ggplot2")
library("sf")

## Linking to GEOS 3.11.0, GDAL 3.5.3, PROJ 9.1.0; sf_use_s2() is TRUE

library("forcats")
library("ggplot2")
library("sf")
library("scales")

```

```

## 
## Attaching package: 'scales'

## The following object is masked from 'package:purrr':
## 
##     discard

library("ggplot2")
library("broom")
library("fields")

## Loading required package: spam

## Spam version 2.10-0 (2023-10-23) is loaded.
## Type 'help( Spam)' or 'demo( spam)' for a short introduction
## and overview of this package.
## Help for individual functions is also obtained by adding the
## suffix '.spam' to the function name, e.g. 'help( chol.spam)'.

## 
## Attaching package: 'spam'

## The following objects are masked from 'package:base':
## 
##     backsolve, forwardsolve

## Loading required package: viridisLite

## 
## Try help(fields) to get started.

library("ggpubr")

## 
## Attaching package: 'ggpubr'

## The following object is masked from 'package:forecast':
## 
##     gghistogram

library("patchwork")
library("ggpattern")
library("cowplot")

## 
## Attaching package: 'cowplot'

## The following object is masked from 'package:patchwork':
## 
##     align_plots

```

```

## The following object is masked from 'package:ggpubr':
##
##     get_legend

library("lme4")

## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following object is masked from 'package:spam':
##
##     det

## The following objects are masked from 'package:tidyverse':
##
##     expand, pack, unpack

library("ez")
library("ggpubr")

```

Loading the results of each model and the shapefiles of the maps.

```

load("models_without_logback/ES_ARIMA/ARIMA_MODELS_influenza_hospitalization_nolog.Rdata")
load("models_without_logback/ES_ADJACENT/ADJACENT_MODELS_influenza_hospitalization_nolog.Rdata")
load("models_without_logback/ES_EPIWEEK/EPIWEEK_MODELS_influenza_hospitalization_nolog.Rdata")
load("models_without_logback/ES_TEMPERATURE/TEMPERATURE_MODELS_influenza_hospitalization_nolog.Rdata")
load("models_without_logback/ES_AVERAGE/AVERAGE_MODELS_influenza_hospitalization_nolog.Rdata")

states <- read_sf("models_without_logback/shapefiles/cb_2018_us_state_500k.shp")

states <- states %>%
  rename(STATE = NAME)

```

---

1 Week ahead

---

```

# Creating new columns with Epidemiological weeks based on target_end_week
AUTO_ARIMA_WEEK1$epiweek <- MMWRweek(AUTO_ARIMA_WEEK1$target_end_date)$MMWRweek
AUTO_ADJACENT_WEEK1$epiweek <- MMWRweek(AUTO_ADJACENT_WEEK1$target_end_date)$MMWRweek
AUTO_EPIWEEK_WEEK1$epiweek <- MMWRweek(AUTO_EPIWEEK_WEEK1$target_end_date)$MMWRweek
AUTO_TEMPERATURE_WEEK1$epiweek <- MMWRweek(AUTO_TEMPERATURE_WEEK1$target_end_date)$MMWRweek
AUTO_AVERAGE_WEEK1$epiweek <- MMWRweek(AUTO_AVERAGE_WEEK1$target_end_date)$MMWRweek

ES27_ARIMA_WEEK1$epiweek <- MMWRweek(ES27_ARIMA_WEEK1$target_end_date)$MMWRweek
ES27_ADJACENT_WEEK1$epiweek <- MMWRweek(ES27_ADJACENT_WEEK1$target_end_date)$MMWRweek
ES27_EPIWEEK_WEEK1$epiweek <- MMWRweek(ES27_EPIWEEK_WEEK1$target_end_date)$MMWRweek
ES27_TEMPERATURE_WEEK1$epiweek <- MMWRweek(ES27_TEMPERATURE_WEEK1$target_end_date)$MMWRweek
ES27_AVERAGE_WEEK1$epiweek <- MMWRweek(ES27_AVERAGE_WEEK1$target_end_date)$MMWRweek

```

```

ES64_ARIMA_WEEK1$epiweek <- MMWRweek(ES64_ARIMA_WEEK1$target_end_date)$MMWRweek
ES64_ADJACENT_WEEK1$epiweek <- MMWRweek(ES64_ADJACENT_WEEK1$target_end_date)$MMWRweek
ES64_EPIWEEK_WEEK1$epiweek <- MMWRweek(ES64_EPIWEEK_WEEK1$target_end_date)$MMWRweek
ES64_TEMPERATURE_WEEK1$epiweek <- MMWRweek(ES64_TEMPERATURE_WEEK1$target_end_date)$MMWRweek
ES64_AVERAGE_WEEK1$epiweek <- MMWRweek(ES64_AVERAGE_WEEK1$target_end_date)$MMWRweek

# Dataframe that will be analysed for 1 Week Ahead
df_W1_NoLg <- data.frame(
  STATE = AUTO_ARIMA_WEEK1$State,
  Julian_date = AUTO_ARIMA_WEEK1$target_end_date,
  epiweek = AUTO_ARIMA_WEEK1$epiweek,
  AUTO_AR=AUTO_ARIMA_WEEK1$WIS,
  AUTO_ADJ=AUTO_ADJACENT_WEEK1$WIS,
  AUTO_EPI=AUTO_EPIWEEK_WEEK1$WIS,
  AUTO_TEMP=AUTO_TEMPERATURE_WEEK1$WIS,
  AUTO_AVG=AUTO_AVERAGE_WEEK1$WIS,
  ES27_AR=ES27_ARIMA_WEEK1$WIS,
  ES27_ADJ=ES27_ADJACENT_WEEK1$WIS,
  ES27_EPI=ES27_EPIWEEK_WEEK1$WIS,
  ES27_TEMP=ES27_TEMPERATURE_WEEK1$WIS,
  ES27_AVG=ES27_AVERAGE_WEEK1$WIS,
)
head(df_W1_NoLg)

```

	STATE	Julian_date	epiweek	AUTO_AR	AUTO_ADJ	AUTO_EPI	AUTO_TEMP
## 1	Alabama	2022-10-29	43	102.94840	101.65743	116.57567	102.44553
## 2	Alabama	2022-11-05	44	79.69032	69.21610	74.91468	79.44376
## 3	Alabama	2022-11-12	45	62.70926	49.99889	35.82769	155.63500
## 4	Alabama	2022-11-19	46	48.13570	37.31865	16.86871	58.72064
## 5	Alabama	2022-11-26	47	62.47143	61.26857	71.46483	61.73031
## 6	Alabama	2022-12-03	48	92.04712	93.99361	98.09041	103.34642
	AUTO_AVG	ES27_AR	ES27_ADJ	ES27_EPI	ES27_TEMP	ES27_AVG	ES64_AR
## 1	111.250523	90.719035	90.791737	92.673037	90.729616	90.986378	88.414739
## 2	47.411877	9.385328	7.062736	9.397495	9.483206	7.024391	3.165721
## 3	78.333058	153.181002	152.552002	153.031292	153.171927	157.087082	170.008450
## 4	37.777085	60.914034	72.241112	61.554400	62.060894	65.839393	68.968221
## 5	66.087165	25.397894	41.902088	24.594584	24.277733	24.098040	23.129469
## 6	6.332483	83.516853	66.325397	77.939015	88.073395	9.997281	81.674040
	ES64_ADJ	ES64_EPI	ES64_TEMP	ES64_AVG			
## 1	89.834739	93.67342	88.803267	90.337167			
## 2	3.068386	3.17232	3.132254	3.094771			
## 3	166.985799	169.74417	170.208228	176.077362			
## 4	78.645976	67.15577	67.642403	70.855246			
## 5	44.816094	23.61011	21.939414	23.700555			
## 6	43.209734	74.32642	84.495765	13.384597			

---

2 Weeks ahead

---

```
# Creating new columns with Epidemiological weeks based on target_end_week
AUTO_ARIMA_WEEK2$epiweek <- MMWRweek(AUTO_ARIMA_WEEK2$target_end_date)$MMWRweek
AUTO_ADJACENT_WEEK2$epiweek <- MMWRweek(AUTO_ADJACENT_WEEK2$target_end_date)$MMWRweek
AUTO_EPIWEEK_WEEK2$epiweek <- MMWRweek(AUTO_EPIWEEK_WEEK2$target_end_date)$MMWRweek
AUTO_TEMPERATURE_WEEK2$epiweek <- MMWRweek(AUTO_TEMPERATURE_WEEK2$target_end_date)$MMWRweek
AUTO_AVERAGE_WEEK2$epiweek <- MMWRweek(AUTO_AVERAGE_WEEK2$target_end_date)$MMWRweek

ES27_ARIMA_WEEK2$epiweek <- MMWRweek(ES27_ARIMA_WEEK2$target_end_date)$MMWRweek
ES27_ADJACENT_WEEK2$epiweek <- MMWRweek(ES27_ADJACENT_WEEK2$target_end_date)$MMWRweek
ES27_EPIWEEK_WEEK2$epiweek <- MMWRweek(ES27_EPIWEEK_WEEK2$target_end_date)$MMWRweek
ES27_TEMPERATURE_WEEK2$epiweek <- MMWRweek(ES27_TEMPERATURE_WEEK2$target_end_date)$MMWRweek
ES27_AVERAGE_WEEK2$epiweek <- MMWRweek(ES27_AVERAGE_WEEK2$target_end_date)$MMWRweek

ES64_ARIMA_WEEK2$epiweek <- MMWRweek(ES64_ARIMA_WEEK2$target_end_date)$MMWRweek
ES64_ADJACENT_WEEK2$epiweek <- MMWRweek(ES64_ADJACENT_WEEK2$target_end_date)$MMWRweek
ES64_EPIWEEK_WEEK2$epiweek <- MMWRweek(ES64_EPIWEEK_WEEK2$target_end_date)$MMWRweek
ES64_TEMPERATURE_WEEK2$epiweek <- MMWRweek(ES64_TEMPERATURE_WEEK2$target_end_date)$MMWRweek
ES64_AVERAGE_WEEK2$epiweek <- MMWRweek(ES64_AVERAGE_WEEK2$target_end_date)$MMWRweek

# Dataframe that will be analysed for 2 Weeks Ahead
df_W2_NoLg <- data.frame(
  STATE = AUTO_ARIMA_WEEK2$State,
  Julian_date = AUTO_ARIMA_WEEK2$target_end_date,
  epiweek = AUTO_ARIMA_WEEK2$epiweek,
  AUTO_AR=AUTO_ARIMA_WEEK2$WIS,
  AUTO_ADJ=AUTO_ADJACENT_WEEK2$WIS,
  AUTO_EPI=AUTO_EPIWEEK_WEEK2$WIS,
  AUTO_TEMP=AUTO_TEMPERATURE_WEEK2$WIS,
  AUTO_AVG=AUTO_AVERAGE_WEEK2$WIS,

  ES27_AR=ES27_ARIMA_WEEK2$WIS,
  ES27_ADJ=ES27_ADJACENT_WEEK2$WIS,
  ES27_EPI=ES27_EPIWEEK_WEEK2$WIS,
  ES27_TEMP=ES27_TEMPERATURE_WEEK2$WIS,
  ES27_AVG=ES27_AVERAGE_WEEK2$WIS,

  ES64_AR=ES64_ARIMA_WEEK2$WIS,
  ES64_ADJ=ES64_ADJACENT_WEEK2$WIS,
  ES64_EPI=ES64_EPIWEEK_WEEK2$WIS,
  ES64_TEMP=ES64_TEMPERATURE_WEEK2$WIS,
  ES64_AVG=ES64_AVERAGE_WEEK2$WIS
)

head(df_W2_NoLg)
```

```
##      STATE Julian_date epiweek    AUTO_AR    AUTO_ADJ    AUTO_EPI    AUTO_TEMP
## 1 Alabama  2022-11-05     44 188.905312 183.765343 228.179513 185.137841
## 2 Alabama  2022-11-12     45  48.034143  39.218917  42.748513  48.165309
## 3 Alabama  2022-11-19     46  81.098716  69.656879  28.840078 357.027657
```

```

## 4 Alabama 2022-11-26      47 34.593933  6.491932 63.573618 67.475787
## 5 Alabama 2022-12-03      48 237.947010 270.813086 251.280702 236.723180
## 6 Alabama 2022-12-10      49   7.307581   7.344853  9.448372  7.854465
##    AUTO_AVG  ES27_AR  ES27_ADJ  ES27_EPI  ES27_TEMP  ES27_AVG  ES64_AR
## 1 213.799620 156.55311 157.25555 157.82259 156.42450 155.52180 150.37049
## 2 11.773647 114.29758 117.34220 114.00493 113.59390 111.78869 155.92213
## 3 100.827045 351.92105 345.19642 353.19759 351.90586 340.90754 401.78478
## 4  8.185654  65.67085  71.09070  66.92643  67.50840  69.30220  82.35596
## 5 254.196723 144.76162 167.68187 143.08761 143.31461 143.52997 133.83584
## 6  85.378319 18.31165 28.74939 25.30183 16.05396  97.09043 15.94977
##    ES64_ADJ  ES64_EPI  ES64_TEMP  ES64_AVG
## 1 154.16654 159.51410 150.98968 154.60212
## 2 151.59357 157.08640 156.57325 158.11796
## 3 394.80115 402.05978 402.16290 403.14849
## 4  88.55030  79.36624  78.91779  81.12383
## 5 175.05363 135.69637 132.76973 137.25642
## 6  64.03856  24.95665  15.18490  83.27090

```

3 Weeks ahead

```

# Creating new columns with Epidemiological weeks based on target_end_week
AUTO_ARIMA_WEEK3$epiweek <- MMWRweek(AUTO_ARIMA_WEEK3$target_end_date)$MMWRweek
AUTO_ADJACENT_WEEK3$epiweek <- MMWRweek(AUTO_ADJACENT_WEEK3$target_end_date)$MMWRweek
AUTO_EPIWEEK_WEEK3$epiweek <- MMWRweek(AUTO_EPIWEEK_WEEK3$target_end_date)$MMWRweek
AUTO_TEMPERATURE_WEEK3$epiweek <- MMWRweek(AUTO_TEMPERATURE_WEEK3$target_end_date)$MMWRweek
AUTO_AVERAGE_WEEK3$epiweek <- MMWRweek(AUTO_AVERAGE_WEEK3$target_end_date)$MMWRweek

ES27_ARIMA_WEEK3$epiweek <- MMWRweek(ES27_ARIMA_WEEK3$target_end_date)$MMWRweek
ES27_ADJACENT_WEEK3$epiweek <- MMWRweek(ES27_ADJACENT_WEEK3$target_end_date)$MMWRweek
ES27_EPIWEEK_WEEK3$epiweek <- MMWRweek(ES27_EPIWEEK_WEEK3$target_end_date)$MMWRweek
ES27_TEMPERATURE_WEEK3$epiweek <- MMWRweek(ES27_TEMPERATURE_WEEK3$target_end_date)$MMWRweek
ES27_AVERAGE_WEEK3$epiweek <- MMWRweek(ES27_AVERAGE_WEEK3$target_end_date)$MMWRweek

ES64_ARIMA_WEEK3$epiweek <- MMWRweek(ES64_ARIMA_WEEK3$target_end_date)$MMWRweek
ES64_ADJACENT_WEEK3$epiweek <- MMWRweek(ES64_ADJACENT_WEEK3$target_end_date)$MMWRweek
ES64_EPIWEEK_WEEK3$epiweek <- MMWRweek(ES64_EPIWEEK_WEEK3$target_end_date)$MMWRweek
ES64_TEMPERATURE_WEEK3$epiweek <- MMWRweek(ES64_TEMPERATURE_WEEK3$target_end_date)$MMWRweek
ES64_AVERAGE_WEEK3$epiweek <- MMWRweek(ES64_AVERAGE_WEEK3$target_end_date)$MMWRweek

# Dataframe that will be analysed for 3 Weeks Ahead
df_W3_NoLg <- data.frame(
  STATE = AUTO_ARIMA_WEEK3$State,
  Julian_date = AUTO_ARIMA_WEEK3$target_end_date,
  epiweek = AUTO_ARIMA_WEEK3$epiweek,
  AUTO_AR=AUTO_ARIMA_WEEK3$WIS,
  AUTO_ADJ=AUTO_ADJACENT_WEEK3$WIS,
  AUTO_EPI=AUTO_EPIWEEK_WEEK3$WIS,
  AUTO_TEMP=AUTO_TEMPERATURE_WEEK3$WIS,
  AUTO_AVG=AUTO_AVERAGE_WEEK3$WIS,
  ES27_AR=ES27_ARIMA_WEEK3$WIS,
  ES27_ADJ=ES27_ADJACENT_WEEK3$WIS,

```

```

ES27_EPI=ES27_EPIWEEK_WEEK3$WIS,
ES27_TEMP=ES27_TEMPERATURE_WEEK3$WIS,
ES27_AVG=ES27_AVERAGE_WEEK3$WIS,

ES64_AR=ES64_ARIMA_WEEK3$WIS,
ES64_ADJ=ES64_ADJACENT_WEEK3$WIS,
ES64_EPI=ES64_EPIWEEK_WEEK3$WIS,
ES64_TEMP=ES64_TEMPERATURE_WEEK3$WIS,
ES64_AVG=ES64_AVERAGE_WEEK3$WIS
)

head(df_W3_NoLg)

##      STATE Julian_date epiweek    AUTO_AR    AUTO_ADJ    AUTO_EPI    AUTO_TEMP
## 1 Alabama 2022-11-12       45 146.78553 132.24099 226.16803 134.66404
## 2 Alabama 2022-11-19       46  48.22303  39.47050  41.58940  48.75787
## 3 Alabama 2022-11-26       47 10.19427 10.10276 41.42233 478.20811
## 4 Alabama 2022-12-03       48 43.25931 137.56318 328.57147 10.67010
## 5 Alabama 2022-12-10       49 146.32655 199.26107 146.15502 144.52135
## 6 Alabama 2022-12-17       50 59.17719 47.79918 62.09687 58.76758
##      AUTO_AVG    ES27_AR    ES27_ADJ    ES27_EPI    ES27_TEMP    ES27_AVG    ES64_AR
## 1 177.978661 83.23148 84.55082 82.15597 82.94497 72.72026 70.96738
## 2 7.153594 289.38002 291.27312 288.86154 288.37530 279.21031 376.54043
## 3 28.768875 470.10331 457.53116 473.27827 470.02991 445.94359 575.24763
## 4 113.515485 12.58747 11.86176 12.64650 12.58217 12.27429 14.35102
## 5 174.117903 30.72114 10.88272 29.74189 30.00309 10.73607 23.73183
## 6 147.677074 83.17768 99.08706 93.65631 78.85432 200.26034 80.44222
##      ES64_ADJ    ES64_EPI    ES64_TEMP    ES64_AVG
## 1 77.42024 77.55526 71.70535 73.02268
## 2 372.94951 379.43593 377.65242 383.92138
## 3 565.90367 578.05058 575.84232 575.00321
## 4 16.34616 14.23757 13.84466 12.75210
## 5 17.03998 26.52609 24.55276 11.48884
## 6 162.09347 92.96110 78.79480 178.76213

```

4 Weeks ahead

```

# Creating new columns with Epidemiological weeks based on target_end_week
AUTO_ARIMA_WEEK4$epiweek <- MMWRweek(AUTO_ARIMA_WEEK4$target_end_date)$MMWRweek
AUTO_ADJACENT_WEEK4$epiweek <- MMWRweek(AUTO_ADJACENT_WEEK4$target_end_date)$MMWRweek
AUTO_EPIWEEK_WEEK4$epiweek <- MMWRweek(AUTO_EPIWEEK_WEEK4$target_end_date)$MMWRweek
AUTO_TEMPERATURE_WEEK4$epiweek <- MMWRweek(AUTO_TEMPERATURE_WEEK4$target_end_date)$MMWRweek
AUTO_AVERAGE_WEEK4$epiweek <- MMWRweek(AUTO_AVERAGE_WEEK4$target_end_date)$MMWRweek

ES27_ARIMA_WEEK4$epiweek <- MMWRweek(ES27_ARIMA_WEEK4$target_end_date)$MMWRweek
ES27_ADJACENT_WEEK4$epiweek <- MMWRweek(ES27_ADJACENT_WEEK4$target_end_date)$MMWRweek
ES27_EPIWEEK_WEEK4$epiweek <- MMWRweek(ES27_EPIWEEK_WEEK4$target_end_date)$MMWRweek
ES27_TEMPERATURE_WEEK4$epiweek <- MMWRweek(ES27_TEMPERATURE_WEEK4$target_end_date)$MMWRweek
ES27_AVERAGE_WEEK4$epiweek <- MMWRweek(ES27_AVERAGE_WEEK4$target_end_date)$MMWRweek

ES64_ARIMA_WEEK4$epiweek <- MMWRweek(ES64_ARIMA_WEEK4$target_end_date)$MMWRweek

```

```

ES64_ADJACENT_WEEK4$epiweek <- MMWRweek(ES64_ADJACENT_WEEK4$target_end_date)$MMWRweek
ES64_EPIWEEK_WEEK4$epiweek <- MMWRweek(ES64_EPIWEEK_WEEK4$target_end_date)$MMWRweek
ES64_TEMPERATURE_WEEK4$epiweek <- MMWRweek(ES64_TEMPERATURE_WEEK4$target_end_date)$MMWRweek
ES64_AVERAGE_WEEK4$epiweek <- MMWRweek(ES64_AVERAGE_WEEK4$target_end_date)$MMWRweek

# Dataframe that will be analysed for 4 Weeks Ahead
df_W4_NoLg <- data.frame(
  STATE = AUTO_ARIMA_WEEK4$State,
  Julian_date = AUTO_ARIMA_WEEK4$target_end_date,
  epiweek = AUTO_ARIMA_WEEK4$epiweek,
  AUTO_AR=AUTO_ARIMA_WEEK4$WIS,
  AUTO_ADJ=AUTO_ADJACENT_WEEK4$WIS,
  AUTO_EPI=AUTO_EPIWEEK_WEEK4$WIS,
  AUTO_TEMP=AUTO_TEMPERATURE_WEEK4$WIS,
  AUTO_AVG=AUTO_AVERAGE_WEEK4$WIS,

  ES27_AR=ES27_ARIMA_WEEK4$WIS,
  ES27_ADJ=ES27_ADJACENT_WEEK4$WIS,
  ES27_EPI=ES27_EPIWEEK_WEEK4$WIS,
  ES27_TEMP=ES27_TEMPERATURE_WEEK4$WIS,
  ES27_AVG=ES27_AVERAGE_WEEK4$WIS,
  
  ES64_AR=ES64_ARIMA_WEEK4$WIS,
  ES64_ADJ=ES64_ADJACENT_WEEK4$WIS,
  ES64_EPI=ES64_EPIWEEK_WEEK4$WIS,
  ES64_TEMP=ES64_TEMPERATURE_WEEK4$WIS,
  ES64_AVG=ES64_AVERAGE_WEEK4$WIS
)
head(df_W4_NoLg)

##      STATE Julian_date epiweek    AUTO_AR    AUTO_ADJ    AUTO_EPI    AUTO_TEMP    AUTO_AVG
## 1 Alabama 2022-11-19      46 77.27004  47.93548 195.75162  51.85643 104.40361
## 2 Alabama 2022-11-26      47 115.99444 106.26135 102.50278 116.05624 39.86127
## 3 Alabama 2022-12-03      48 192.02549 176.92592 265.49305 472.69279 126.51362
## 4 Alabama 2022-12-10      49  29.55947  46.55813 232.08536  91.59221 22.18855
## 5 Alabama 2022-12-17      50  81.71423  93.68834  53.08411  79.57274 115.23051
## 6 Alabama 2022-12-24      51  60.75983  37.79366  58.29653  55.23814 136.53276
##      ES27_AR   ES27_ADJ   ES27_EPI   ES27_TEMP   ES27_AVG   ES64_AR   ES64_ADJ
## 1    7.254855  7.16035  7.617808  7.312711  14.40510  11.59783  9.069458
## 2 386.488083 388.30036 385.697901 384.899766 369.92347 537.93279 538.339518
## 3 461.672199 443.53294 466.645841 461.529587 425.18023 639.83705 629.159196
## 4  84.869169  81.88044  87.162162  87.610216  84.65971 114.14164 115.632787
## 5 21.531612 113.58265 21.857114 21.762429  75.31410  29.71893  99.180700
## 6  83.471365  98.93852  94.066825  78.769468 226.31181  83.21651 181.867637
##      ES64_EPI   ES64_TEMP   ES64_AVG
## 1    8.877403  11.48946  13.89780
## 2  543.121394 539.66233 557.08945
## 3  643.495810 640.74775 642.76341
## 4 110.388493 108.19323 105.29785
## 5  26.189449  27.70742  70.39027
## 6  95.721917  81.27869 202.11325

```

Filter only the flu season

```
# Filter the dataframe for epiweek >= 40 or epiweek <= 20
filtered_df_W1_NoLg <- df_W1_NoLg %>%
  filter(epiweek >= 40 | epiweek <= 20)
```

```
# Display the filtered dataset
head(filtered_df_W1_NoLg)
```

```
##      STATE Julian_date epiweek    AUTO_AR    AUTO_ADJ    AUTO_EPI    AUTO_TEMP
## 1 Alabama 2022-10-29       43 102.94840 101.65743 116.57567 102.44553
## 2 Alabama 2022-11-05       44  79.69032  69.21610  74.91468  79.44376
## 3 Alabama 2022-11-12       45  62.70926  49.99889  35.82769 155.63500
## 4 Alabama 2022-11-19       46  48.13570  37.31865  16.86871  58.72064
## 5 Alabama 2022-11-26       47  62.47143  61.26857  71.46483  61.73031
## 6 Alabama 2022-12-03       48  92.04712  93.99361  98.09041 103.34642
##      AUTO_AVG   ES27_AR   ES27_ADJ   ES27_EPI   ES27_TEMP   ES27_AVG   ES64_AR
## 1 111.250523  90.719035  90.791737  92.673037  90.729616  90.986378  88.414739
## 2 47.411877  9.385328   7.062736   9.397495   9.483206   7.024391   3.165721
## 3 78.333058 153.181002 152.552002 153.031292 153.171927 157.087082 170.008450
## 4 37.777085  60.914034  72.241112  61.554400  62.060894  65.839393  68.968221
## 5 66.087165  25.397894  41.902088  24.594584  24.277733  24.098040  23.129469
## 6 6.332483   83.516853  66.325397  77.939015  88.073395  9.997281  81.674040
##      ES64_ADJ   ES64_EPI   ES64_TEMP   ES64_AVG
## 1 89.834739  93.67342  88.803267  90.337167
## 2 3.068386   3.17232  3.132254  3.094771
## 3 166.985799 169.74417 170.208228 176.077362
## 4 78.645976  67.15577  67.642403  70.855246
## 5 44.816094  23.61011  21.939414  23.700555
## 6 43.209734  74.32642  84.495765  13.384597
```

```
# Filter the dataframe for epiweek >= 40 or epiweek <= 20
filtered_df_W2_NoLg <- df_W2_NoLg %>%
  filter(epiweek >= 40 | epiweek <= 20)
```

```
# Display the filtered dataset
head(filtered_df_W2_NoLg)
```

```
##      STATE Julian_date epiweek    AUTO_AR    AUTO_ADJ    AUTO_EPI    AUTO_TEMP
## 1 Alabama 2022-11-05       44 188.905312 183.765343 228.179513 185.137841
## 2 Alabama 2022-11-12       45  48.034143  39.218917  42.748513  48.165309
## 3 Alabama 2022-11-19       46  81.098716  69.656879  28.840078 357.027657
## 4 Alabama 2022-11-26       47  34.593933  6.491932  63.573618  67.475787
## 5 Alabama 2022-12-03       48 237.947010 270.813086 251.280702 236.723180
## 6 Alabama 2022-12-10       49   7.307581  7.344853  9.448372  7.854465
##      AUTO_AVG   ES27_AR   ES27_ADJ   ES27_EPI   ES27_TEMP   ES27_AVG   ES64_AR
## 1 213.799620 156.55311 157.25555 157.82259 156.42450 155.52180 150.37049
## 2 11.773647 114.29758 117.34220 114.00493 113.59390 111.78869 155.92213
## 3 100.827045 351.92105 345.19642 353.19759 351.90586 340.90754 401.78478
## 4  8.185654  65.67085  71.09070  66.92643  67.50840  69.30220  82.35596
## 5 254.196723 144.76162 167.68187 143.08761 143.31461 143.52997 133.83584
## 6  85.378319 18.31165  28.74939  25.30183  16.05396  97.09043  15.94977
##      ES64_ADJ   ES64_EPI   ES64_TEMP   ES64_AVG
```

```

## 1 154.16654 159.51410 150.98968 154.60212
## 2 151.59357 157.08640 156.57325 158.11796
## 3 394.80115 402.05978 402.16290 403.14849
## 4 88.55030 79.36624 78.91779 81.12383
## 5 175.05363 135.69637 132.76973 137.25642
## 6 64.03856 24.95665 15.18490 83.27090

# Filter the dataframe for epiweek >= 40 or epiweek <= 20
filtered_df_W3_NoLg <- df_W3_NoLg %>%
  filter(epiweek >= 40 | epiweek <= 20)

# Display the filtered dataset
head(filtered_df_W3_NoLg)

##      STATE Julian_date epiweek AUTO_AR AUTO_ADJ AUTO_EPI AUTO_TEMP
## 1 Alabama 2022-11-12      45 146.78553 132.24099 226.16803 134.66404
## 2 Alabama 2022-11-19      46  48.22303  39.47050  41.58940  48.75787
## 3 Alabama 2022-11-26      47 10.19427 10.10276 41.42233 478.20811
## 4 Alabama 2022-12-03      48 43.25931 137.56318 328.57147 10.67010
## 5 Alabama 2022-12-10      49 146.32655 199.26107 146.15502 144.52135
## 6 Alabama 2022-12-17      50 59.17719 47.79918 62.09687 58.76758
##      AUTO_AVG ES27_AR ES27_ADJ ES27_EPI ES27_TEMP ES27_AVG ES64_AR
## 1 177.978661  83.23148  84.55082  82.15597  82.94497  72.72026 70.96738
## 2  7.153594 289.38002 291.27312 288.86154 288.37530 279.21031 376.54043
## 3 28.768875 470.10331 457.53116 473.27827 470.02991 445.94359 575.24763
## 4 113.515485 12.58747 11.86176 12.64650 12.58217 12.27429 14.35102
## 5 174.117903 30.72114 10.88272 29.74189 30.00309 10.73607 23.73183
## 6 147.677074 83.17768 99.08706 93.65631 78.85432 200.26034 80.44222
##      ES64_ADJ ES64_EPI ES64_TEMP ES64_AVG
## 1    77.42024  77.55526  71.70535  73.02268
## 2 372.94951 379.43593 377.65242 383.92138
## 3 565.90367 578.05058 575.84232 575.00321
## 4 16.34616 14.23757 13.84466 12.75210
## 5 17.03998 26.52609 24.55276 11.48884
## 6 162.09347 92.96110 78.79480 178.76213

# Filter the dataframe for epiweek >= 40 or epiweek <= 20
filtered_df_W4_NoLg <- df_W4_NoLg %>%
  filter(epiweek >= 40 | epiweek <= 20)

# Display the filtered dataset
head(filtered_df_W4_NoLg)

##      STATE Julian_date epiweek AUTO_AR AUTO_ADJ AUTO_EPI AUTO_TEMP AUTO_AVG
## 1 Alabama 2022-11-19      46 77.27004 47.93548 195.75162 51.85643 104.40361
## 2 Alabama 2022-11-26      47 115.99444 106.26135 102.50278 116.05624 39.86127
## 3 Alabama 2022-12-03      48 192.02549 176.92592 265.49305 472.69279 126.51362
## 4 Alabama 2022-12-10      49 29.55947 46.55813 232.08536 91.59221 22.18855
## 5 Alabama 2022-12-17      50 81.71423 93.68834 53.08411 79.57274 115.23051
## 6 Alabama 2022-12-24      51 60.75983 37.79366 58.29653 55.23814 136.53276
##      ES27_AR ES27_ADJ ES27_EPI ES27_TEMP ES27_AVG ES64_AR ES64_ADJ
## 1    7.254855  7.16035  7.617808  7.312711 14.40510 11.59783  9.069458
## 2 386.488083 388.30036 385.697901 384.899766 369.92347 537.93279 538.339518

```

```

## 3 461.672199 443.53294 466.645841 461.529587 425.18023 639.83705 629.159196
## 4 84.869169 81.88044 87.162162 87.610216 84.65971 114.14164 115.632787
## 5 21.531612 113.58265 21.857114 21.762429 75.31410 29.71893 99.180700
## 6 83.471365 98.93852 94.066825 78.769468 226.31181 83.21651 181.867637
##   ES64_EPI ES64_TEMP ES64_AVG
## 1 8.877403 11.48946 13.89780
## 2 543.121394 539.66233 557.08945
## 3 643.495810 640.74775 642.76341
## 4 110.388493 108.19323 105.29785
## 5 26.189449 27.70742 70.39027
## 6 95.721917 81.27869 202.11325

```

Computing total weekly hospitalization to include in the plot bellow.

```

hosp_by_date <- AUTO_ARIMA_WEEK1 %>%
  group_by(target_end_date) %>%
  summarise(total_hosp = sum(cases, na.rm = TRUE)) %>%
  rename(Julian_date = target_end_date)

```

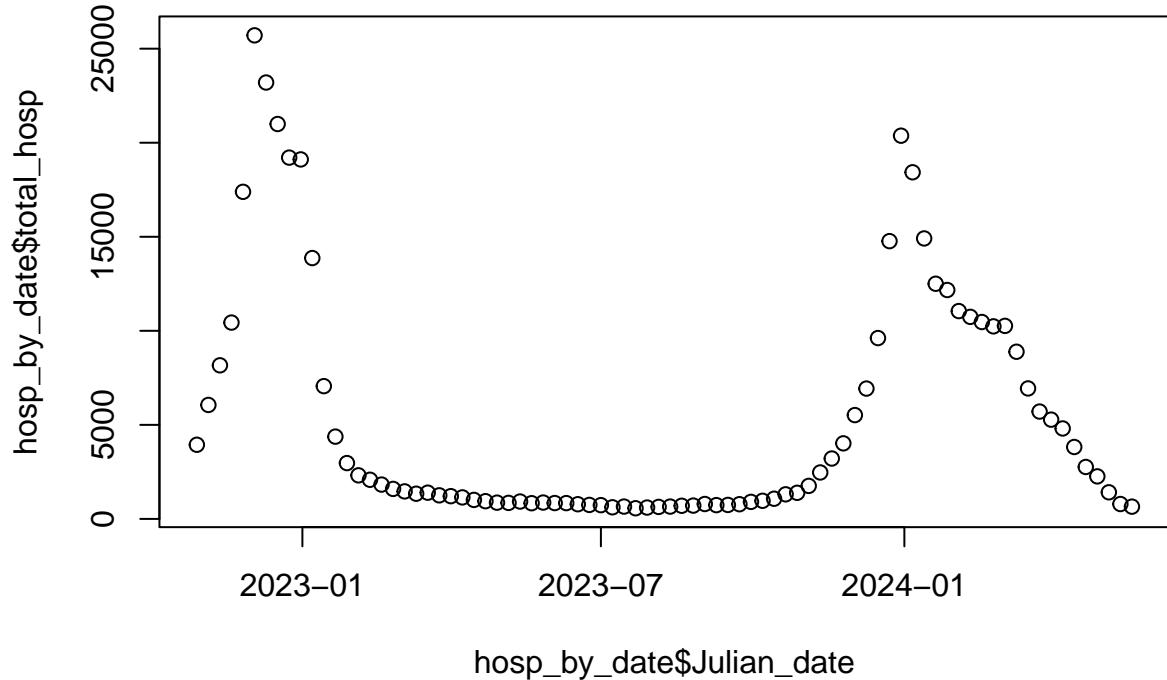
```
hosp_by_date
```

```

## # A tibble: 82 x 2
##   Julian_date total_hosp
##   <date>        <int>
## 1 2022-10-29     3942
## 2 2022-11-05     6059
## 3 2022-11-12     8166
## 4 2022-11-19    10439
## 5 2022-11-26    17390
## 6 2022-12-03    25709
## 7 2022-12-10    23201
## 8 2022-12-17    20995
## 9 2022-12-24    19209
## 10 2022-12-31   19112
## # i 72 more rows

```

```
plot(hosp_by_date$Julian_date, hosp_by_date$total_hosp)
```



Weighted interval scores for each state and each target week.

```
# Combine all datasets and add a column to identify them
df_combined_ <- bind_rows(
  df_W1_NoLg %>% mutate(Dataset = "1 week ahead"),
  df_W2_NoLg %>% mutate(Dataset = "2 weeks ahead"),
  df_W3_NoLg %>% mutate(Dataset = "3 weeks ahead"),
  df_W4_NoLg %>% mutate(Dataset = "4 weeks ahead")
) %>%
  select(STATE, epiweek, AUTO_AR, Dataset, Julian_date) %>%
  rename(WIS = AUTO_AR) # Rename for consistency

#####
df_combined_ <- df_combined_ %>%
  left_join(hosp_by_date, by = "Julian_date")

#####
# Define the epiweek values to highlight
highlight_weeks <- c(21, 41)

# Get corresponding Julian dates for these epiweeks
highlight_dates <- df_combined_ %>%
  filter(epiweek %in% highlight_weeks) %>%
```

```

select(epiweek, Julian_date) %>%
distinct()

sec_breaks<-seq(0, max(df_combined_$total_hosp), by=10000)

plt0 <- ggplot(df_combined_, aes(x = Julian_date, y = WIS, group = STATE)) +
  geom_vline(
    data = highlight_dates,
    aes(xintercept = Julian_date),
    color = "red",
    linetype = "dashed",
    size = 1
  ) +
  geom_line(color = "black", size = 0.15) +
  facet_wrap(~ Dataset, ncol = 2) +
  theme_minimal() +
  labs(
    x = "",
    y = "WIS"
  ) +
  geom_line(
    aes(y = total_hosp / 4),
    color = "blue",
    size = 0.2,
    linetype = "dashed"
  ) +
  theme(
    axis.text.x = element_text(angle = 45, hjust = 1),
    legend.position = "none",
    axis.title.y.right = element_text(color = "blue"),
    axis.text.y.right = element_text(color = "blue"),
    axis.ticks.y.right = element_line(color = "blue"),
    panel.grid.major.x = element_blank(),
    panel.grid.minor =element_blank()
  ) +
  scale_x_date(
    date_breaks = "1 month",
    date_labels = "%b %y"
  ) +
  scale_y_continuous(
    sec.axis = sec_axis(
      ~ . * 4,
      name = "Influenza-related hospitalizations"
    )
  )
)

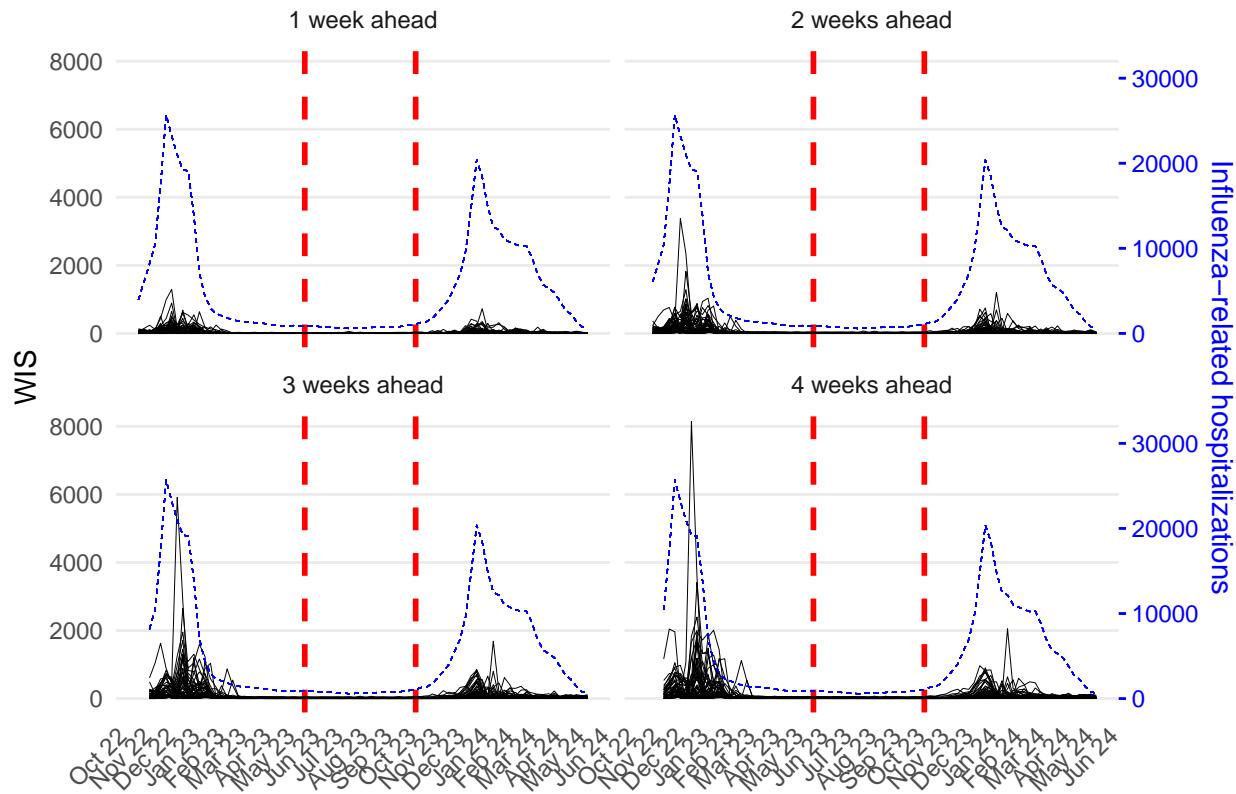
```

```

## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

```

```
# Print the plot
plt0
```



```
ggsave(plt0, height = 5, width = 8, filename="Fig2.jpg")
```

Computing mean weighted interval score for all target week for each model.

```
# Define the function
calculate_mean_wis <- function(data) {
  data %>%
    group_by(STATE) %>%
    summarize(
      AUTO_AR = mean(AUTO_AR, na.rm = TRUE),
      AUTO_ADJ = mean(AUTO_ADJ, na.rm = TRUE),
      AUTO_EPI = mean(AUTO_EPI, na.rm = TRUE),
      AUTO_TMP = mean(AUTO_TEMP, na.rm = TRUE),
      AUTO_AVG = mean(AUTO_AVG, na.rm = TRUE),

      ES27_AR = mean(ES27_AR, na.rm = TRUE),
      ES27_ADJ = mean(ES27_ADJ, na.rm = TRUE),
      ES27_EPI = mean(ES27_EPI, na.rm = TRUE),
      ES27_TMP = mean(ES27_TEMP, na.rm = TRUE),
      ES27_AVG = mean(ES27_AVG, na.rm = TRUE),

      ES64_AR = mean(ES64_AR, na.rm = TRUE),
```

```

    ES64_ADJ = mean(ES64_ADJ, na.rm = TRUE),
    ES64_EPI = mean(ES64_EPI, na.rm = TRUE),
    ES64_TMP = mean(ES64_TEMP, na.rm = TRUE),
    ES64_AVG = mean(ES64_AVG, na.rm = TRUE)
  )
}

# Now you can use the function with any dataframe
W1_NoLg <- calculate_mean_wis(filtered_df_W1_NoLg)
W2_NoLg <- calculate_mean_wis(filtered_df_W2_NoLg)
W3_NoLg <- calculate_mean_wis(filtered_df_W3_NoLg)
W4_NoLg <- calculate_mean_wis(filtered_df_W4_NoLg)

# Display the resulting dataframe
head(W1_NoLg)

```

```

## # A tibble: 6 x 16
##   STATE    AUTO_AR AUTO_ADJ AUTO_EPI AUTO_TMP AUTO_AVG ES27_AR ES27_ADJ ES27_EPI
##   <chr>     <dbl>   <dbl>   <dbl>   <dbl>   <dbl>     <dbl>   <dbl>   <dbl>
## 1 Alabama    23.8    19.4    24.2    25.4    19.5    23.7    21.7    23.8
## 2 Arizona    44.6    42.6    46.7    43.5    47.4    46.0    40.6    46.5
## 3 Arkansas   21.5    21.8    21.2    23.0    21.8    21.2    24.3    21.2
## 4 Califor~   117.    93.5   108.    112.    139.    112.    114.    114.
## 5 Colorado   18.1    17.3    17.3    18.6    17.9    17.3    17.7    16.5
## 6 Connect~   24.1    26.6    23.3    24.6    21.5    25.4    24.5    25.7
## # i 7 more variables: ES27_TMP <dbl>, ES27_AVG <dbl>, ES64_AR <dbl>,
## #   ES64_ADJ <dbl>, ES64_EPI <dbl>, ES64_TMP <dbl>, ES64_AVG <dbl>

```

```
head(W2_NoLg)
```

```

## # A tibble: 6 x 16
##   STATE    AUTO_AR AUTO_ADJ AUTO_EPI AUTO_TMP AUTO_AVG ES27_AR ES27_ADJ ES27_EPI
##   <chr>     <dbl>   <dbl>   <dbl>   <dbl>   <dbl>     <dbl>   <dbl>   <dbl>
## 1 Alabama    36.0    30.0    36.4    41.8    32.7    42.1    38.8    42.3
## 2 Arizona    101.    105.    105.    98.3    103.    107.    98.9    109.
## 3 Arkansas   37.6    37.7    36.6    39.0    36.5    37.0    42.0    37.2
## 4 Califor~   250.    168.    217.    245.    263.    240.    243.    242.
## 5 Colorado   30.6    27.5    29.9    31.0    27.8    28.7    29.2    27.4
## 6 Connect~   38.9    39.7    36.3    39.9    35.2    43.9    38.3    43.7
## # i 7 more variables: ES27_TMP <dbl>, ES27_AVG <dbl>, ES64_AR <dbl>,
## #   ES64_ADJ <dbl>, ES64_EPI <dbl>, ES64_TMP <dbl>, ES64_AVG <dbl>

```

```
head(W3_NoLg)
```

```

## # A tibble: 6 x 16
##   STATE    AUTO_AR AUTO_ADJ AUTO_EPI AUTO_TMP AUTO_AVG ES27_AR ES27_ADJ ES27_EPI
##   <chr>     <dbl>   <dbl>   <dbl>   <dbl>   <dbl>     <dbl>   <dbl>   <dbl>
## 1 Alabama    38.8    35.6    44.2    46.1    39.3    53.1    47.8    53.4
## 2 Arizona    139.    152.    140.    135.    151.    151.    146.    155.
## 3 Arkansas   52.5    51.2    47.6    53.6    48.4    51.1    57.0    51.3
## 4 Califor~   379.    256.    285.    376.    326.    388.    389.    391.

```

```

## 5 Colorado    44.4    39.0    43.6    45.2    38.2    43.4    42.6    42.3
## 6 Connect~   56.2    53.8    51.8    57.1    52.9    64.6    52.6    64.6
## # i 7 more variables: ES27_TMP <dbl>, ES27_AVG <dbl>, ES64_AR <dbl>,
## #   ES64_ADJ <dbl>, ES64_EPI <dbl>, ES64_TMP <dbl>, ES64_AVG <dbl>

```

```
head(W4_NoLg)
```

```

## # A tibble: 6 x 16
##   STATE    AUTO_AR  AUTO_ADJ  AUTO_EPI  AUTO_TMP  AUTO_AVG  ES27_AR  ES27_ADJ  ES27_EPI
##   <chr>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>
## 1 Alabama    45.5     39.9     49.3     50.0     42.5     61.8     57.3     62.0
## 2 Arizona   163.     180.     162.     162.     177.     188.     176.     193.
## 3 Arkansas   67.1     60.1     57.5     68.3     57.1     68.3     70.4     68.3
## 4 Califor~  485.     306.     343.     498.     354.     537.     529.     538.
## 5 Colorado   55.3     47.0     56.1     56.1     45.1     56.6     53.7     55.8
## 6 Connect~   71.1     64.5     62.8     69.0     62.4     83.5     62.2     83.3
## # i 7 more variables: ES27_TMP <dbl>, ES27_AVG <dbl>, ES64_AR <dbl>,
## #   ES64_ADJ <dbl>, ES64_EPI <dbl>, ES64_TMP <dbl>, ES64_AVG <dbl>

```

Boxplots of the mean(WIS) by each state.

```

# Combine the data into one data frame
combined_data <- data.frame(
  week = rep(c("W1", "W2", "W3", "W4"), each = 48 * 15),
  Models = rep(c("AUTO_AR", "ES27_AR", "ES64_AR", "AUTO_ADJ", "ES27_ADJ", "ES64_ADJ", "AUTO_TMP", "ES27_TMP",
  value = c(W1_NoLg$AUTO_AR, W1_NoLg$ES27_AR, W1_NoLg$ES64_AR,
  W1_NoLg$AUTO_ADJ, W1_NoLg$ES27_ADJ, W1_NoLg$ES64_ADJ,
  W1_NoLg$AUTO_TMP, W1_NoLg$ES27_TMP, W1_NoLg$ES64_TMP,
  W1_NoLg$AUTO_EPI, W1_NoLg$ES27_EPI, W1_NoLg$ES64_EPI,
  W1_NoLg$AUTO_AVG, W1_NoLg$ES27_AVG, W1_NoLg$ES64_AVG,
  W2_NoLg$AUTO_AR, W2_NoLg$ES27_AR, W2_NoLg$ES64_AR,
  W2_NoLg$AUTO_ADJ, W2_NoLg$ES27_ADJ, W2_NoLg$ES64_ADJ,
  W2_NoLg$AUTO_TMP, W2_NoLg$ES27_TMP, W2_NoLg$ES64_TMP,
  W2_NoLg$AUTO_EPI, W2_NoLg$ES27_EPI, W2_NoLg$ES64_EPI,
  W2_NoLg$AUTO_AVG, W2_NoLg$ES27_AVG, W2_NoLg$ES64_AVG,
  W3_NoLg$AUTO_AR, W3_NoLg$ES27_AR, W3_NoLg$ES64_AR,
  W3_NoLg$AUTO_ADJ, W3_NoLg$ES27_ADJ, W3_NoLg$ES64_ADJ,
  W3_NoLg$AUTO_TMP, W3_NoLg$ES27_TMP, W3_NoLg$ES64_TMP,
  W3_NoLg$AUTO_EPI, W3_NoLg$ES27_EPI, W3_NoLg$ES64_EPI,
  W3_NoLg$AUTO_AVG, W3_NoLg$ES27_AVG, W3_NoLg$ES64_AVG,
  W4_NoLg$AUTO_AR, W4_NoLg$ES27_AR, W4_NoLg$ES64_AR,
  W4_NoLg$AUTO_ADJ, W4_NoLg$ES27_ADJ, W4_NoLg$ES64_ADJ,
  W4_NoLg$AUTO_TMP, W4_NoLg$ES27_TMP, W4_NoLg$ES64_TMP,
  W4_NoLg$AUTO_EPI, W4_NoLg$ES27_EPI, W4_NoLg$ES64_EPI,
  W4_NoLg$AUTO_AVG, W4_NoLg$ES27_AVG, W4_NoLg$ES64_AVG
)

```

```

# Create a new column for categories
combined_data <- combined_data %>%
  mutate(category = case_when(
    grepl("AR", Models) ~ "AR",
    grepl("EPI", Models) ~ "EPI",
    grepl("TMP", Models) ~ "TMP",
    grepl("ADJ", Models) ~ "ADJ",
    grepl("AVG", Models) ~ "AVG",
  ))

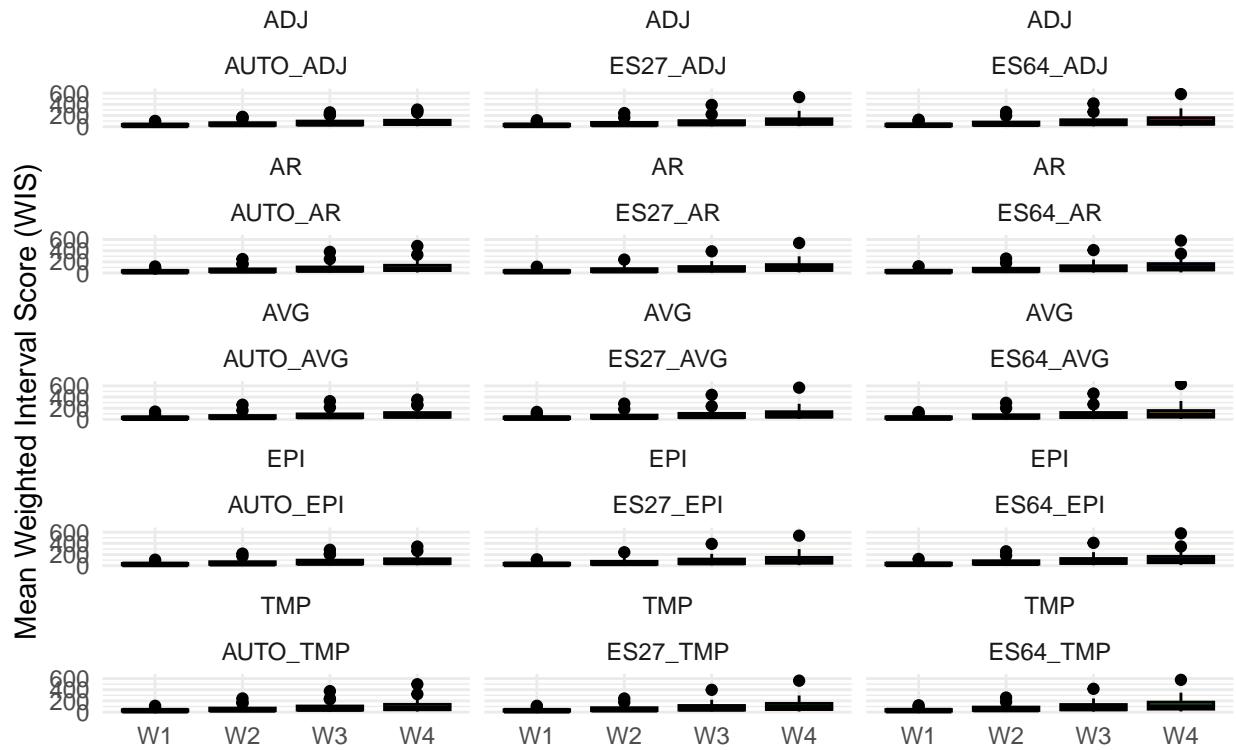
# Define color ramps for each category
colors_ar <- c("#a6cee3", "#226e83", "#08306b")
colors_adj <- c("#fb9a99", "red3", "#a11c3e")
colors_epi <- c("#cab2d6", "purple2", "#5e2b7b")
colors_tmp <- c("lightgreen", "green3", "#319045")
colors_avg <- c("orange", "orange2", "orange3")

# Create a custom color mapping for each variable
custom_colors <- c(
  "AUTO_AR" = colors_ar[1], "ES27_AR" = colors_ar[2], "ES64_AR" = colors_ar[3],
  "AUTO_EPI" = colors_epi[1], "ES27_EPI" = colors_epi[2], "ES64_EPI" = colors_epi[3],
  "AUTO_TMP" = colors_tmp[1], "ES27_TMP" = colors_tmp[2], "ES64_TMP" = colors_tmp[3],
  "AUTO_ADJ" = colors_adj[1], "ES27_ADJ" = colors_adj[2], "ES64_ADJ" = colors_adj[3],
  "AUTO_AVG" = colors_avg[1], "ES27_AVG" = colors_avg[2], "ES64_AVG" = colors_avg[3]
)

# Create the box plot with facet for categories
plt1<- ggplot(combined_data, aes(x = week, y = value, fill = Models)) +
  geom_boxplot(color = "black") +
  scale_fill_manual(values = custom_colors) +
  facet_wrap(category ~ Models , nrow = 5) +
  labs(
    title = "A) Models without log-back transformations",
    x = "",
    y = "Mean Weighted Interval Score (WIS)"
  ) +
  scale_y_continuous(limits = c(0, 650)) +
  theme_minimal()+
  theme(legend.position = "none")
plt1

```

### A) Models without log-back transformations



```
#ggsave(plt1, height = 12, width = 8, filename="fig1A.jpg")
```

Loading model with log-back transformations.

```
load("models_with_logback/ES_ARIMA/ARIMA_MODELS_influenza_hospitalization.Rdata")
load("models_with_logback/ES_ADJACENT/ADJACENT_MODELS_influenza_hospitalization.Rdata")
load("models_with_logback/ES_EPIWEEK/EPIWEEK_MODELS_influenza_hospitalization.Rdata")
load("models_with_logback/ES_TEMPERATURE/TEMPERATURE_MODELS_influenza_hospitalization.Rdata")
load("models_with_logback/ES_AVERAGE/AVERAGE_MODELS_influenza_hospitalization.Rdata")
```

1 Week ahead

```
# Creating new columns with Epidemiological weeks based on target_end_week
AUTO_ARIMA_WEEK1$epiweek <- MMWRweek(AUTO_ARIMA_WEEK1$target_end_date)$MMWRweek
AUTO_ADJACENT_WEEK1$epiweek <- MMWRweek(AUTO_ADJACENT_WEEK1$target_end_date)$MMWRweek
AUTO_EPIWEEK_WEEK1$epiweek <- MMWRweek(AUTO_EPIWEEK_WEEK1$target_end_date)$MMWRweek
AUTO_TEMPERATURE_WEEK1$epiweek <- MMWRweek(AUTO_TEMPERATURE_WEEK1$target_end_date)$MMWRweek
AUTO_AVERAGE_WEEK1$epiweek <- MMWRweek(AUTO_AVERAGE_WEEK1$target_end_date)$MMWRweek

ES27_ARIMA_WEEK1$epiweek <- MMWRweek(ES27_ARIMA_WEEK1$target_end_date)$MMWRweek
ES27_ADJACENT_WEEK1$epiweek <- MMWRweek(ES27_ADJACENT_WEEK1$target_end_date)$MMWRweek
ES27_EPIWEEK_WEEK1$epiweek <- MMWRweek(ES27_EPIWEEK_WEEK1$target_end_date)$MMWRweek
ES27_TEMPERATURE_WEEK1$epiweek <- MMWRweek(ES27_TEMPERATURE_WEEK1$target_end_date)$MMWRweek
```

```

ES27_AVERAGE_WEEK1$epiweek <- MMWRweek(ES27_AVERAGE_WEEK1$target_end_date)$MMWRweek

ES64_ARIMA_WEEK1$epiweek <- MMWRweek(ES64_ARIMA_WEEK1$target_end_date)$MMWRweek
ES64_ADJACENT_WEEK1$epiweek <- MMWRweek(ES64_ADJACENT_WEEK1$target_end_date)$MMWRweek
ES64_EPIWEEK_WEEK1$epiweek <- MMWRweek(ES64_EPIWEEK_WEEK1$target_end_date)$MMWRweek
ES64_TEMPERATURE_WEEK1$epiweek <- MMWRweek(ES64_TEMPERATURE_WEEK1$target_end_date)$MMWRweek
ES64_AVERAGE_WEEK1$epiweek <- MMWRweek(ES64_AVERAGE_WEEK1$target_end_date)$MMWRweek

# Dataframe that will be analysed for 1 Week Ahead
df_W1 <- data.frame(
  STATE = AUTO_ARIMA_WEEK1$State,
  Julian_date = AUTO_ARIMA_WEEK1$target_end_date,
  epiweek = AUTO_ARIMA_WEEK1$epiweek,
  AUTO_AR_LB=AUTO_ARIMA_WEEK1$WIS,
  AUTO_ADJ_LB=AUTO_ADJACENT_WEEK1$WIS,
  AUTO_EPI_LB=AUTO_EPIWEEK_WEEK1$WIS,
  AUTO_TEMP_LB=AUTO_TEMPERATURE_WEEK1$WIS,
  AUTO_AVG_LB=AUTO_AVERAGE_WEEK1$WIS,
  ES27_AR_LB=ES27_ARIMA_WEEK1$WIS,
  ES27_ADJ_LB=ES27_ADJACENT_WEEK1$WIS,
  ES27_EPI_LB=ES27_EPIWEEK_WEEK1$WIS,
  ES27_TEMP_LB=ES27_TEMPERATURE_WEEK1$WIS,
  ES27_AVG_LB=ES27_AVERAGE_WEEK1$WIS,
  ES64_AR_LB=ES64_ARIMA_WEEK1$WIS,
  ES64_ADJ_LB=ES64_ADJACENT_WEEK1$WIS,
  ES64_EPI_LB=ES64_EPIWEEK_WEEK1$WIS,
  ES64_TEMP_LB=ES64_TEMPERATURE_WEEK1$WIS,
  ES64_AVG_LB=ES64_AVERAGE_WEEK1$WIS
)
head(df_W1)

##      STATE Julian_date epiweek AUTO_AR_LB AUTO_ADJ_LB AUTO_EPI_LB AUTO_TEMP_LB
## 1 Alabama 2022-10-29     43 164.97469   154.82262   167.16881   158.73461
## 2 Alabama 2022-11-05     44 152.25677   125.66999   147.56571   217.70136
## 3 Alabama 2022-11-12     45 42.76549    33.63300    42.10368   41.93627
## 4 Alabama 2022-11-19     46 27.64761    64.94758    27.33367   28.21504
## 5 Alabama 2022-11-26     47 30.84363    28.07577    29.48502   53.76514
## 6 Alabama 2022-12-03     48 70.42607    81.54591   130.42912   69.58822
##      AUTO_AVG_LB ES27_AR_LB ES27_ADJ_LB ES27_EPI_LB ES27_TEMP_LB ES27_AVG_LB
## 1    111.40345  138.67842  125.21088  160.28714   183.38737   91.96030
## 2     58.82784  100.18764   78.71110  105.33944   101.77054   46.03894
## 3     69.47533   34.89792   39.88211   34.55781   34.77695   87.57530
## 4    138.85755   60.46462  104.07048   59.09603   59.51544  170.95917
## 5     63.33534   34.33307   50.43779   33.09650   41.61223   95.06813
## 6     37.97748   65.48558   47.01762   63.64209   65.18264  41.62150
##      ES64_AR_LB ES64_ADJ_LB ES64_EPI_LB ES64_TEMP_LB ES64_AVG_LB
## 1    138.58970  133.18630  152.65640  180.91343   95.46299
## 2    104.66467   79.03814  109.79832  112.65048   46.02587
## 3     35.59296   40.77665   34.91187   35.43623   91.62425
## 4     65.23764  116.71887   64.70152   64.23071  187.23402

```

```

## 5 34.67596 39.50396 33.76894 43.53779 79.26431
## 6 65.56808 48.47490 60.83552 65.06090 40.40643

```

2 Weeks ahead

```

# Creating new columns with Epidemiological weeks based on target_end_week
AUTO_ARIMA_WEEK2$epiweek <- MMWRweek(AUTO_ARIMA_WEEK2$target_end_date)$MMWRweek
AUTO_ADJACENT_WEEK2$epiweek <- MMWRweek(AUTO_ADJACENT_WEEK2$target_end_date)$MMWRweek
AUTO_EPIWEEK_WEEK2$epiweek <- MMWRweek(AUTO_EPIWEEK_WEEK2$target_end_date)$MMWRweek
AUTO_TEMPERATURE_WEEK2$epiweek <- MMWRweek(AUTO_TEMPERATURE_WEEK2$target_end_date)$MMWRweek
AUTO_AVERAGE_WEEK2$epiweek <- MMWRweek(AUTO_AVERAGE_WEEK2$target_end_date)$MMWRweek

ES27_ARIMA_WEEK2$epiweek <- MMWRweek(ES27_ARIMA_WEEK2$target_end_date)$MMWRweek
ES27_ADJACENT_WEEK2$epiweek <- MMWRweek(ES27_ADJACENT_WEEK2$target_end_date)$MMWRweek
ES27_EPIWEEK_WEEK2$epiweek <- MMWRweek(ES27_EPIWEEK_WEEK2$target_end_date)$MMWRweek
ES27_TEMPERATURE_WEEK2$epiweek <- MMWRweek(ES27_TEMPERATURE_WEEK2$target_end_date)$MMWRweek
ES27_AVERAGE_WEEK2$epiweek <- MMWRweek(ES27_AVERAGE_WEEK2$target_end_date)$MMWRweek

ES64_ARIMA_WEEK2$epiweek <- MMWRweek(ES64_ARIMA_WEEK2$target_end_date)$MMWRweek
ES64_ADJACENT_WEEK2$epiweek <- MMWRweek(ES64_ADJACENT_WEEK2$target_end_date)$MMWRweek
ES64_EPIWEEK_WEEK2$epiweek <- MMWRweek(ES64_EPIWEEK_WEEK2$target_end_date)$MMWRweek
ES64_TEMPERATURE_WEEK2$epiweek <- MMWRweek(ES64_TEMPERATURE_WEEK2$target_end_date)$MMWRweek
ES64_AVERAGE_WEEK2$epiweek <- MMWRweek(ES64_AVERAGE_WEEK2$target_end_date)$MMWRweek

# Dataframe that will be analysed for 2 Week Ahead
df_W2 <- data.frame(
  STATE = AUTO_ARIMA_WEEK2$State,
  Julian_date = AUTO_ARIMA_WEEK2$target_end_date,
  epiweek = AUTO_ARIMA_WEEK2$epiweek,
  AUTO_AR_LB=AUTO_ARIMA_WEEK2$WIS,
  AUTO_ADJ_LB=AUTO_ADJACENT_WEEK2$WIS,
  AUTO_EPI_LB=AUTO_EPIWEEK_WEEK2$WIS,
  AUTO_TEMP_LB=AUTO_TEMPERATURE_WEEK2$WIS,
  AUTO_AVG_LB=AUTO_AVERAGE_WEEK2$WIS,

  ES27_AR_LB=ES27_ARIMA_WEEK2$WIS,
  ES27_ADJ_LB=ES27_ADJACENT_WEEK2$WIS,
  ES27_EPI_LB=ES27_EPIWEEK_WEEK2$WIS,
  ES27_TEMP_LB=ES27_TEMPERATURE_WEEK2$WIS,
  ES27_AVG_LB=ES27_AVERAGE_WEEK2$WIS,

  ES64_AR_LB=ES64_ARIMA_WEEK2$WIS,
  ES64_ADJ_LB=ES64_ADJACENT_WEEK2$WIS,
  ES64_EPI_LB=ES64_EPIWEEK_WEEK2$WIS,
  ES64_TEMP_LB=ES64_TEMPERATURE_WEEK2$WIS,
  ES64_AVG_LB=ES64_AVERAGE_WEEK2$WIS
)

head(df_W2)

```

```

##      STATE Julian_date epiweek AUTO_AR_LB AUTO_ADJ_LB AUTO_EPI_LB AUTO_TEMP_LB
## 1 Alabama 2022-11-05        44   277.52451   260.09905   285.18405   272.93892

```

```

## 2 Alabama 2022-11-12      45 123.76206   98.81682   119.43244   225.73550
## 3 Alabama 2022-11-19      46 27.65801    30.68134   27.41752    28.12358
## 4 Alabama 2022-11-26      47 29.02424   52.51845   28.08553   29.14046
## 5 Alabama 2022-12-03      48 168.36465  90.69261   158.96411   46.58232
## 6 Alabama 2022-12-10      49 28.16100   32.72021   80.13328   28.28261
##   AUTO_AVG_LB ES27_AR_LB ES27_ADJ_LB ES27_EPI_LB ES27_TEMP_LB ES27_AVG_LB
## 1 198.89474 231.81375  211.64222  267.41322  304.55539  162.85351
## 2 36.72868 63.48153   50.33202  67.64182   65.26766  35.19653
## 3 108.01831 59.66592   75.06981  56.00013   58.12578  149.80369
## 4 98.55272 65.08632   103.25240  62.44626   64.43724  153.72638
## 5 44.42643 56.73039   46.35015  60.59899   51.23473  42.80849
## 6 32.65251 29.66896   35.66245  30.87440  29.65345  70.94583
##   ES64_AR_LB ES64_ADJ_LB ES64_EPI_LB ES64_TEMP_LB ES64_AVG_LB
## 1 238.53237 222.99366  258.96427  299.73039  166.07500
## 2 69.05467 51.88106   72.42017  78.11781   36.04737
## 3 62.96120 76.63837   57.82458  60.28029  159.58156
## 4 56.75389 85.65467   56.00229  56.38154  135.42546
## 5 56.78263 47.00848   59.27148  50.22071  41.59863
## 6 29.36865 34.02798   30.95575  29.43398  72.17442

```

3 Weeks ahead

```

# Creating new columns with Epidemiological weeks based on target_end_week
AUTO_ARIMA_WEEK3$epiweek <- MMWRweek(AUTO_ARIMA_WEEK3$target_end_date)$MMWRweek
AUTO_ADJACENT_WEEK3$epiweek <- MMWRweek(AUTO_ADJACENT_WEEK3$target_end_date)$MMWRweek
AUTO_EPIWEEK_WEEK3$epiweek <- MMWRweek(AUTO_EPIWEEK_WEEK3$target_end_date)$MMWRweek
AUTO_TEMPERATURE_WEEK3$epiweek <- MMWRweek(AUTO_TEMPERATURE_WEEK3$target_end_date)$MMWRweek
AUTO_AVERAGE_WEEK3$epiweek <- MMWRweek(AUTO_AVERAGE_WEEK3$target_end_date)$MMWRweek

ES27_ARIMA_WEEK3$epiweek <- MMWRweek(ES27_ARIMA_WEEK3$target_end_date)$MMWRweek
ES27_ADJACENT_WEEK3$epiweek <- MMWRweek(ES27_ADJACENT_WEEK3$target_end_date)$MMWRweek
ES27_EPIWEEK_WEEK3$epiweek <- MMWRweek(ES27_EPIWEEK_WEEK3$target_end_date)$MMWRweek
ES27_TEMPERATURE_WEEK3$epiweek <- MMWRweek(ES27_TEMPERATURE_WEEK3$target_end_date)$MMWRweek
ES27_AVERAGE_WEEK3$epiweek <- MMWRweek(ES27_AVERAGE_WEEK3$target_end_date)$MMWRweek

ES64_ARIMA_WEEK3$epiweek <- MMWRweek(ES64_ARIMA_WEEK3$target_end_date)$MMWRweek
ES64_ADJACENT_WEEK3$epiweek <- MMWRweek(ES64_ADJACENT_WEEK3$target_end_date)$MMWRweek
ES64_EPIWEEK_WEEK3$epiweek <- MMWRweek(ES64_EPIWEEK_WEEK3$target_end_date)$MMWRweek
ES64_TEMPERATURE_WEEK3$epiweek <- MMWRweek(ES64_TEMPERATURE_WEEK3$target_end_date)$MMWRweek
ES64_AVERAGE_WEEK3$epiweek <- MMWRweek(ES64_AVERAGE_WEEK3$target_end_date)$MMWRweek

# Dataframe that will be analysed for 2 Week Ahead
df_W3 <- data.frame(
  STATE = AUTO_ARIMA_WEEK3$State,
  Julian_date = AUTO_ARIMA_WEEK3$target_end_date,
  epiweek = AUTO_ARIMA_WEEK3$epiweek,
  AUTO_AR_LB=AUTO_ARIMA_WEEK3$WIS,
  AUTO_ADJ_LB=AUTO_ADJACENT_WEEK3$WIS,
  AUTO_EPI_LB=AUTO_EPIWEEK_WEEK3$WIS,
  AUTO_TEMP_LB=AUTO_TEMPERATURE_WEEK3$WIS,
  AUTO_AVG_LB=AUTO_AVERAGE_WEEK3$WIS,
  ES27_AR_LB=ES27_ARIMA_WEEK3$WIS,

```

```

ES27_ADJ_LB=ES27_ADJACENT_WEEK3$WIS,
ES27_EPI_LB=ES27_EPIWEEK_WEEK3$WIS,
ES27_TEMP_LB=ES27_TEMPERATURE_WEEK3$WIS,
ES27_AVG_LB=ES27_AVERAGE_WEEK3$WIS,

ES64_AR_LB=ES64_ARIMA_WEEK3$WIS,
ES64_ADJ_LB=ES64_ADJACENT_WEEK3$WIS,
ES64_EPI_LB=ES64_EPIWEEK_WEEK3$WIS,
ES64_TEMP_LB=ES64_TEMPERATURE_WEEK3$WIS,
ES64_AVG_LB=ES64_AVERAGE_WEEK3$WIS
)

head(df_W3)

```

	STATE	Julian_date	epiweek	AUTO_AR_LB	AUTO_ADJ_LB	AUTO_EPI_LB	AUTO_TEMP_LB
## 1	Alabama	2022-11-12	45	255.67207	241.03987	259.01418	251.72006
## 2	Alabama	2022-11-19	46	96.15237	72.82592	93.74152	183.44917
## 3	Alabama	2022-11-26	47	39.85273	31.57650	40.18449	39.08883
## 4	Alabama	2022-12-03	48	99.81193	48.50555	137.20756	98.21383
## 5	Alabama	2022-12-10	49	110.10063	48.22772	102.41015	44.48408
## 6	Alabama	2022-12-17	50	46.25364	20.84313	52.68966	43.73901
	AUTO_AVG_LB	ES27_AR_LB	ES27_ADJ_LB	ES27_EPI_LB	ES27_TEMP_LB	ES27_AVG_LB	
## 1	171.21369	196.06526	172.30067	234.10599	281.50630	123.97151	
## 2	28.87514	38.74124	35.84129	39.54435	39.32222	47.37365	
## 3	88.49128	65.89910	79.19608	64.75946	64.52593	147.40097	
## 4	46.81402	54.26322	64.53571	55.82159	54.26457	83.05521	
## 5	34.92782	39.63831	48.66583	39.05724	44.12713	74.54001	
## 6	56.01685	47.13537	63.33375	49.56314	46.42059	119.68466	
	ES64_AR_LB	ES64_ADJ_LB	ES64_EPI_LB	ES64_TEMP_LB	ES64_AVG_LB		
## 1	201.04627	187.07056	217.42038	276.50884	125.77841		
## 2	42.95249	37.63603	42.86805	48.73175	50.21087		
## 3	69.16687	80.13844	67.19970	66.66957	147.74137		
## 4	54.91965	64.83746	55.47049	55.00449	84.74953		
## 5	39.02389	48.27718	38.64341	44.30995	71.72136		
## 6	45.95904	58.36465	48.61270	44.94409	109.02039		

4 Weeks ahead

```

# Creating new columns with Epidemiological weeks based on target_end_week
AUTO_ARIMA_WEEK4$epiweek <- MMWRweek(AUTO_ARIMA_WEEK4$target_end_date)$MMWRweek
AUTO_ADJACENT_WEEK4$epiweek <- MMWRweek(AUTO_ADJACENT_WEEK4$target_end_date)$MMWRweek
AUTO_EPIWEEK_WEEK4$epiweek <- MMWRweek(AUTO_EPIWEEK_WEEK4$target_end_date)$MMWRweek
AUTO_TEMPERATURE_WEEK4$epiweek <- MMWRweek(AUTO_TEMPERATURE_WEEK4$target_end_date)$MMWRweek
AUTO_AVERAGE_WEEK4$epiweek <- MMWRweek(AUTO_AVERAGE_WEEK4$target_end_date)$MMWRweek

ES27_ARIMA_WEEK4$epiweek <- MMWRweek(ES27_ARIMA_WEEK4$target_end_date)$MMWRweek
ES27_ADJACENT_WEEK4$epiweek <- MMWRweek(ES27_ADJACENT_WEEK4$target_end_date)$MMWRweek
ES27_EPIWEEK_WEEK4$epiweek <- MMWRweek(ES27_EPIWEEK_WEEK4$target_end_date)$MMWRweek
ES27_TEMPERATURE_WEEK4$epiweek <- MMWRweek(ES27_TEMPERATURE_WEEK4$target_end_date)$MMWRweek
ES27_AVERAGE_WEEK4$epiweek <- MMWRweek(ES27_AVERAGE_WEEK4$target_end_date)$MMWRweek

```

```

ES64_ARIMA_WEEK4$epiweek <- MMWRweek(ES64_ARIMA_WEEK4$target_end_date)$MMWRweek
ES64_ADJACENT_WEEK4$epiweek <- MMWRweek(ES64_ADJACENT_WEEK4$target_end_date)$MMWRweek
ES64_EPIWEEK_WEEK4$epiweek <- MMWRweek(ES64_EPIWEEK_WEEK4$target_end_date)$MMWRweek
ES64_TEMPERATURE_WEEK4$epiweek <- MMWRweek(ES64_TEMPERATURE_WEEK4$target_end_date)$MMWRweek
ES64_AVERAGE_WEEK4$epiweek <- MMWRweek(ES64_AVERAGE_WEEK4$target_end_date)$MMWRweek

# Dataframe that will be analysed for 2 Week Ahead
df_W4 <- data.frame(
  STATE = AUTO_ARIMA_WEEK4$State,
  Julian_date = AUTO_ARIMA_WEEK4$target_end_date,
  epiweek = AUTO_ARIMA_WEEK4$epiweek,
  AUTO_AR_LB=AUTO_ARIMA_WEEK4$WIS,
  AUTO_ADJ_LB=AUTO_ADJACENT_WEEK4$WIS,
  AUTO_EPI_LB=AUTO_EPIWEEK_WEEK4$WIS,
  AUTO_TEMP_LB=AUTO_TEMPERATURE_WEEK4$WIS,
  AUTO_AVG_LB=AUTO_AVERAGE_WEEK4$WIS,
  
  ES27_AR_LB=ES27_ARIMA_WEEK4$WIS,
  ES27_ADJ_LB=ES27_ADJACENT_WEEK4$WIS,
  ES27_EPI_LB=ES27_EPIWEEK_WEEK4$WIS,
  ES27_TEMP_LB=ES27_TEMPERATURE_WEEK4$WIS,
  ES27_AVG_LB=ES27_AVERAGE_WEEK4$WIS,
  
  ES64_AR_LB=ES64_ARIMA_WEEK4$WIS,
  ES64_ADJ_LB=ES64_ADJACENT_WEEK4$WIS,
  ES64_EPI_LB=ES64_EPIWEEK_WEEK4$WIS,
  ES64_TEMP_LB=ES64_TEMPERATURE_WEEK4$WIS,
  ES64_AVG_LB=ES64_AVERAGE_WEEK4$WIS
)
head(df_W4)

```

	STATE	Julian_date	epiweek	AUTO_AR_LB	AUTO_ADJ_LB	AUTO_EPI_LB	AUTO_TEMP_LB
## 1	Alabama	2022-11-19	46	195.02213	188.18432	191.28263	184.49068
## 2	Alabama	2022-11-26	47	119.84900	93.23656	117.90577	206.13596
## 3	Alabama	2022-12-03	48	140.12527	106.19007	142.84636	145.43076
## 4	Alabama	2022-12-10	49	44.74843	39.45008	88.53987	44.36605
## 5	Alabama	2022-12-17	50	73.72079	27.81180	68.22627	94.55459
## 6	Alabama	2022-12-24	51	53.99946	21.96318	77.33404	53.39699
	AUTO_AVG_LB	ES27_AR_LB	ES27_ADJ_LB	ES27_EPI_LB	ES27_TEMP_LB	ES27_AVG_LB	
## 1	108.69715	133.28250	110.64096	168.33451	214.95541	68.27622	
## 2	30.53778	44.01597	40.89044	45.21651	44.79821	49.47867	
## 3	49.11299	65.53483	65.59596	69.66975	65.34986	91.51636	
## 4	72.19681	78.71868	113.28678	75.15470	78.07503	154.77010	
## 5	55.57844	66.11022	83.59744	60.04147	76.71141	127.32916	
## 6	48.81717	51.57630	65.28205	55.34849	51.35922	118.96839	
	ES64_AR_LB	ES64_ADJ_LB	ES64_EPI_LB	ES64_TEMP_LB	ES64_AVG_LB		
## 1	140.62946	128.11925	153.57953	210.85208	72.38258		
## 2	50.17820	43.45508	49.86112	57.17021	51.78548		
## 3	68.83669	67.34827	70.94611	68.72892	96.78787		
## 4	79.49211	115.03138	77.78400	78.82837	159.23288		
## 5	65.00889	69.31000	61.05206	77.46774	110.23667		
## 6	50.68669	61.17782	56.36945	50.28846	117.30146		

Filter only the flu season

```
# Filter the dataframe for epiweek >= 40 or epiweek <= 20
filtered_df_W1 <- df_W1 %>%
  filter(epiweek >= 40 | epiweek <= 20)

# Display the filtered dataset
head(filtered_df_W1)

##      STATE Julian_date epiweek AUTO_AR_LB AUTO_ADJ_LB AUTO_EPI_LB AUTO_TEMP_LB
## 1 Alabama 2022-10-29      43 164.97469 154.82262 167.16881 158.73461
## 2 Alabama 2022-11-05      44 152.25677 125.66999 147.56571 217.70136
## 3 Alabama 2022-11-12      45 42.76549 33.63300 42.10368 41.93627
## 4 Alabama 2022-11-19      46 27.64761 64.94758 27.33367 28.21504
## 5 Alabama 2022-11-26      47 30.84363 28.07577 29.48502 53.76514
## 6 Alabama 2022-12-03      48 70.42607 81.54591 130.42912 69.58822
##      AUTO_AVG_LB ES27_AR_LB ES27_ADJ_LB ES27_EPI_LB ES27_TEMP_LB ES27_AVG_LB
## 1    111.40345 138.67842 125.21088 160.28714 183.38737 91.96030
## 2     58.82784 100.18764  78.71110 105.33944 101.77054 46.03894
## 3     69.47533  34.89792  39.88211 34.55781 34.77695 87.57530
## 4    138.85755  60.46462 104.07048 59.09603 59.51544 170.95917
## 5     63.33534  34.33307  50.43779 33.09650 41.61223 95.06813
## 6     37.97748  65.48558  47.01762 63.64209 65.18264 41.62150
##      ES64_AR_LB ES64_ADJ_LB ES64_EPI_LB ES64_TEMP_LB ES64_AVG_LB
## 1    138.58970 133.18630 152.65640 180.91343 95.46299
## 2    104.66467  79.03814 109.79832 112.65048 46.02587
## 3     35.59296  40.77665  34.91187 35.43623 91.62425
## 4     65.23764 116.71188  64.70152 64.23071 187.23402
## 5     34.67596  39.50396  33.76894 43.53779 79.26431
## 6     65.56808  48.47490  60.83552 65.06090 40.40643

# Filter the dataframe for epiweek >= 40 or epiweek <= 20
filtered_df_W2 <- df_W2 %>%
  filter(epiweek >= 40 | epiweek <= 20)

# Display the filtered dataset
head(filtered_df_W2)

##      STATE Julian_date epiweek AUTO_AR_LB AUTO_ADJ_LB AUTO_EPI_LB AUTO_TEMP_LB
## 1 Alabama 2022-11-05      44 277.52451 260.09905 285.18405 272.93892
## 2 Alabama 2022-11-12      45 123.76206  98.81682 119.43244 225.73550
## 3 Alabama 2022-11-19      46 27.65801 30.68134 27.41752 28.12358
## 4 Alabama 2022-11-26      47 29.02424 52.51845 28.08553 29.14046
## 5 Alabama 2022-12-03      48 168.36465 90.69261 158.96411 46.58232
## 6 Alabama 2022-12-10      49 28.16100 32.72021 80.13328 28.28261
##      AUTO_AVG_LB ES27_AR_LB ES27_ADJ_LB ES27_EPI_LB ES27_TEMP_LB ES27_AVG_LB
## 1    198.89474 231.81375 211.64222 267.41322 304.55539 162.85351
## 2     36.72868  63.48153  50.33202  67.64182  65.26766 35.19653
## 3    108.01831  59.66592  75.06981  56.00013  58.12578 149.80369
## 4     98.55272  65.08632 103.25240  62.44626  64.43724 153.72638
## 5     44.42643  56.73039  46.35015  60.59899  51.23473 42.80849
## 6     32.65251  29.66896  35.66245  30.87440  29.65345 70.94583
##      ES64_AR_LB ES64_ADJ_LB ES64_EPI_LB ES64_TEMP_LB ES64_AVG_LB
```

```

## 1 238.53237 222.99366 258.96427 299.73039 166.07500
## 2 69.05467 51.88106 72.42017 78.11781 36.04737
## 3 62.96120 76.63837 57.82458 60.28029 159.58156
## 4 56.75389 85.65467 56.00229 56.38154 135.42546
## 5 56.78263 47.00848 59.27148 50.22071 41.59863
## 6 29.36865 34.02798 30.95575 29.43398 72.17442

# Filter the dataframe for epiweek >= 40 or epiweek <= 20
filtered_df_W3 <- df_W3 %>%
  filter(epiweek >= 40 | epiweek <= 20)

# Display the filtered dataset
head(filtered_df_W3)

##      STATE Julian_date epiweek AUTO_AR_LB AUTO_ADJ_LB AUTO_EPI_LB AUTO_TEMP_LB
## 1 Alabama 2022-11-12      45 255.67207 241.03987 259.01418 251.72006
## 2 Alabama 2022-11-19      46  96.15237  72.82592  93.74152 183.44917
## 3 Alabama 2022-11-26      47 39.85273  31.57650  40.18449 39.08883
## 4 Alabama 2022-12-03      48 99.81193  48.50555 137.20756 98.21383
## 5 Alabama 2022-12-10      49 110.10063  48.22772 102.41015 44.48408
## 6 Alabama 2022-12-17      50 46.25364  20.84313  52.68966 43.73901
##      AUTO_AVG_LB ES27_AR_LB ES27_ADJ_LB ES27_EPI_LB ES27_TEMP_LB ES27_AVG_LB
## 1    171.21369 196.06526   172.30067  234.10599  281.50630 123.97151
## 2     28.87514 38.74124   35.84129  39.54435  39.32222 47.37365
## 3     88.49128 65.89910   79.19608  64.75946  64.52593 147.40097
## 4     46.81402 54.26322   64.53571  55.82159  54.26457 83.05521
## 5     34.92782 39.63831   48.66583  39.05724  44.12713 74.54001
## 6     56.01685 47.13537   63.33375  49.56314  46.42059 119.68466
##      ES64_AR_LB ES64_ADJ_LB ES64_EPI_LB ES64_TEMP_LB ES64_AVG_LB
## 1    201.04627 187.07056   217.42038  276.50884 125.77841
## 2     42.95249 37.63603   42.86805  48.73175  50.21087
## 3     69.16687 80.13844   67.19970  66.66957 147.74137
## 4     54.91965 64.83746   55.47049  55.00449  84.74953
## 5     39.02389 48.27718   38.64341  44.30995  71.72136
## 6     45.95904 58.36465   48.61270  44.94409 109.02039

# Filter the dataframe for epiweek >= 40 or epiweek <= 20
filtered_df_W4 <- df_W4 %>%
  filter(epiweek >= 40 | epiweek <= 20)

# Display the filtered dataset
head(filtered_df_W4)

##      STATE Julian_date epiweek AUTO_AR_LB AUTO_ADJ_LB AUTO_EPI_LB AUTO_TEMP_LB
## 1 Alabama 2022-11-19      46 195.02213 188.18432 191.28263 184.49068
## 2 Alabama 2022-11-26      47 119.84900  93.23656 117.90577 206.13596
## 3 Alabama 2022-12-03      48 140.12527 106.19007 142.84636 145.43076
## 4 Alabama 2022-12-10      49  44.74843  39.45008  88.53987 44.36605
## 5 Alabama 2022-12-17      50  73.72079  27.81180  68.22627 94.55459
## 6 Alabama 2022-12-24      51 53.99946  21.96318  77.33404 53.39699
##      AUTO_AVG_LB ES27_AR_LB ES27_ADJ_LB ES27_EPI_LB ES27_TEMP_LB ES27_AVG_LB
## 1    108.69715 133.28250 110.64096 168.33451 214.95541 68.27622
## 2     30.53778 44.01597  40.89044  45.21651  44.79821 49.47867

```

```

## 3    49.11299   65.53483   65.59596   69.66975   65.34986   91.51636
## 4    72.19681   78.71868   113.28678   75.15470   78.07503   154.77010
## 5    55.57844   66.11022   83.59744   60.04147   76.71141   127.32916
## 6    48.81717   51.57630   65.28205   55.34849   51.35922   118.96839
##   ES64_AR_LB ES64_ADJ_LB ES64_EPI_LB ES64_TEMP_LB ES64_AVG_LB
## 1    140.62946   128.11925   153.57953   210.85208   72.38258
## 2    50.17820    43.45508   49.86112   57.17021   51.78548
## 3    68.83669   67.34827   70.94611   68.72892   96.78787
## 4    79.49211   115.03138   77.78400   78.82837   159.23288
## 5    65.00889   69.31000   61.05206   77.46774   110.23667
## 6    50.68669   61.17782   56.36945   50.28846   117.30146

```

Fitting data.

```

i <- 57 # forecasted week
ii <- 8 # state number

my_data = read.csv("models_with_logback/treated_influenza_hosp_dataframe_v2.csv")
my_data$target_end_date<-as.Date(my_data$target_end_date) # set the dates as dates

list_of_states <- split(my_data, my_data$state_name)

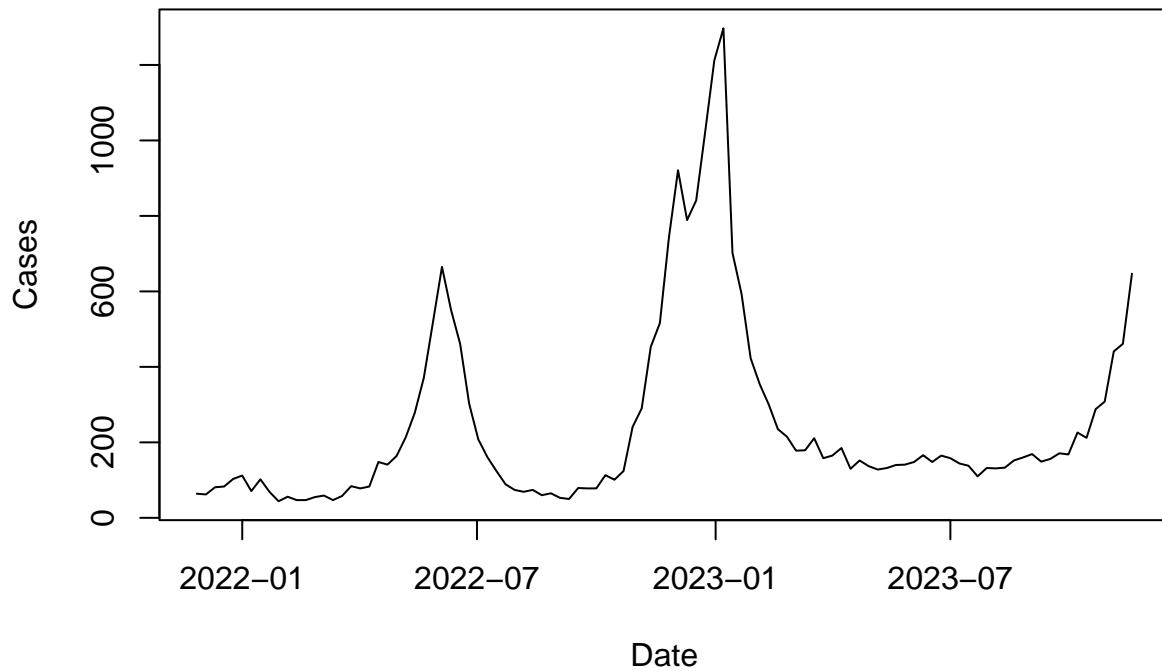
# Define the date range
end_date <- as.Date(AUTO_AVERAGE_WEEK1_list[[ii]]$target_end_date[i]-7)
start_date <- end_date - (7*103) # 2 years before

# Subset the data
subset_data <- subset(list_of_states[[ii]],
                       target_end_date >= start_date & target_end_date <= end_date)

# Plot
plot(subset_data$target_end_date, subset_data$cases, type = "l",
      xlab = "Date", ylab = "Cases", main = "Cases - Last 2 Years")

```

## Cases – Last 2 Years



Plotting conceptual image

```
i <- 57 # forecast week
ii <- 8 # state number

forecast_df <- data.frame(
  Dates = c(
    AUTO_AVERAGE_WEEK1_list[[ii]]$target_end_date[i] ,
    AUTO_AVERAGE_WEEK2_list[[ii]]$target_end_date[i] ,
    AUTO_AVERAGE_WEEK3_list[[ii]]$target_end_date[i] ,
    AUTO_AVERAGE_WEEK4_list[[ii]]$target_end_date[i]
  ),
  Hosp   = c(
    AUTO_AVERAGE_WEEK1_list[[ii]]$cases[i] ,
    AUTO_AVERAGE_WEEK2_list[[ii]]$cases[i] ,
    AUTO_AVERAGE_WEEK3_list[[ii]]$cases[i] ,
    AUTO_AVERAGE_WEEK4_list[[ii]]$cases[i]
  ),
  Value = c(
    AUTO_AVERAGE_WEEK1_list[[ii]]$forecasts[i] ,
    AUTO_AVERAGE_WEEK2_list[[ii]]$forecasts[i] ,
    AUTO_AVERAGE_WEEK3_list[[ii]]$forecasts[i] ,
    AUTO_AVERAGE_WEEK4_list[[ii]]$forecasts[i]
  ),
  Lower = c(
    AUTO_AVERAGE_WEEK1_list[[ii]][["0.025"]][i] ,
```

```

    AUTO_AVERAGE_WEEK2_list[[ii]][["0.025"]][i],
    AUTO_AVERAGE_WEEK3_list[[ii]][["0.025"]][i],
    AUTO_AVERAGE_WEEK4_list[[ii]][["0.025"]][i]
),
Upper = c(
    AUTO_AVERAGE_WEEK1_list[[ii]][["0.975"]][i],
    AUTO_AVERAGE_WEEK2_list[[ii]][["0.975"]][i],
    AUTO_AVERAGE_WEEK3_list[[ii]][["0.975"]][i],
    AUTO_AVERAGE_WEEK4_list[[ii]][["0.975"]][i]
)
)

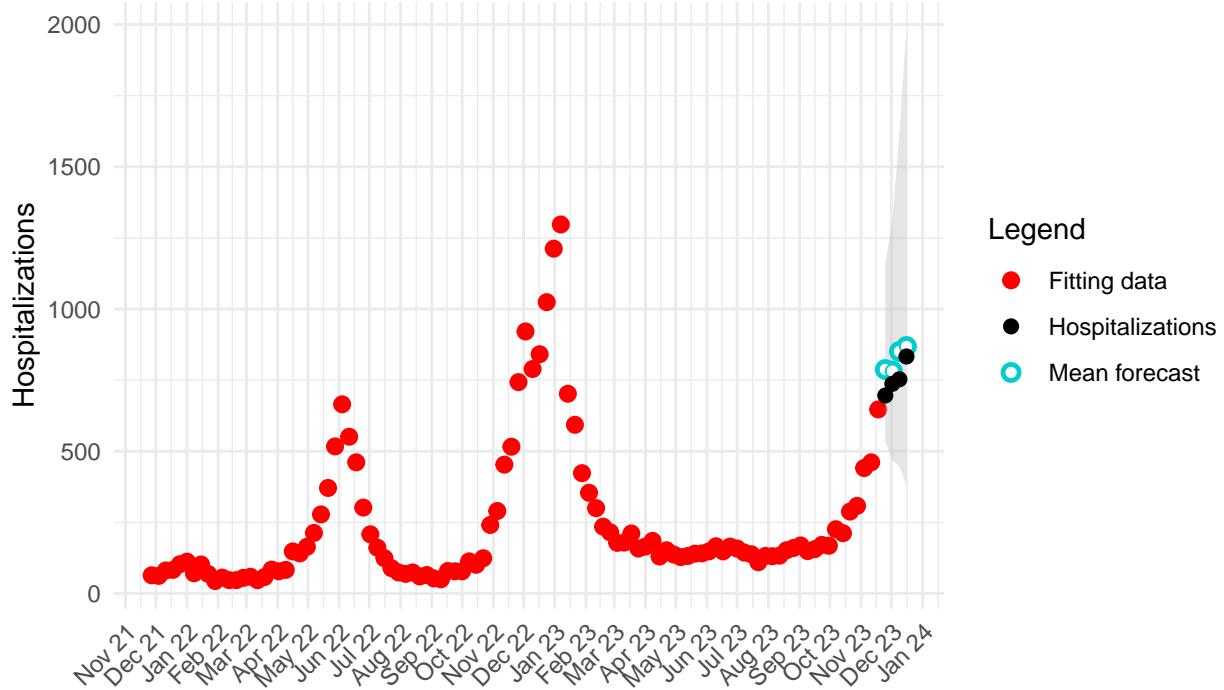
conceptual_figure = ggplot() +
# two-year fitting data
geom_point(data = subset_data,
            aes(x = target_end_date, y = cases, color = "Fitting data"),
            size = 2.5) +
# grey ribbon
geom_ribbon(data = forecast_df,
            aes(x = Dates, ymin = Lower, ymax = Upper),
            fill = "gray", alpha = 0.4) +
# mean forecast
geom_point(data = forecast_df,
            aes(x = Dates, y = Value, color = "Mean forecast"),
            size = 3) +
geom_point(data = forecast_df,
            aes(x = Dates, y = Value, color = "Mean forecast"),
            shape = 21, fill = "white", size = 2) +
# actual hospitalizations
geom_point(data = forecast_df,
            aes(x = Dates, y = Hosp, color = "Hospitalizations"),
            shape = 16, size = 2.5) +
# Manually define legend entries
scale_color_manual(name = "Legend",
                    values = c("Fitting data" = "red",
                               "Mean forecast" = "cyan3",
                               "Hospitalizations" = "black")) +
scale_x_date(
date_breaks = "1 month",
date_labels = "%b %y"
) +
labs(
    title     = "Example of probabilistic forecasts",
    subtitle = "Gray ribbon represents 95% confidence interval",
    x        = "",
    y        = "Hospitalizations"
) +
theme_minimal() +
theme(plot.title = element_text(size = 18, face = "bold"),
      plot.subtitle = element_text(size = 14),
      axis.text.x = element_text(angle = 45, hjust = 1))

```

```
conceptual_figure
```

## Example of probabilistic forecasts

Gray ribbon represents 95% confidence interval



```
# Save the combined plot
```

```
ggsave("Fig1.jpg", conceptual_figure, width = 7, height = 4, dpi = 600)
```

Calculate mean weighted interval score for influenza seasons on each model and each state.

```
# Define the function
calculate_mean_wis <- function(data) {
  data %>%
    group_by(STATE) %>%
    summarize(
      AUTO_AR_LB = mean(AUTO_AR_LB, na.rm = TRUE),
      AUTO_ADJ_LB = mean(AUTO_ADJ_LB, na.rm = TRUE),
      AUTO_EPI_LB = mean(AUTO_EPI_LB, na.rm = TRUE),
      AUTO_TMP_LB = mean(AUTO_TEMP_LB, na.rm = TRUE),
      AUTO_AVG_LB = mean(AUTO_AVG_LB, na.rm = TRUE),

      ES27_AR_LB = mean(ES27_AR_LB, na.rm = TRUE),
      ES27_ADJ_LB = mean(ES27_ADJ_LB, na.rm = TRUE),
      ES27_EPI_LB = mean(ES27_EPI_LB, na.rm = TRUE),
      ES27_TMP_LB = mean(ES27_TEMP_LB, na.rm = TRUE),
      ES27_AVG_LB = mean(ES27_AVG_LB, na.rm = TRUE),
```

```

    ES64_AR_LB = mean(ES64_AR_LB, na.rm = TRUE),
    ES64_ADJ_LB = mean(ES64_ADJ_LB, na.rm = TRUE),
    ES64_EPI_LB = mean(ES64_EPI_LB, na.rm = TRUE),
    ES64_TMP_LB = mean(ES64_TEMP_LB, na.rm = TRUE),
    ES64_AVG_LB = mean(ES64_AVG_LB, na.rm = TRUE),
)
}

# Now we can use the function with any dataframe
W1_Lg <- calculate_mean_wis(filtered_df_W1)
W2_Lg <- calculate_mean_wis(filtered_df_W2)
W3_Lg <- calculate_mean_wis(filtered_df_W3)
W4_Lg <- calculate_mean_wis(filtered_df_W4)

# Display the resulting dataframe
head(W1_Lg)

```

```

## # A tibble: 6 x 16
##   STATE    AUTO_AR_LB AUTO_ADJ_LB AUTO_EPI_LB AUTO_TMP_LB AUTO_AVG_LB ES27_AR_LB
##   <chr>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>
## 1 Alabama    21.5      18.7      22.4      24.0      19.4      20.9
## 2 Arizona    45.5      35.2      45.3      45.5      57.5      44.5
## 3 Arkansas   20.0      17.0      19.7      21.5      19.9      21.7
## 4 Califor~   81.6      81.0      94.9      94.6     111.      89.5
## 5 Colorado   16.8      17.8      17.2      18.1      18.4      17.4
## 6 Connect~   19.3      15.8      18.8      19.0      15.2      19.8
## # i 9 more variables: ES27_ADJ_LB <dbl>, ES27_EPI_LB <dbl>, ES27_TMP_LB <dbl>,
## #   ES27_AVG_LB <dbl>, ES64_AR_LB <dbl>, ES64_ADJ_LB <dbl>, ES64_EPI_LB <dbl>,
## #   ES64_TMP_LB <dbl>, ES64_AVG_LB <dbl>

```

```
head(W2_Lg)
```

```

## # A tibble: 6 x 16
##   STATE    AUTO_AR_LB AUTO_ADJ_LB AUTO_EPI_LB AUTO_TMP_LB AUTO_AVG_LB ES27_AR_LB
##   <chr>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>
## 1 Alabama    31.2      25.7      30.9      30.8      26.4      28.7
## 2 Arizona    77.9      64.9      76.6      76.8      95.7      72.5
## 3 Arkansas   30.4      27.7      30.6      32.4      29.6      35.7
## 4 Califor~   160.      156.      190.      155.      194.      166.
## 5 Colorado   26.7      28.2      28.6      28.4      26.0      28.5
## 6 Connect~   31.4      25.2      29.2      30.2      24.0      31.9
## # i 9 more variables: ES27_ADJ_LB <dbl>, ES27_EPI_LB <dbl>, ES27_TMP_LB <dbl>,
## #   ES27_AVG_LB <dbl>, ES64_AR_LB <dbl>, ES64_ADJ_LB <dbl>, ES64_EPI_LB <dbl>,
## #   ES64_TMP_LB <dbl>, ES64_AVG_LB <dbl>

```

```
head(W3_Lg)
```

```

## # A tibble: 6 x 16
##   STATE    AUTO_AR_LB AUTO_ADJ_LB AUTO_EPI_LB AUTO_TMP_LB AUTO_AVG_LB ES27_AR_LB
##   <chr>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>
## 1 Alabama    38.1      28.9      36.9      39.4      31.4      33.5

```

```

## 2 Arizona      106.       92.8      104.       104.       133.       100.
## 3 Arkansas     40.2       37.8      39.0       42.0       38.3       48.5
## 4 Califor~    249.      229.      285.      219.      278.      245.
## 5 Colorado     35.7       40.0      40.4       38.8       35.8       39.9
## 6 Connect~     46.1       37.7      40.8       43.9       35.3       45.9
## # i 9 more variables: ES27_ADJ_LB <dbl>, ES27_EPI_LB <dbl>, ES27_TMP_LB <dbl>,
## #   ES27_AVG_LB <dbl>, ES64_AR_LB <dbl>, ES64_ADJ_LB <dbl>, ES64_EPI_LB <dbl>,
## #   ES64_TMP_LB <dbl>, ES64_AVG_LB <dbl>

```

```
head(W4_Lg)
```

```

## # A tibble: 6 x 16
##   STATE    AUTO_AR_LB AUTO_ADJ_LB AUTO_EPI_LB AUTO_TMP_LB AUTO_AVG_LB ES27_AR_LB
##   <chr>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>
## 1 Alabama    43.4      33.5      41.9      45.6      36.8      38.4
## 2 Arizona    132.      120.      130.      129.      171.      129.
## 3 Arkansas   50.3       47.8      48.9      52.0      48.8      64.6
## 4 Califor~   325.      290.      362.      270.      351.      315.
## 5 Colorado   44.1       51.5      51.5      48.2      45.5      50.7
## 6 Connect~   59.0       48.1      52.6      56.0      46.0      59.0
## # i 9 more variables: ES27_ADJ_LB <dbl>, ES27_EPI_LB <dbl>, ES27_TMP_LB <dbl>,
## #   ES27_AVG_LB <dbl>, ES64_AR_LB <dbl>, ES64_ADJ_LB <dbl>, ES64_EPI_LB <dbl>,
## #   ES64_TMP_LB <dbl>, ES64_AVG_LB <dbl>

```

Here we have boxplots of the mean(WIS) by each state.

```

# Combine the data into one dataframe
combined_data <- data.frame(
  week = rep(c("W1", "W2", "W3", "W4"), each = 48 * 15),
  Models = rep(c("AUTO_AR_LB", "ES27_AR_LB", "ES64_AR_LB", "AUTO_ADJ_LB", "ES27_ADJ_LB", "ES64_ADJ_LB",
  value = c(W1_Lg$AUTO_AR_LB, W1_Lg$ES27_AR_LB, W1_Lg$ES64_AR_LB,
  W1_Lg$AUTO_ADJ_LB, W1_Lg$ES27_ADJ_LB, W1_Lg$ES64_ADJ_LB,
  W1_Lg$AUTO_TMP_LB, W1_Lg$ES27_TMP_LB, W1_Lg$ES64_TMP_LB,
  W1_Lg$AUTO_EPI_LB, W1_Lg$ES27_EPI_LB, W1_Lg$ES64_EPI_LB,
  W1_Lg$AUTO_AVG_LB, W1_Lg$ES27_AVG_LB, W1_Lg$ES64_AVG_LB,
  W2_Lg$AUTO_AR_LB, W2_Lg$ES27_AR_LB, W2_Lg$ES64_AR_LB,
  W2_Lg$AUTO_ADJ_LB, W2_Lg$ES27_ADJ_LB, W2_Lg$ES64_ADJ_LB,
  W2_Lg$AUTO_TMP_LB, W2_Lg$ES27_TMP_LB, W2_Lg$ES64_TMP_LB,
  W2_Lg$AUTO_EPI_LB, W2_Lg$ES27_EPI_LB, W2_Lg$ES64_EPI_LB,
  W2_Lg$AUTO_AVG_LB, W2_Lg$ES27_AVG_LB, W2_Lg$ES64_AVG_LB,
  W3_Lg$AUTO_AR_LB, W3_Lg$ES27_AR_LB, W3_Lg$ES64_AR_LB,
  W3_Lg$AUTO_ADJ_LB, W3_Lg$ES27_ADJ_LB, W3_Lg$ES64_ADJ_LB,
  W3_Lg$AUTO_TMP_LB, W3_Lg$ES27_TMP_LB, W3_Lg$ES64_TMP_LB,
  W3_Lg$AUTO_EPI_LB, W3_Lg$ES27_EPI_LB, W3_Lg$ES64_EPI_LB,
  W3_Lg$AUTO_AVG_LB, W3_Lg$ES27_AVG_LB, W3_Lg$ES64_AVG_LB,
  W4_Lg$AUTO_AR_LB, W4_Lg$ES27_AR_LB, W4_Lg$ES64_AR_LB,
  W4_Lg$AUTO_ADJ_LB, W4_Lg$ES27_ADJ_LB, W4_Lg$ES64_ADJ_LB,
  W4_Lg$AUTO_TMP_LB, W4_Lg$ES27_TMP_LB, W4_Lg$ES64_TMP_LB,

```

```

W4_Lg$AUTO_EPI_LB, W4_Lg$ES27_EPI_LB, W4_Lg$ES64_EPI_LB,
W4_Lg$AUTO_AVG_LB, W4_Lg$ES27_AVG_LB, W4_Lg$ES64_AVG_LB)
)

# Create a new column for model families
combined_data <- combined_data %>%
  mutate(category = case_when(
    grepl("AR", Models) ~ "AR",
    grepl("EPI", Models) ~ "EPI",
    grepl("TMP", Models) ~ "TMP",
    grepl("ADJ", Models) ~ "ADJ",
    grepl("AVG", Models) ~ "AVG",
  ))

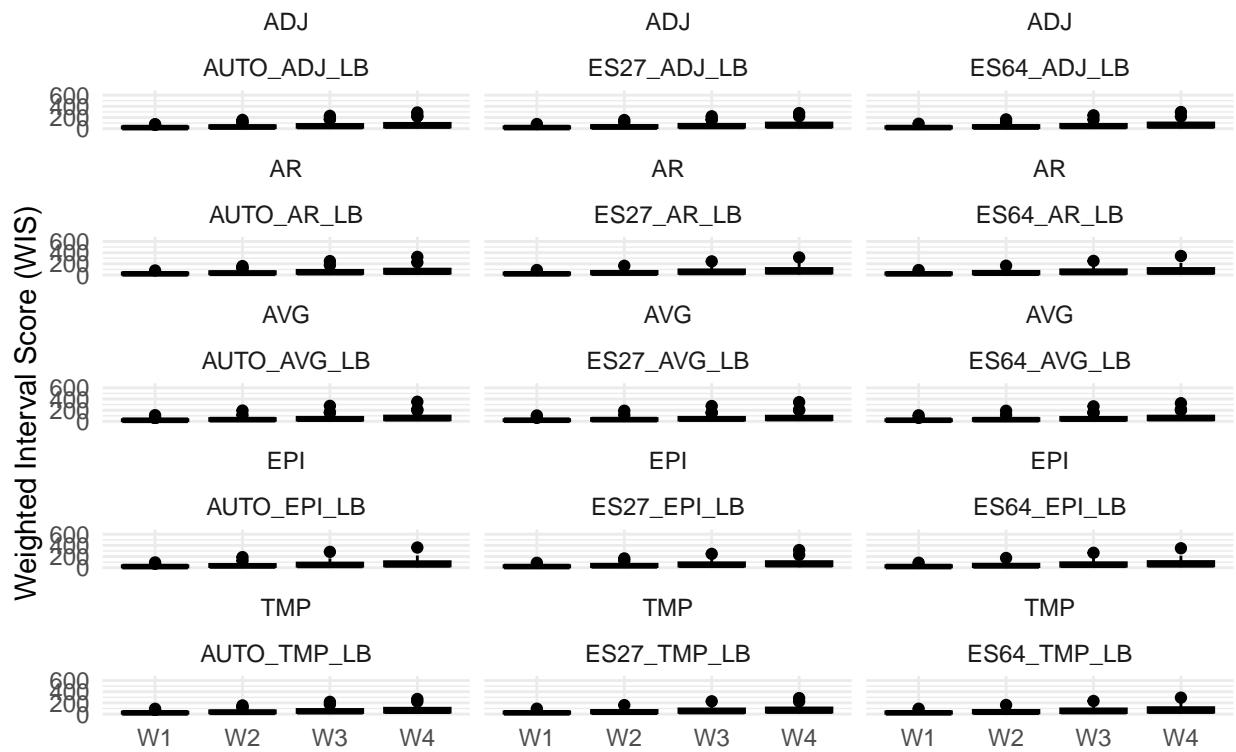
# Define color ramps for each model family
colors_ar <- c("#a6cee3", "#226e83", "#08306b")
colors_adj <- c("#fb9a99", "red3", "#a11c3e")
colors_epi <- c("#cab2d6", "purple2", "#5e2b7b")
colors_tmp <- c("lightgreen", "green3", "#319045")
colors_avg <- c("orange", "orange2", "orange3")

# Use a custom color on each model
custom_colors <- c(
  "AUTO_AR_LB" = colors_ar[1], "ES27_AR_LB" = colors_ar[2], "ES64_AR_LB" = colors_ar[3],
  "AUTO_EPI_LB" = colors_epi[1], "ES27_EPI_LB" = colors_epi[2], "ES64_EPI_LB" = colors_epi[3],
  "AUTO_TMP_LB" = colors_tmp[1], "ES27_TMP_LB" = colors_tmp[2], "ES64_TMP_LB" = colors_tmp[3],
  "AUTO_ADJ_LB" = colors_adj[1], "ES27_ADJ_LB" = colors_adj[2], "ES64_ADJ_LB" = colors_adj[3],
  "AUTO_AVG_LB" = colors_avg[1], "ES27_AVG_LB" = colors_avg[2], "ES64_AVG_LB" = colors_avg[3]
)

# Create the box plot with additional facet for categories
plt2<- ggplot(combined_data, aes(x = week, y = value, fill = Models)) +
  geom_boxplot(color = "black") +
  scale_fill_manual(values = custom_colors) +
  facet_wrap(category ~ Models , nrow = 5) +
  labs(
    title = "B) Models with log-back transformations",
    x = "",
    y = "Weighted Interval Score (WIS)"
  ) +
  scale_y_continuous(limits = c(0, 650)) + # Set y-axis range
  theme_minimal()+
  theme(legend.position = "none")
plt2

```

## B) Models with log–back transformations



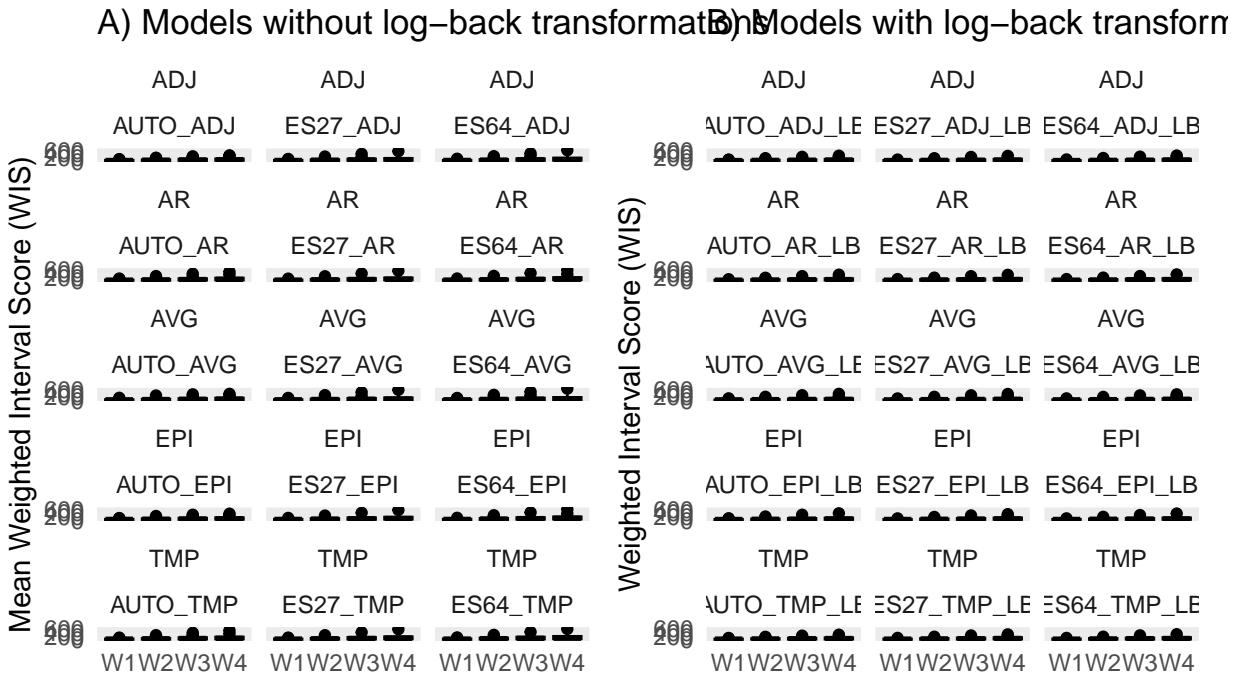
```
#ggsave(plt2, height = 8, width = 10, filename="Fig1B.jpg")
```

Figure 3

```
# Combine plot in a 2x2 grid
figure3 <- (plt1 | plt2) +
  plot_annotation(
    title = "Mean WIS for the 24 models",
    subtitle = "Results by target week",
    theme = theme(plot.title = element_text(size = 16, face = "bold"),
                  plot.subtitle = element_text(size = 14)))
  )
# Print
print(figure3)
```

## Mean WIS for the 24 models

Results by target week



```
# Save to file
ggsave("Fig3.jpg", figure3, width =9, height =10)
```

COMBINING THE RESULTS WITH LOG-BACK AND NO LOG-BACK TRANSFORMATION

```
W1<-merge(W1_Lg,W1_NoLg, by = "STATE")
W2<-merge(W2_Lg,W2_NoLg, by = "STATE")
W3<-merge(W3_Lg,W3_NoLg, by = "STATE")
W4<-merge(W4_Lg,W4_NoLg, by = "STATE")
```

Here I include a column with the best model based on the lowest mean(WIS) of each state.

```
# BEST RESULT

cols <- colnames(W1)[-1] # get states names
W1$Best_Result <- character(nrow(W1)) # create an empty column for the best models

# Give me the model with lower WIS value
for (i in 1:nrow(W1)) {
  # Find the model with the minimum value on each row
  # based on the column name
  # save it in the best result
  W1$Best_Result[i] <- cols[which.min(W1[i, cols])]
}
```

```

# REORDER BY FREQUENCY
W1$Best_Result <- fct_infreq(W1$Best_Result)
# Create a new column to indicate if model has a log-back
# transformation or not
W1$Model_Type <- ifelse(grepl("_LB$", W1$Best_Result), "With log-back transformation", "Without log-back transformation")

# Print the first rows
head(W1)

##          STATE AUTO_AR_LB AUTO_ADJ_LB AUTO_EPI_LB AUTO_TMP_LB AUTO_AVG_LB
## 1      Alabama   21.50088   18.71138   22.39703   23.98760   19.35844
## 2     Arizona    45.52219   35.18194   45.26514   45.54241   57.53076
## 3   Arkansas    19.96738   16.96916   19.65110   21.45397   19.86030
## 4 California   81.55021   80.96661   94.94076   94.56612  110.61367
## 5 Colorado     16.79576   17.77995   17.17619   18.14275   18.36797
## 6 Connecticut   19.25440   15.82928   18.75026   18.98928   15.19050
##   ES27_AR_LB ES27_ADJ_LB ES27_EPI_LB ES27_TMP_LB ES27_AVG_LB ES64_AR_LB
## 1   20.85880   19.25938   21.33316   21.86869   21.13189   20.99726
## 2   44.49070   36.40411   45.11440   43.39661   56.03339   42.60622
## 3   21.69890   17.41815   21.30174   22.17678   20.30253   21.23517
## 4   89.51986   82.96145   88.05622   96.47274   105.01014   89.10840
## 5   17.36470   16.52006   16.82767   18.19789   16.54575   16.61554
## 6   19.78691   17.76911   19.17224   19.24697   15.88075   19.24724
##   ES64_ADJ_LB ES64_EPI_LB ES64_TMP_LB ES64_AVG_LB AUTO_AR AUTO_ADJ AUTO_EPI
## 1   19.52444   21.33758   22.01204   21.30777   23.82641  19.39934  24.22010
## 2   36.35690   45.63922   43.18101   55.27515   44.64733  42.56012  46.74881
## 3   16.90517   21.00537   21.89732   18.97059   21.50138  21.82260  21.19340
## 4   88.40400   89.59663   96.06782   108.38833  116.56227  93.49131  107.92167
## 5   16.72593   16.40725   17.27189   16.59354   18.07303  17.27311  17.34732
## 6   18.63364   19.23542   18.82657   17.07286   24.10891  26.58338  23.31798
##          AUTO_TMP AUTO_AVG ES27_AR ES27_ADJ ES27_EPI ES27_TMP ES27_AVG
## 1   25.41574  19.49120  23.72509  21.68002  23.80977  24.33474  23.14540
## 2   43.48828  47.41803  45.98882  40.55615  46.50543  44.75483  45.97291
## 3   22.97657  21.78262  21.15178  24.31616  21.22730  21.75788  23.67186
## 4 112.41001 139.25447 112.28469 114.45865 114.18650 112.30299 133.07351
## 5   18.63101  17.91991  17.30915  17.67074  16.54699  17.92531  17.71241
## 6   24.55721  21.47748  25.41463  24.47919  25.66267  25.75117  21.18974
##   ES64_AR ES64_ADJ ES64_EPI ES64_TMP ES64_AVG Best_Result
## 1   24.60068  23.38985  24.23985  24.98227  23.98900 AUTO_ADJ_LB
## 2   47.32760  41.10185  47.88236  46.69213  47.08806 AUTO_ADJ_LB
## 3   22.81367  26.02472  22.92309  24.40946  25.80619 ES64_ADJ_LB
## 4 119.97893 124.88327 121.08555 121.35980 130.50423 AUTO_ADJ_LB
## 5   17.11431  17.59973  16.38063  17.69002  17.94483   ES64_EPI
## 6   26.77442  27.33642  26.98980  28.43992  21.76993 AUTO_AVG_LB
##          Model_Type
## 1 With log-back transformation
## 2 With log-back transformation
## 3 With log-back transformation
## 4 With log-back transformation
## 5 Without log-back transformation
## 6 With log-back transformation

```

```

#####
# Extract the columns of interest
cols <- colnames(W2)[-1] # get states names
W2$Best_Result <- character(nrow(W2)) # create an empty column for the best models
# Give me the model with lower WIS value
for (i in 1:nrow(W2)) {
  # Find the model with the minimum value on each row
  # based on the column name
  # save it in the best result
  W2$Best_Result[i] <- cols[which.min(W2[i, cols])]
}

# REORDER BY FREQUENCY
W2$Best_Result <- fct_infreq(W2$Best_Result)
# Create a new column to indicate if model has a log-back
# transformation or not
W2$Model_Type <- ifelse(grepl("_LB$", W2$Best_Result), "With log-back transformation", "Without log-back")

# Print merged results
head(W2)

```

	STATE	AUTO_AR_LB	AUTO_ADJ_LB	AUTO_EPI_LB	AUTO_TMP_LB	AUTO_AVG_LB	
## 1	Alabama	31.16976	25.68193	30.87672	30.83478	26.42760	
## 2	Arizona	77.94043	64.87303	76.61272	76.77889	95.66276	
## 3	Arkansas	30.37586	27.70001	30.62213	32.40354	29.58848	
## 4	California	160.08310	156.43030	189.78418	155.17087	193.66267	
## 5	Colorado	26.72887	28.24473	28.60433	28.36833	25.99055	
## 6	Connecticut	31.41674	25.18517	29.20407	30.21597	23.97753	
##	ES27_AR_LB	ES27_ADJ_LB	ES27_EPI_LB	ES27_TMP_LB	ES27_AVG_LB	ES64_AR_LB	
## 1	28.71067	26.55882	28.86954	29.18227	28.70554	28.55843	
## 2	72.48261	65.31287	72.64752	69.27474	90.58515	69.83565	
## 3	35.66330	28.47459	34.83340	35.30356	31.42918	34.36039	
## 4	165.75755	154.26221	166.46913	161.08032	190.87587	166.77175	
## 5	28.47549	26.23565	27.25269	29.09701	23.57854	26.25568	
## 6	31.92099	27.19555	30.36840	30.02038	23.77804	29.72233	
##	ES64_ADJ_LB	ES64_EPI_LB	ES64_TMP_LB	ES64_AVG_LB	AUTO_AR	AUTO_ADJ	AUTO_EPI
## 1	26.72802	28.54364	28.94711	28.49263	36.00316	29.97398	36.38449
## 2	64.67340	72.26294	68.97624	87.16214	101.01153	104.67278	104.56702
## 3	27.65420	34.20213	34.79972	29.46256	37.56866	37.66000	36.64999
## 4	165.77120	174.51730	162.96468	189.74315	250.10777	168.19945	216.81694
## 5	27.03370	26.21377	26.85938	24.10903	30.56762	27.51315	29.91979
## 6	27.82723	29.27228	28.55528	24.63026	38.86048	39.70454	36.29880
##	AUTO_TMP	AUTO_AVG	ES27_AR	ES27_ADJ	ES27_EPI	ES27_TMP	ES27_AVG
## 1	41.80543	32.73492	42.08275	38.83911	42.32867	42.38859	43.75290
## 2	98.30048	103.21981	106.97999	98.93783	109.11005	106.30188	101.71638
## 3	39.01924	36.50152	36.96728	41.95746	37.21486	37.77238	40.39599
## 4	244.65152	262.81249	240.32979	243.04099	241.58930	242.91313	282.04360
## 5	31.01883	27.77390	28.68397	29.22517	27.39527	29.13544	29.00367
## 6	39.90806	35.17583	43.93562	38.25072	43.70789	46.07414	34.88137
##	ES64_AR	ES64_ADJ	ES64_EPI	ES64_TMP	ES64_AVG	Best_Result	
## 1	44.54860	43.92800	44.88587	44.86200	44.87401	AUTO_ADJ_LB	
## 2	111.04490	100.54368	111.99654	111.43491	101.78799	ES64_ADJ_LB	

```

## 3 39.76156 45.64826 39.72121 42.19981 44.18069 ES64_ADJ_LB
## 4 259.45167 263.40632 258.20380 260.15284 296.77384 ES27_ADJ_LB
## 5 28.35024 29.14537 27.25058 28.66675 29.54493 ES27_AVG_LB
## 6 48.33919 44.61704 48.15219 49.64837 36.21101 ES27_AVG_LB
## Model_Type
## 1 With log-back transformation
## 2 With log-back transformation
## 3 With log-back transformation
## 4 With log-back transformation
## 5 With log-back transformation
## 6 With log-back transformation

#####
# BEST RESULT
# Extract the columns of interest
cols <- colnames(W3)[-1] # get states names
W3$Best_Result <- character(nrow(W3))# create an empty column for the best models
# Give me the model with lower WIS value
for (i in 1:nrow(W3)) {
  # Find the model with the minimum value on each row
  # based on the column name
  # save it in the best result
  W3$Best_Result[i] <- cols[which.min(W3[i, cols])]
}

# REORDER BY FREQUENCY
W3$Best_Result <- fct_infreq(W3$Best_Result)
# Create a new column to indicate if model has a log-back
# transformation or not
W3$Model_Type <- ifelse(grepl("_LB$", W3$Best_Result), "With log-back transformation", "Without log-back transformation")

# Print merged results
head(W3)

##          STATE AUTO_AR_LB AUTO_ADJ_LB AUTO_EPI_LB AUTO_TMP_LB AUTO_AVG_LB
## 1      Alabama   38.06767   28.85802   36.94506   39.35297   31.36858
## 2     Arizona   105.68808   92.84931  103.83354  103.95012  132.50762
## 3    Arkansas   40.16564   37.83843   38.98530   41.96317   38.34471
## 4 California   249.33610  228.93146  284.58703  218.78418  277.57507
## 5 Colorado    35.72667   40.00171   40.35217   38.82044   35.81360
## 6 Connecticut   46.05532   37.72490   40.79870   43.91198   35.32860
##          ES27_AR_LB ES27_ADJ_LB ES27_EPI_LB ES27_TMP_LB ES27_AVG_LB ES64_AR_LB
## 1       33.45276   30.53513   33.55950   34.18829   34.61166   33.28668
## 2      100.07908   91.84994   95.59706   96.00889  122.10859   92.95885
## 3       48.46757   38.40733   46.55857   46.64195   41.18293   45.25104
## 4      244.51737  221.19022  248.02803  229.05807  274.51717  252.39978
## 5       39.94547   37.34637   37.76580   40.46168   32.31140   35.34731
## 6       45.88978   39.08301   44.32235   42.91971   34.63863   42.37364
##          ES64_ADJ_LB ES64_EPI_LB ES64_TMP_LB ES64_AVG_LB AUTO_AR AUTO_ADJ AUTO_EPI
## 1        30.79499   32.88516   34.00872   34.47802   38.77036   35.62976  44.16947
## 2        90.05179   94.39553   92.66650  117.28995  138.99966  152.44372  140.19591
## 3       36.40443   44.53144   45.03869   38.47283   52.51509   51.24779  47.60087
## 4      238.57143  267.09273  234.42772  269.05311  379.07745  255.92721  285.23593
## 5       38.77150   36.02127   36.51507   33.53203   44.40351   38.95800  43.56443

```

```

## 6 39.59984 41.61795 40.30865 34.91718 56.23350 53.78087 51.78381
## AUTO_TMP AUTO_AVG ES27_AR ES27_ADJ ES27_EPI ES27_TMP ES27_AVG
## 1 46.05607 39.25860 53.05708 47.77327 53.43002 54.26636 54.87899
## 2 135.37528 151.11178 150.74098 145.74845 154.68408 150.05604 148.77660
## 3 53.62337 48.35269 51.07920 57.01549 51.29766 51.66749 54.81546
## 4 375.50795 326.25154 388.35449 388.89710 390.76645 397.99830 438.92134
## 5 45.20783 38.22422 43.44209 42.64511 42.28405 43.91010 42.01012
## 6 57.14880 52.91577 64.64686 52.62770 64.57240 67.41036 50.40027
## ES64_AR ES64_ADJ ES64_EPI ES64_TMP ES64_AVG Best_Result
## 1 57.76643 55.91689 58.28371 58.87958 58.30163 AUTO_ADJ_LB
## 2 157.70389 148.17646 158.24416 157.63183 147.96381 ES64_ADJ_LB
## 3 53.31634 60.43716 52.70552 55.09474 59.46949 ES64_ADJ_LB
## 4 409.87994 415.11647 409.29568 416.00723 462.00180 AUTO_TMP_LB
## 5 42.75558 42.65950 41.73564 42.90381 42.79269 ES27_AVG_LB
## 6 74.42901 61.34672 74.46252 77.61053 55.22415 ES27_AVG_LB
## Model_Type
## 1 With log-back transformation
## 2 With log-back transformation
## 3 With log-back transformation
## 4 With log-back transformation
## 5 With log-back transformation
## 6 With log-back transformation

#####
# BEST RESULT
# Extract the columns of interest
cols <- colnames(W4)[-1] # get states names
W4$Best_Result <- character(nrow(W4)) # create an empty column for the best models

# Give me the model with lower WIS value
for (i in 1:nrow(W4)) {
  # Find the model with the minimum value on each row
  # based on the column name
  # save it in the best result
  W4$Best_Result[i] <- cols[which.min(W4[i, cols])]
}
W4$Best_Result <- fct_infreq(W4$Best_Result)
# Create a new column to indicate if model has a log-back
# transformation or not
W4$Model_Type <- ifelse(grepl("_LB$", W4$Best_Result), "With log-back transformation", "Without log-back")
# Print merged results
head(W4)

## STATE AUTO_AR_LB AUTO_ADJ_LB AUTO_EPI_LB AUTO_TMP_LB AUTO_AVG_LB
## 1 Alabama 43.42794 33.46697 41.85936 45.64649 36.77960
## 2 Arizona 131.50684 120.25171 129.57882 129.16855 170.59572
## 3 Arkansas 50.30320 47.78956 48.89116 52.00687 48.76410
## 4 California 324.71650 289.72878 361.88210 269.99206 351.12200
## 5 Colorado 44.12714 51.52363 51.46275 48.20536 45.52945
## 6 Connecticut 58.96374 48.12606 52.61345 56.02897 46.04524
## ES27_AR_LB ES27_ADJ_LB ES27_EPI_LB ES27_TMP_LB ES27_AVG_LB ES64_AR_LB
## 1 38.39827 35.18243 38.88707 39.27315 40.92104 38.62733
## 2 129.08500 118.44168 121.14001 120.93762 155.75597 116.23067
## 3 64.58626 48.81809 61.34403 60.73860 52.45312 58.06495

```

```

## 4 315.24701 279.34082 316.04186 286.16079 344.18840 340.47004
## 5 50.73779 48.04605 47.84586 50.76351 42.03216 43.67938
## 6 58.95177 49.29686 57.15871 55.37101 44.83746 53.82269
##   ES64_ADJ_LB ES64_EPI_LB ES64_TMP_LB ES64_AVG_LB AUTO_AR AUTO_ADJ AUTO_EPI
## 1 35.36200 38.61357 39.33035 41.06098 45.45221 39.93102 49.28308
## 2 115.06789 119.63801 116.76110 146.54151 163.21412 179.90852 161.64565
## 3 44.66368 56.52417 57.58076 47.41751 67.11706 60.05783 57.47855
## 4 297.92714 348.51372 293.91844 325.37360 484.62060 306.08016 342.73309
## 5 49.35154 44.92123 44.63028 43.15576 55.31168 46.98355 56.09067
## 6 49.51952 52.83570 51.37682 44.92182 71.10327 64.49127 62.79066
##   AUTO_TMP AUTO_AVG ES27_AR ES27_ADJ ES27_EPI ES27_TMP ES27_AVG
## 1 50.04248 42.45901 61.79601 57.30286 62.00075 63.49449 65.02397
## 2 162.11291 177.33227 187.73436 175.68197 193.36014 187.62784 181.94556
## 3 68.25641 57.05509 68.26235 70.37725 68.32286 68.97580 66.19615
## 4 498.37785 353.67732 536.84721 528.65652 537.78179 561.87951 567.14627
## 5 56.14517 45.05636 56.57592 53.72223 55.78747 56.60697 52.44099
## 6 69.02256 62.40533 83.53432 62.22517 83.30342 88.39400 61.55655
##   ES64_AR ES64_ADJ ES64_EPI ES64_TMP ES64_AVG Best_Result
## 1 69.27650 68.13969 69.50792 70.84880 71.31850 AUTO_ADJ_LB
## 2 199.30810 179.19030 200.95464 199.88634 181.99879 ES64_ADJ_LB
## 3 73.05803 75.95411 72.30468 76.37572 72.01390 ES64_ADJ_LB
## 4 580.63156 582.91973 579.32705 576.78219 633.52470 AUTO_TMP_LB
## 5 55.62434 53.71907 54.88341 55.41115 53.07662 ES27_AVG_LB
## 6 100.87372 70.62766 100.98942 104.22873 66.36596 ES27_AVG_LB
##
##           Model_Type
## 1 With log-back transformation
## 2 With log-back transformation
## 3 With log-back transformation
## 4 With log-back transformation
## 5 With log-back transformation
## 6 With log-back transformation

```

Now, let's plot the best models by each state for each forecast horizon (1-4 weeks ahead).

```

# define colors for the plot
custom_colors <- c(
  "AUTO_AR_LB" = colors_ar[1], "ES27_AR_LB" = colors_ar[1], "ES64_AR_LB" = colors_ar[1],
  "AUTO_EPI_LB" = colors_epi[1], "ES27_EPI_LB" = colors_epi[1], "ES64_EPI_LB" = colors_epi[1],
  "AUTO_TMP_LB" = colors_tmp[1], "ES27_TMP_LB" = colors_tmp[1], "ES64_TMP_LB" = colors_tmp[1],
  "AUTO_ADJ_LB" = colors_adj[1], "ES27_ADJ_LB" = colors_adj[1], "ES64_ADJ_LB" = colors_adj[1],
  "AUTO_AVG_LB" = colors_avg[1], "ES27_AVG_LB" = colors_avg[1], "ES64_AVG_LB" = colors_avg[1],

  "AUTO_AR" = colors_ar[1], "ES27_AR" = colors_ar[1], "ES64_AR" = colors_ar[1],
  "AUTO_EPI" = colors_epi[1], "ES27_EPI" = colors_epi[1], "ES64_EPI" = colors_epi[1],
  "AUTO_TMP" = colors_tmp[1], "ES27_TMP" = colors_tmp[1], "ES64_TMP" = colors_tmp[1],
  "AUTO_ADJ" = colors_adj[1], "ES27_ADJ" = colors_adj[1], "ES64_ADJ" = colors_adj[1],
  "AUTO_AVG" = colors_avg[1], "ES27_AVG" = colors_avg[1], "ES64_AVG" = colors_avg[1]
)

# define patterns for the plot
custom_patterns <- c(
  # AR
  "AUTO_AR"      = "circle",
  "AUTO_AR_LB"    = "circle",

```

```

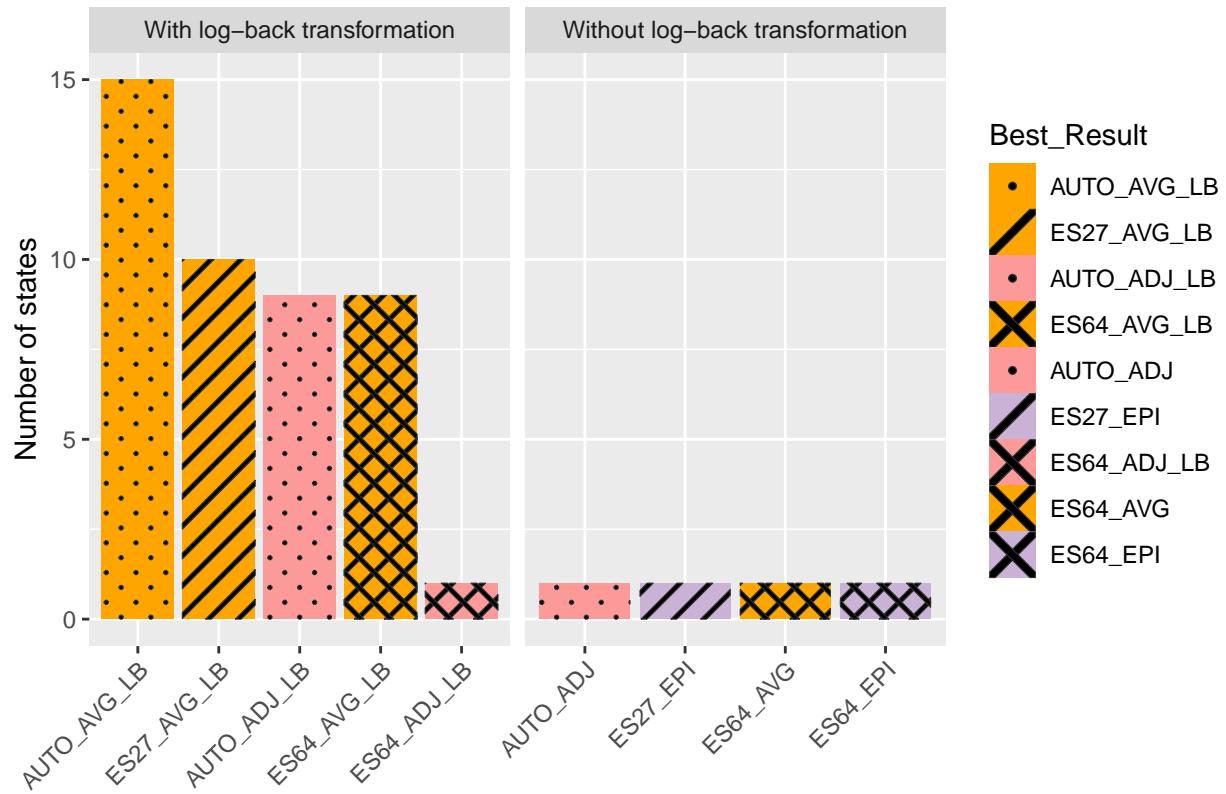
"ES27_AR"      = "stripe",      "ES27_AR_LB"      = "stripe",
"ES64_AR"      = "crosshatch",   "ES64_AR_LB"      = "crosshatch",
# EPI
"AUTO_EPI"     = "circle",      "AUTO_EPI_LB"     = "circle",
"ES27_EPI"     = "stripe",      "ES27_EPI_LB"     = "stripe",
"ES64_EPI"     = "crosshatch",   "ES64_EPI_LB"     = "crosshatch",
# TMP
"AUTO_TMP"     = "circle",      "AUTO_TMP_LB"     = "circle",
"ES27_TMP"     = "stripe",      "ES27_TMP_LB"     = "stripe",
"ES64_TMP"     = "crosshatch",   "ES64_TMP_LB"     = "crosshatch",
# ADJ
"AUTO_ADJ"     = "circle",      "AUTO_ADJ_LB"     = "circle",
"ES27_ADJ"     = "stripe",      "ES27_ADJ_LB"     = "stripe",
"ES64_ADJ"     = "crosshatch",   "ES64_ADJ_LB"     = "crosshatch",
# AVG
"AUTO_AVG"     = "circle",      "AUTO_AVG_LB"     = "circle",
"ES27_AVG"     = "stripe",      "ES27_AVG_LB"     = "stripe",
"ES64_AVG"     = "crosshatch",   "ES64_AVG_LB"     = "crosshatch"
)

plot1 <- ggplot(W1, aes(x = Best_Result,
                        pattern = Best_Result,
                        fill     = Best_Result)) +
  geom_bar_pattern(
    colour      = NA,
    pattern_fill = "black",
    pattern_size = 0.2,
    pattern_angle = 45
  ) +
  scale_fill_manual(values = custom_colors) +
  scale_pattern_manual(values = custom_patterns) +
  labs(title = "Best model on the 48 U.S. states (W1)",
       x = NULL, y = "Number of states") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  facet_wrap(~ Model_Type, scales = "free_x")

plot1

```

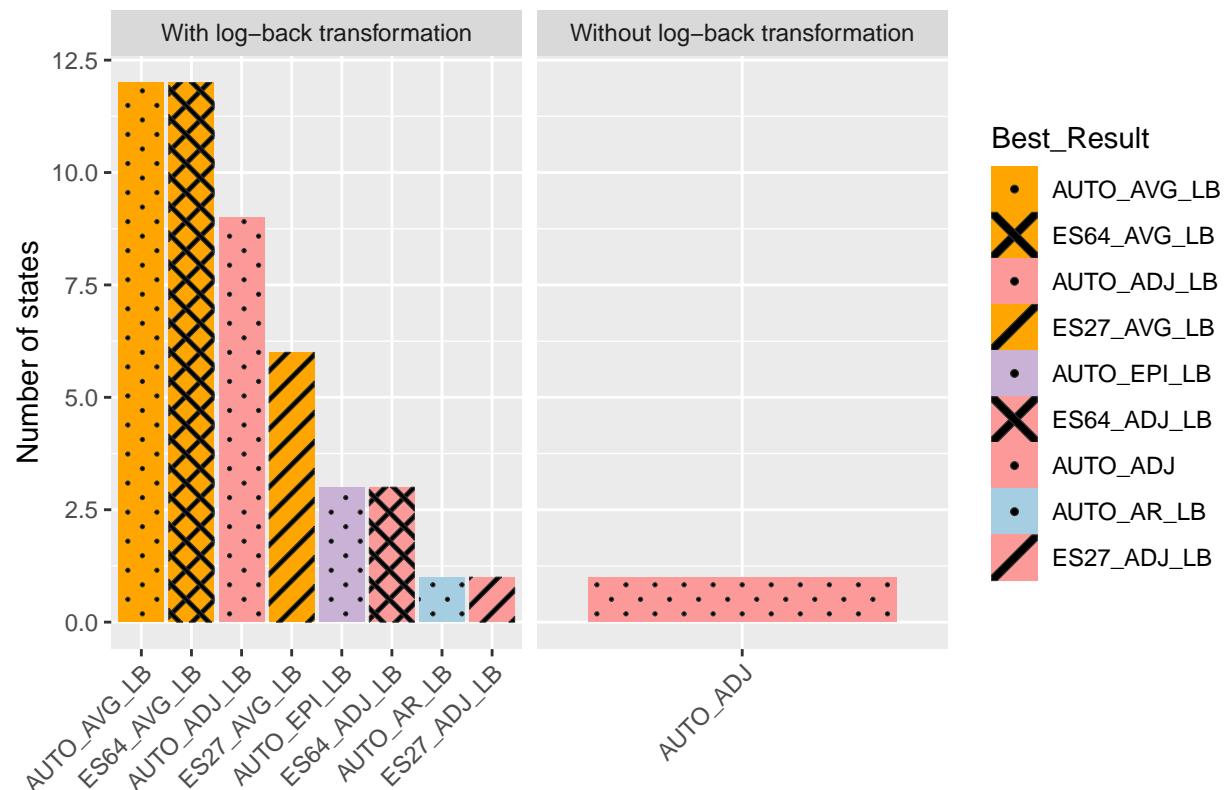
## Best model on the 48 U.S. states (W1)



```
plot2 <- ggplot(W2, aes(x = Best_Result,
                         pattern = Best_Result,
                         fill     = Best_Result)) +
  geom_bar_pattern(
    colour      = NA,
    pattern_fill = "black",
    pattern_size = 0.2,
    pattern_angle = 45
  ) +
  scale_fill_manual(values = custom_colors) +
  scale_pattern_manual(values = custom_patterns) +
  labs(title = "Best model on the 48 U.S. states (W2)",
       x = NULL, y = "Number of states") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  facet_wrap(~ Model_Type, scales = "free_x")

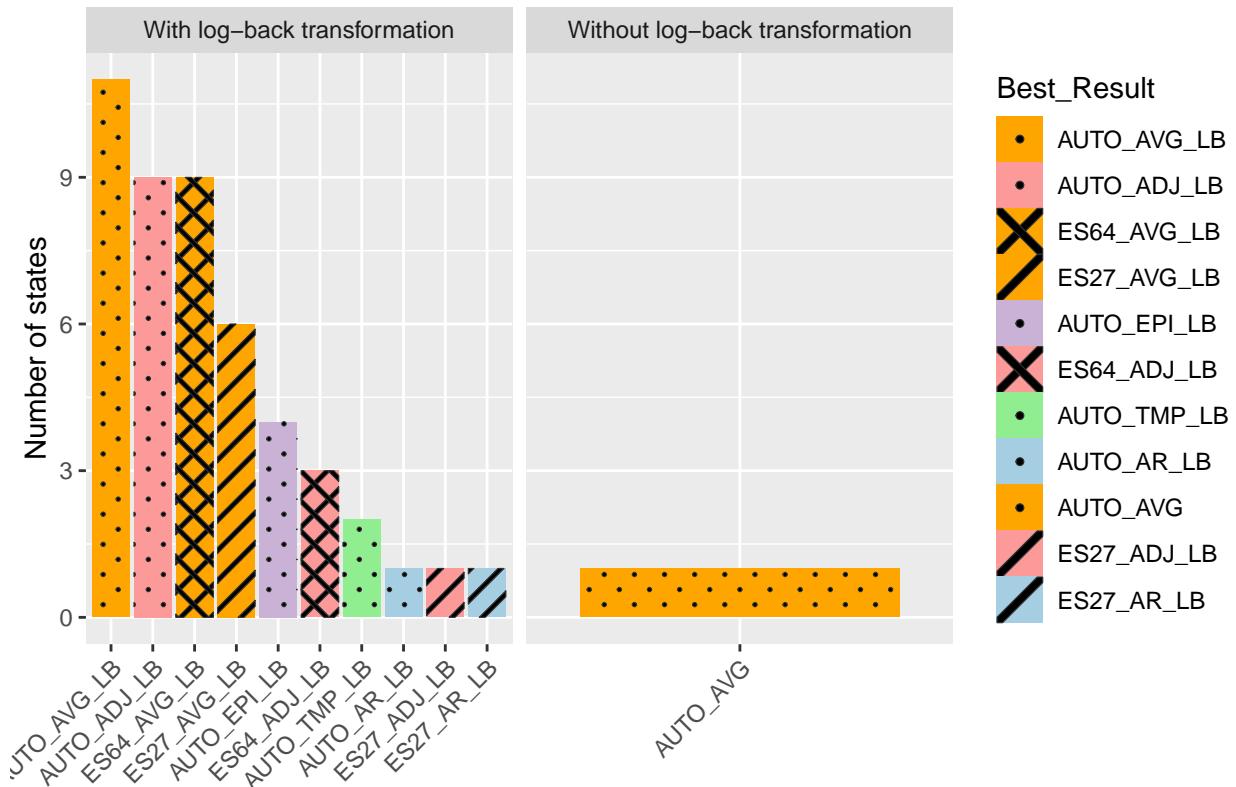
plot2
```

## Best model on the 48 U.S. states (W2)



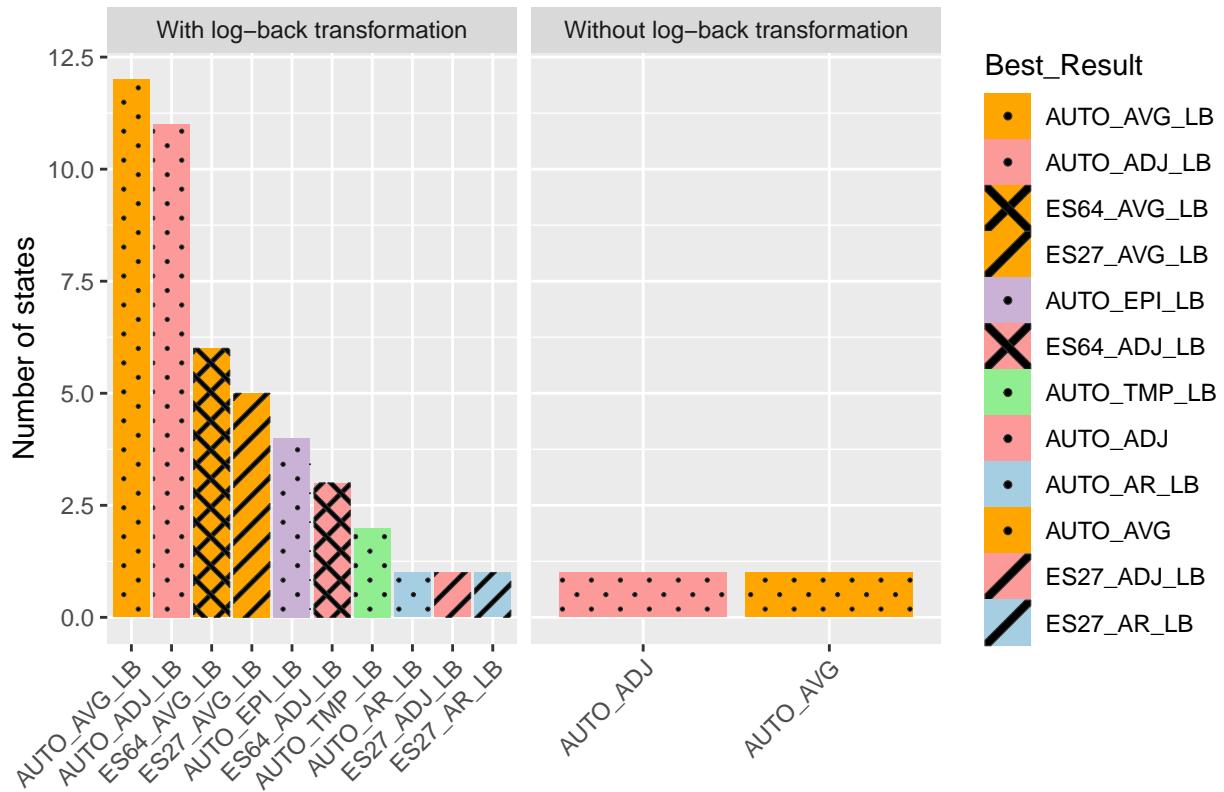
```
plot3 <- ggplot(W3, aes(x = Best_Result,
                        pattern = Best_Result,
                        fill      = Best_Result)) +
  geom_bar_pattern(
    colour      = NA,
    pattern_fill = "black",
    pattern_size = 0.2,
    pattern_angle = 45
  ) +
  scale_fill_manual(values = custom_colors) +
  scale_pattern_manual(values = custom_patterns) +
  labs(title = "Best model on the 48 U.S. states (W3)",
       x = NULL, y = "Number of states") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  facet_wrap(~ Model_Type, scales = "free_x")
plot3
```

## Best model on the 48 U.S. states (W3)



```
plot4 <- ggplot(W4, aes(x = Best_Result,
                         pattern = Best_Result,
                         fill      = Best_Result)) +
  geom_bar_pattern(
    colour      = NA,
    pattern_fill = "black",
    pattern_size = 0.2,
    pattern_angle = 45
  ) +
  scale_fill_manual(values = custom_colors) +
  scale_pattern_manual(values = custom_patterns) +
  labs(title = "Best model on the 48 U.S. states (W4)",
       x = NULL, y = "Number of states") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  facet_wrap(~ Model_Type, scales = "free_x")
plot4
```

## Best model on the 48 U.S. states (W4)



Now I will combine all these plots 2x2.

```
# Include columns with target week labels
W1$Week_Ahead <- "W1"
W2$Week_Ahead <- "W2"
W3$Week_Ahead <- "W3"
W4$Week_Ahead <- "W4"

# Combine all dataframes
all_weeks <- bind_rows(W1, W2, W3, W4)

# Create combined plot
combined_plot <- ggplot(all_weeks, aes(x = Best_Result,
                                         fill = Best_Result,
                                         pattern = Best_Result)) +
  geom_bar_pattern(
    pattern_fill = "black",
    pattern_size = 0.1,
    pattern_angle = 45,
    colour       = NA
  ) +
  scale_fill_manual(values = custom_colors) +
  scale_pattern_manual(values = custom_patterns) +
  labs(title = "Best model based on the lowest mean WIS",
       subtitle = "Results for the 48 states on the contiguous U.S. by target week",
       x = "", y = "Number of states",
```

```

    fill = "Best model",
    pattern = "Best model") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1),
        plot.title = element_text(size = 16, face = "bold"), legend.position = "left") +
  facet_grid(Week_Ahead ~ Model_Type, scales = "free_x")

# Save and display the plot
#ggsave("Fig4.jpg", combined_plot, width = 7, height = 6)
print(combined_plot)

```

Best model



## Best model based on the lowest mean WIS

Results for the 48 states on the contiguous U.S. by target week



These legends from ggplot don't work well. I will create an legend for a random plot and I will use this legend in the previous plot.

```

# Define the custom patterns
custom_patterns <- c("AUTO" = "circle", "ES27" = "stripe", "ES64" = "crosshatch")
custom_fills <- c("AUTO" = "white", "ES27" = "white", "ES64" = "white")

# Create a example of plot
legend_data <- data.frame(
  Model_Type = c("AUTO", "ES27", "ES64", "ES27", "ES64"),
  Model_Family = c("AR", "TMP", "EPI", "ADJ", "AVG"),
  x = 1:5,
  y = 1
)

# Define patterns for model types

```

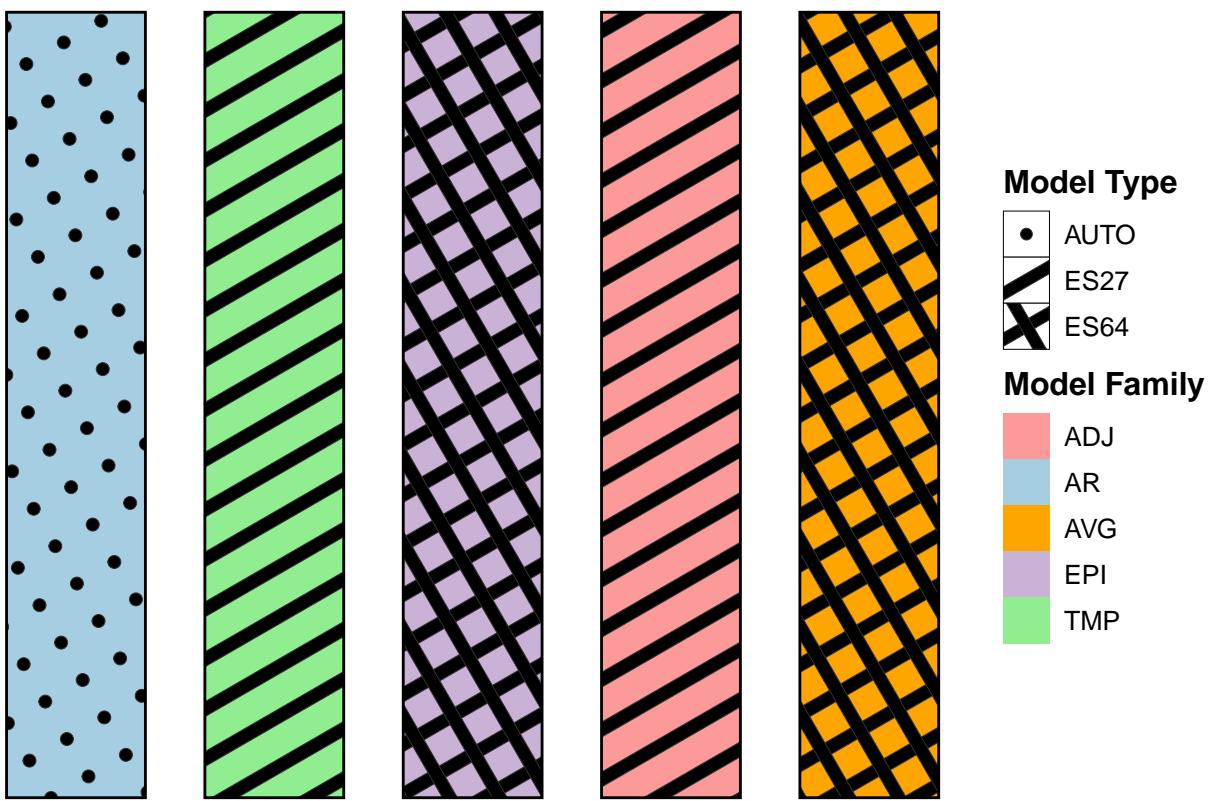
```

custom_patterns <- c("AUTO" = "circle", "ES27" = "stripe", "ES64" = "crosshatch")

# Define colors
custom_fills <- c("AR" = colors_ar[1], "TMP"=colors_tmp[1], "EPI" = colors_epi[1], "ADJ" = colors_adj[1]

# Visualizing an example
custom_legend <- ggplot(legend_data, aes(x = factor(x), y =
                                         pattern = Model_Type, fill = Model_Family)) +
  geom_bar_pattern(stat = "identity",
                   pattern_fill = "black",
                   pattern_density = 0.3,
                   pattern_spacing = 0.05,
                   pattern_size = 0.15,
                   colour = "black",
                   width = 0.7) +
  scale_fill_manual(values = custom_fills, name = "Model Family") +
  scale_pattern_manual(values = custom_patterns, name = "Model Type") +
  theme_void() +
  theme(
    legend.position = "right",
    legend.title = element_text(size = 12, face = "bold"),
    legend.text = element_text(size = 10)
  ) +
  guides(
    fill = guide_legend(
      override.aes = list(pattern = "none", pattern_fill = NA, colour = NA)
    ),
    pattern = guide_legend(
      override.aes = list(fill = "white", pattern_fill = "black")
    )
  )
custom_legend

```



Here is the final plot with custom legend.

```
# Remove legends from initial plot
combined_plot_clean <- combined_plot + theme(legend.position = "none")

# Extract only the custom legend
legend_only <- cowplot::get_legend(custom_legend)

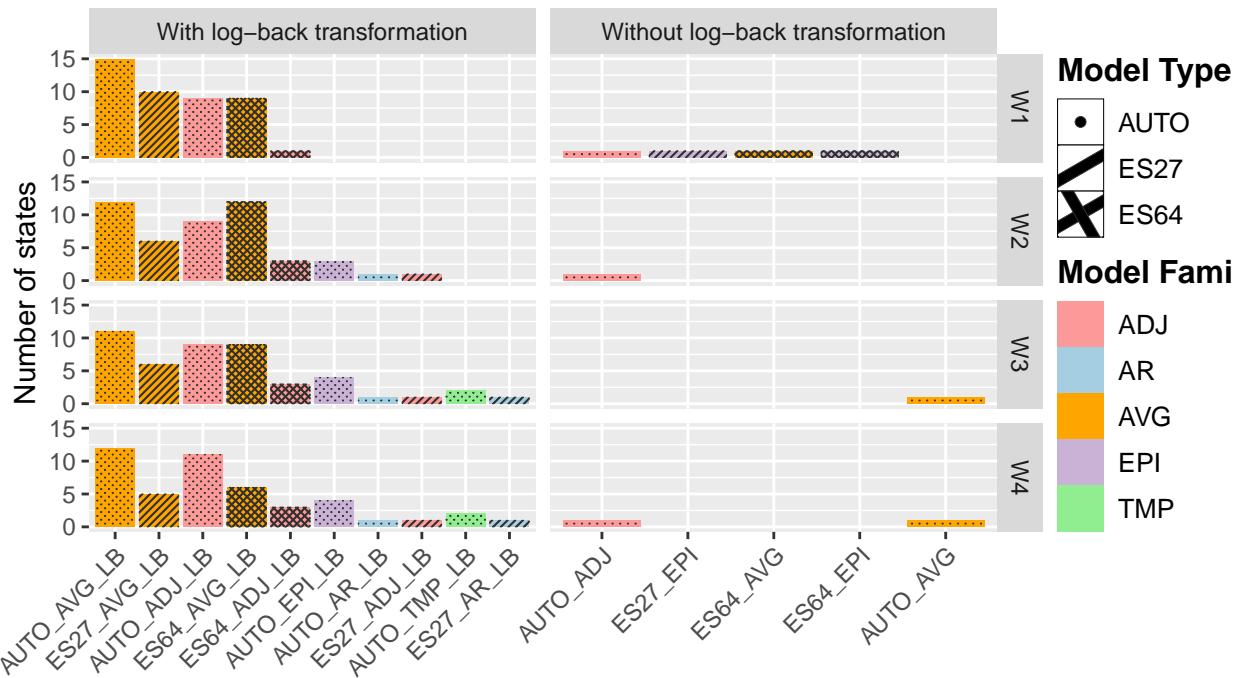
## Warning in get_plot_component(plot, "guide-box"): Multiple components found;
## returning the first one. To return all, use 'return_all = TRUE'.

# Combine main plot with the custom legend
figure4 <- combined_plot_clean +
  patchwork::plot_spacer() +
  patchwork::wrap_elements(legend_only) +
  plot_layout(widths = c(4, 0.05, 0.4))

# Display the result
figure4
```

## Best model based on the lowest mean WIS

Results for the 48 states on the contiguous U.S. by target week



```
ggsave("Fig4.jpg", figure4, width = 10, height = 6)
```

Now we use a Wilcoxon test with Holm p-value adjustment for repeated comparisons to evaluate the differences in model performances.

1 WEEK AHEAD - Wilcoxon test with Holm p-value adjustment

```
# Get all models names except baseline
model_types <- setdiff(names(W1), c("STATE", "AUTO_AR", "Best_Result", "Model_Type", "Week_Ahead"))

# Perform Wilcoxon test for each model type against baseline
wilcox_results1 <- map_df(model_types, function(model) {
  test <- wilcox.test(W1[[model]], W1$AUTO_AR, paired = TRUE)
  data.frame(Model = model, p_value = test$p.value, W = test$statistic)
})

# p values adjustment
p_values<-wilcox_results1$p_value
p_values<-p.adjust(p_values, method = "holm")
p_values<-data.frame(p_values)

#####
p_values_wk1 <- data.frame(Model = model_types, PValue = p_values, WeekAhead=1)
p_values_wk1 <-drop_na(p_values_wk1)
p_values_wk1
```

```

##          Model      p_values WeekAhead
## 1    AUTO_AR_LB 9.231208e-07      1
## 2    AUTO_ADJ_LB 3.482370e-11      1
## 3    AUTO_EPI_LB 2.567875e-07      1
## 4    AUTO_TMP_LB 2.507425e-04      1
## 5    AUTO_AVG_LB 1.809289e-08      1
## 6    ES27_AR_LB 1.044917e-04      1
## 7    ES27_ADJ_LB 2.257359e-08      1
## 8    ES27_EPI_LB 4.817238e-05      1
## 9    ES27_TMP_LB 1.804422e-02      1
## 10   ES27_AVG_LB 2.101837e-08      1
## 11   ES64_AR_LB 6.763713e-05      1
## 12   ES64_ADJ_LB 5.091761e-08      1
## 13   ES64_EPI_LB 5.546505e-05      1
## 14   ES64_TMP_LB 6.011928e-03      1
## 15   ES64_AVG_LB 1.952345e-08      1
## 16    AUTO_ADJ 1.000000e+00      1
## 17    AUTO_EPI 1.532974e-02      1
## 18    AUTO_TMP 1.000000e+00      1
## 19    AUTO_AVG 1.161461e-01      1
## 20    ES27_AR 6.422568e-01      1
## 21    ES27_ADJ 1.000000e+00      1
## 22    ES27_EPI 1.000000e+00      1
## 23    ES27_TMP 1.000000e+00      1
## 24    ES27_AVG 1.000000e+00      1
## 25    ES64_AR 1.161461e-01      1
## 26    ES64_ADJ 1.161461e-01      1
## 27    ES64_EPI 3.663236e-02      1
## 28    ES64_TMP 4.741434e-05      1
## 29    ES64_AVG 1.000000e+00      1

```

2 WEEKS AHEAD - Wilcoxon test with Holm p-value adjustment

```

# Get all models names except baseline
model_types <- setdiff(names(W2), c("STATE", "AUTO_AR", "Best_Result", "Model_Type", "Week_Ahead"))

# Perform Wilcoxon test for each model type against baseline
wilcox_results2 <- map_df(model_types, function(model) {
  test <- wilcox.test(W2[[model]], W2$AUTO_AR, paired = TRUE)
  data.frame(Model = model, p_value = test$p.value, W = test$statistic)
})

# p values adjustment
p_values<-wilcox_results2$p_value
p_values<-p.adjust(p_values, method = "holm")
p_values<-data.frame(p_values)

#####
p_values_wk2 <- data.frame(Model = model_types, PValue = p_values, WeekAhead=2)
p_values_wk2 <- drop_na(p_values_wk2)
p_values_wk2

```

```

##          Model      p_values WeekAhead

```

```

## 1    AUTO_AR_LB 4.263256e-12      2
## 2    AUTO_ADJ_LB 5.542233e-13     2
## 3    AUTO_EPI_LB 5.542233e-13     2
## 4    AUTO_TMP_LB 2.594831e-10     2
## 5    AUTO_AVG_LB 2.060574e-13     2
## 6    ES27_AR_LB 7.335643e-09      2
## 7    ES27_ADJ_LB 1.045919e-10     2
## 8    ES27_EPI_LB 2.586795e-09      2
## 9    ES27_TMP_LB 7.335643e-09      2
## 10   ES27_AVG_LB 2.060574e-13     2
## 11   ES64_AR_LB 3.571415e-09      2
## 12   ES64_ADJ_LB 1.045919e-10     2
## 13   ES64_EPI_LB 2.891909e-10     2
## 14   ES64_TMP_LB 3.814620e-09      2
## 15   ES64_AVG_LB 2.060574e-13     2
## 16   AUTO_ADJ 5.094992e-03       2
## 17   AUTO_EPI 5.430881e-04       2
## 18   AUTO_TMP 1.000000e+00       2
## 19   AUTO_AVG 2.629954e-04       2
## 20   ES27_AR 1.000000e+00       2
## 21   ES27_ADJ 1.000000e+00       2
## 22   ES27_EPI 1.000000e+00       2
## 23   ES27_TMP 1.000000e+00       2
## 24   ES27_AVG 1.000000e+00       2
## 25   ES64_AR 2.629954e-04       2
## 26   ES64_ADJ 4.419932e-01       2
## 27   ES64_EPI 1.363122e-04       2
## 28   ES64_TMP 2.131640e-05       2
## 29   ES64_AVG 1.000000e+00       2

```

3 WEEKS AHEAD - Wilcoxon test with Holm p-value adjustment

```

# Get all models names except baseline
model_types <- setdiff(names(W3), c("STATE", "AUTO_AR", "Best_Result", "Model_Type", "Week_Ahead"))

# Perform Wilcoxon test for each model type against baseline
wilcox_results3 <- map_df(model_types, function(model) {
  test <- wilcox.test(W3[[model]], W3$AUTO_AR, paired = TRUE)
  data.frame(Model = model, p_value = test$p.value, W = test$statistic)
})

# p values adjustment
p_values<-wilcox_results3$p_value
p_values<-p.adjust(p_values, method = "holm")
p_values<-data.frame(p_values)

#####
p_values_wk3 <- data.frame(Model = model_types, PValue = p_values, WeekAhead=3)
p_values_wk3 <-drop_na(p_values_wk3)
p_values_wk3

##          Model      p_values WeekAhead
## 1    AUTO_AR_LB 2.060574e-13      3

```

```

## 2 AUTO_ADJ_LB 2.060574e-13      3
## 3 AUTO_EPI_LB 3.552714e-13      3
## 4 AUTO_TMP_LB 5.393019e-12      3
## 5 AUTO_AVG_LB 5.115908e-13      3
## 6 ES27_AR_LB 2.458123e-10      3
## 7 ES27_ADJ_LB 2.141576e-11      3
## 8 ES27_EPI_LB 7.505605e-10      3
## 9 ES27_TMP_LB 8.083134e-10      3
## 10 ES27_AVG_LB 2.060574e-13     3
## 11 ES64_AR_LB 2.224141e-10     3
## 12 ES64_ADJ_LB 2.141576e-11     3
## 13 ES64_EPI_LB 1.082867e-10     3
## 14 ES64_TMP_LB 1.702389e-10     3
## 15 ES64_AVG_LB 2.060574e-13     3
## 16 AUTO_ADJ 1.318634e-03       3
## 17 AUTO_EPI 1.453191e-04       3
## 18 AUTO_TMP 1.000000e+00       3
## 19 AUTO_AVG 8.570441e-04       3
## 20 ES27_AR 1.000000e+00       3
## 21 ES27_ADJ 1.000000e+00       3
## 22 ES27_EPI 1.000000e+00       3
## 23 ES27_TMP 1.000000e+00       3
## 24 ES27_AVG 1.000000e+00       3
## 25 ES64_AR 1.145431e-06       3
## 26 ES64_ADJ 1.626837e-01       3
## 27 ES64_EPI 1.145242e-06       3
## 28 ES64_TMP 9.779156e-08       3
## 29 ES64_AVG 1.000000e+00       3

```

4 WEEKS AHEAD - Wilcoxon test with Holm p-value adjustment

```

# Get all models names except baseline
model_types <- setdiff(names(W4), c("STATE", "AUTO_AR", "Best_Result", "Model_Type", "Week_Ahead"))

# Perform Wilcoxon test for each model type against baseline
wilcox_results4 <- map_df(model_types, function(model) {
  test <- wilcox.test(W4[[model]], W4$AUTO_AR, paired = TRUE)
  data.frame(Model = model, p_value = test$p.value, W = test$statistic)
})

# Perform Wilcoxon test for each model type against AUTO_AAR
p_values<-wilcox_results4$p_value
p_values<-p.adjust(p_values, method = "holm")
p_values<-data.frame(p_values)

#####
p_values_wk4 <- data.frame(Model = model_types, PValue = p_values, WeekAhead=4)
p_values_wk4 <-drop_na(p_values_wk4)
p_values_wk4

##          Model      p_values WeekAhead
## 1    AUTO_AR_LB 2.060574e-13        4
## 2    AUTO_ADJ_LB 1.293188e-12        4

```

```

## 3 AUTO_EPI_LB 1.797673e-11      4
## 4 AUTO_TMP_LB 7.617018e-11      4
## 5 AUTO_AVG_LB 8.185452e-11      4
## 6 ES27_AR_LB 1.012994e-07      4
## 7 ES27_ADJ_LB 4.263256e-12      4
## 8 ES27_EPI_LB 3.365841e-09      4
## 9 ES27_TMP_LB 2.970194e-08      4
## 10 ES27_AVG_LB 9.592327e-13     4
## 11 ES64_AR_LB 3.387709e-08      4
## 12 ES64_ADJ_LB 1.776357e-12     4
## 13 ES64_EPI_LB 3.174591e-09     4
## 14 ES64_TMP_LB 7.974705e-10     4
## 15 ES64_AVG_LB 3.979039e-13     4
## 16 AUTO_ADJ 2.799617e-05        4
## 17 AUTO_EPI 9.690508e-07        4
## 18 AUTO_TMP 5.914043e-01        4
## 19 AUTO_AVG 1.982266e-05        4
## 20 ES27_AR 2.645923e-02        4
## 21 ES27_ADJ 1.000000e+00        4
## 22 ES27_EPI 4.762452e-02        4
## 23 ES27_TMP 7.764381e-03        4
## 24 ES27_AVG 1.000000e+00        4
## 25 ES64_AR 7.617018e-11        4
## 26 ES64_ADJ 3.263910e-01        4
## 27 ES64_EPI 6.669865e-11        4
## 28 ES64_TMP 2.141576e-11        4
## 29 ES64_AVG 1.000000e+00        4

```

Let's calculate the mean(WIS) improvement relative to the AUTO ARIMA model. We will compare each model with the AUTO ARIMA model for the same states. Later we sum the results of these comparisons. Negative results indicate that there was a general improvement in the mean(WIS) for a given model type among all states.

```

calculate_percentage_of_improvement <- function(data) {
  return(data.frame(
    AUTO_AR = (((data$AUTO_AR / data$AUTO_AR)-1) * 100),
    ES27_AR = (((data$ES27_AR/data$AUTO_AR)-1) * 100),
    ES64_AR = (((data$ES64_AR/data$AUTO_AR )-1) * 100),

    AUTO_ADJ = (((data$AUTO_ADJ/data$AUTO_AR)-1) * 100),
    ES27_ADJ = (((data$ES27_ADJ/data$AUTO_AR)-1) * 100),
    ES64_ADJ = (((data$ES64_ADJ/data$AUTO_AR)-1) * 100),

    AUTO_TMP = (((data$AUTO_TMP/data$AUTO_AR)-1) * 100),
    ES27_TMP = (((data$ES27_TMP/data$AUTO_AR)-1) * 100),
    ES64_TMP = (((data$ES64_TMP/data$AUTO_AR)-1) * 100),

    AUTO_EPI = (((data$AUTO_EPI/data$AUTO_AR)-1) * 100),
    ES27_EPI = (((data$ES27_EPI/data$AUTO_AR)-1) * 100),
    ES64_EPI = (((data$ES64_EPI/data$AUTO_AR)-1) * 100),

    AUTO_AVG = (((data$AUTO_AVG/data$AUTO_AR)-1) * 100),
    ES27_AVG = (((data$ES27_AVG/data$AUTO_AR)-1) * 100),
    ES64_AVG = (((data$ES64_AVG/data$AUTO_AR)-1) * 100),
  ))
}

```

```

AUTO_AR_LB = (((data$AUTO_AR_LB/data$AUTO_AR)-1) * 100),
ES27_AR_LB = (((data$ES27_AR_LB/data$AUTO_AR)-1) * 100),
ES64_AR_LB = (((data$ES64_AR_LB/data$AUTO_AR)-1) * 100),

AUTO_ADJ_LB = (((data$AUTO_ADJ_LB/data$AUTO_AR)-1) * 100),
ES27_ADJ_LB = (((data$ES27_ADJ_LB/data$AUTO_AR)-1) * 100),
ES64_ADJ_LB = (((data$ES64_ADJ_LB/data$AUTO_AR)-1) * 100),

AUTO_TMP_LB = (((data$AUTO_TMP_LB/data$AUTO_AR)-1) * 100),
ES27_TMP_LB = (((data$ES27_TMP_LB/data$AUTO_AR)-1) * 100),
ES64_TMP_LB = (((data$ES64_TMP_LB/data$AUTO_AR)-1) * 100),

AUTO_EPI_LB = (((data$AUTO_EPI_LB/data$AUTO_AR)-1) * 100),
ES27_EPI_LB = (((data$ES27_EPI_LB/data$AUTO_AR)-1) * 100),
ES64_EPI_LB = (((data$ES64_EPI_LB/data$AUTO_AR)-1) * 100),

AUTO_AVG_LB = (((data$AUTO_AVG_LB/data$AUTO_AR)-1) * 100),
ES27_AVG_LB = (((data$ES27_AVG_LB/data$AUTO_AR)-1) * 100),
ES64_AVG_LB = (((data$ES64_AVG_LB/data$AUTO_AR)-1) * 100)
))

}

# Calculate percentage of improvement
W1_percentage_of_improvement <- calculate_percentage_of_improvement(W1)
W2_percentage_of_improvement <- calculate_percentage_of_improvement(W2)
W3_percentage_of_improvement <- calculate_percentage_of_improvement(W3)
W4_percentage_of_improvement <- calculate_percentage_of_improvement(W4)

head(W1_percentage_of_improvement)

##    AUTO_AR    ES27_AR    ES64_AR    AUTO_ADJ   ES27_ADJ   ES64_ADJ    AUTO_TMP
## 1      0 -0.425261  3.249636 -18.580514 -9.008468 -1.832247  6.670445
## 2      0  3.004647  6.003209 -4.674881 -9.163326 -7.941074 -2.596000
## 3      0 -1.625961  6.103246  1.493953 13.091159 21.037418  6.860865
## 4      0 -3.669779  2.931193 -19.792816 -1.804717  7.138679 -3.562265
## 5      0 -4.226605 -5.304692 -4.426017 -2.225894 -2.618824  3.087349
## 6      0  5.415933 11.056141 10.263731  1.535857 13.387209  1.859511
##    ES27_TMP  ES64_TMP  AUTO_EPI    ES27_EPI  ES64_EPI    AUTO_AVG  ES27_AVG
## 1  2.1334807  4.851158  1.652310 -0.06985519  1.735219 -18.195002 -2.858213
## 2  0.2407755  4.579908  4.706856  4.16173010  7.245755  6.205750  2.969020
## 3  1.1929343 13.525039 -1.432388 -1.27474377  6.612140  1.307974 10.094602
## 4 -3.6540744  4.115855 -7.412860 -2.03819125  3.880569 19.467884 14.165173
## 5 -0.8173495 -2.119214 -4.015449 -8.44371205 -9.364204 -0.847198 -1.995320
## 6  6.8118691 17.964388 -3.280627  6.44475710 11.949480 -10.914737 -12.108253
##    ES64_AVG AUTO_AR_LB  ES27_AR_LB  ES64_AR_LB  AUTO_ADJ_LB  ES27_ADJ_LB
## 1  0.6823754 -9.760318 -12.4551532 -11.874029 -21.467910 -19.167943
## 2  5.4667044  1.959505 -0.3508059 -4.571610 -21.200352 -18.462948
## 3 20.0210515 -7.134463  0.9186063 -1.238116 -21.078765 -18.990555
## 4 11.9609611 -30.037211 -23.1999627 -23.552961 -30.537890 -28.826495
## 5 -0.7093309 -7.067280 -3.9192809 -8.064414 -1.621618 -8.592726
## 6 -9.7017097 -20.135752 -17.9269656 -20.165452 -34.342624 -26.296486
##    ES64_ADJ_LB AUTO_TMP_LB  ES27_TMP_LB  ES64_TMP_LB  AUTO_EPI_LB  ES27_EPI_LB
## 1 -18.055461   0.6765080   -8.216595   -7.614982   -5.999171 -10.4642322

```

```

## 2 -18.568689 2.0047833 -2.801332 -3.284220 1.383757 1.0461390
## 3 -21.376380 -0.2205393 3.141167 1.841462 -8.605417 -0.9285115
## 4 -24.157275 -18.8707296 -17.235020 -17.582404 -18.549317 -24.4556346
## 5 -7.453615 0.3857786 0.690887 -4.432802 -4.962318 -6.8906948
## 6 -22.710575 -21.2354299 -20.166577 -21.910300 -22.226851 -20.4765288
## ES64_EPI_LB AUTO_AVG_LB ES27_AVG_LB ES64_AVG_LB
## 1 -10.445696 -18.752183 -11.308977 -10.570787
## 2 2.221616 28.856007 25.502225 23.803936
## 3 -2.306907 -7.632462 -5.575696 -11.770385
## 4 -23.134105 -5.103359 -9.910694 -7.012507
## 5 -9.216949 1.631934 -8.450567 -8.186177
## 6 -20.214482 -36.992150 -34.129116 -29.184450

```

```
head(W2_percentage_of_improvement)
```

```

## AUTO_AR ES27_AR ES64_AR AUTO_ADJ ES27_ADJ ES64_ADJ AUTO_TMP
## 1 0 16.886258 23.735253 -16.7462534 7.876956 22.0115048 16.116006
## 2 0 5.908693 9.932897 3.6245881 -2.052938 -0.4631657 -2.683907
## 3 0 -1.600747 5.837064 0.2431436 11.682101 21.5062433 3.861168
## 4 0 -3.909505 3.735949 -32.7492101 -2.825492 5.3171280 -2.181561
## 5 0 -6.162255 -7.254021 -9.9925249 -4.391768 -4.6528217 1.476075
## 6 0 13.059910 24.391642 2.1720376 -1.569108 14.8134141 2.695759
## ES27_TMP ES64_TMP AUTO_EPI ES27_EPI ES64_EPI AUTO_AVG ES27_AVG
## 1 17.735758 24.605724 1.059166 17.5693125 24.672035 -9.077659 21.5251716
## 2 5.237373 10.318996 3.519885 8.0174240 10.875007 2.186168 0.6977928
## 3 0.5422777 12.327177 -2.445300 -0.9417194 5.729662 -2.840491 7.5257679
## 4 -2.876615 4.016296 -13.310595 -3.4059202 3.237016 5.079698 12.7688274
## 5 -4.685289 -6.218589 -2.119355 -10.3781430 -10.851485 -9.139501 -5.1163831
## 6 18.562990 27.760565 -6.591992 12.4738772 23.910432 -9.481741 -10.2394682
## ES64_AVG AUTO_AR_LB ES27_AR_LB ES64_AR_LB AUTO_ADJ_LB ES27_ADJ_LB
## 1 24.6390880 -13.42493 -20.255127 -20.677985 -28.667568 -26.23196
## 2 0.7686793 -22.84006 -28.243232 -30.863691 -35.776606 -35.34117
## 3 17.5998692 -19.14573 -5.071651 -8.539729 -26.268293 -24.20651
## 4 18.6583851 -35.99435 -33.725551 -33.320044 -37.454841 -38.32170
## 5 -3.3456903 -12.55823 -6.844298 -14.106241 -7.599204 -14.17176
## 6 -6.8179010 -19.15503 -17.857454 -23.515267 -35.190774 -30.01745
## ES64_ADJ_LB AUTO_TMP_LB ES27_TMP_LB ES64_TMP_LB AUTO_EPI_LB ES27_EPI_LB
## 1 -25.76201 -14.355347 -18.945261 -19.598433 -14.238871 -19.813860
## 2 -35.97425 -23.989978 -31.418975 -31.714492 -24.154486 -28.079974
## 3 -26.39022 -13.748461 -6.029218 -7.370336 -18.490207 -7.280689
## 4 -33.72009 -37.958398 -35.595637 -34.842218 -24.119037 -33.441039
## 5 -11.56099 -7.194834 -4.811023 -12.131293 -6.422775 -10.844605
## 6 -28.39194 -22.244998 -22.748296 -26.518446 -24.848920 -21.852739
## ES64_EPI_LB AUTO_AVG_LB ES27_AVG_LB ES64_AVG_LB
## 1 -20.719062 -26.596439 -20.26938 -20.86077
## 2 -28.460708 -5.295206 -10.32198 -13.71070
## 3 -8.961002 -21.241583 -16.34202 -21.57675
## 4 -30.223161 -22.568310 -23.68255 -24.13544
## 5 -14.243348 -14.973596 -22.86433 -21.12888
## 6 -24.673398 -38.298430 -38.81178 -36.61874

```

```
head(W3_percentage_of_improvement)
```

```

##    AUTO_AR   ES27_AR   ES64_AR   AUTO_ADJ   ES27_ADJ   ES64_ADJ   AUTO_TMP
## 1      0 36.849584 48.996357 -8.100518 23.221102 44.225871 18.7919535
## 2      0 8.447011 13.456310  9.672009  4.855258  6.602029 -2.6074731
## 3      0 -2.734258 1.525751 -2.413220  8.569726 15.085302  2.1103928
## 4      0 2.447269 8.125647 -32.486829  2.590408  9.507033 -0.9416265
## 5      0 -2.165188 -3.711248 -12.263692 -3.960037 -3.927639  1.8113846
## 6      0 14.961474 32.357065 -4.361521 -6.412204  9.092825  1.6276807
##    ES27_TMP   ES64_TMP   AUTO_EPI   ES27_EPI   ES64_EPI   AUTO_AVG   ES27_AVG
## 1 39.968653 51.867483 13.9258548 37.811505 50.3305780 1.259305 41.548820
## 2 7.954252 13.404469  0.8606105 11.283784 13.8449989 8.713778 7.033783
## 3 -1.614029 4.912206 -9.3577461 -2.318251 0.3626149 -7.926111 4.380391
## 4 4.991289 9.742016 -24.7552354 3.083538 7.9715196 -13.935387 15.786719
## 5 -1.111190 -3.377426 -1.8896608 -4.773180 -6.0082418 -13.916207 -5.390076
## 6 19.875792 38.014750 -7.9128899 14.829057 32.4166579 -5.899913 -10.373238
##    ES64_AVG AUTO_AR_LB ES27_AR_LB ES64_AR_LB AUTO_ADJ_LB ES27_ADJ_LB
## 1 50.376786 -1.81246 -13.715639 -14.14400 -25.56681 -21.24105
## 2 6.449045 -23.96522 -28.000488 -33.12297 -33.20178 -33.92074
## 3 13.242659 -23.51601 -7.707364 -13.83233 -27.94751 -26.86420
## 4 21.875306 -34.22555 -35.496725 -33.41736 -39.60826 -41.65039
## 5 -3.627689 -19.54089 -10.039828 -20.39523 -9.91318 -15.89320
## 6 -1.794932 -18.09986 -18.394228 -24.64698 -32.91384 -30.49871
##    ES64_ADJ_LB AUTO_TMP_LB ES27_TMP_LB ES64_TMP_LB AUTO_EPI_LB ES27_EPI_LB
## 1 -20.57079 1.502706 -11.818499 -12.28165 -4.70798 -13.44033
## 2 -35.21438 -25.215557 -30.928686 -33.33329 -25.29943 -31.22497
## 3 -30.67817 -20.093120 -11.183735 -14.23669 -25.76362 -11.34249
## 4 -37.06525 -42.285099 -39.574863 -38.15836 -24.92641 -34.57062
## 5 -12.68369 -12.573484 -8.877297 -17.76536 -9.12391 -14.94860
## 6 -29.57964 -21.911361 -23.675909 -28.31915 -27.44769 -21.18160
##    ES64_EPI_LB AUTO_AVG_LB ES27_AVG_LB ES64_AVG_LB
## 1 -15.17965 -19.091334 -10.72650 -11.07119
## 2 -32.08938 -4.670541 -12.15188 -15.61854
## 3 -15.20259 -26.983450 -21.57887 -26.73948
## 4 -29.54138 -26.776158 -27.58283 -29.02424
## 5 -18.87743 -19.345108 -27.23233 -24.48337
## 6 -25.99083 -37.175166 -38.40215 -37.90679

```

```
head(W4_percentage_of_improvement)
```

```

##    AUTO_AR   ES27_AR   ES64_AR   AUTO_ADJ   ES27_ADJ   ES64_ADJ   AUTO_TMP
## 1      0 35.958209 52.4161443 -12.147244 26.072790 49.9150372 10.0991133
## 2      0 15.023357 22.1144913 10.228521  7.638951  9.7884796 -0.6747037
## 3      0 1.706412 8.8516560 -10.517790  4.857466 13.1666203  1.6975557
## 4      0 10.776804 19.8115734 -36.841281  9.086680 20.2837304  2.8387683
## 5      0 2.285655 0.5652681 -15.056744 -2.873634 -2.8793372  1.5068882
## 6      0 17.483084 41.8693070 -9.299149 -12.486211 -0.6689095 -2.9263236
##    ES27_TMP   ES64_TMP   AUTO_EPI   ES27_EPI   ES64_EPI   AUTO_AVG   ES27_AVG
## 1 39.695070 55.875384 8.4283592 36.4086787 52.9252831 -6.585378 43.060099
## 2 14.958090 22.468778 -0.9609937 18.4702242 23.1233149  8.650076 11.476605
## 3 2.769402 13.794797 -14.3607437 1.7965720 7.7292108 -14.991665 -1.372094
## 4 15.942143 19.017267 -29.2780605 10.9696509 19.5423906 -27.019750 17.028925
## 5 2.341794 0.179837 1.4083662 0.8601834 -0.7742941 -18.540964 -5.190029
## 6 24.317764 46.587810 -11.6909025 17.1583521 42.0320302 -12.232829 -13.426559
##    ES64_AVG AUTO_AR_LB ES27_AR_LB ES64_AR_LB AUTO_ADJ_LB ES27_ADJ_LB
## 1 56.908767 -4.453616 -15.519451 -15.01550 -26.368875 -22.59468

```

```

## 2 11.509213 -19.426802 -20.910643 -28.78639 -26.322729 -27.43172
## 3 7.295969 -25.051536 -3.770726 -13.48704 -28.796701 -27.26425
## 4 30.725912 -32.995729 -34.949730 -29.74503 -40.215340 -42.35886
## 5 -4.040856 -20.220938 -8.269317 -21.03047 -6.848562 -13.13581
## 6 -6.662587 -17.073097 -17.089943 -24.30351 -32.315266 -30.66866
##   ES64_ADJ_LB AUTO_TMP_LB ES27_TMP_LB ES64_TMP_LB AUTO_EPI_LB ES27_EPI_LB
## 1 -22.19959 0.4274503 -13.594633 -13.46879 -7.904662 -14.444049
## 2 -29.49882 -20.8594512 -25.902482 -28.46140 -20.608085 -25.778474
## 3 -33.45406 -22.5131931 -9.503486 -14.20845 -27.155383 -8.601425
## 4 -38.52363 -44.2879527 -40.951583 -39.35082 -25.326718 -34.785714
## 5 -10.77557 -12.8477760 -8.222805 -19.31130 -6.958636 -13.497728
## 6 -30.35551 -21.2005823 -22.125936 -27.74338 -26.004187 -19.611707
##   ES64_EPI_LB AUTO_AVG_LB ES27_AVG_LB ES64_AVG_LB
## 1 -15.04577 -19.080727 -9.969083 -9.661199
## 2 -26.69874 4.522644 -4.569552 -10.215176
## 3 -15.78271 -27.344704 -21.848298 -29.351035
## 4 -28.08524 -27.547033 -28.977762 -32.860139
## 5 -18.78527 -17.685658 -24.008530 -21.977128
## 6 -25.69160 -35.241742 -36.940366 -36.821729

```

Now let's plot histograms of percentage of WIS improvement for each state including mean and standard deviation of percentage of improvement compared to auto\_arima (baseline).

```

# List of models names
models_names <- c(names(W1_percentage_of_improvement))

# Create an empty dataframe to store results
summary_impr <- data.frame(WeekAhead = character(), Model = character(), m = numeric(), sd = numeric(),)

# List of datasets. One dataset for each target week.
datasets_list <- list("1" = W1_percentage_of_improvement,
                      "2" = W2_percentage_of_improvement,
                      "3" = W3_percentage_of_improvement,
                      "4" = W4_percentage_of_improvement)

# Here I have 2 for loops. One take into account the target week and the other the model.
# Loop through target weeks
for (target_week_ in names(datasets_list)) {
  dataset <- datasets_list[[target_week_]] # Get the dataset based on target week
  # Loop through models
  for (given_model in models_names) {
    data <- dataset[[given_model]]
    mean_val <- mean(data, na.rm = TRUE)
    sd_val <- sd(data, na.rm = TRUE)

    # Append results to dataframe
    summary_impr <- rbind(summary_impr, data.frame(WeekAhead = target_week_, Model = given_model, m = mean_val, sd = sd_val))

    # Generate histogram
    p <- ggplot(dataset, aes(x = .data[[given_model]])) +
      geom_histogram(color = "black", fill = "lightblue", bins = 30, alpha = 0.7) +
      geom_vline(aes(xintercept = mean_val), color = "red", linetype = "dashed", linewidth = 1) +
      geom_vline(aes(xintercept = mean_val - sd_val), color = "blue", linetype = "dotted", linewidth = 1) +
      geom_vline(aes(xintercept = mean_val + sd_val), color = "blue", linetype = "dotted", linewidth = 1)
  }
}

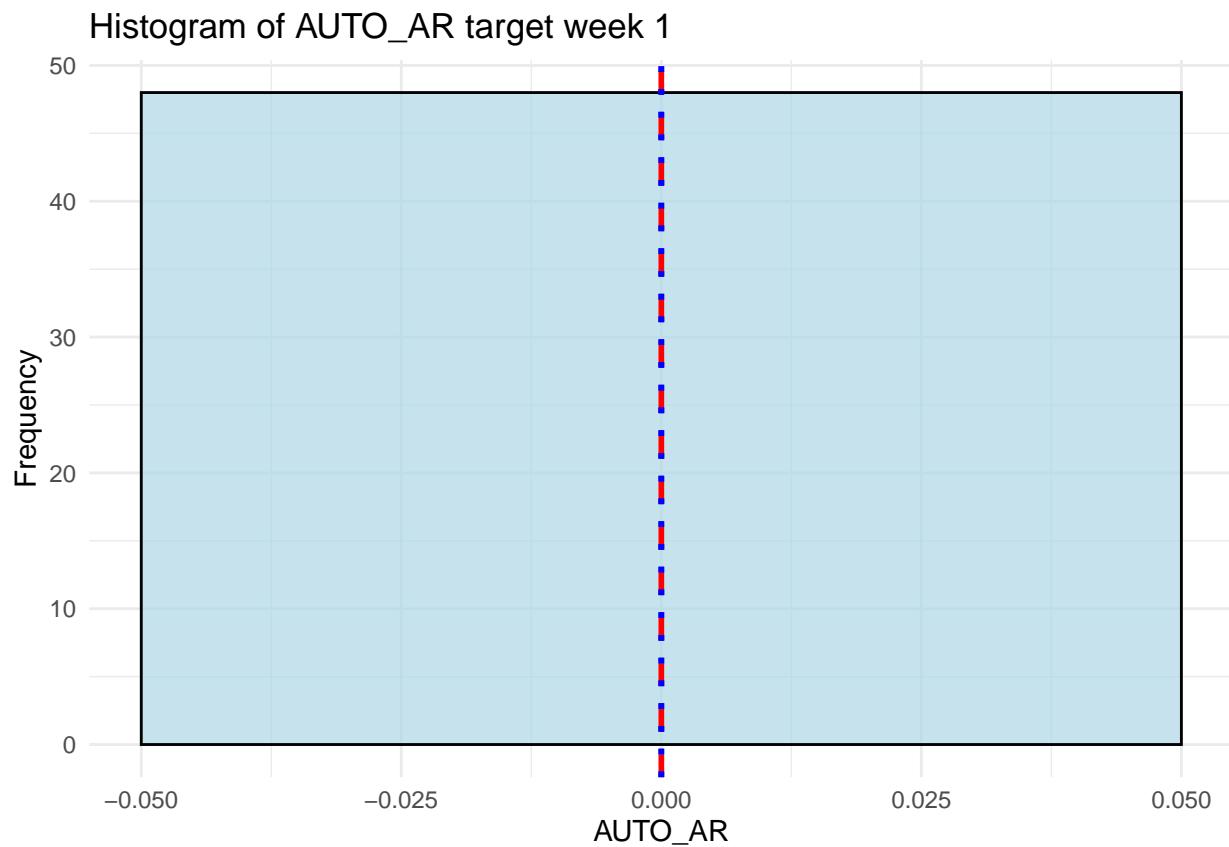
```

```

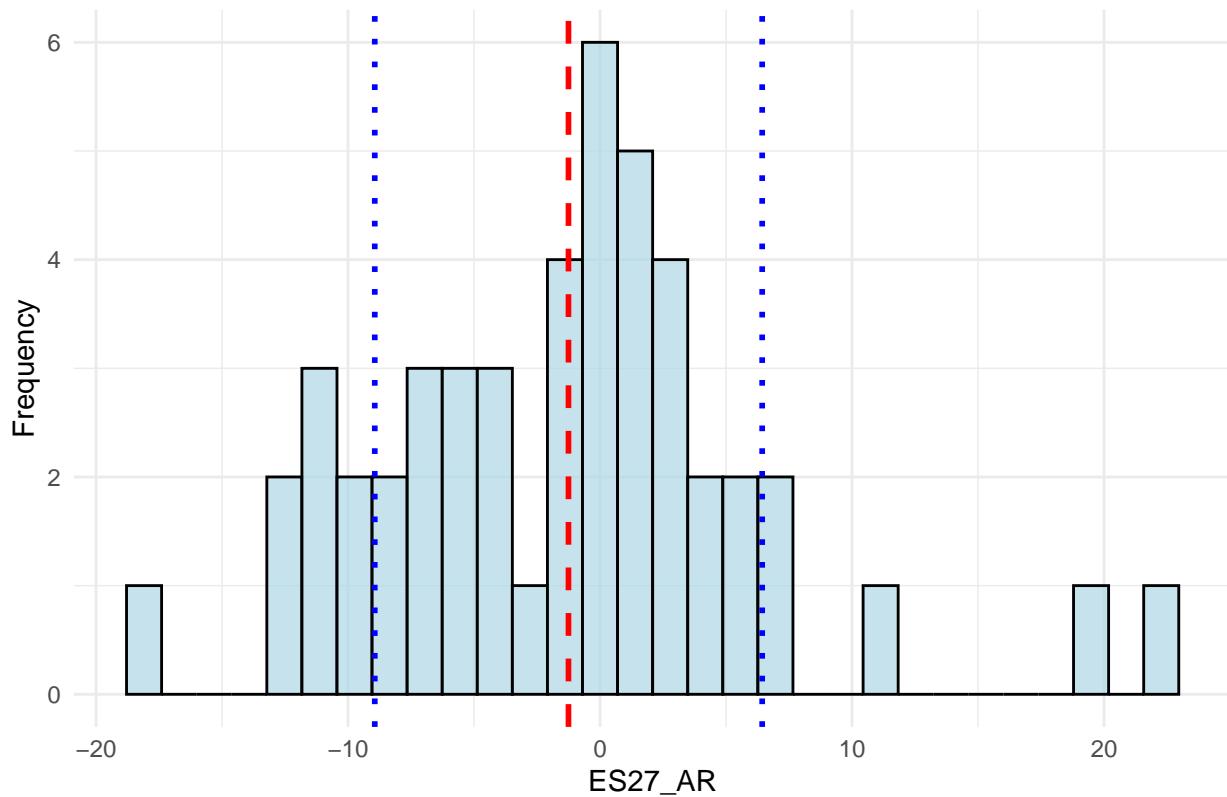
    labs(title = paste("Histogram of", given_model, "target week", target_week_),
         x = given_model,
         y = "Frequency") +
    theme_minimal()

  print(p) # Display the plot
}
}

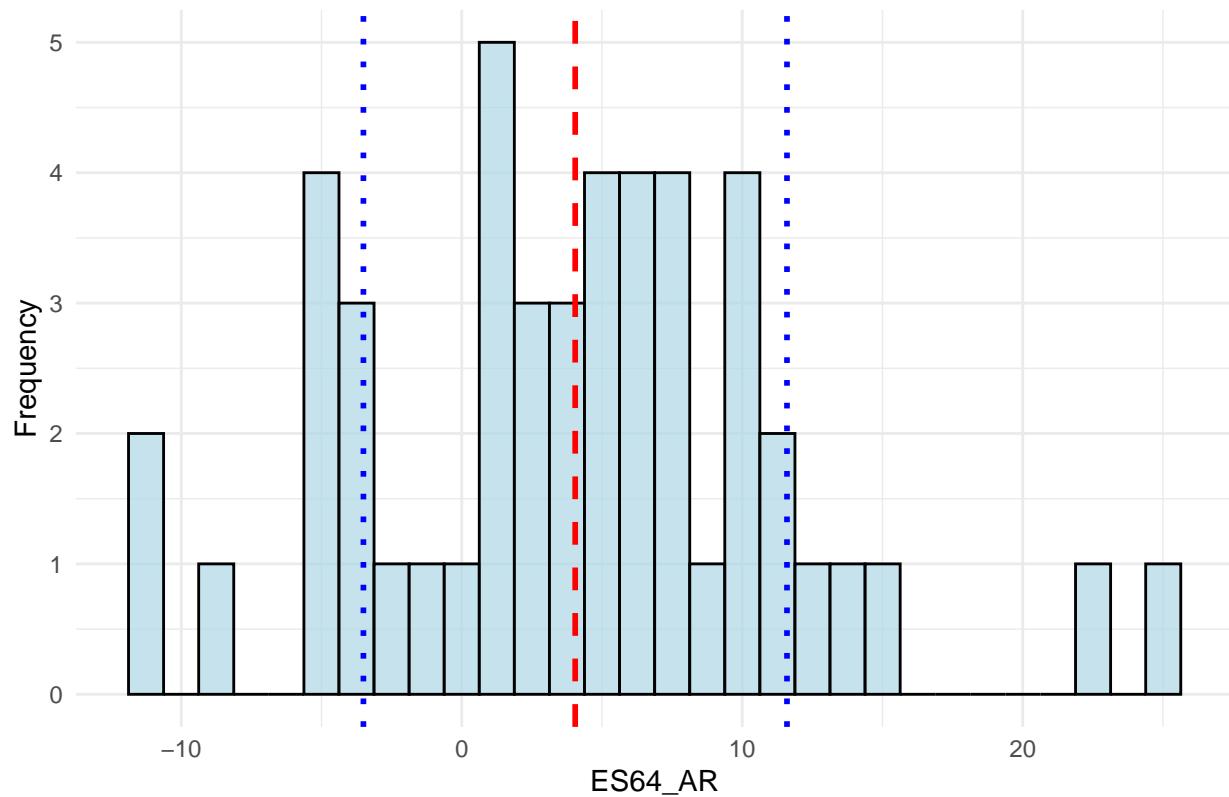
```



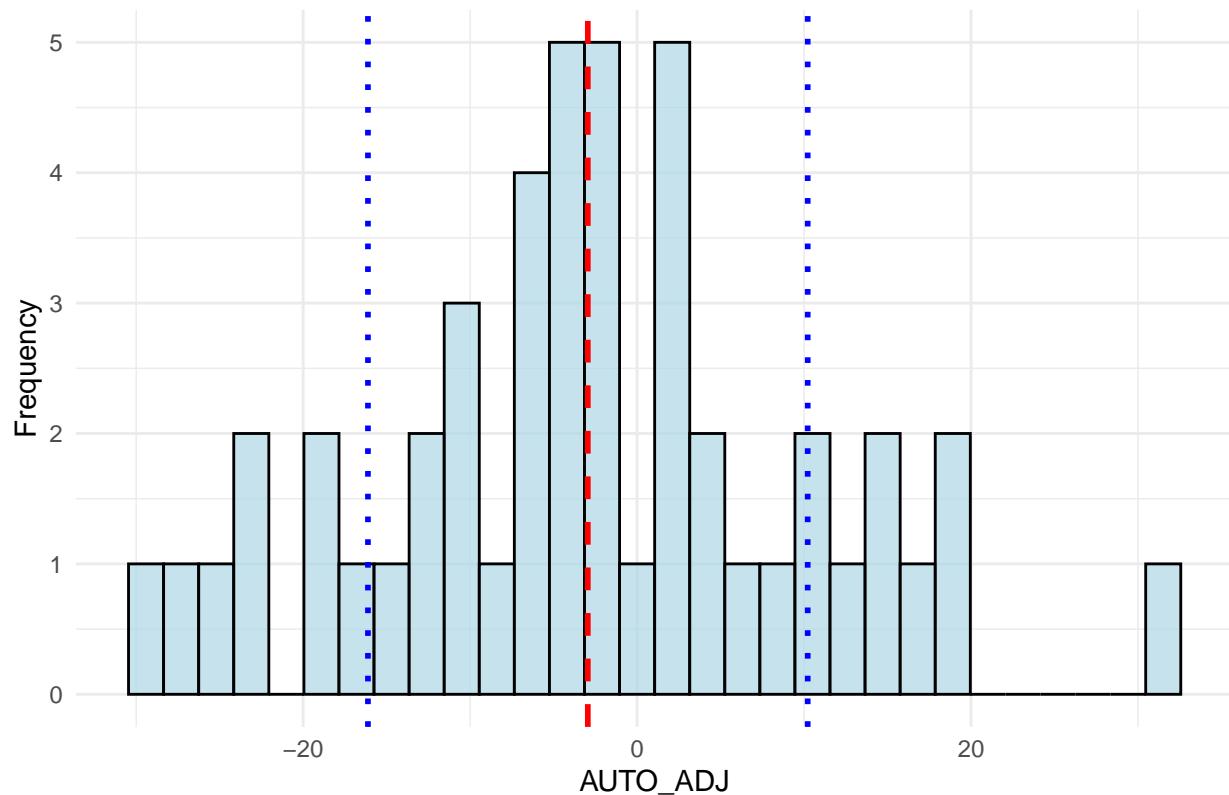
Histogram of ES27\_AR target week 1



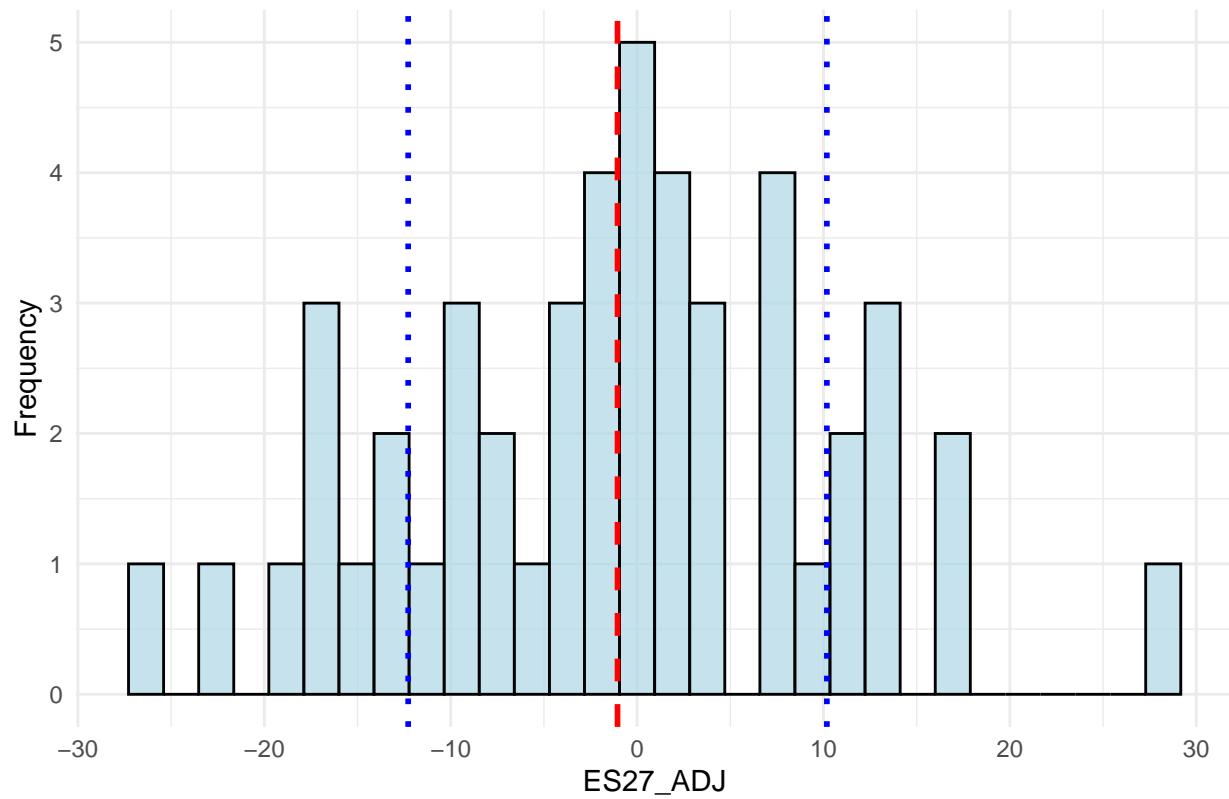
Histogram of ES64\_AR target week 1



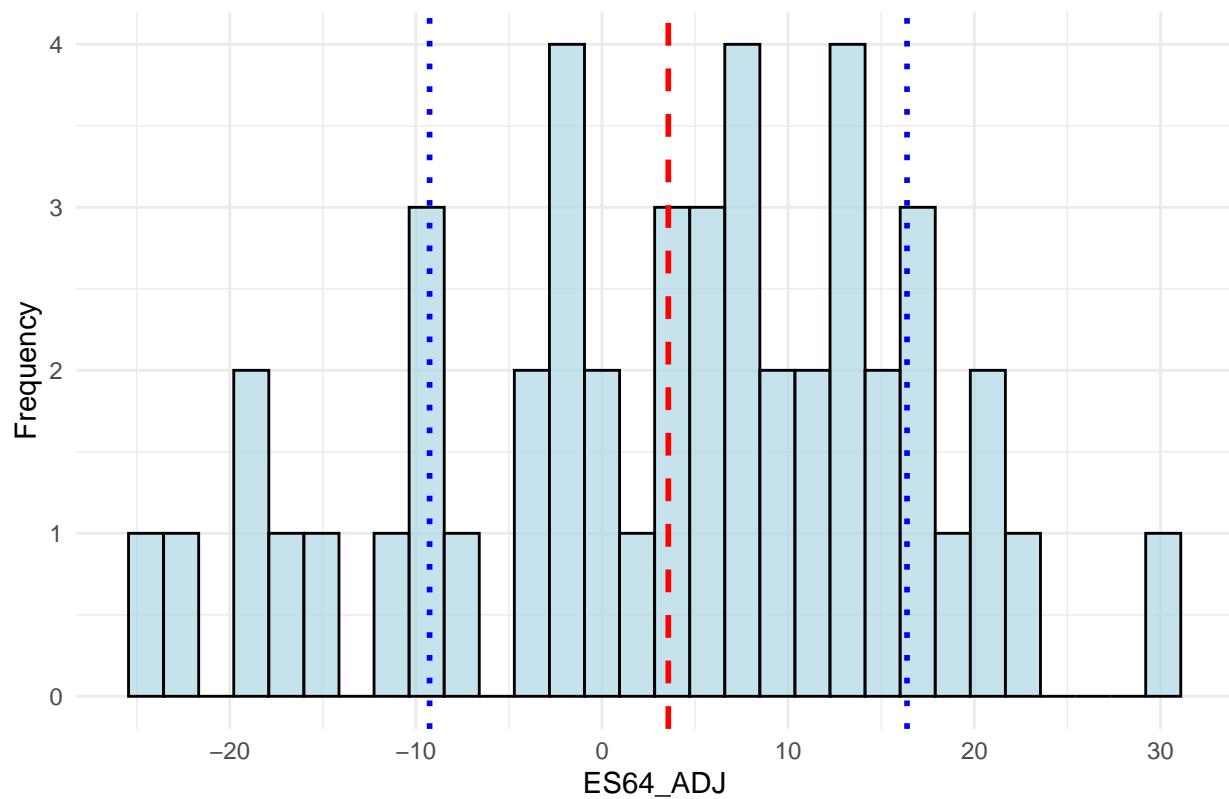
Histogram of AUTO\_ADJ target week 1



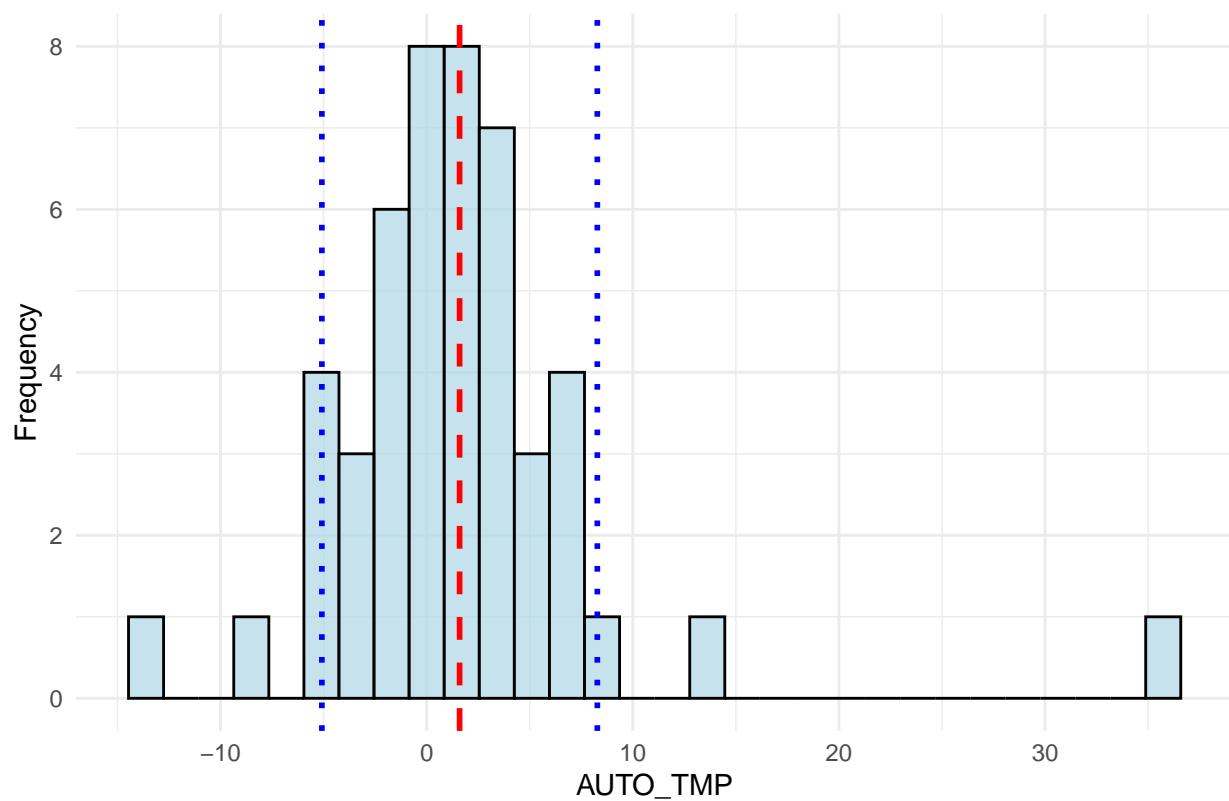
Histogram of ES27\_ADJ target week 1



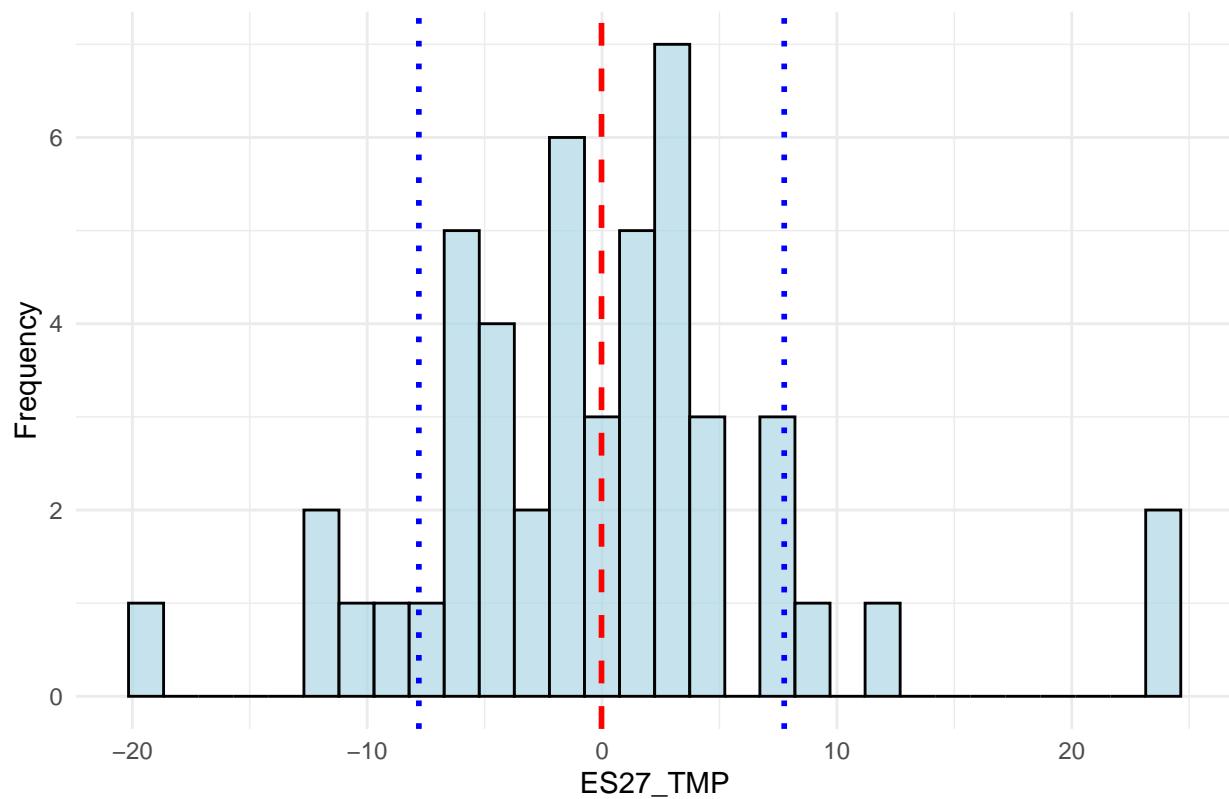
Histogram of ES64\_ADJ target week 1



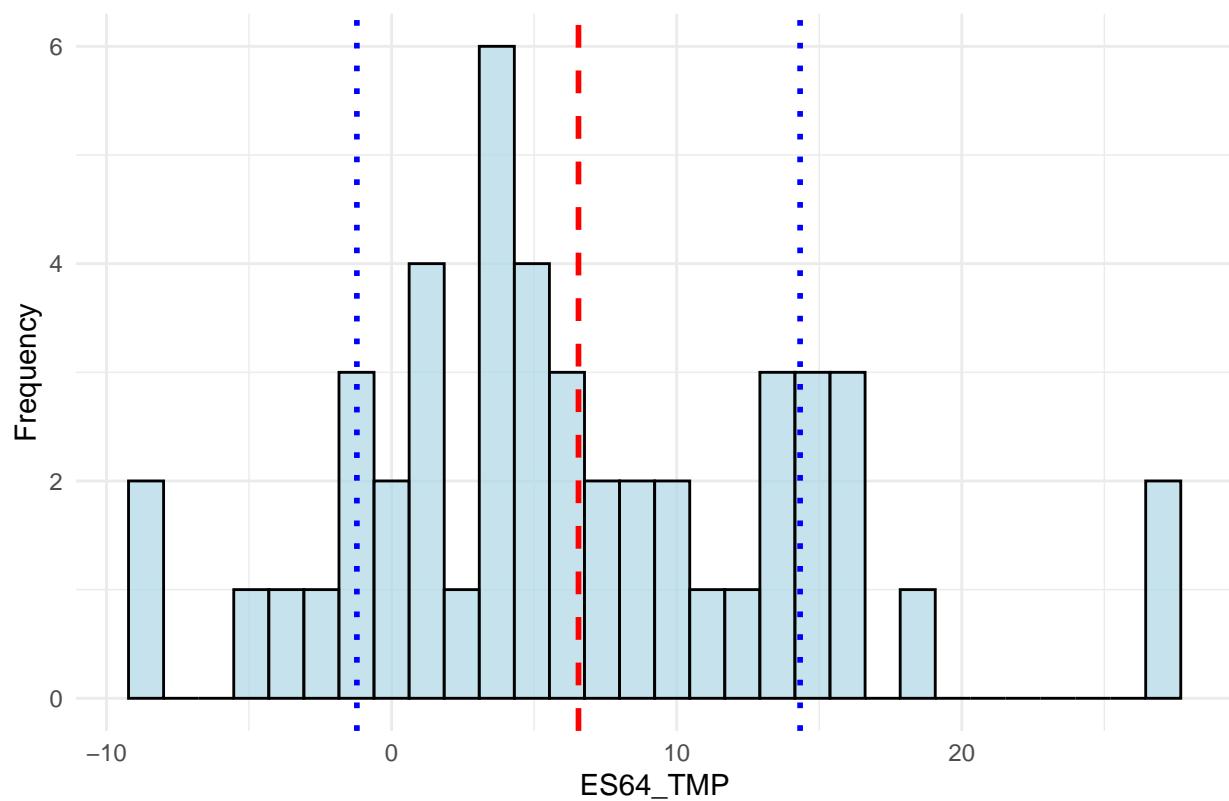
Histogram of AUTO\_TMP target week 1



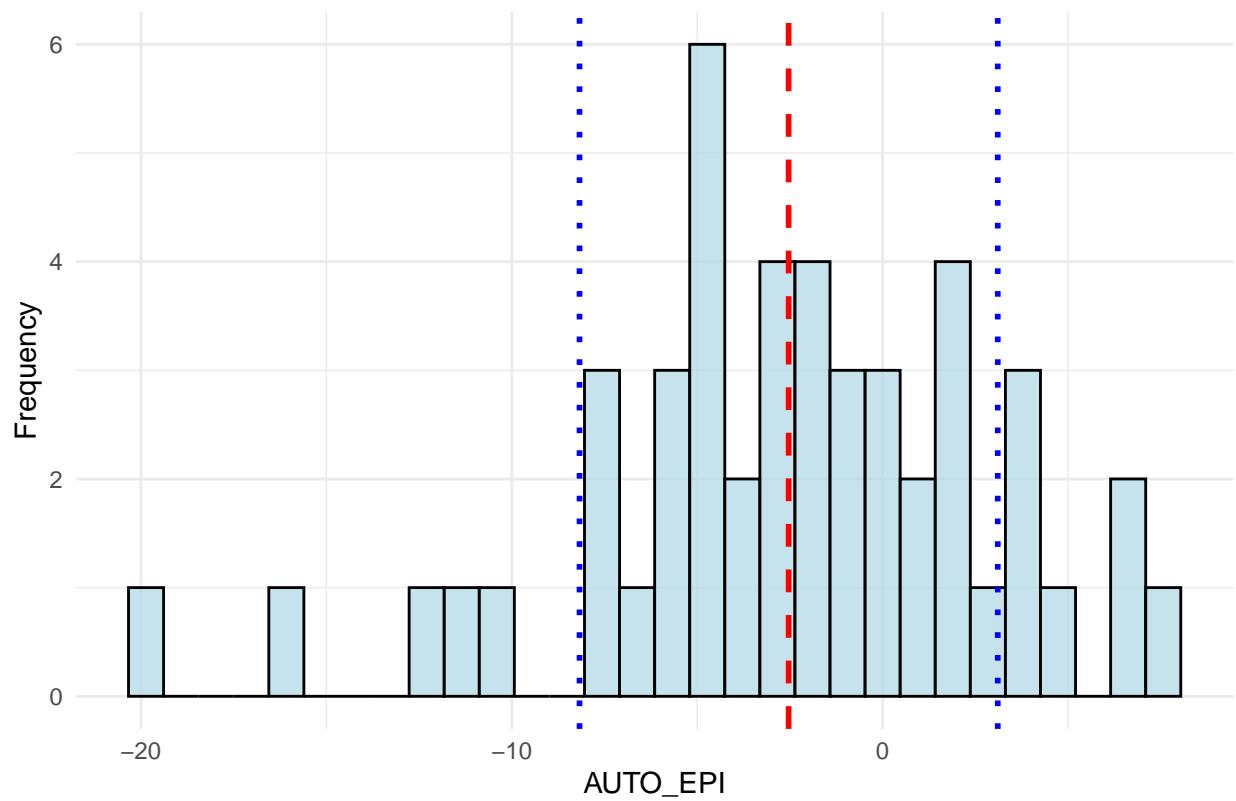
Histogram of ES27\_TMP target week 1



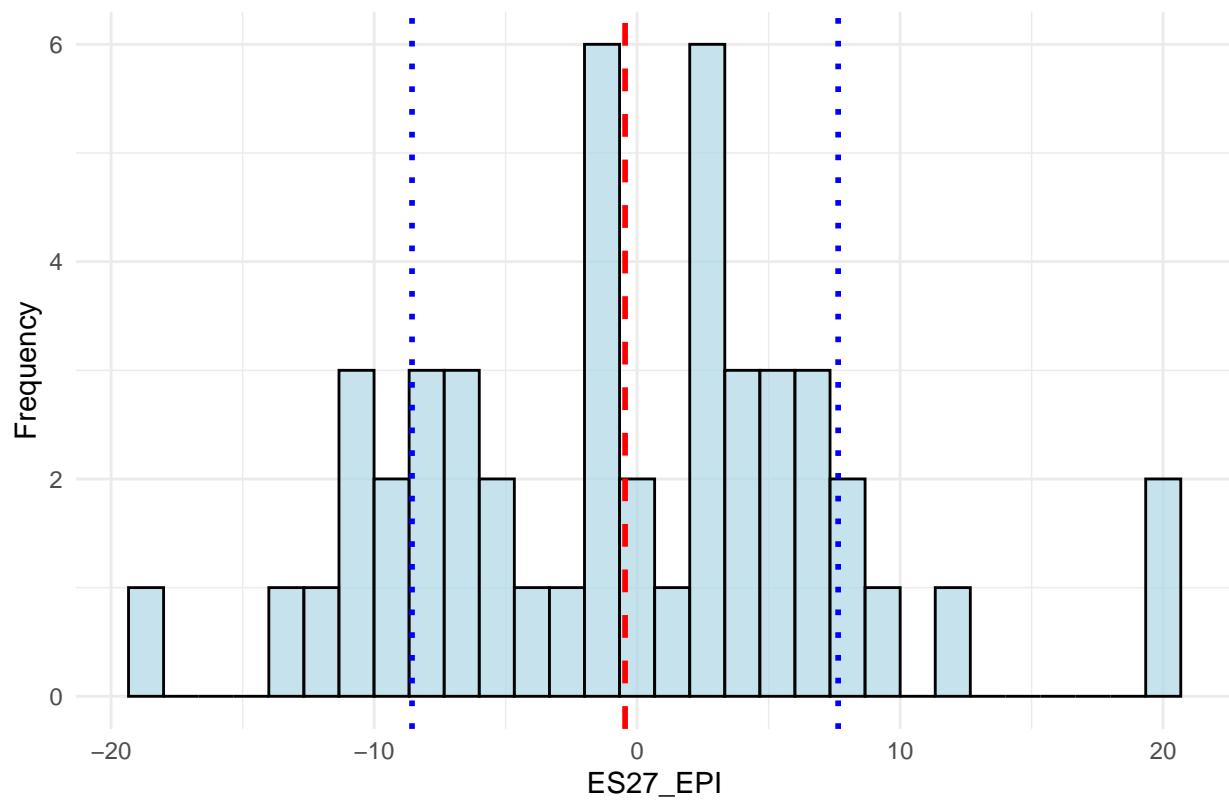
Histogram of ES64\_TMP target week 1



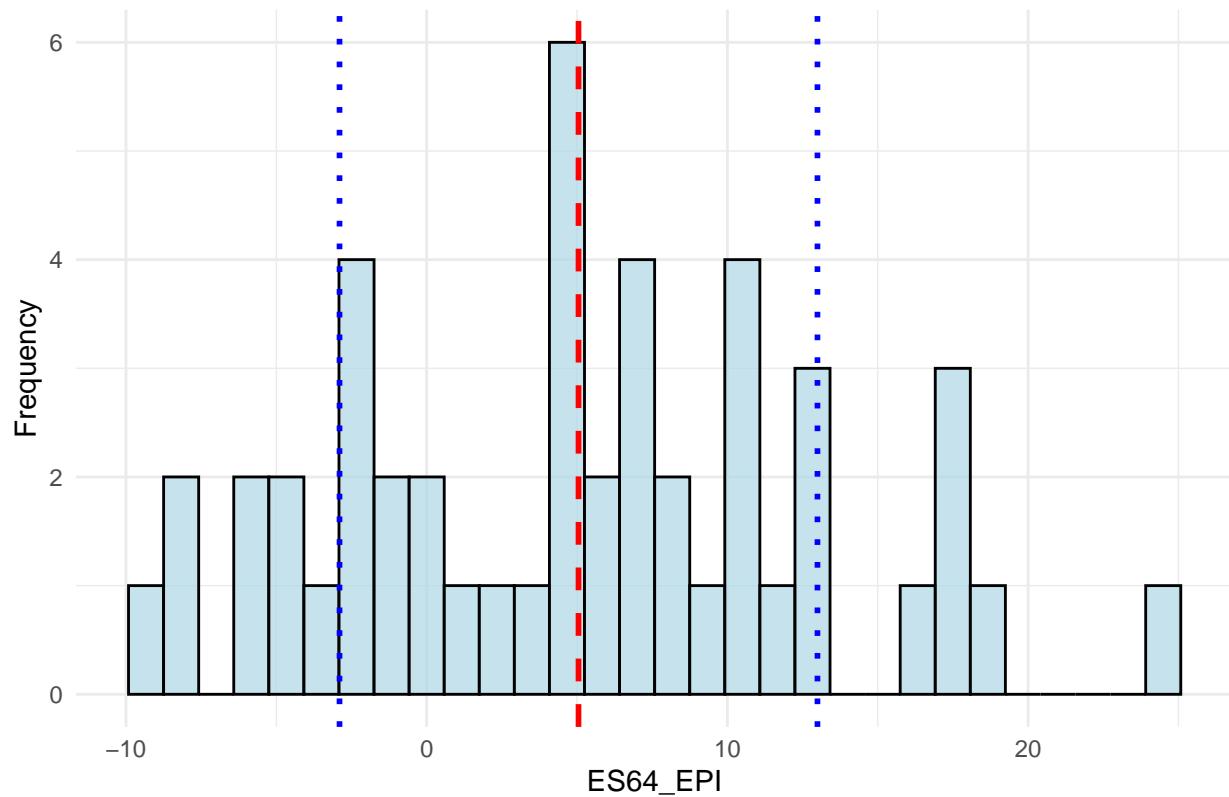
Histogram of AUTO\_EPI target week 1



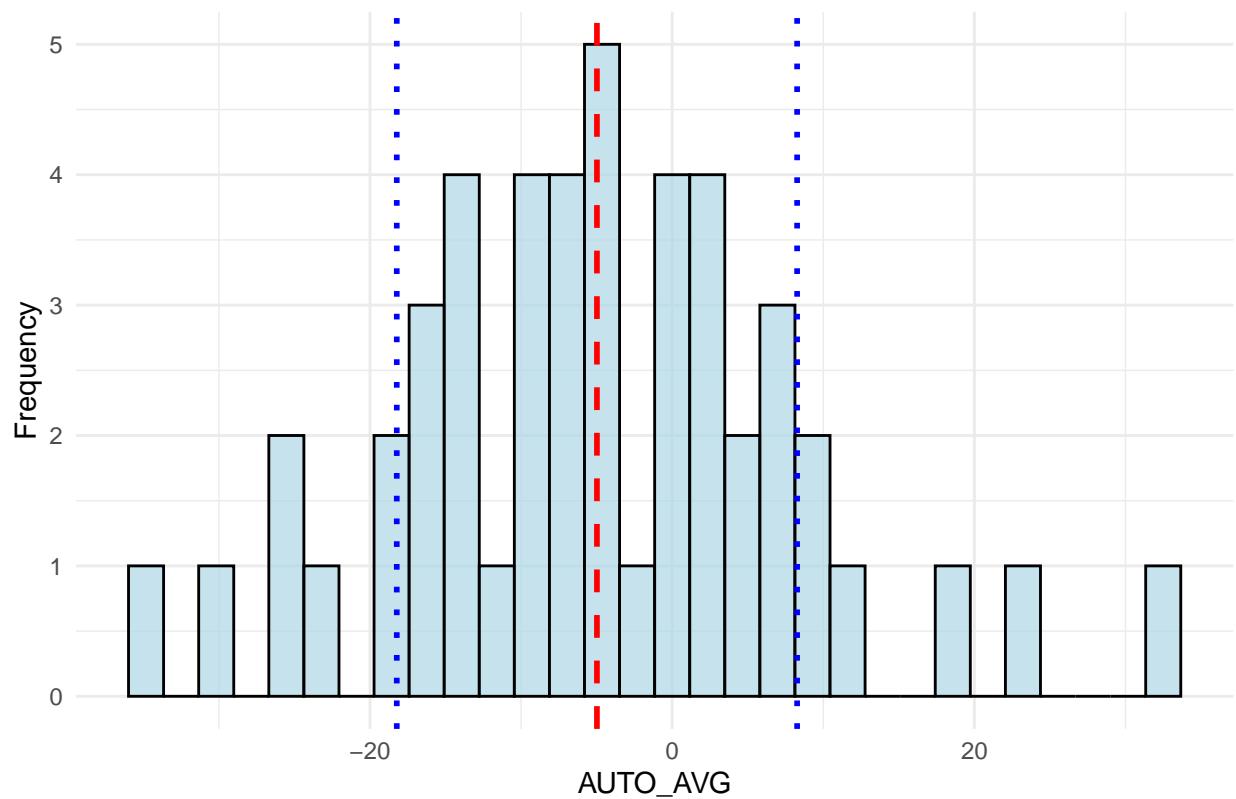
Histogram of ES27\_EPI target week 1



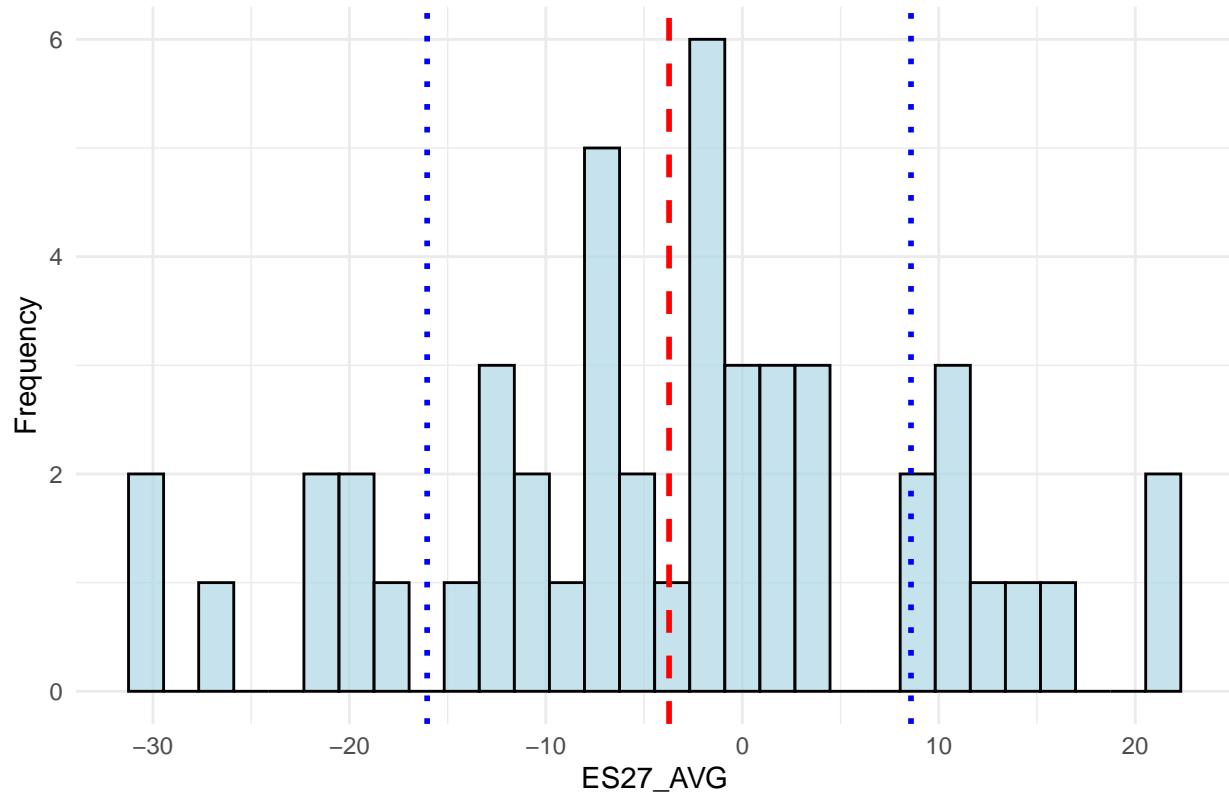
Histogram of ES64\_EPI target week 1



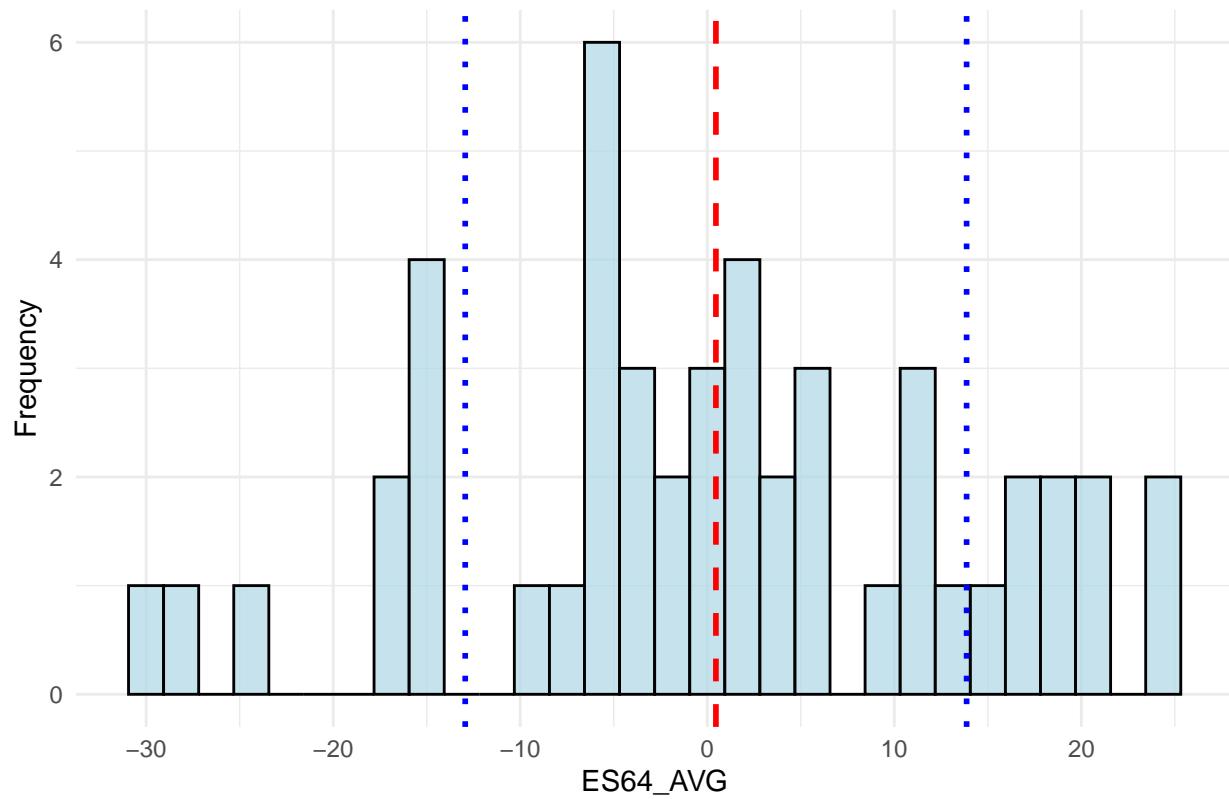
Histogram of AUTO\_AVG target week 1



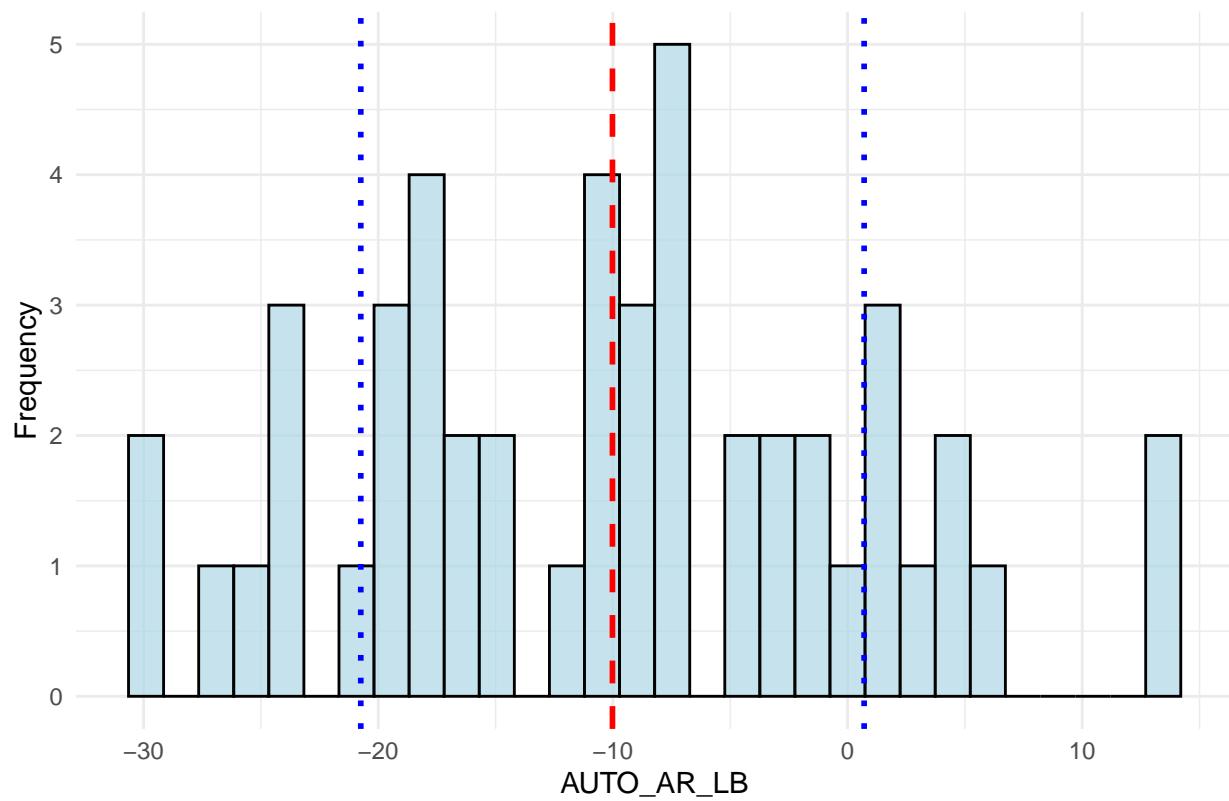
Histogram of ES27\_AVG target week 1



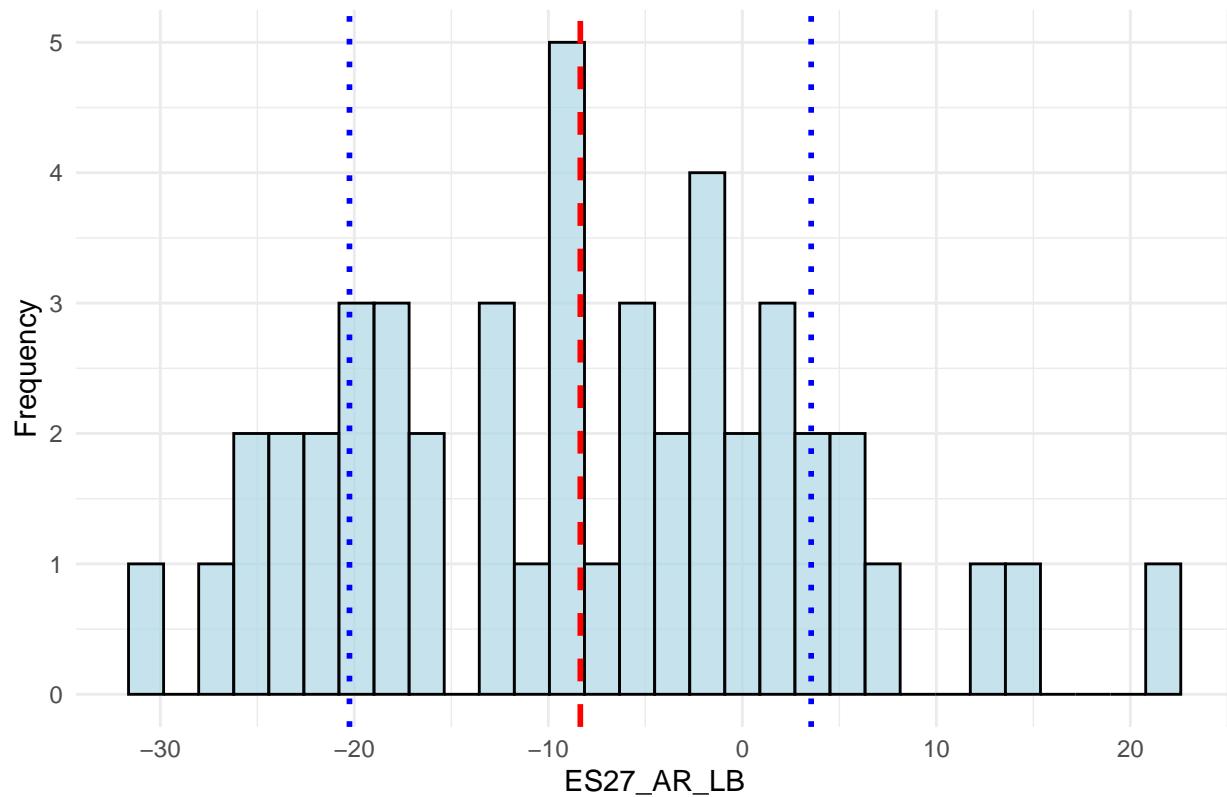
Histogram of ES64\_AVG target week 1



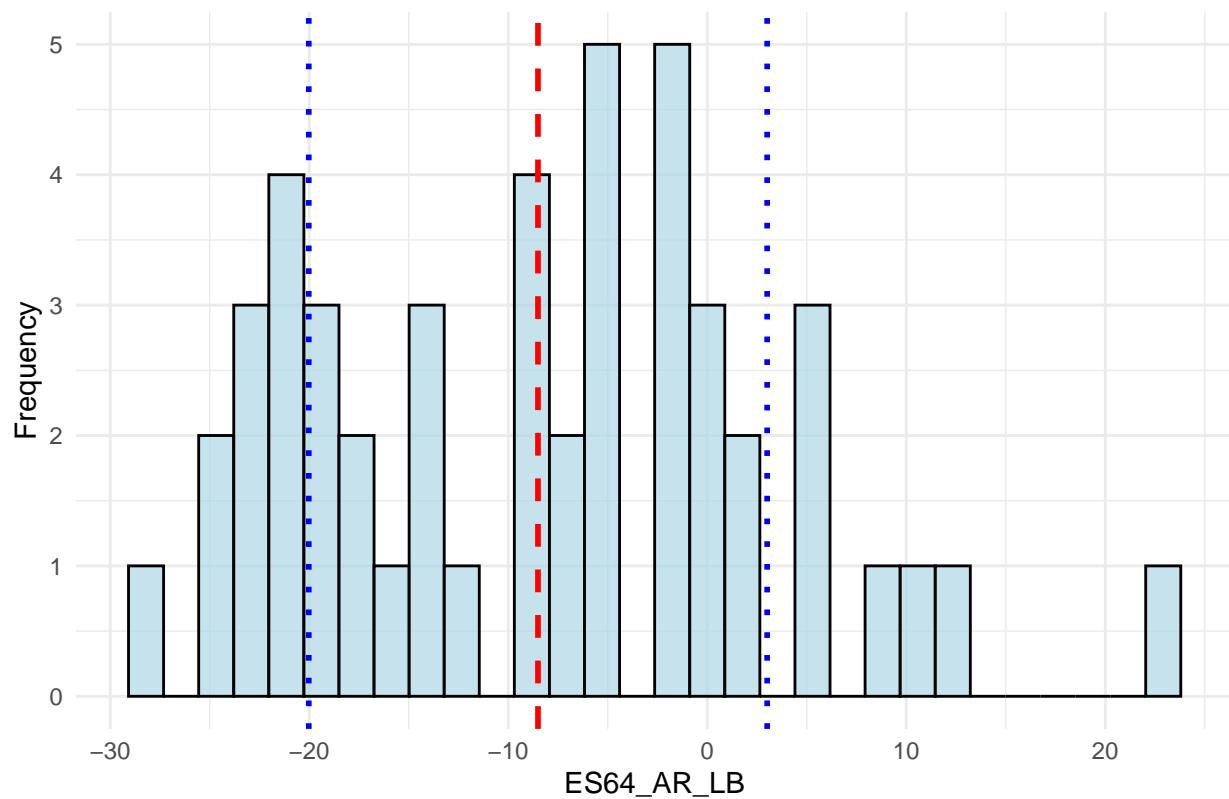
Histogram of AUTO\_AR\_LB target week 1



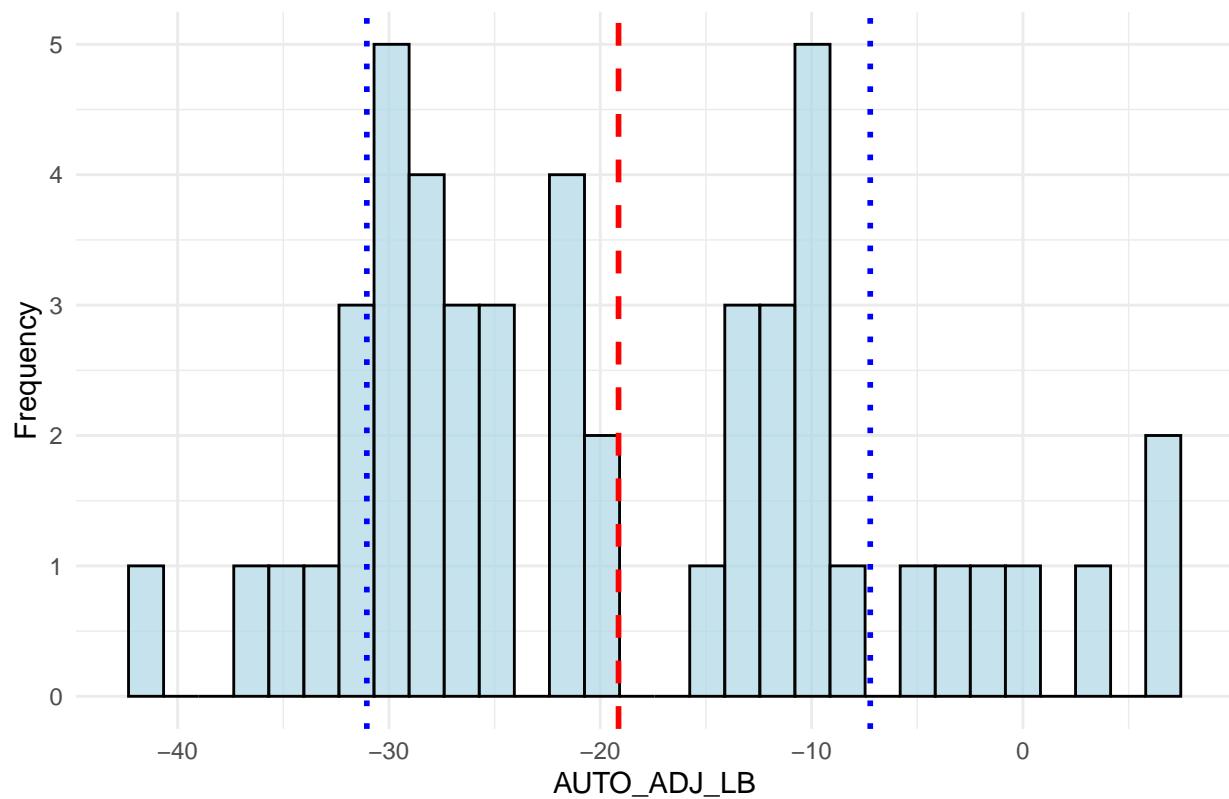
Histogram of ES27\_AR\_LB target week 1



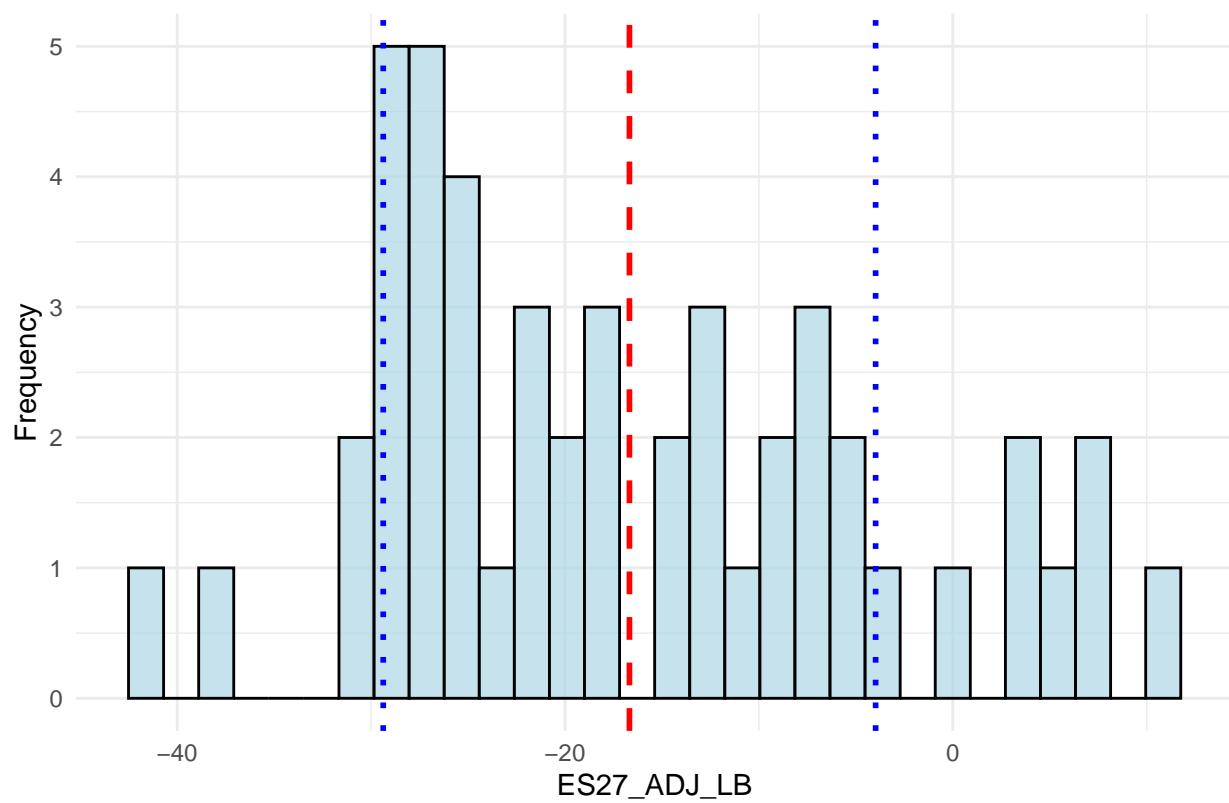
Histogram of ES64\_AR\_LB target week 1



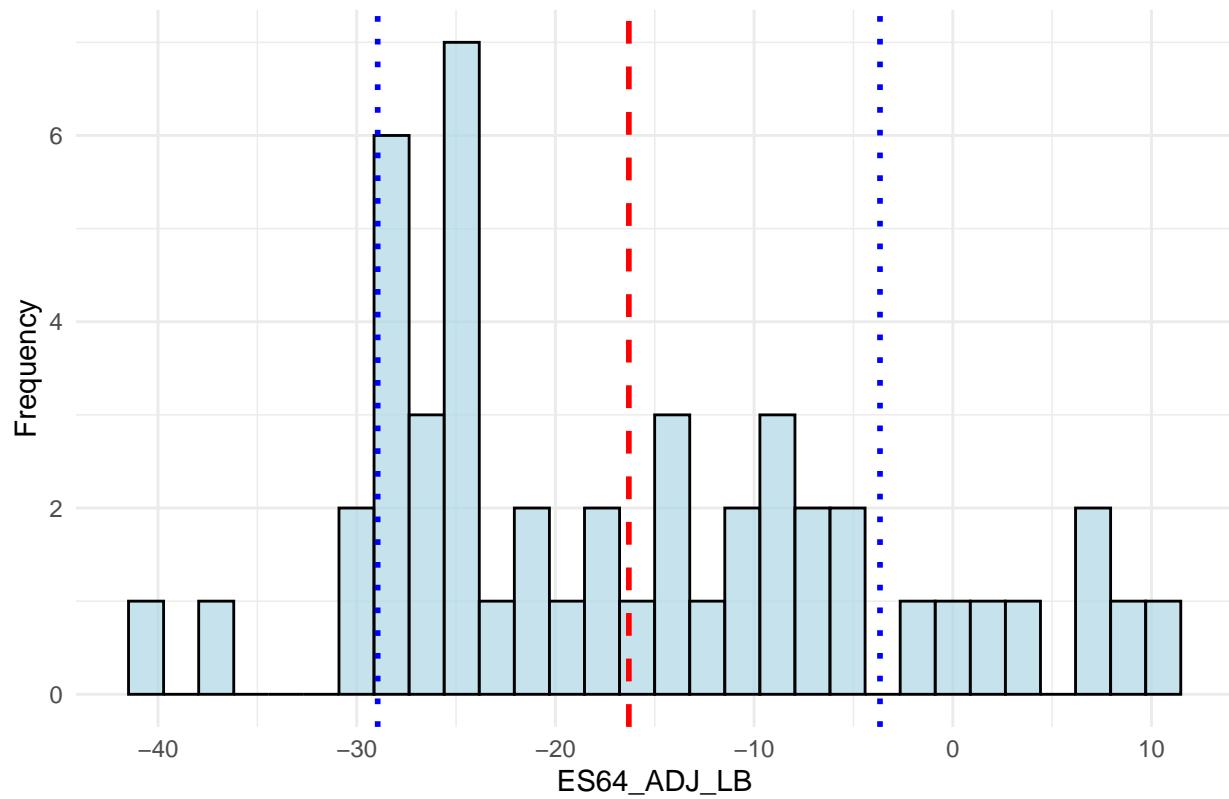
Histogram of AUTO\_ADJ\_LB target week 1



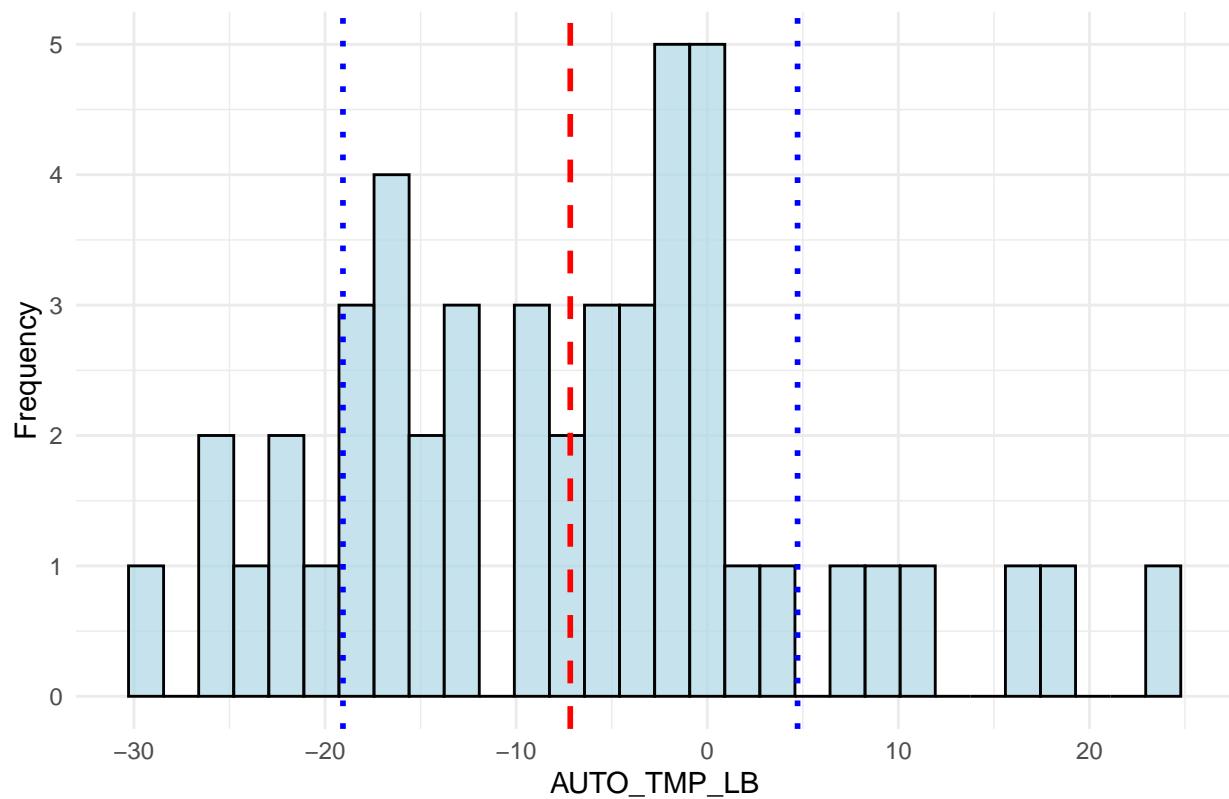
Histogram of ES27\_ADJ\_LB target week 1



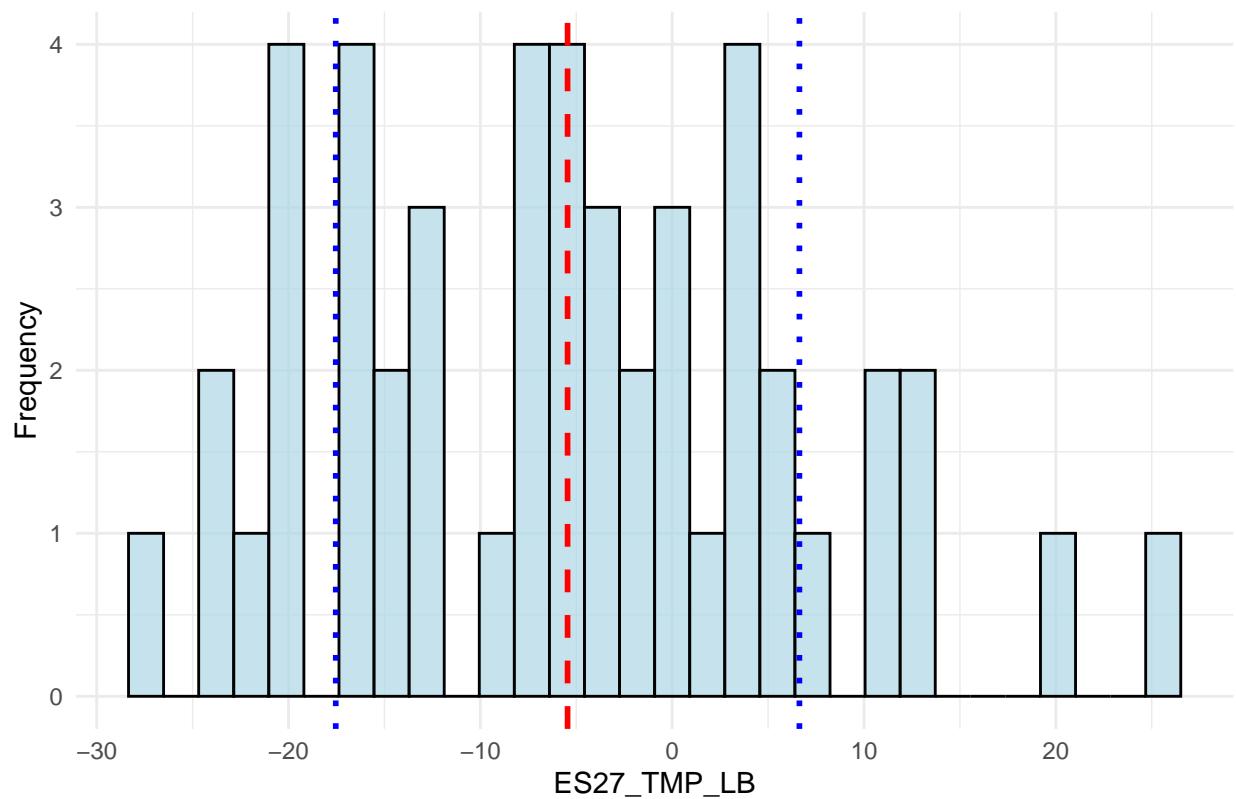
Histogram of ES64\_ADJ\_LB target week 1



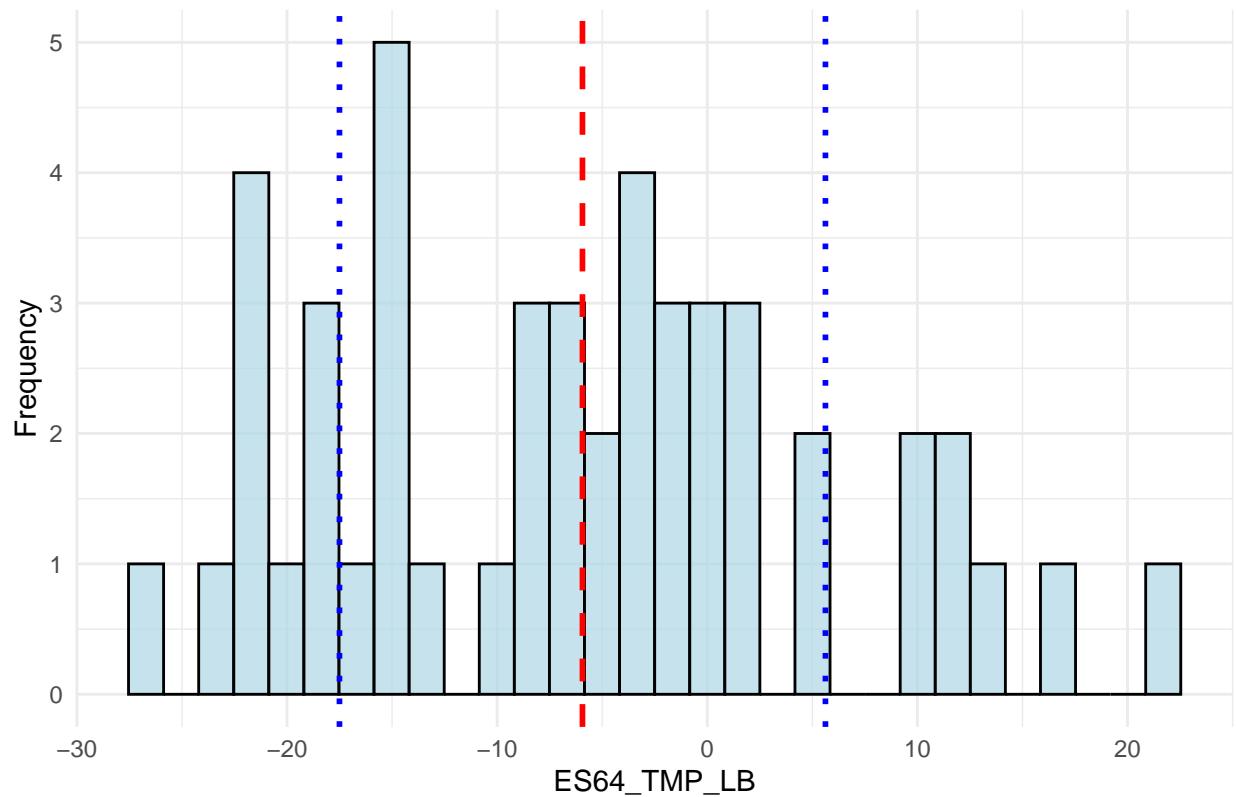
Histogram of AUTO\_TMP\_LB target week 1



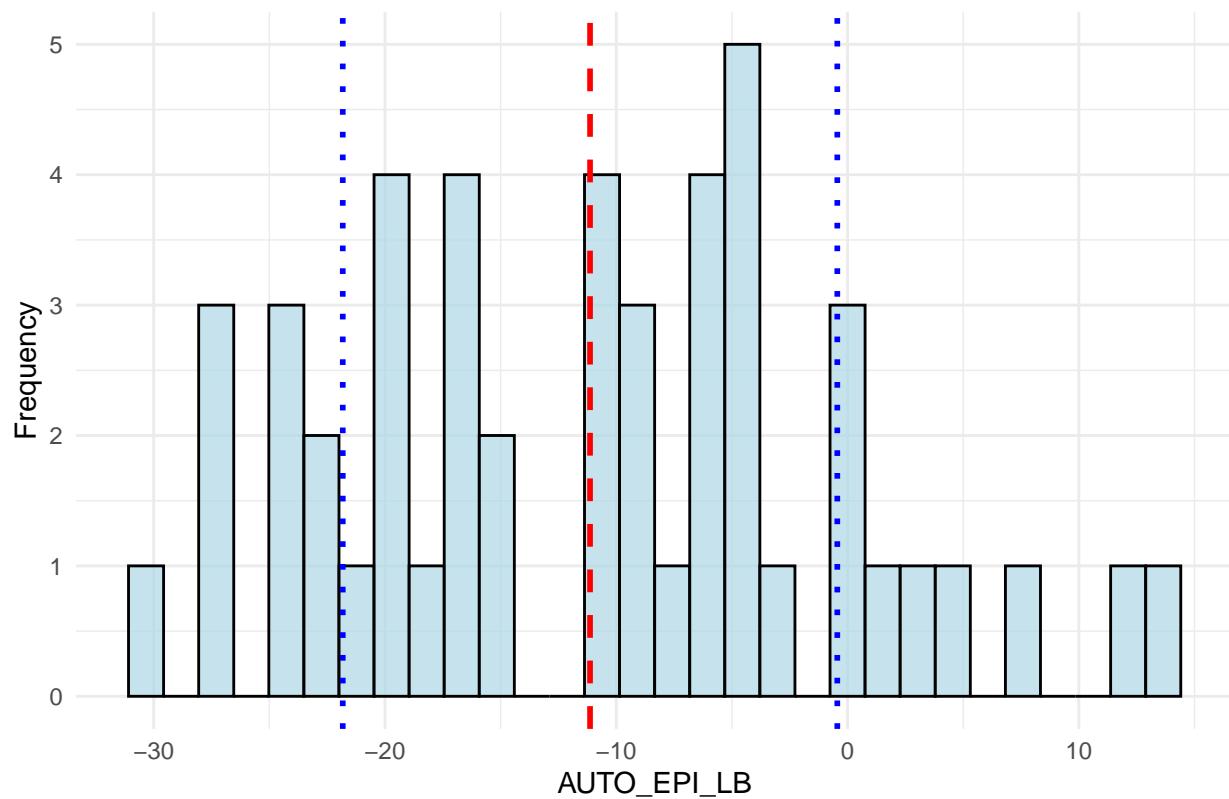
Histogram of ES27\_TMP\_LB target week 1



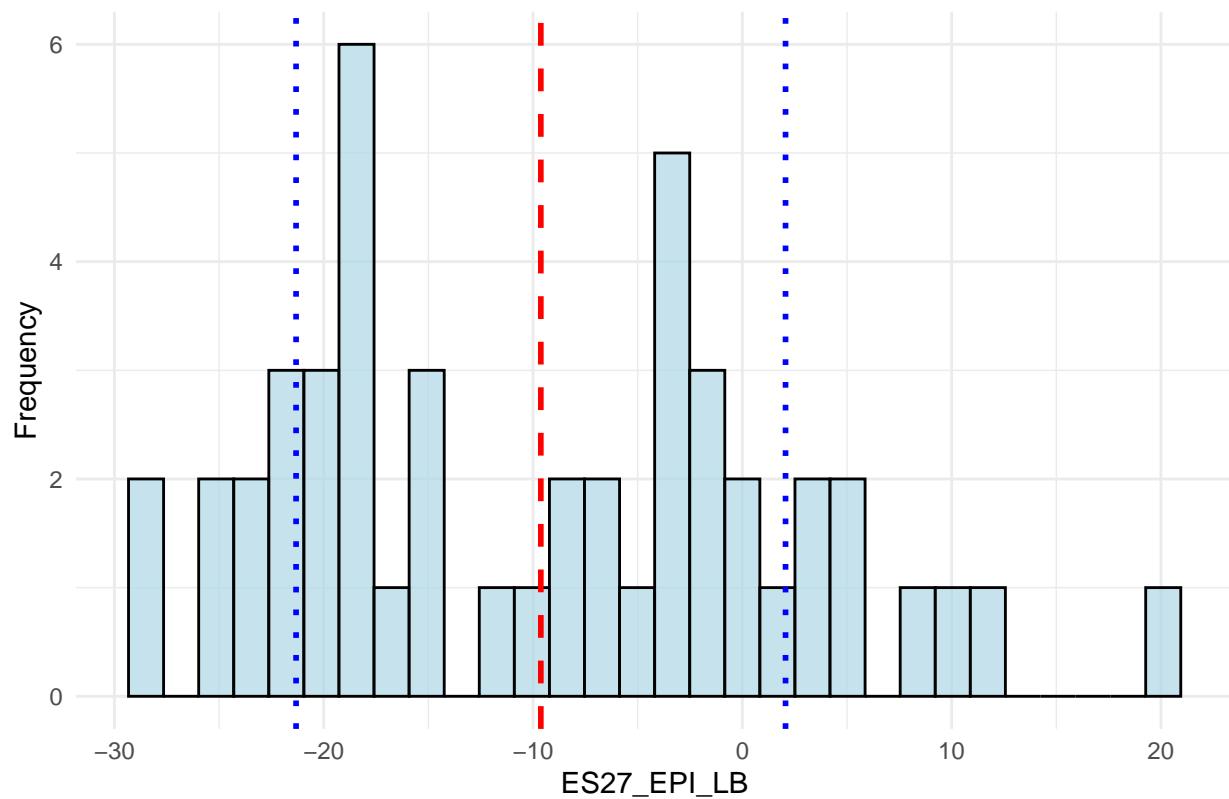
Histogram of ES64\_TMP\_LB target week 1



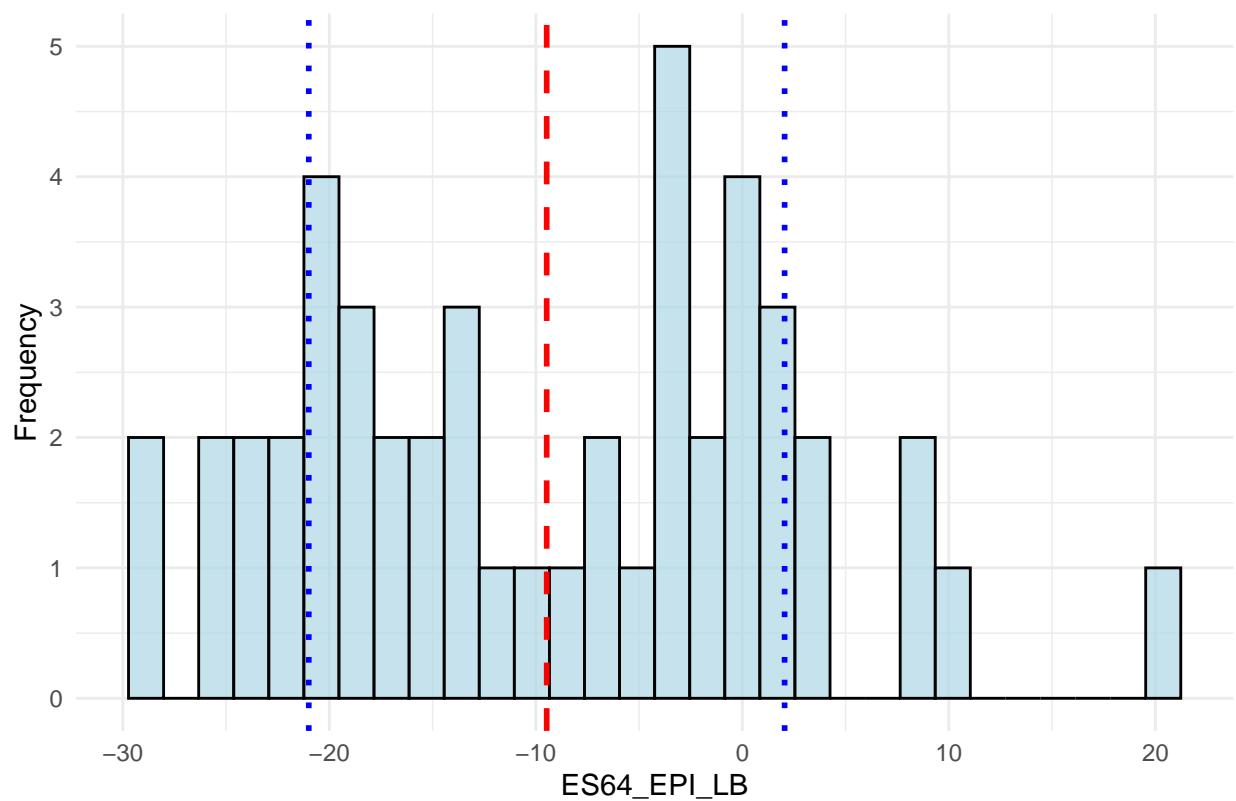
Histogram of AUTO\_EPI\_LB target week 1



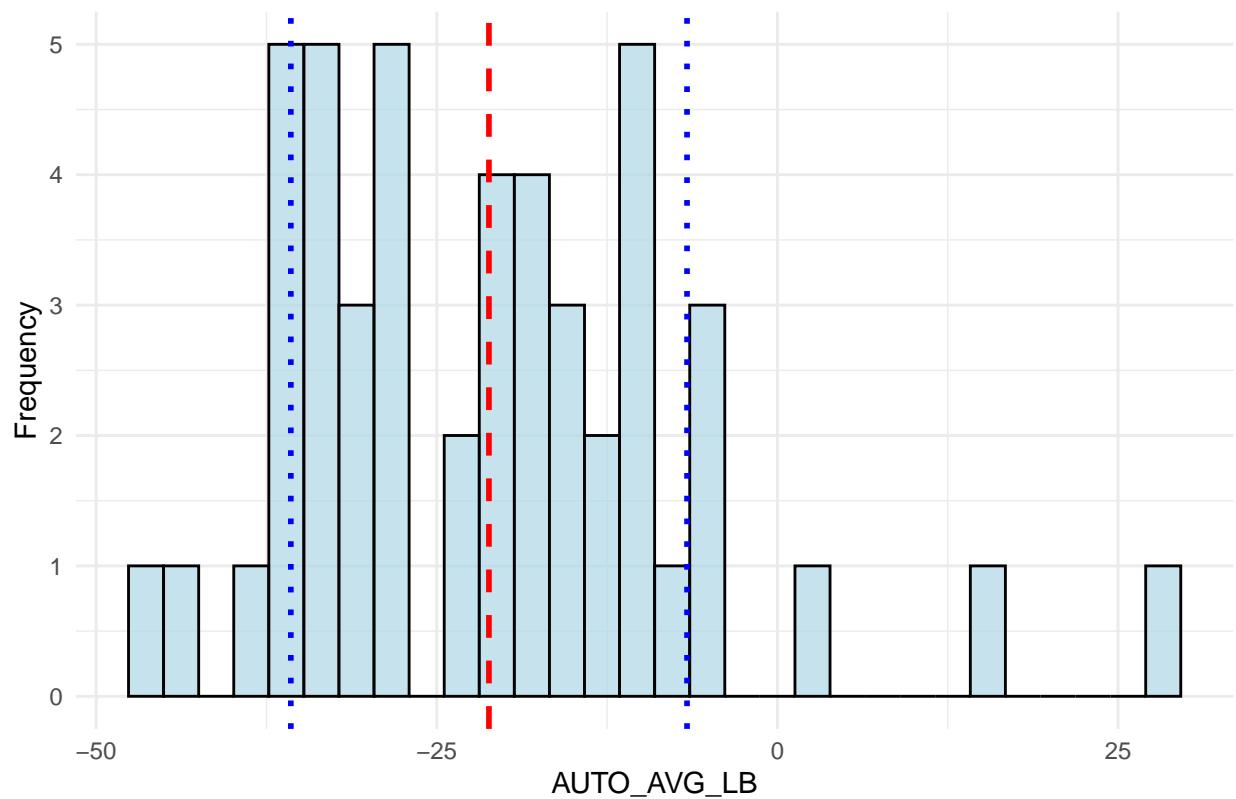
Histogram of ES27\_EPI\_LB target week 1



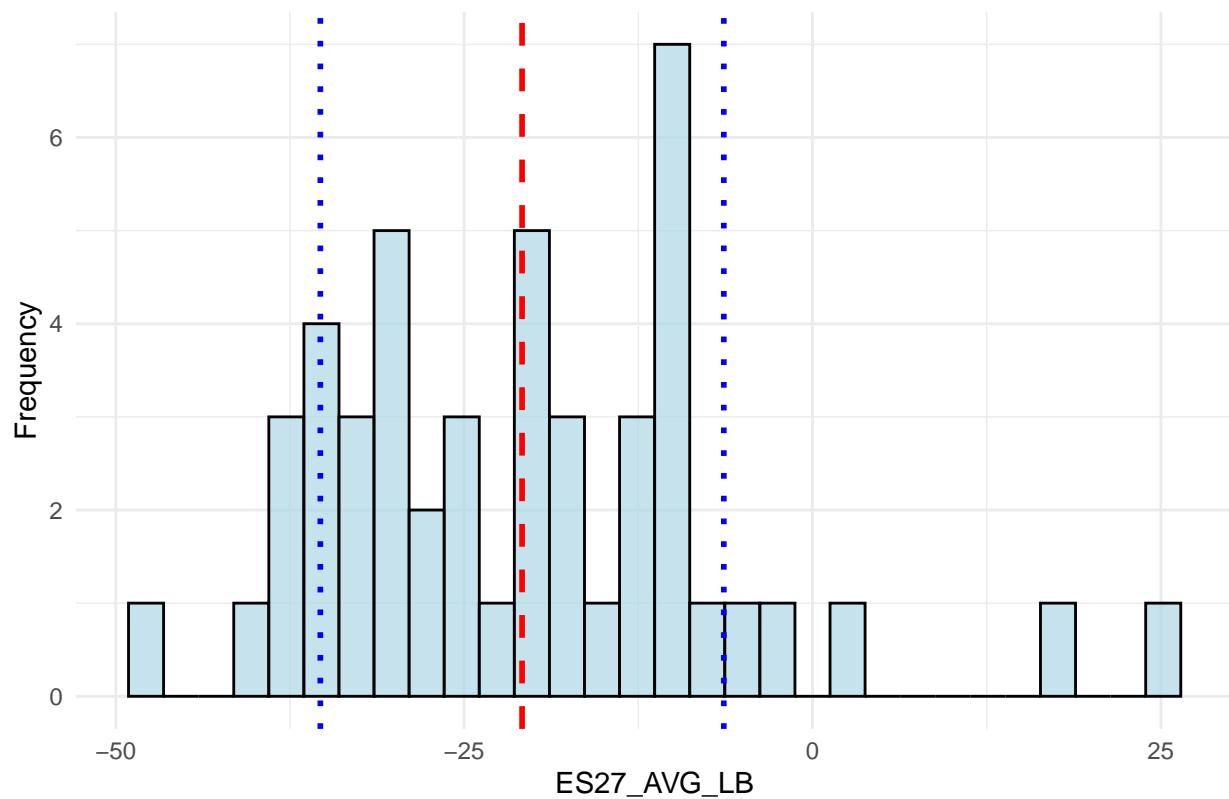
Histogram of ES64\_EPI\_LB target week 1



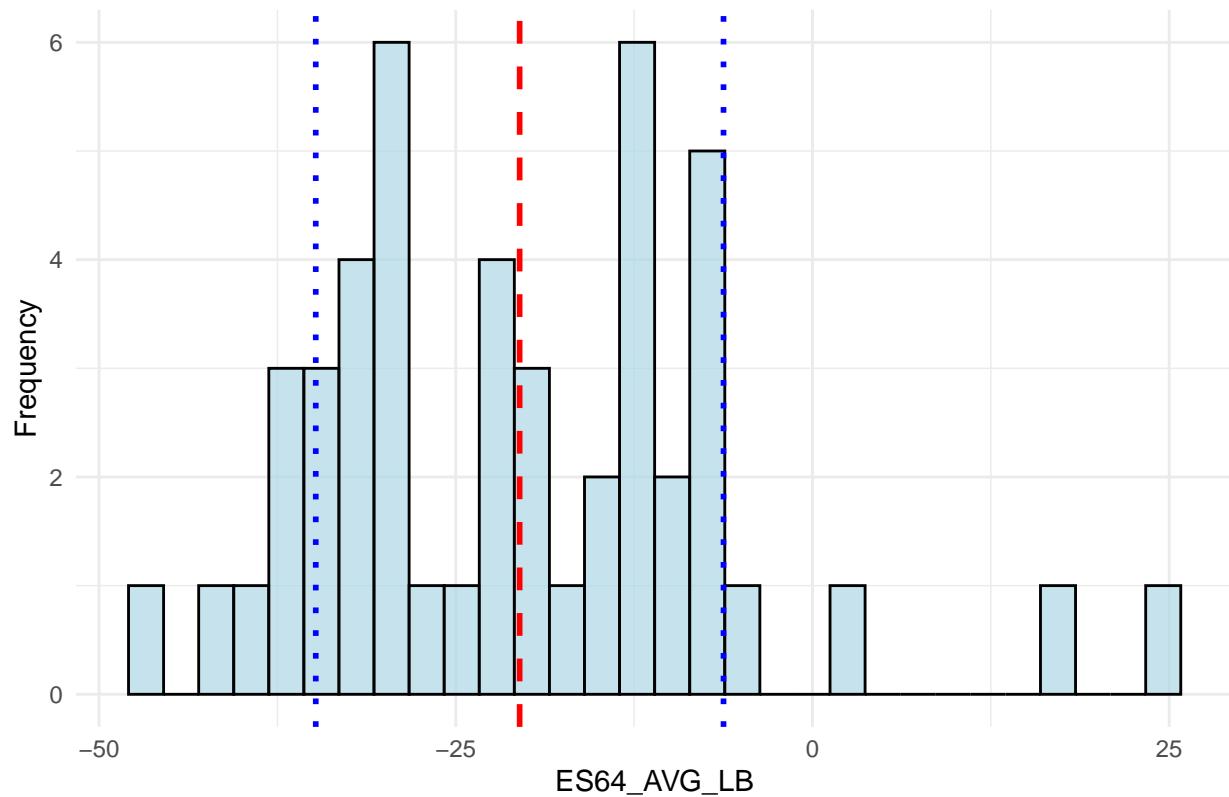
Histogram of AUTO\_AVG\_LB target week 1



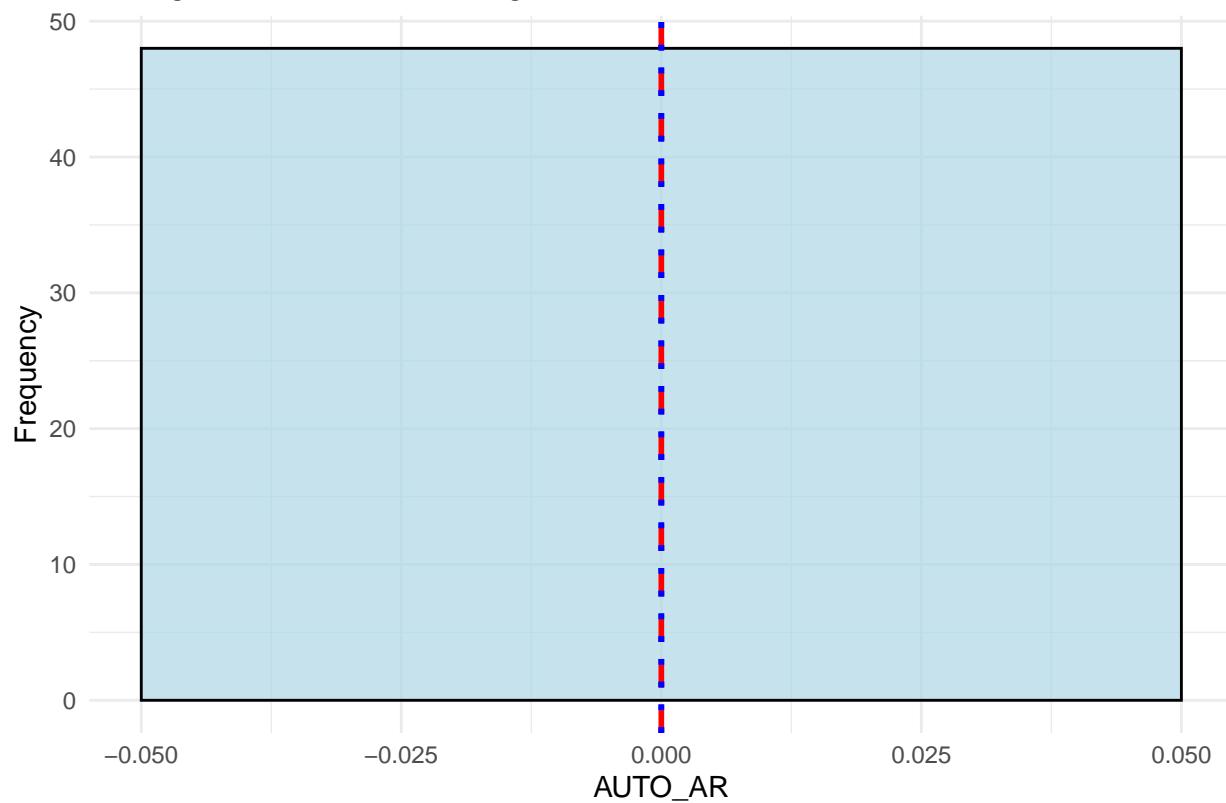
Histogram of ES27\_AVG\_LB target week 1



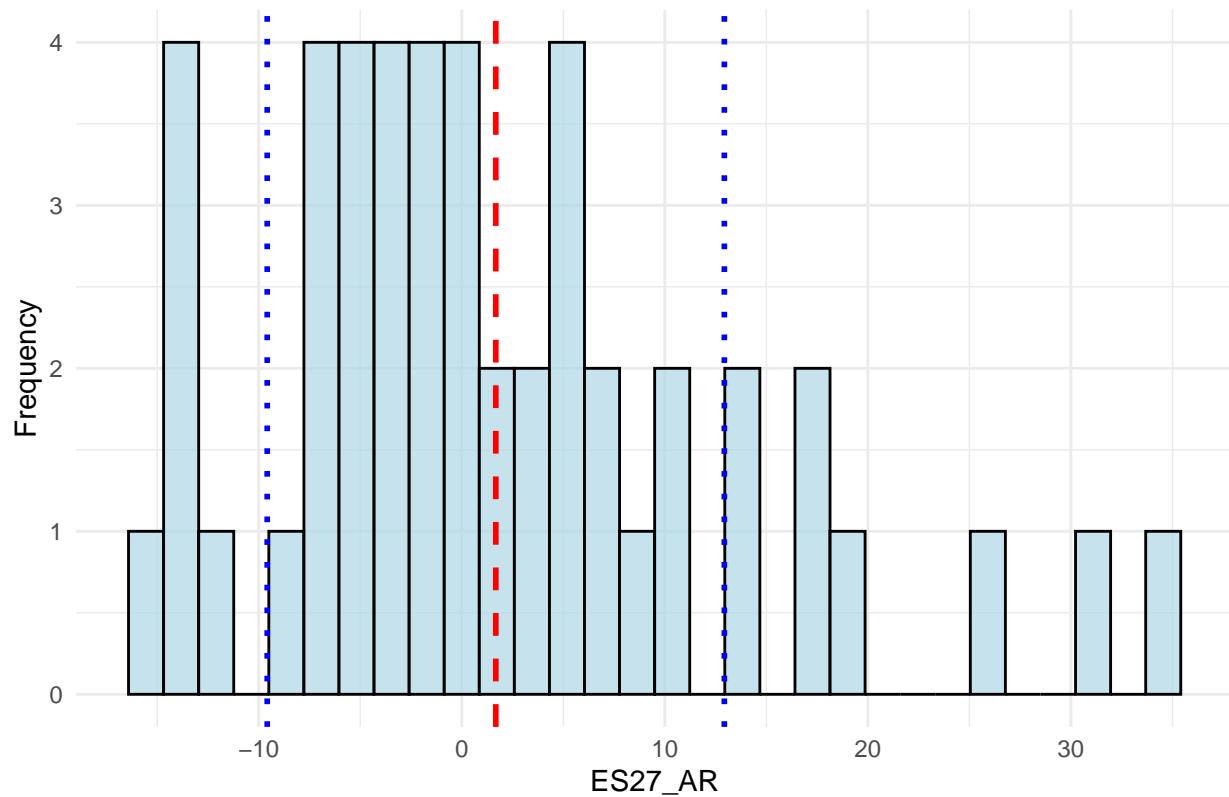
Histogram of ES64\_AVG\_LB target week 1



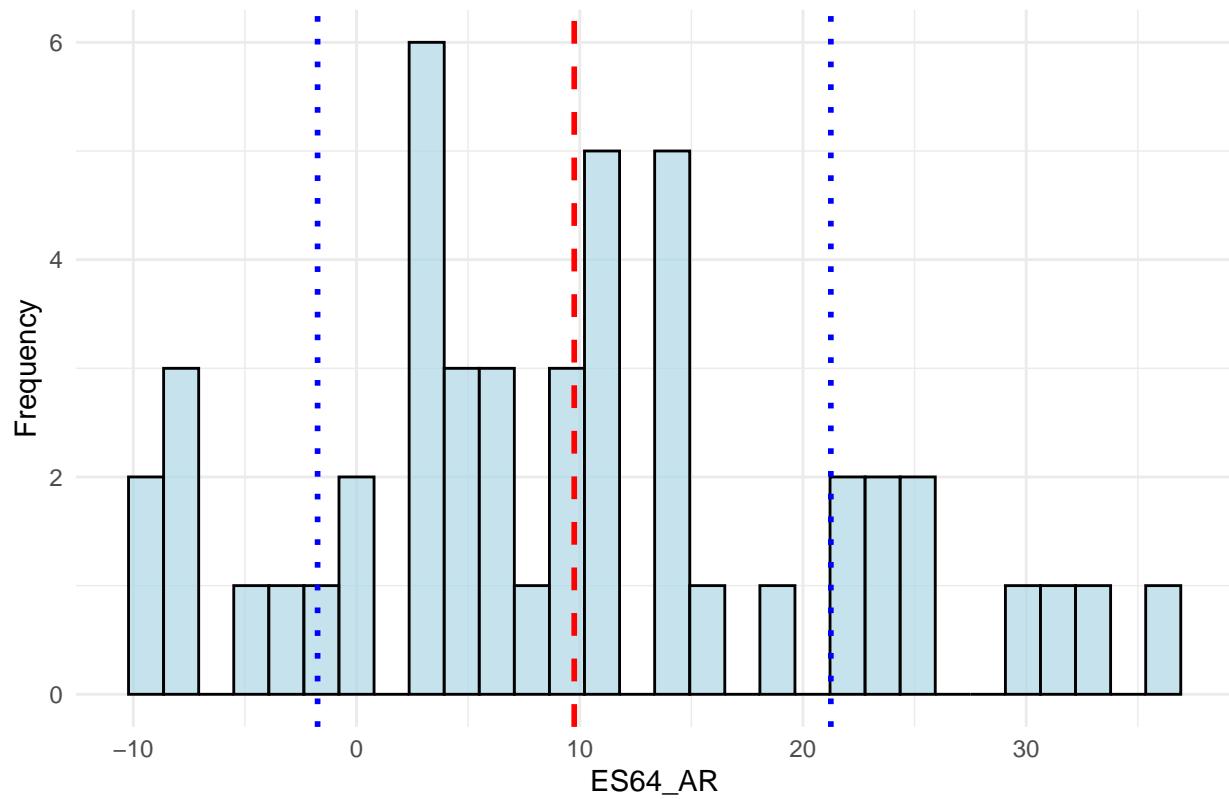
Histogram of AUTO\_AR target week 2



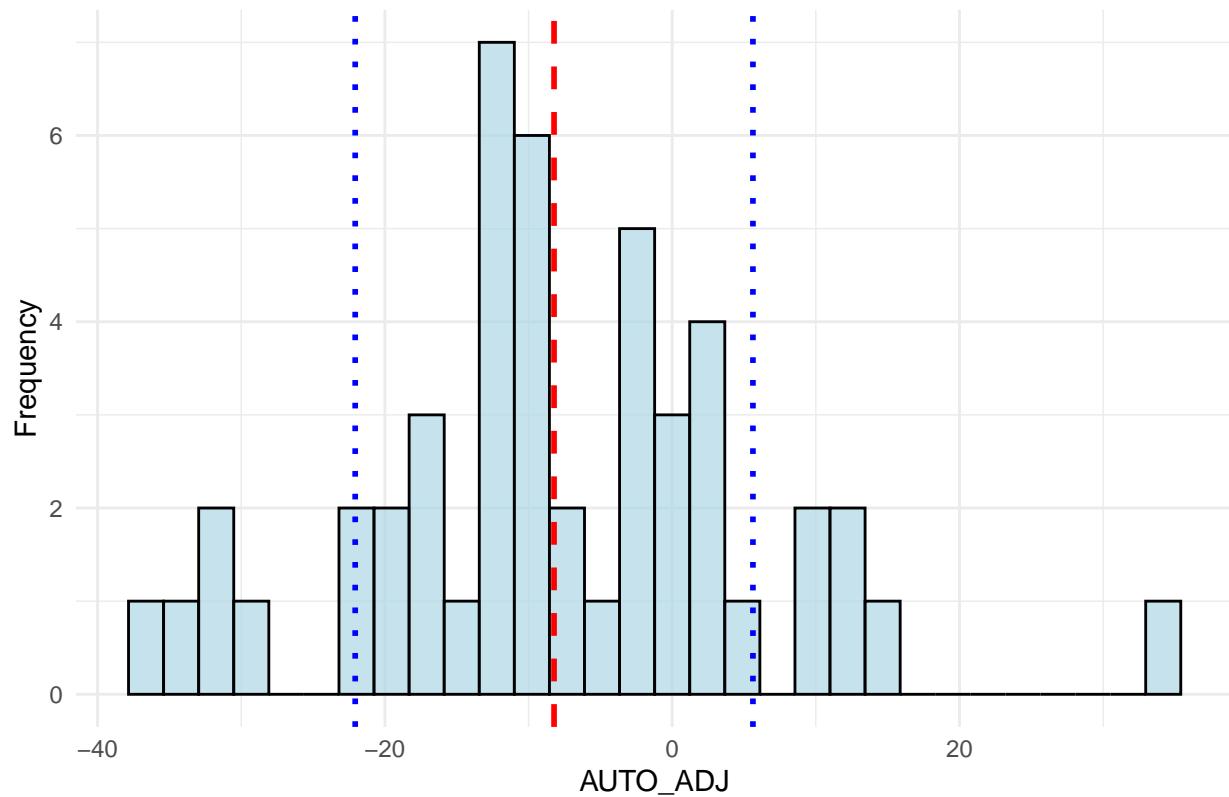
Histogram of ES27\_AR target week 2



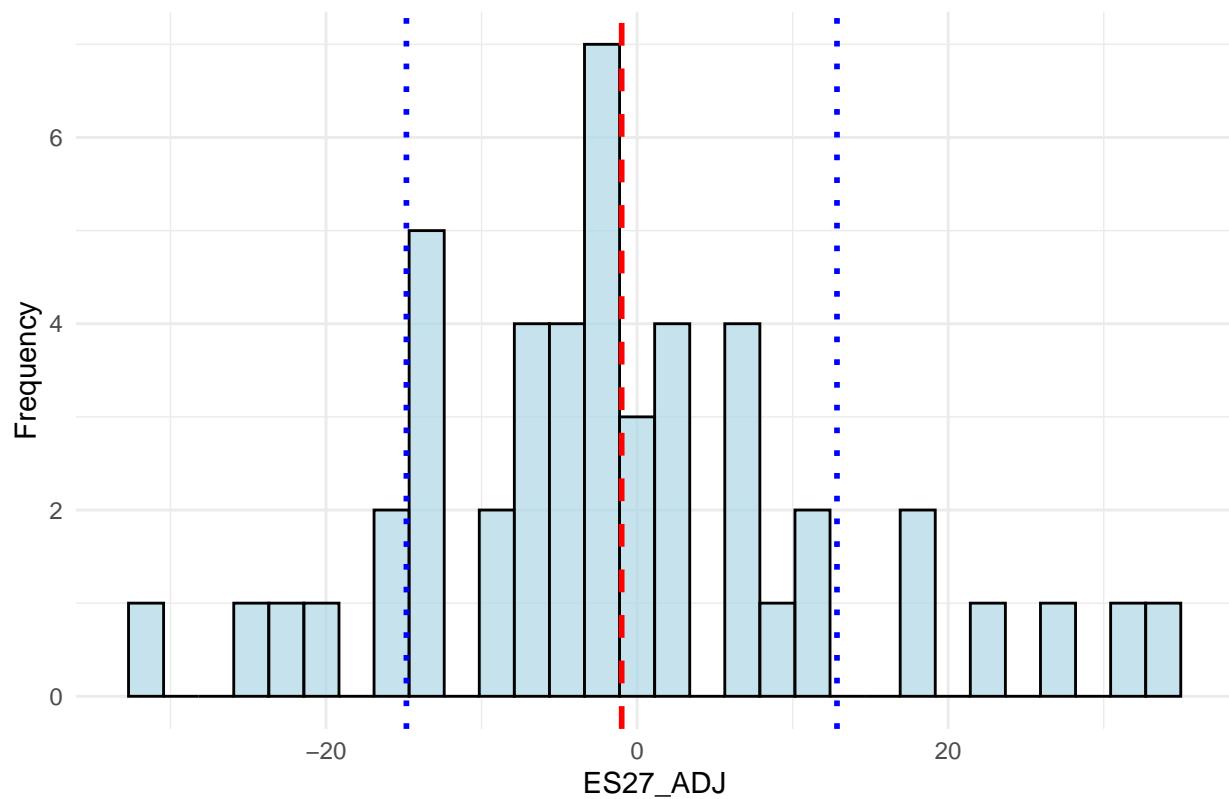
Histogram of ES64\_AR target week 2



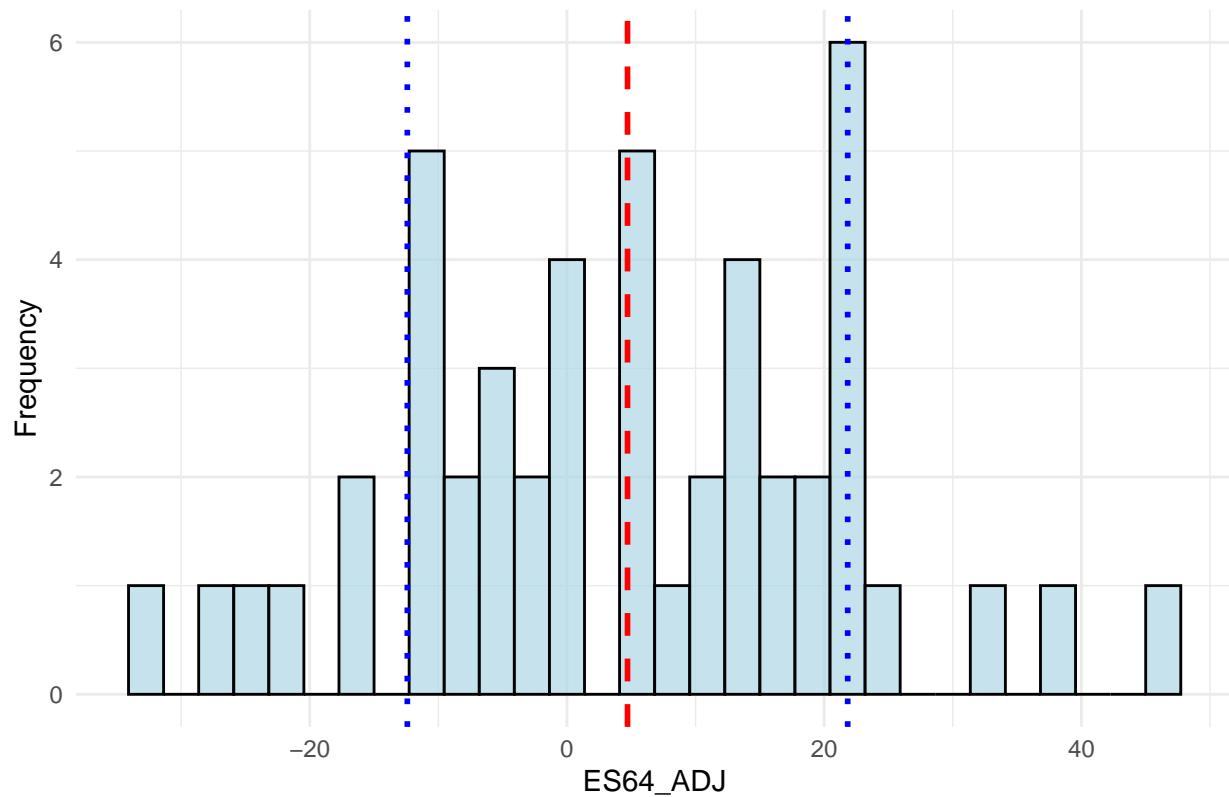
Histogram of AUTO\_ADJ target week 2



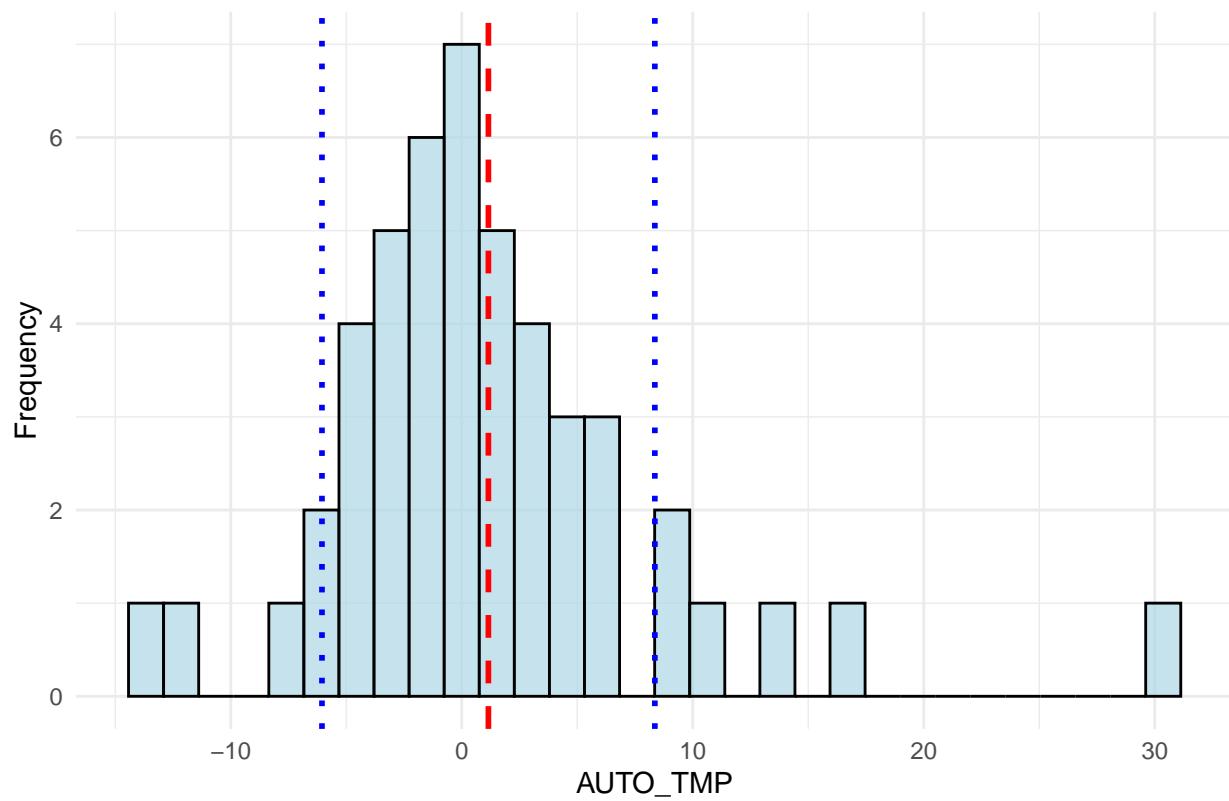
Histogram of ES27\_ADJ target week 2



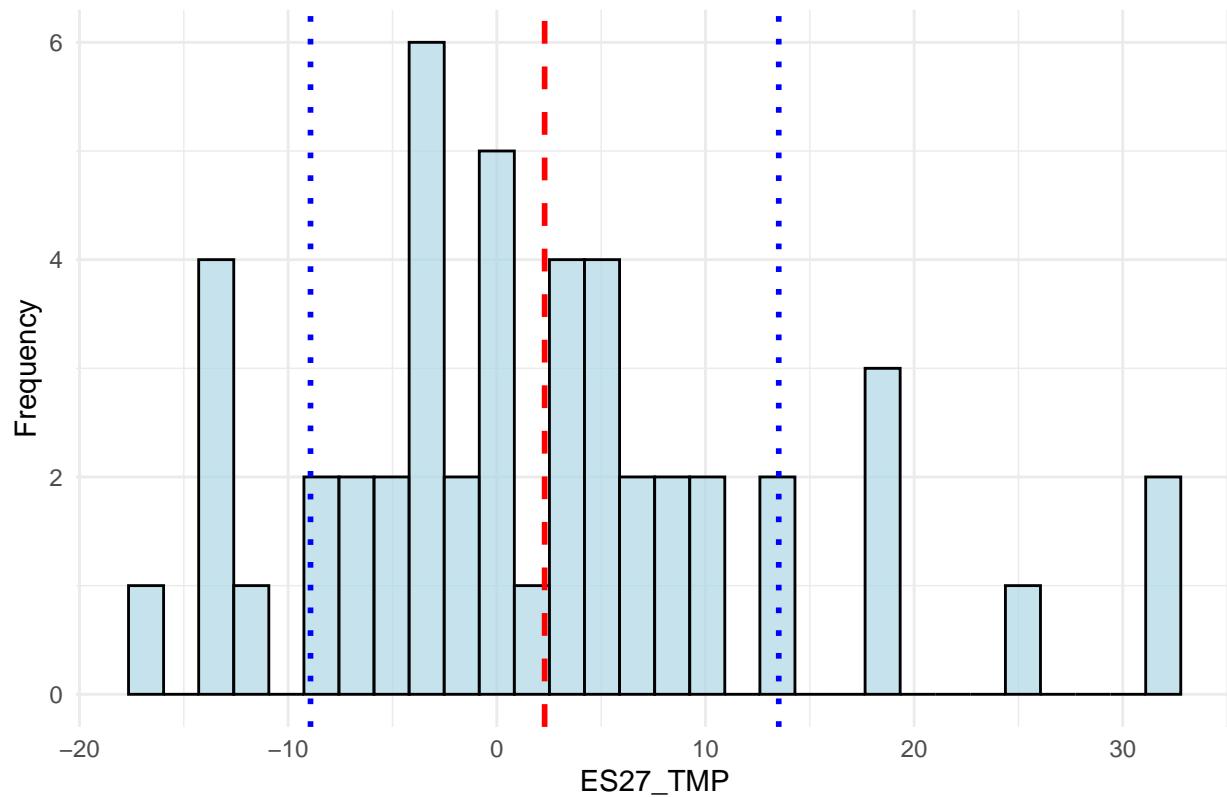
Histogram of ES64\_ADJ target week 2



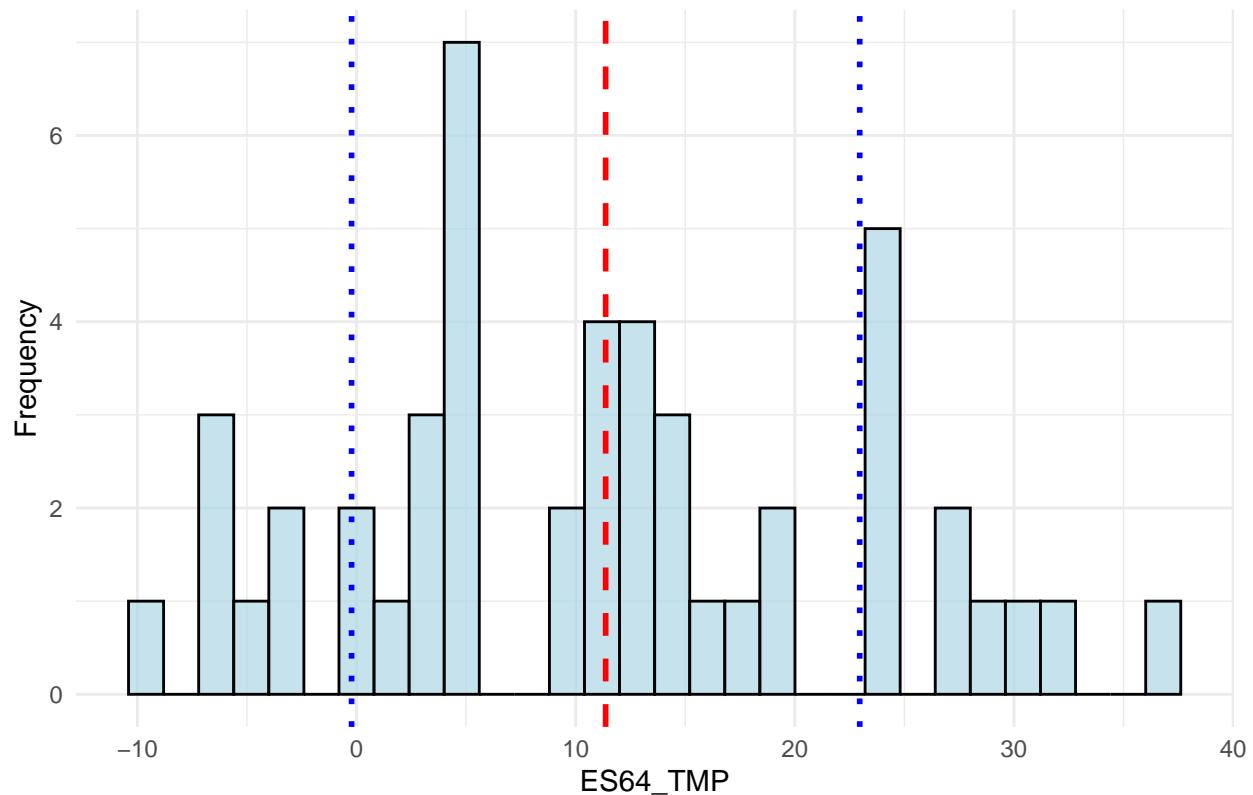
Histogram of AUTO\_TMP target week 2



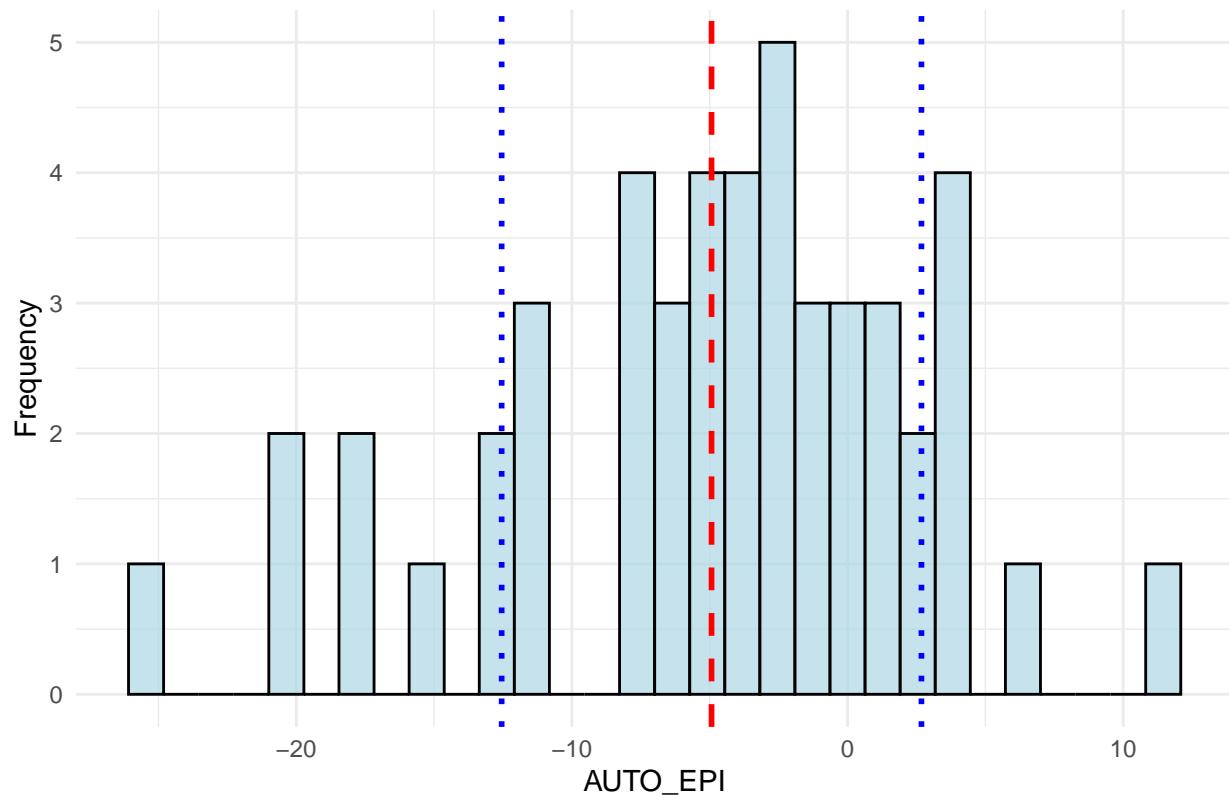
Histogram of ES27\_TMP target week 2



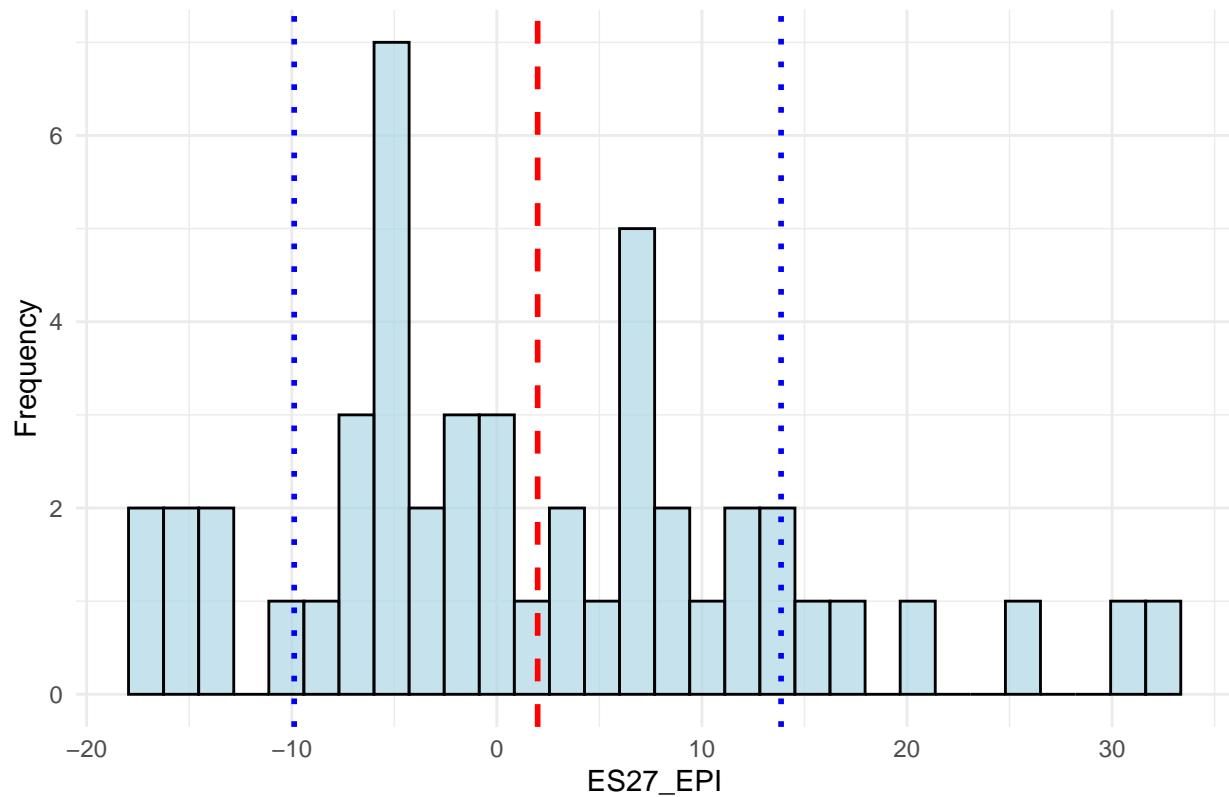
Histogram of ES64\_TMP target week 2



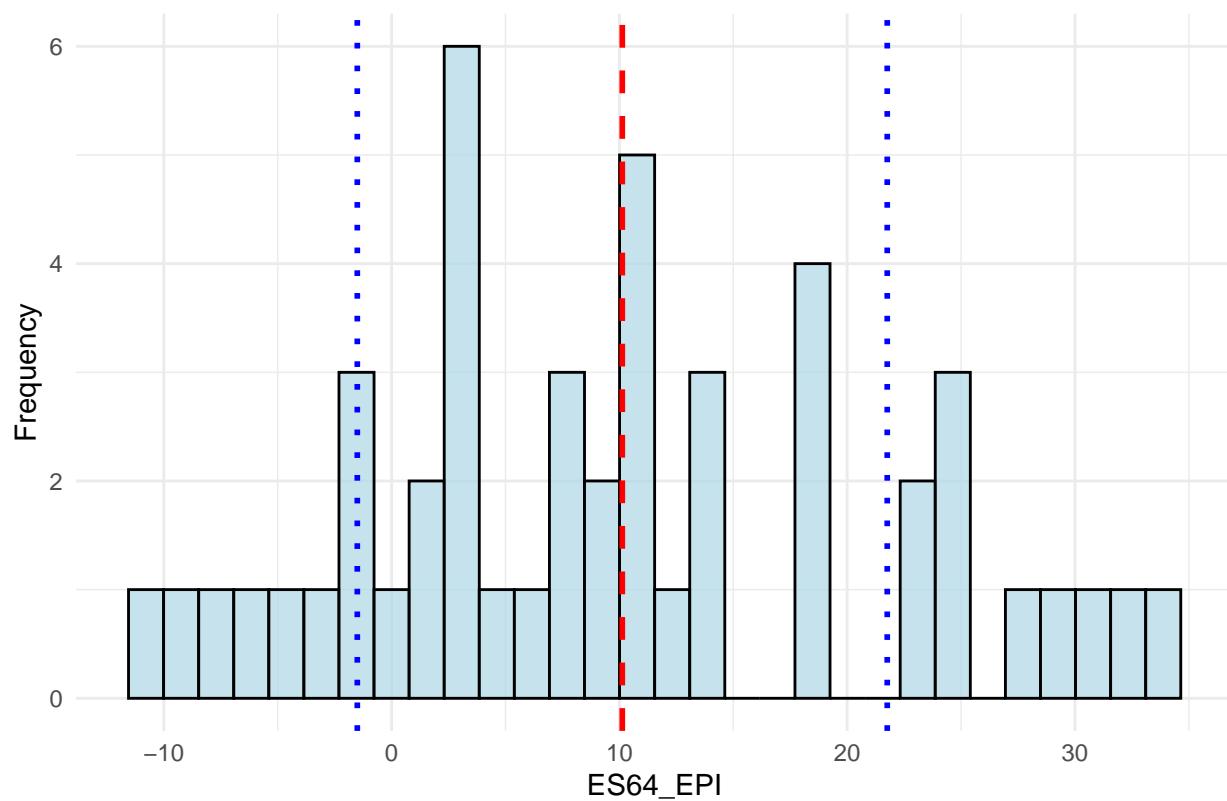
Histogram of AUTO\_EPI target week 2



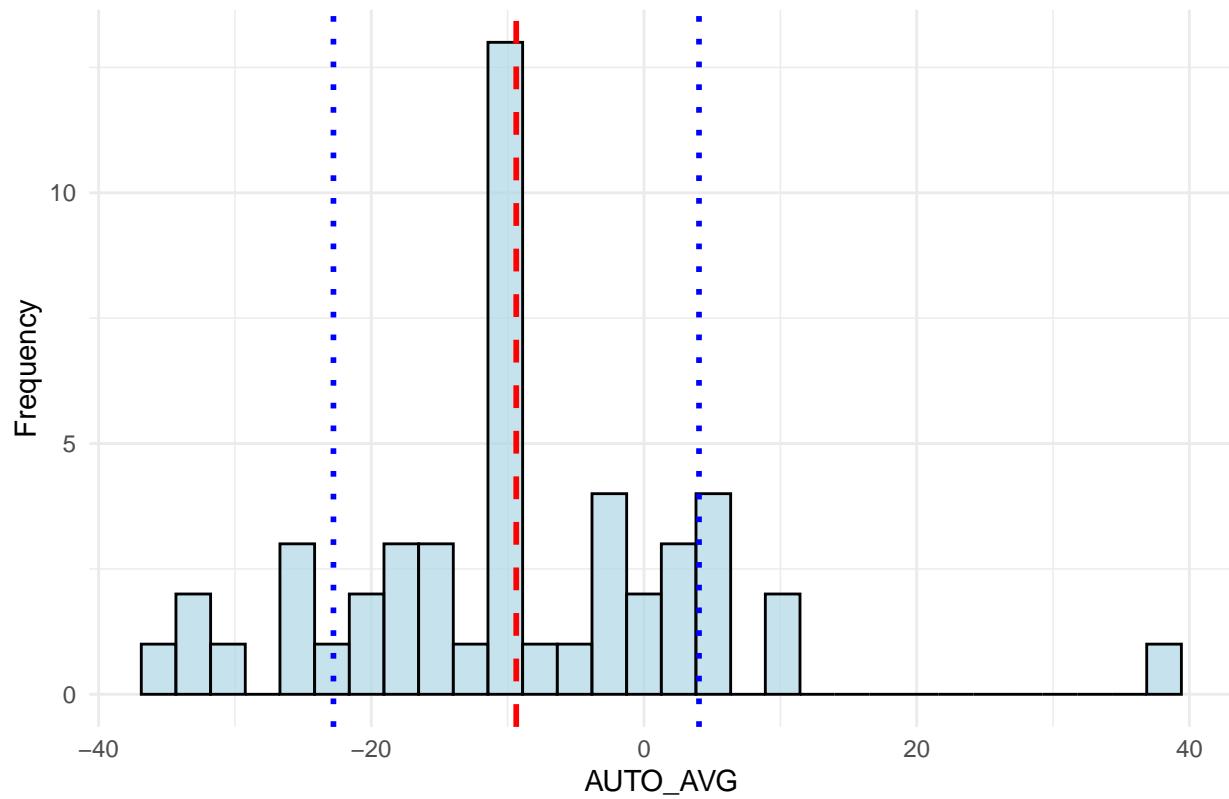
Histogram of ES27\_EPI target week 2



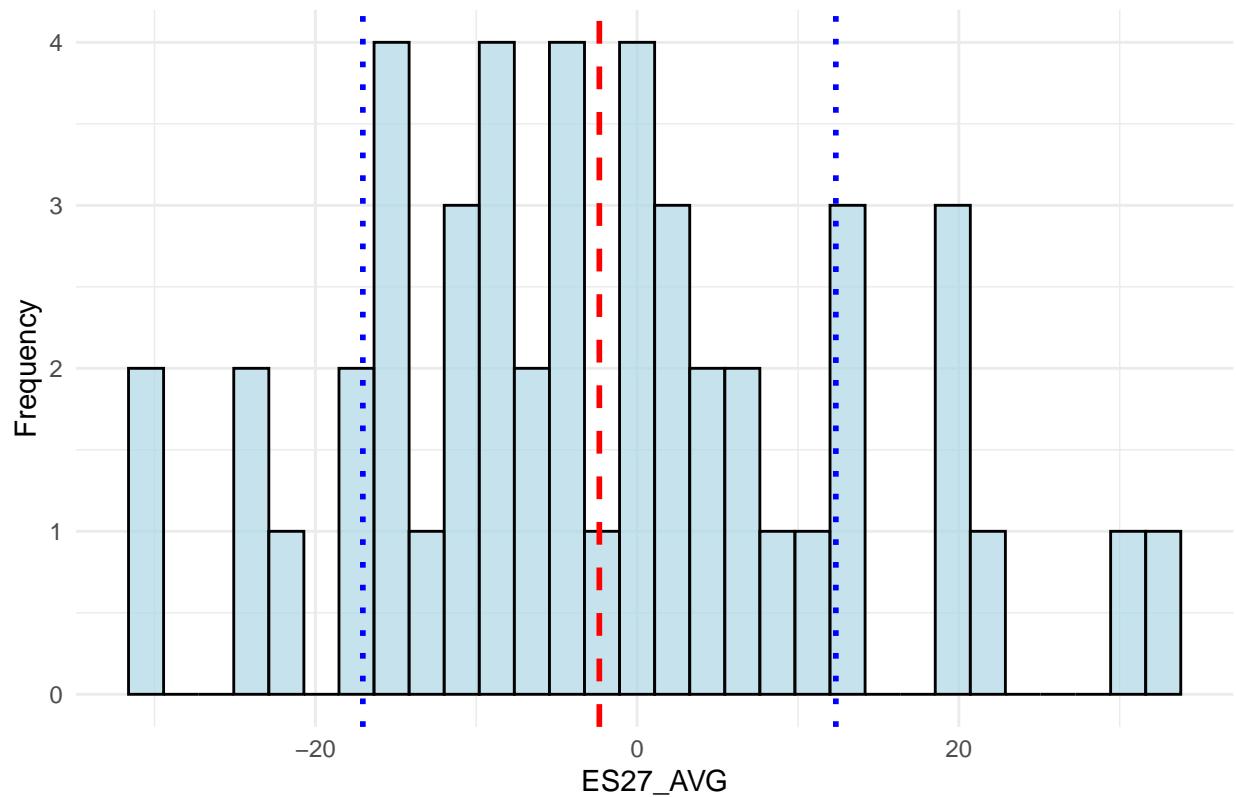
Histogram of ES64\_EPI target week 2



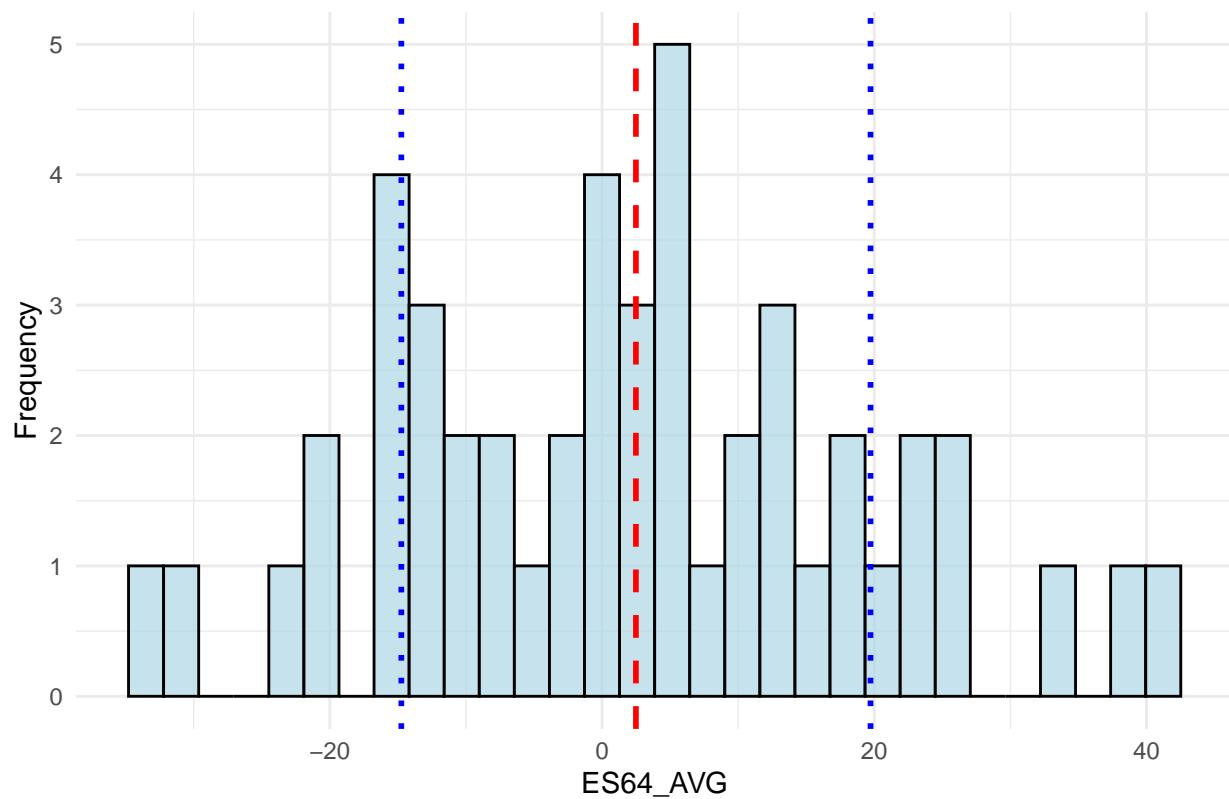
Histogram of AUTO\_AVG target week 2



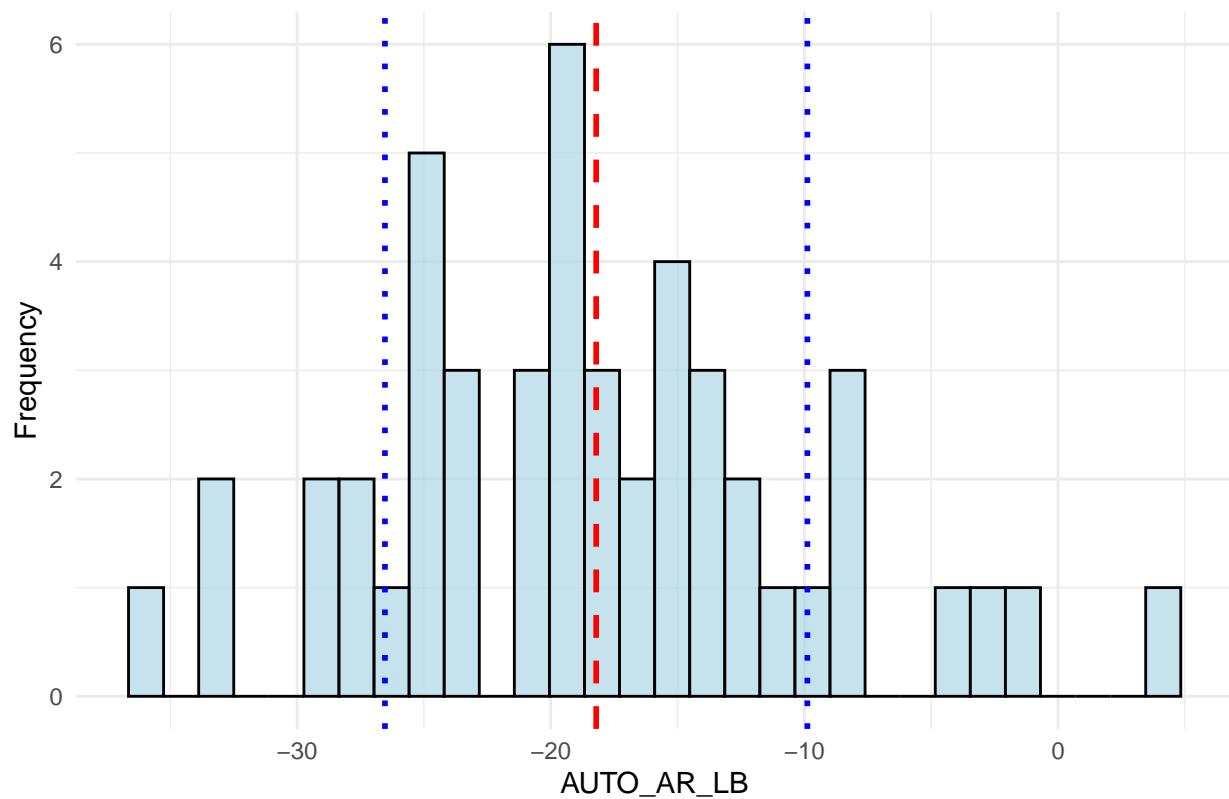
Histogram of ES27\_AVG target week 2



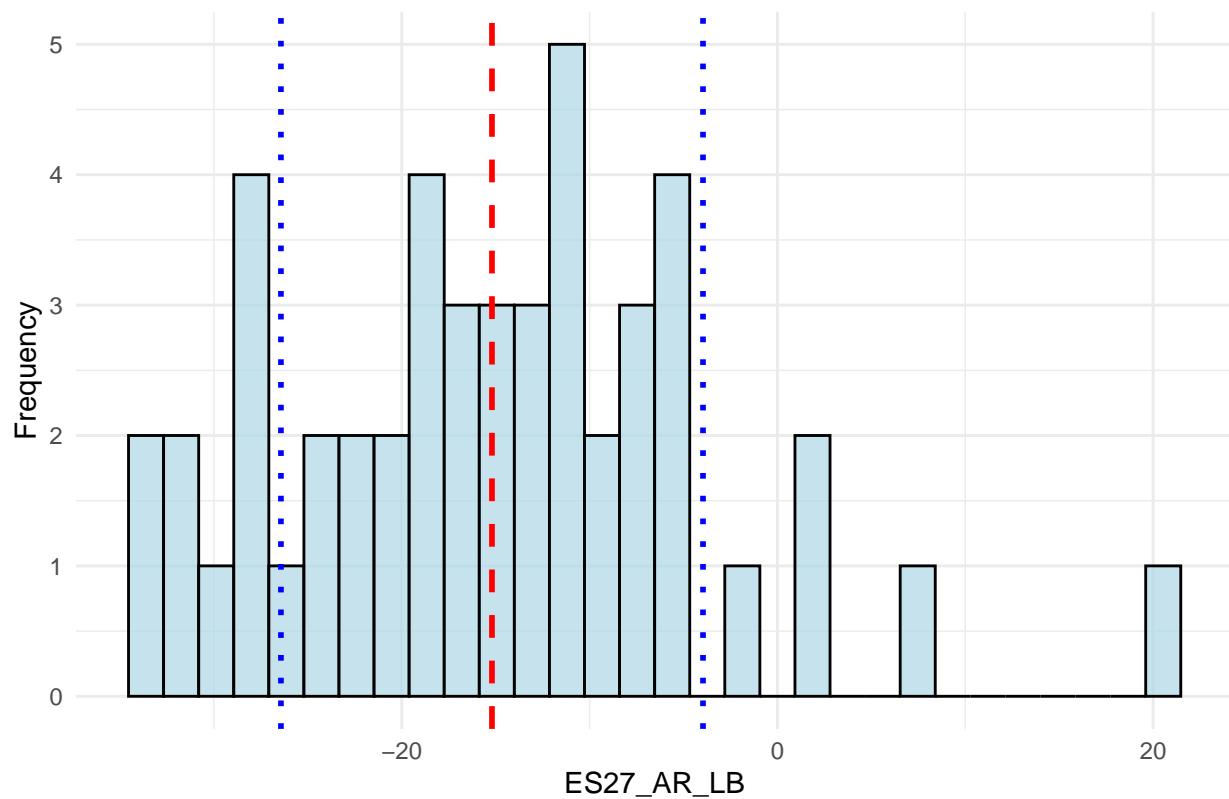
Histogram of ES64\_AVG target week 2



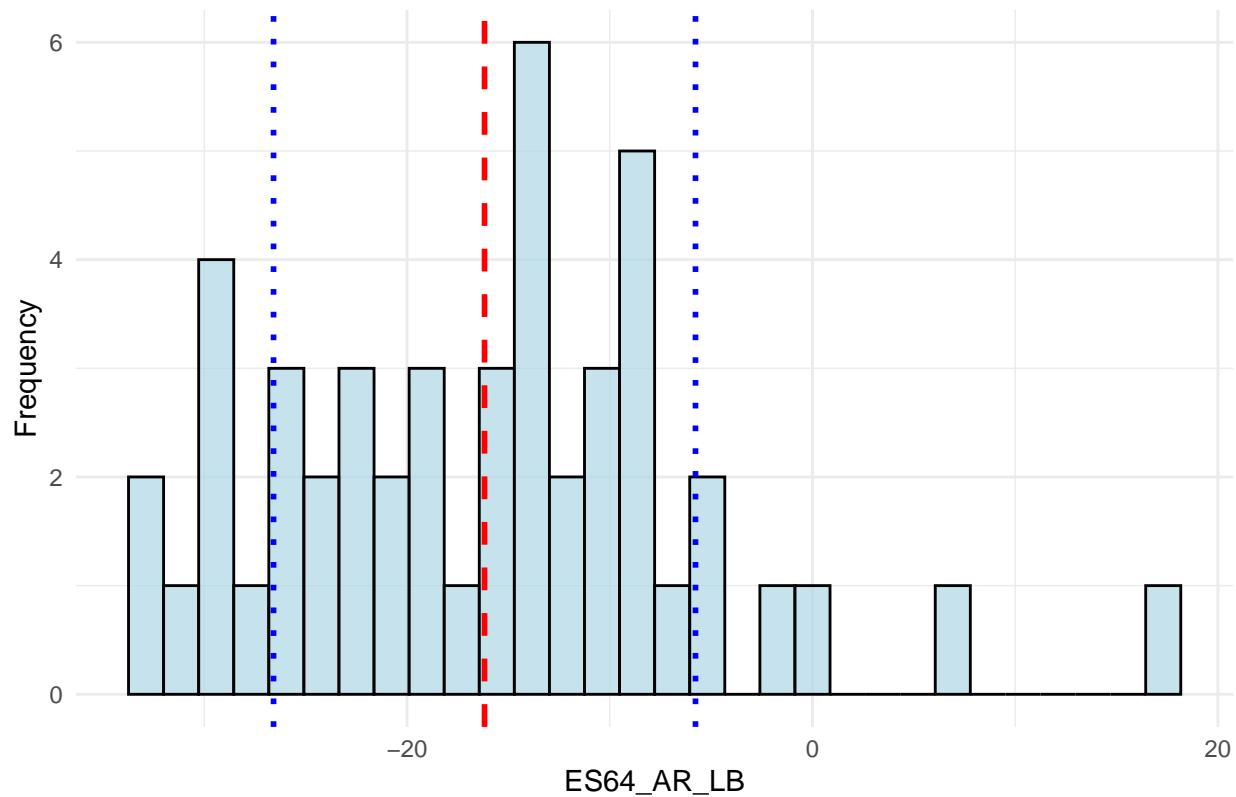
Histogram of AUTO\_AR\_LB target week 2



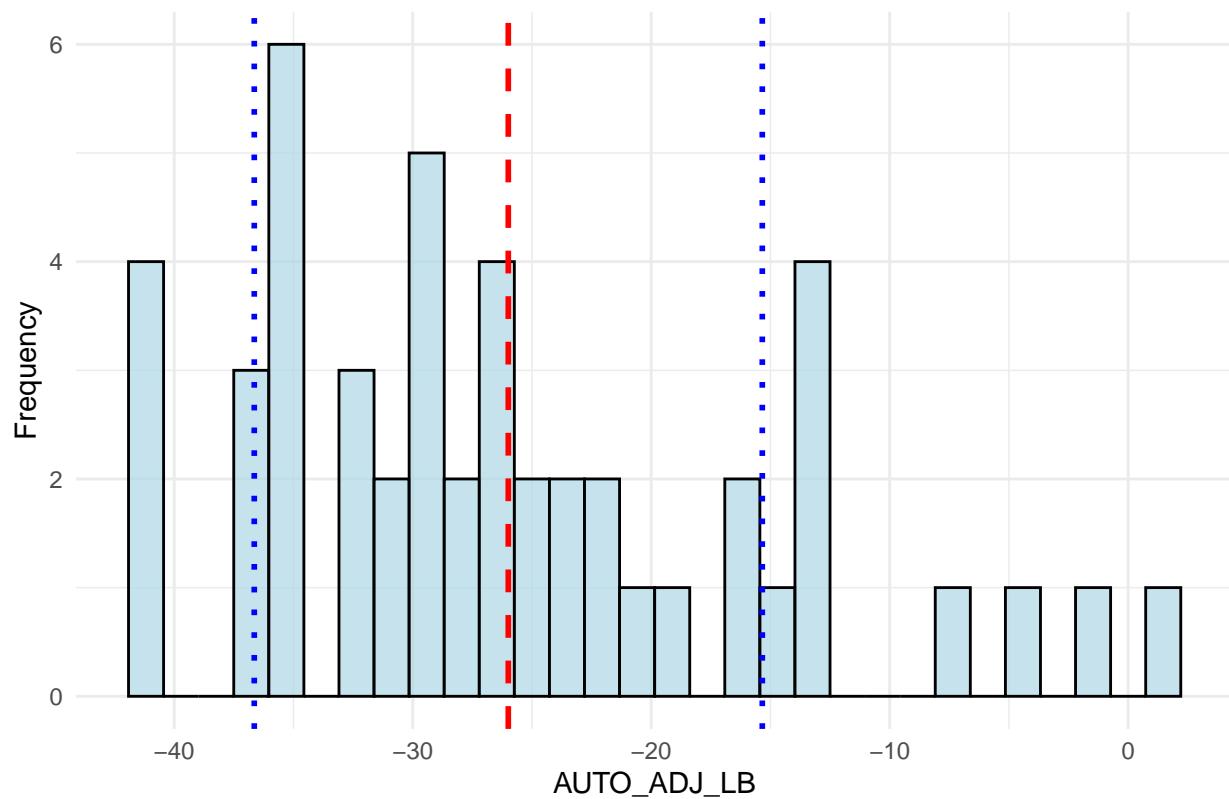
Histogram of ES27\_AR\_LB target week 2



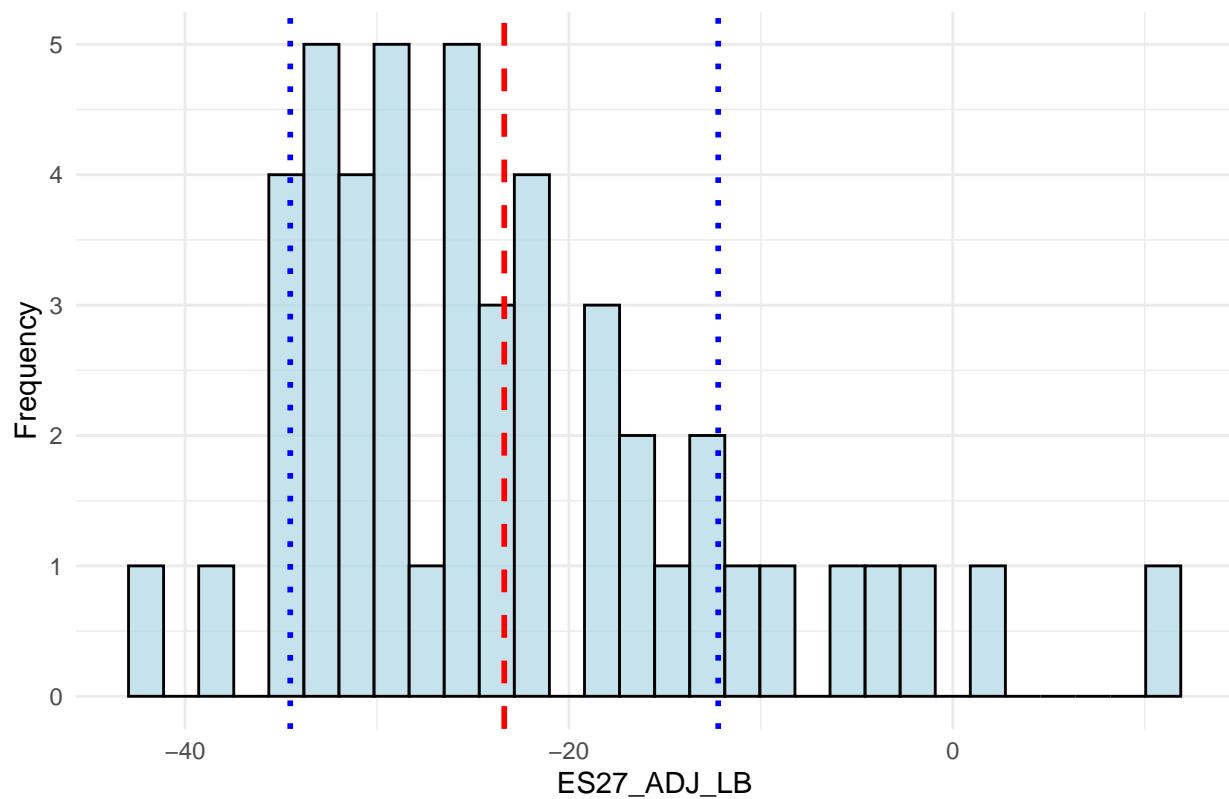
Histogram of ES64\_AR\_LB target week 2



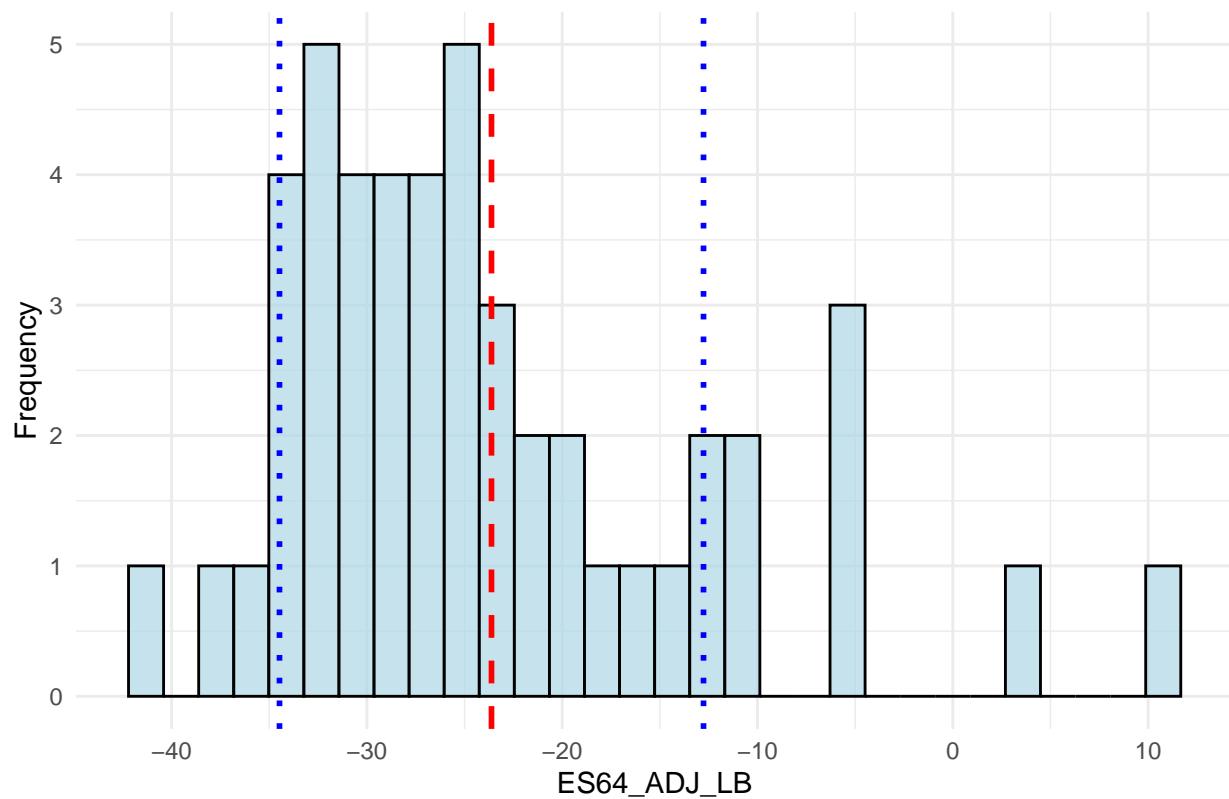
Histogram of AUTO\_ADJ\_LB target week 2



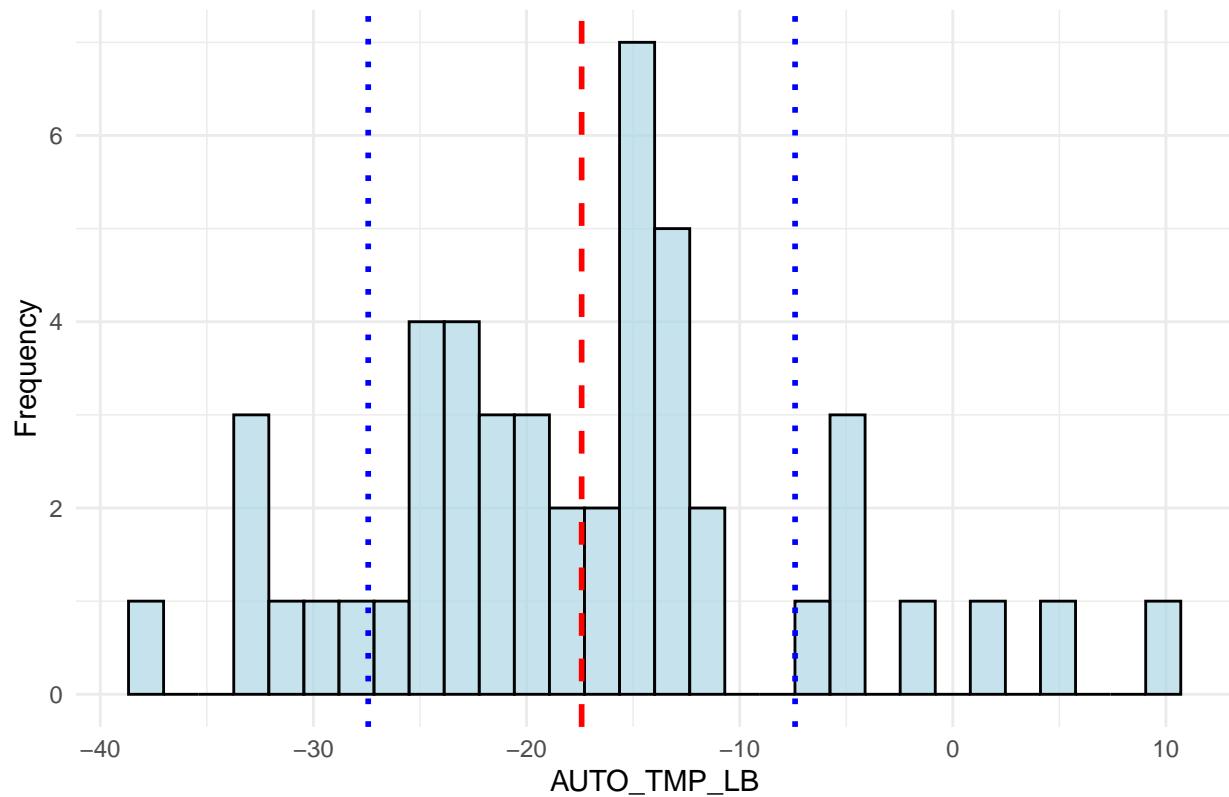
Histogram of ES27\_ADJ\_LB target week 2



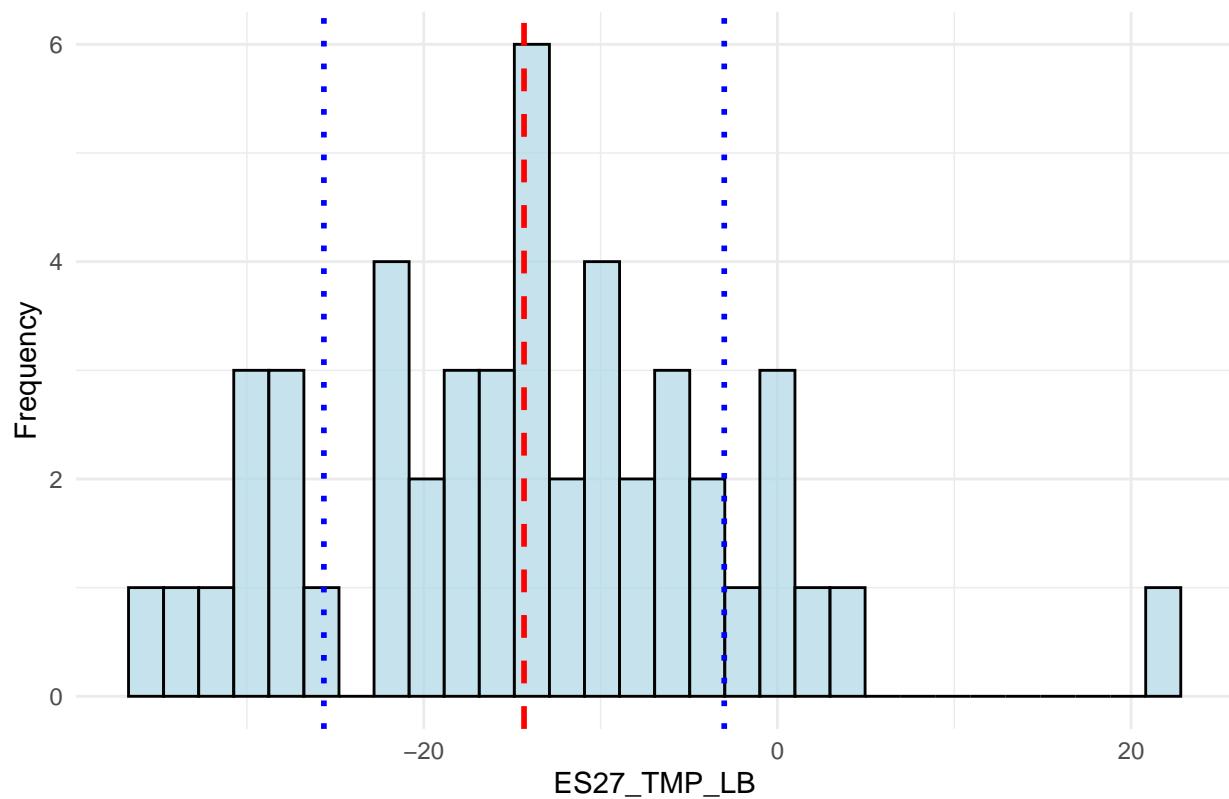
Histogram of ES64\_ADJ\_LB target week 2



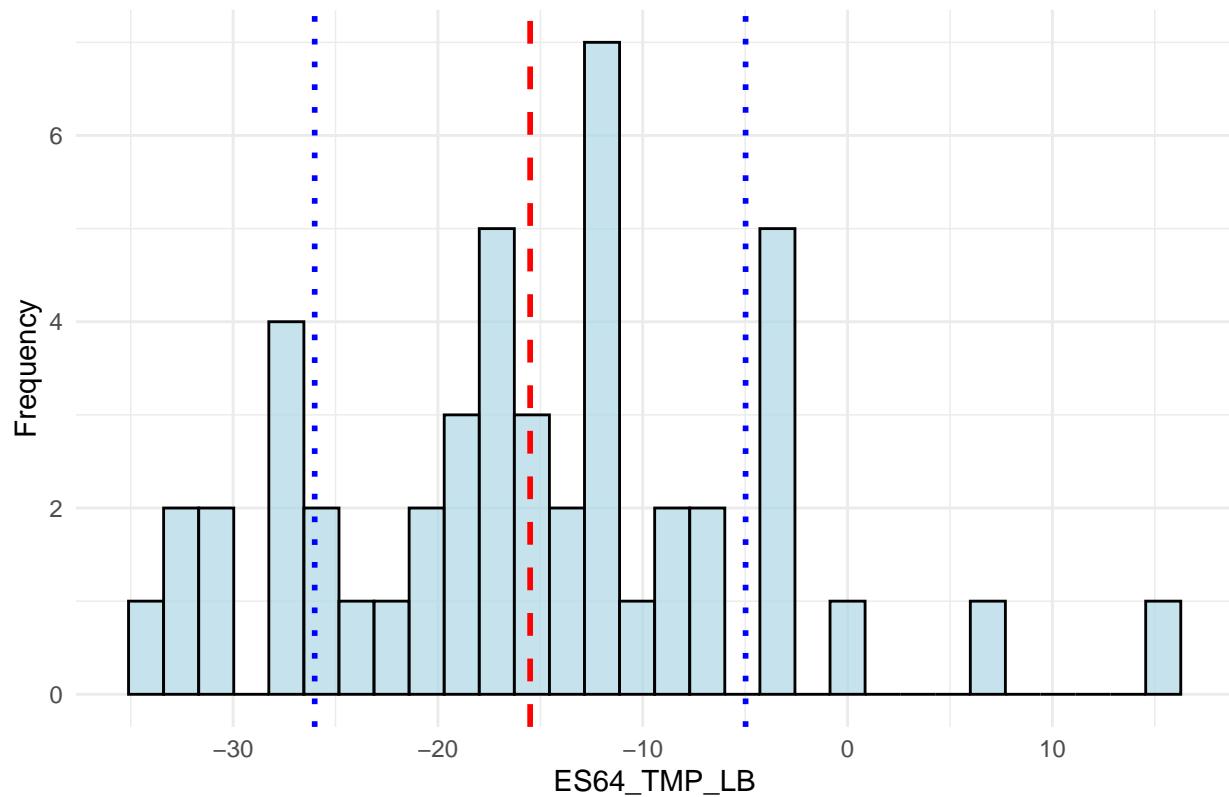
Histogram of AUTO\_TMP\_LB target week 2



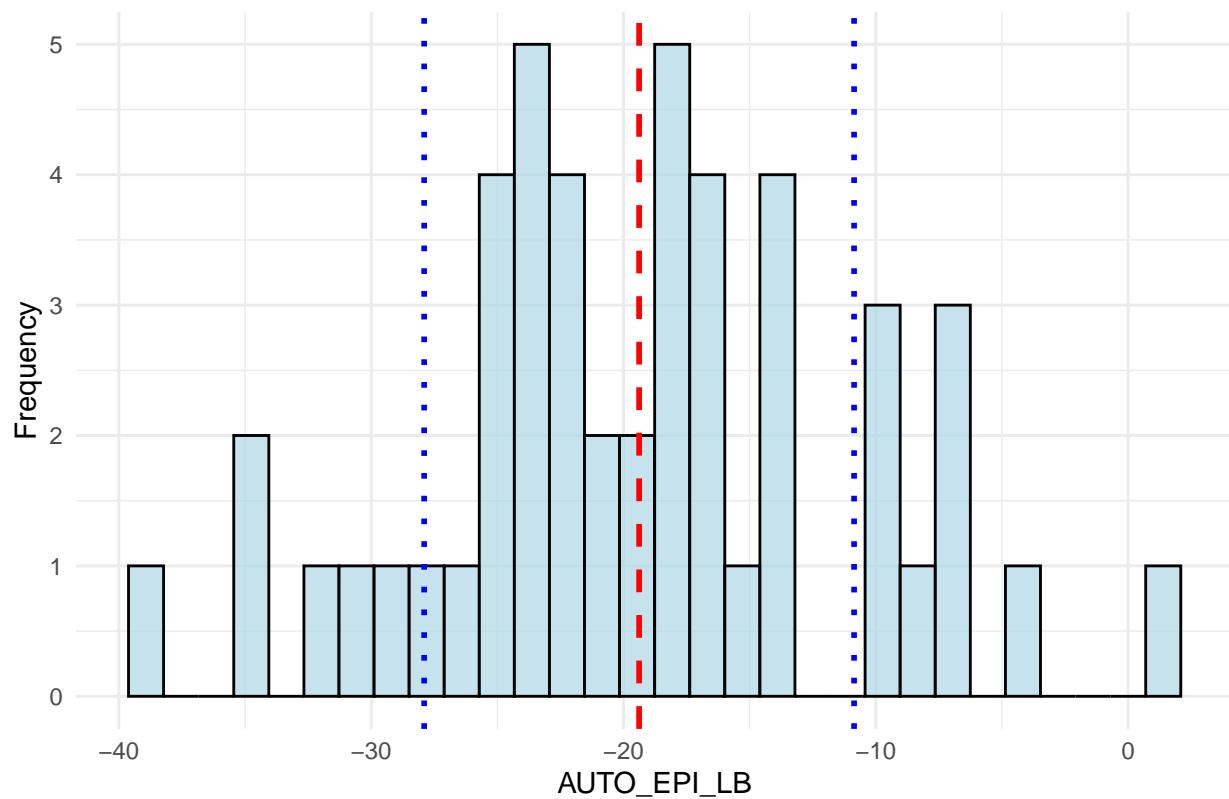
Histogram of ES27\_TMP\_LB target week 2



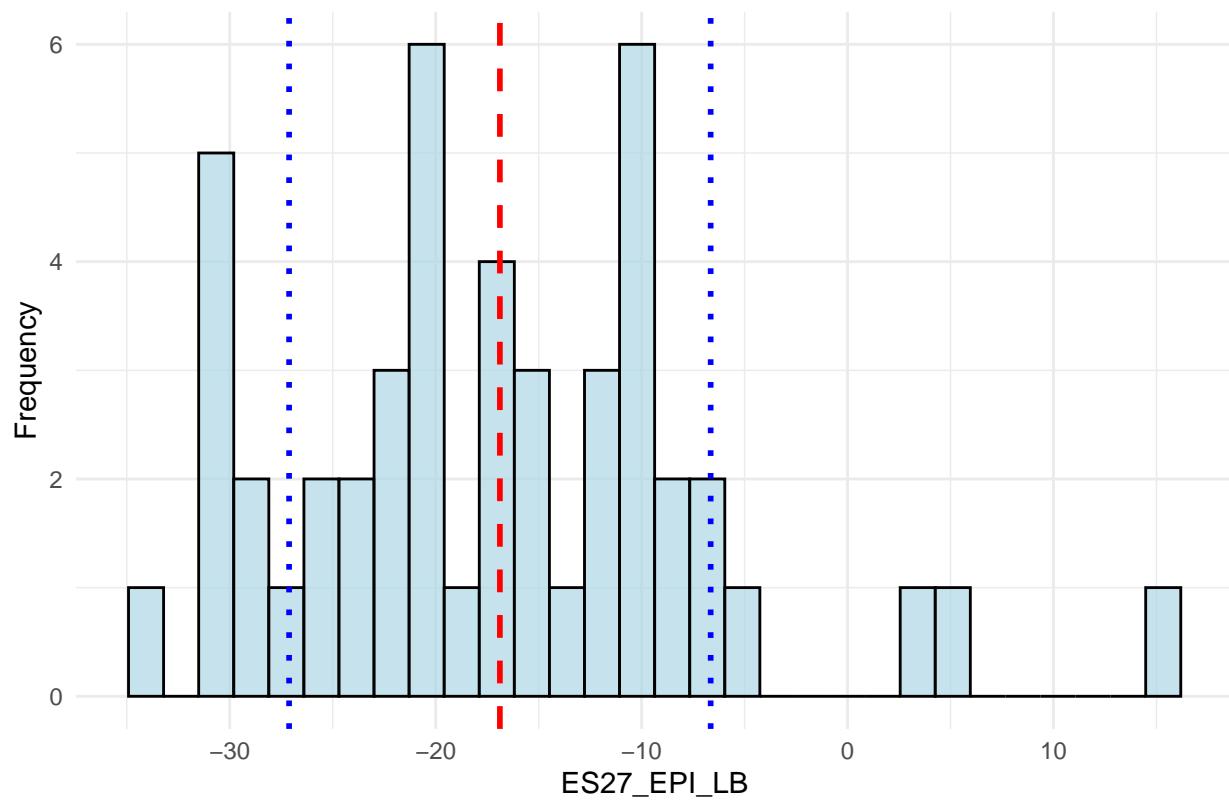
Histogram of ES64\_TMP\_LB target week 2



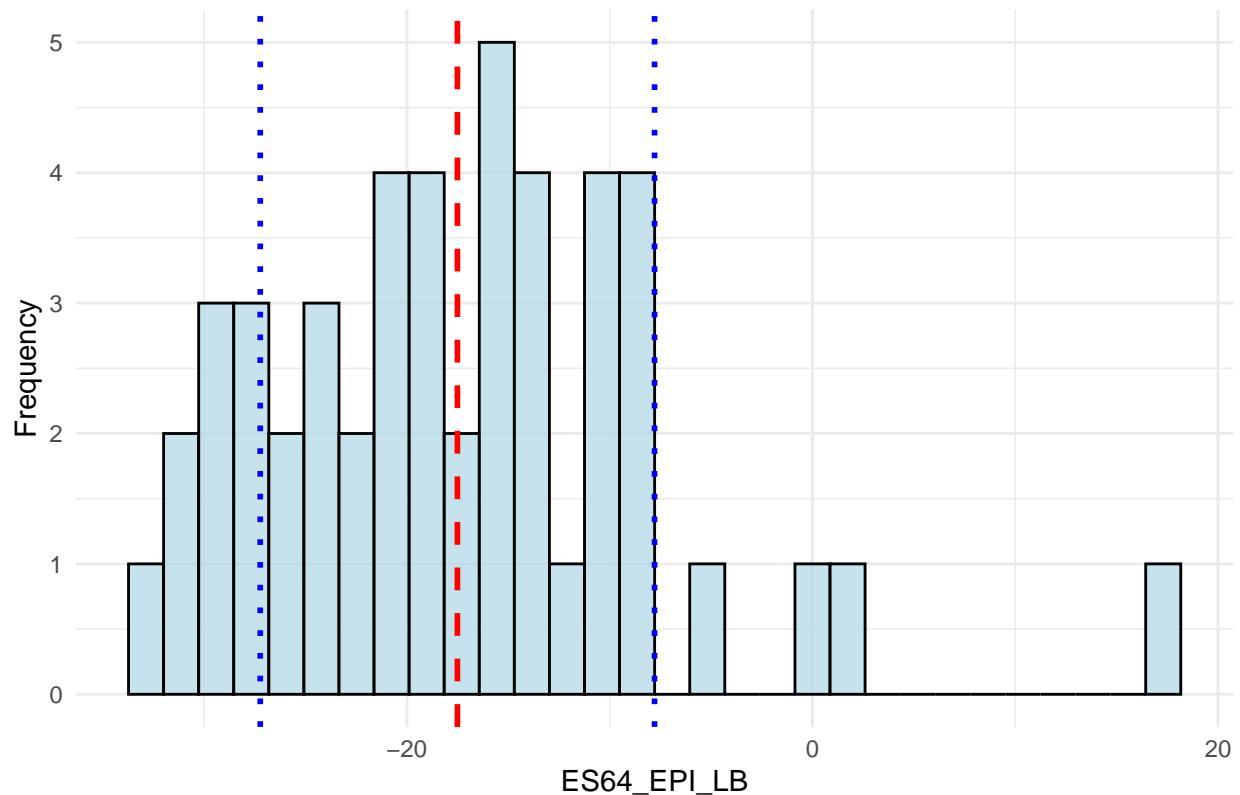
Histogram of AUTO\_EPI\_LB target week 2



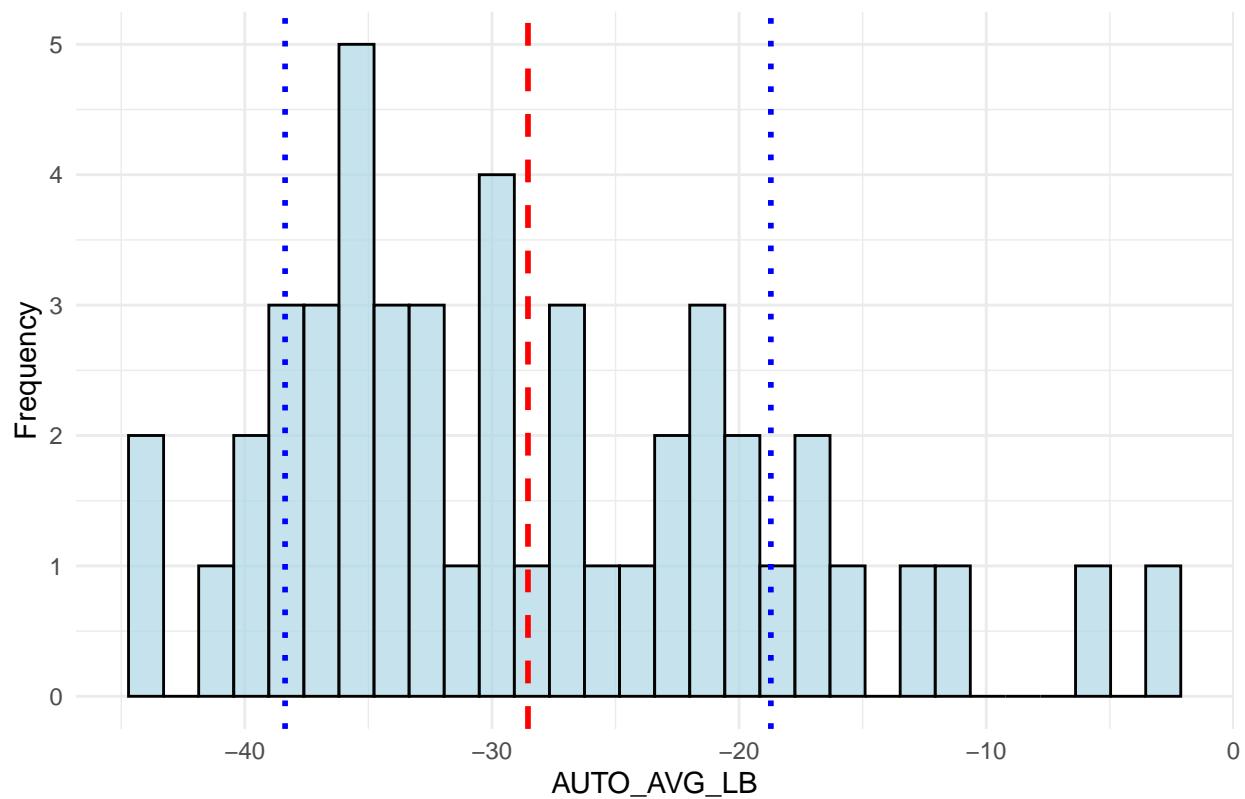
Histogram of ES27\_EPI\_LB target week 2



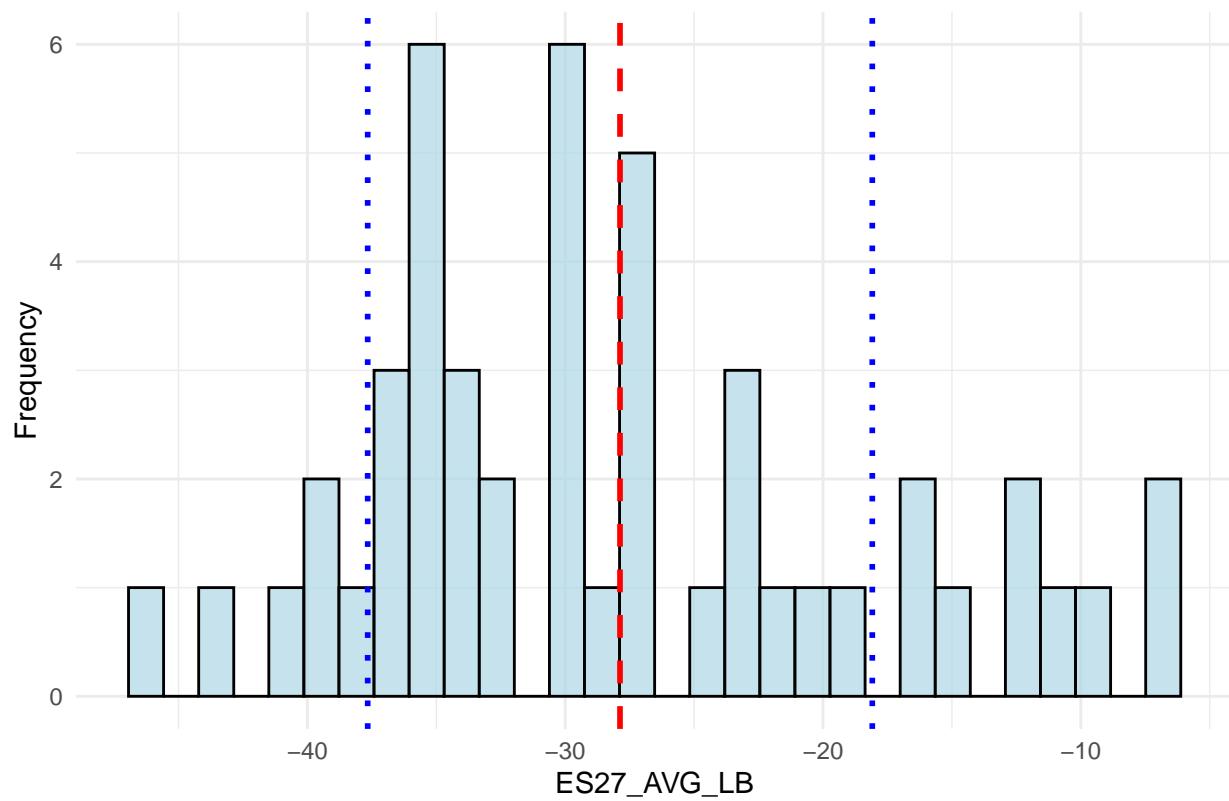
Histogram of ES64\_EPI\_LB target week 2



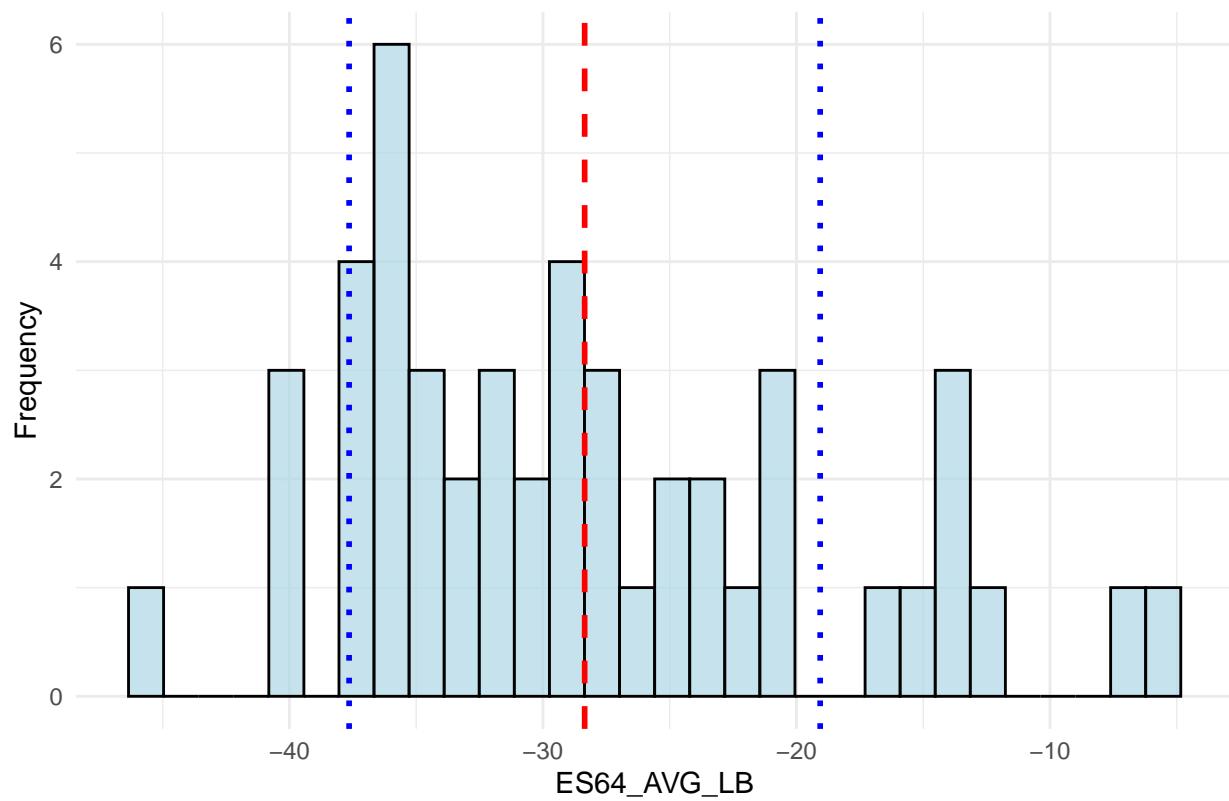
Histogram of AUTO\_AVG\_LB target week 2



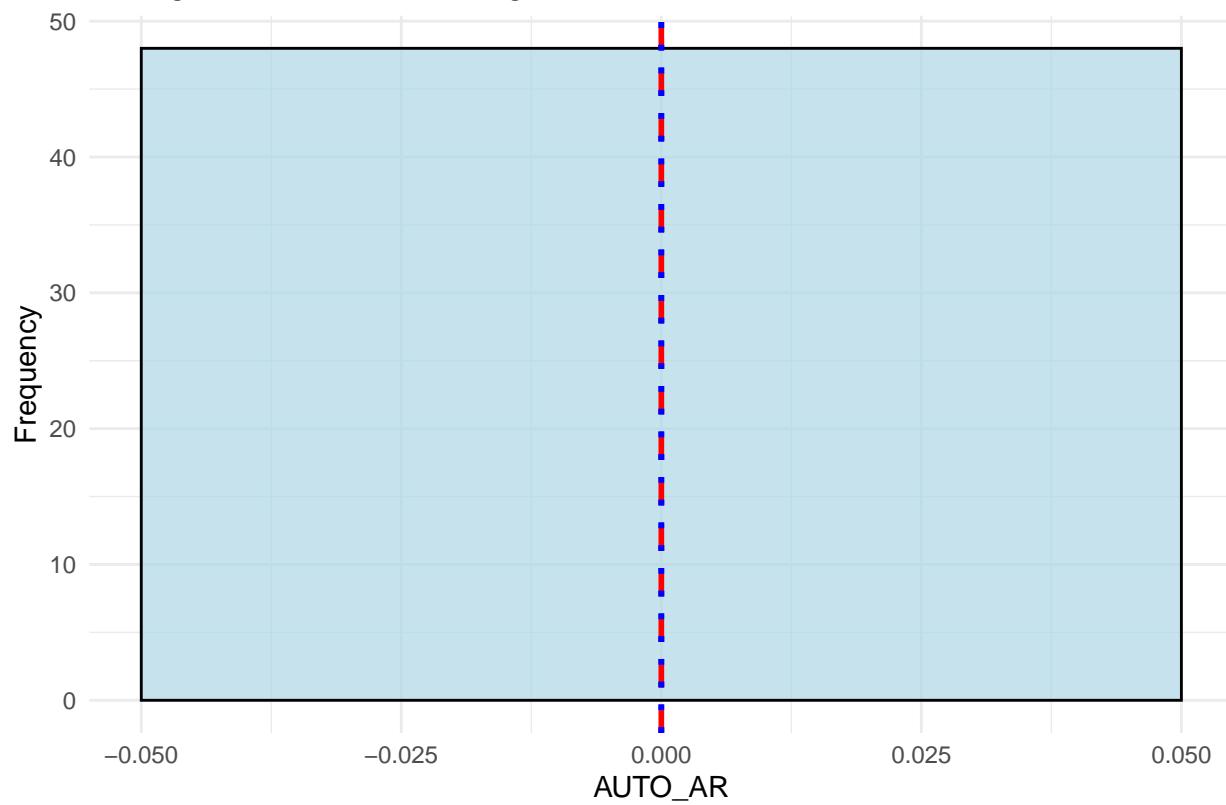
Histogram of ES27\_AVG\_LB target week 2



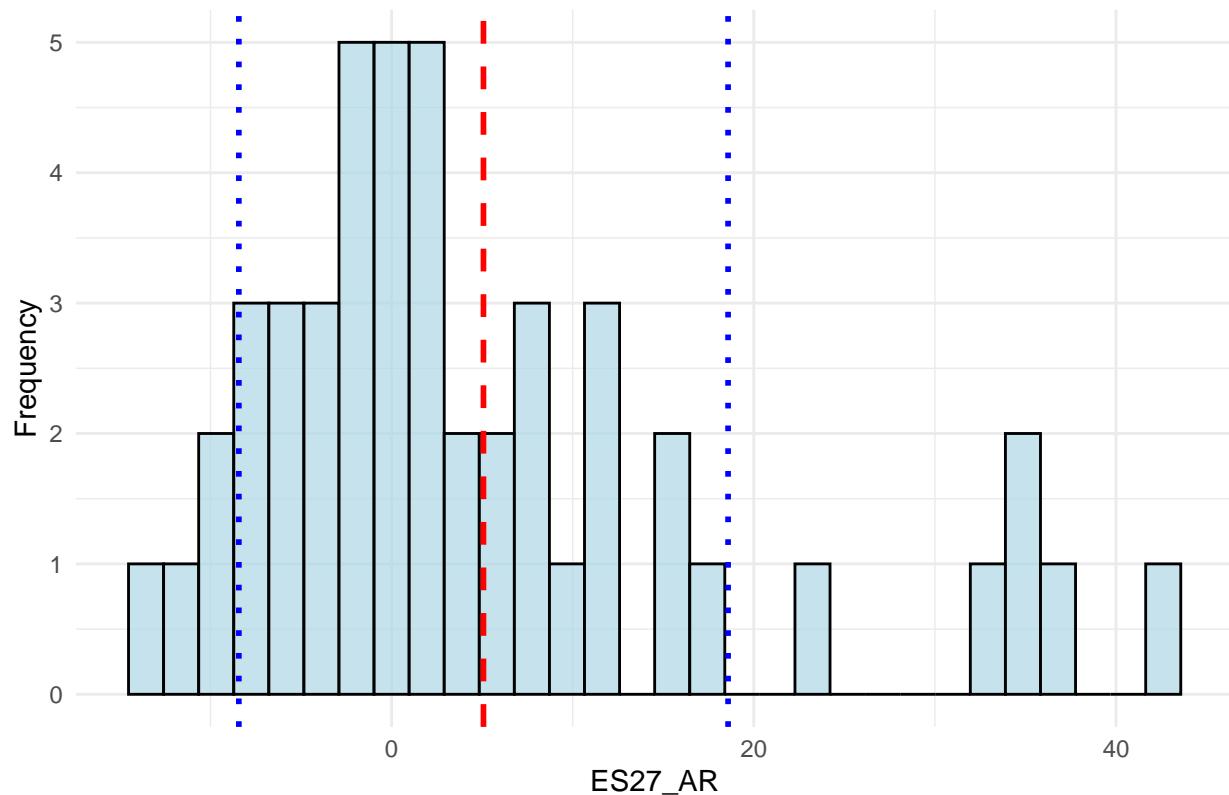
Histogram of ES64\_AVG\_LB target week 2



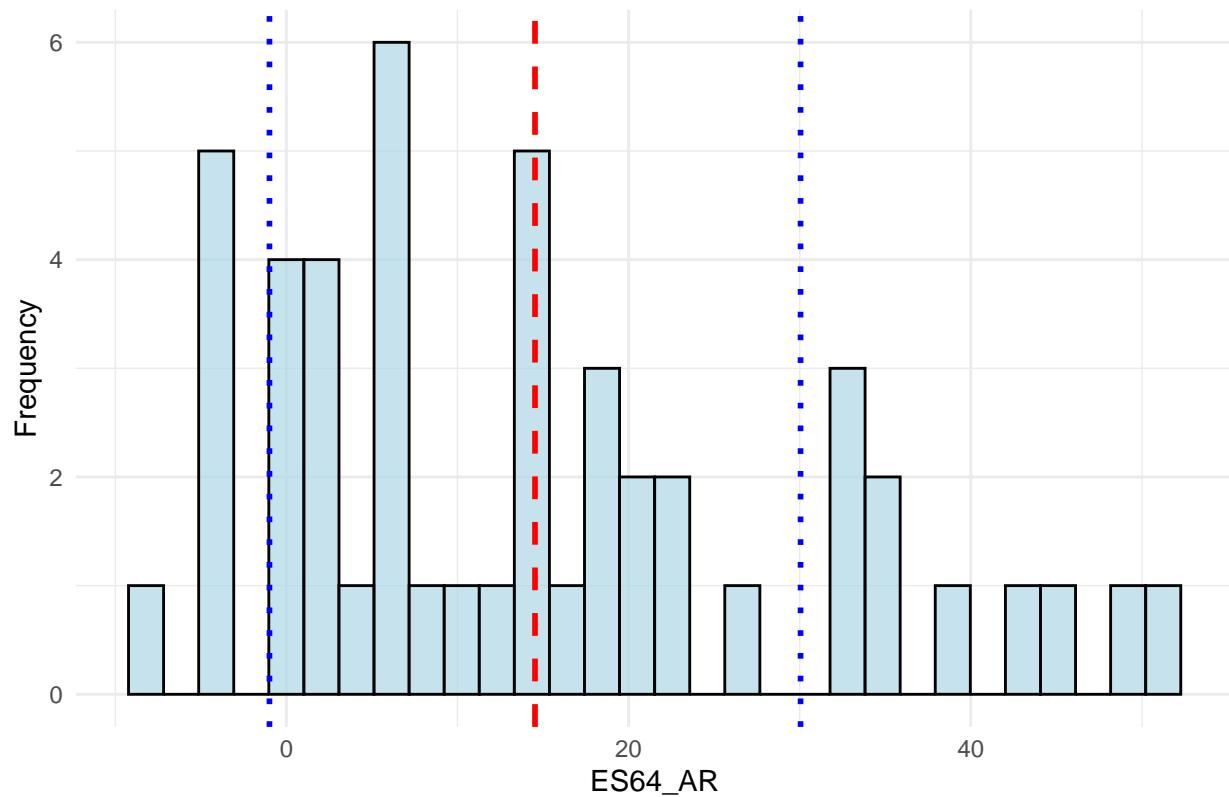
Histogram of AUTO\_AR target week 3



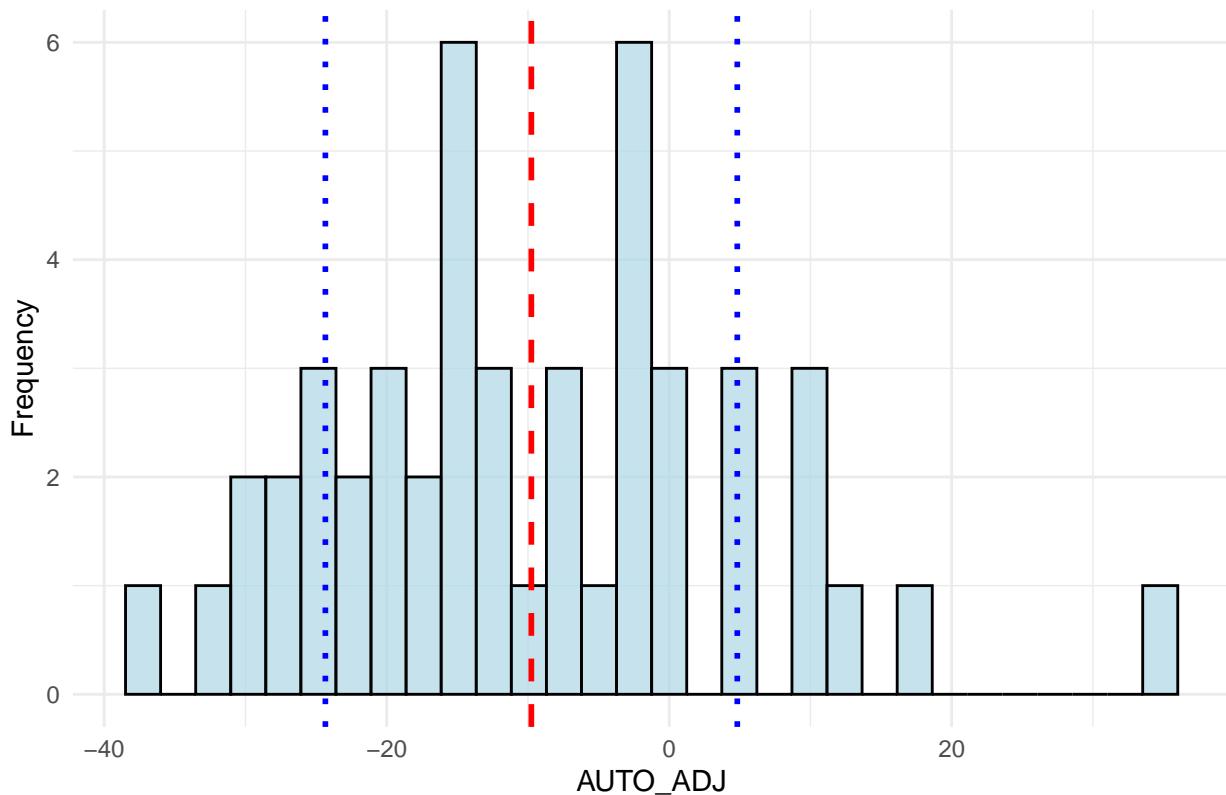
Histogram of ES27\_AR target week 3



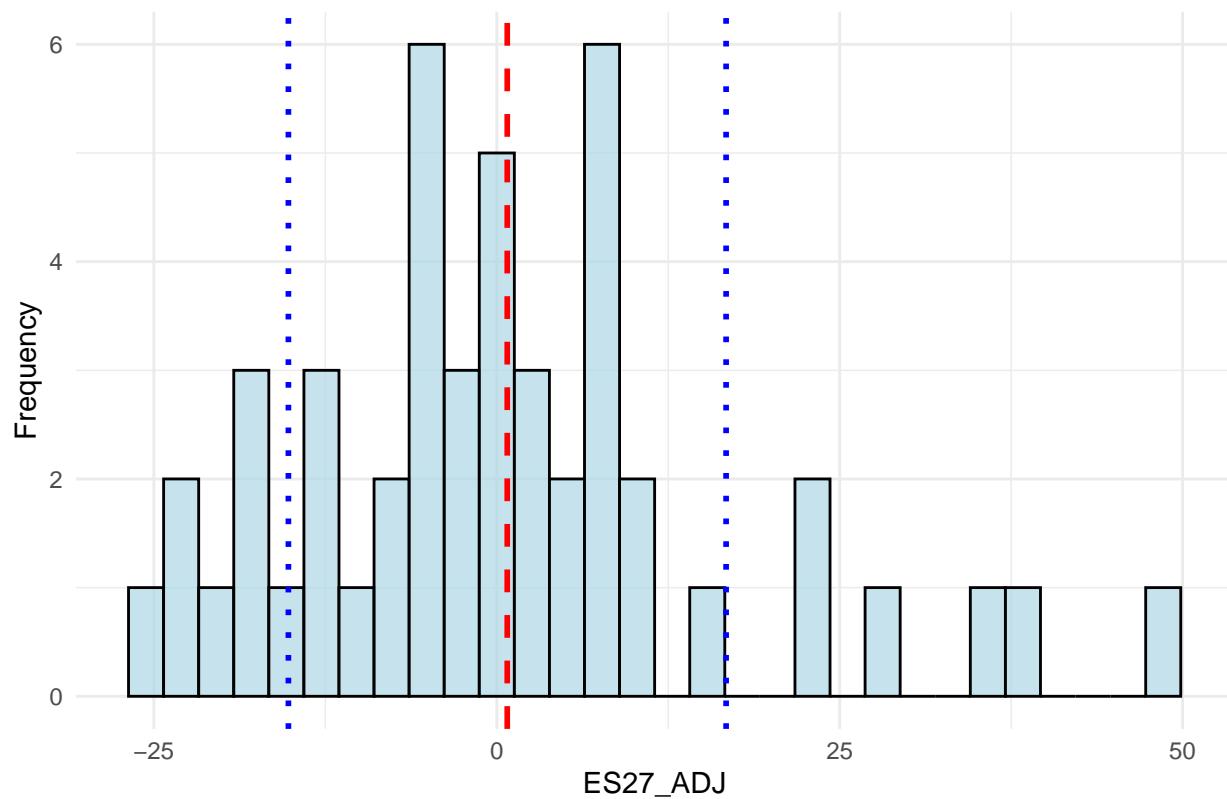
Histogram of ES64\_AR target week 3



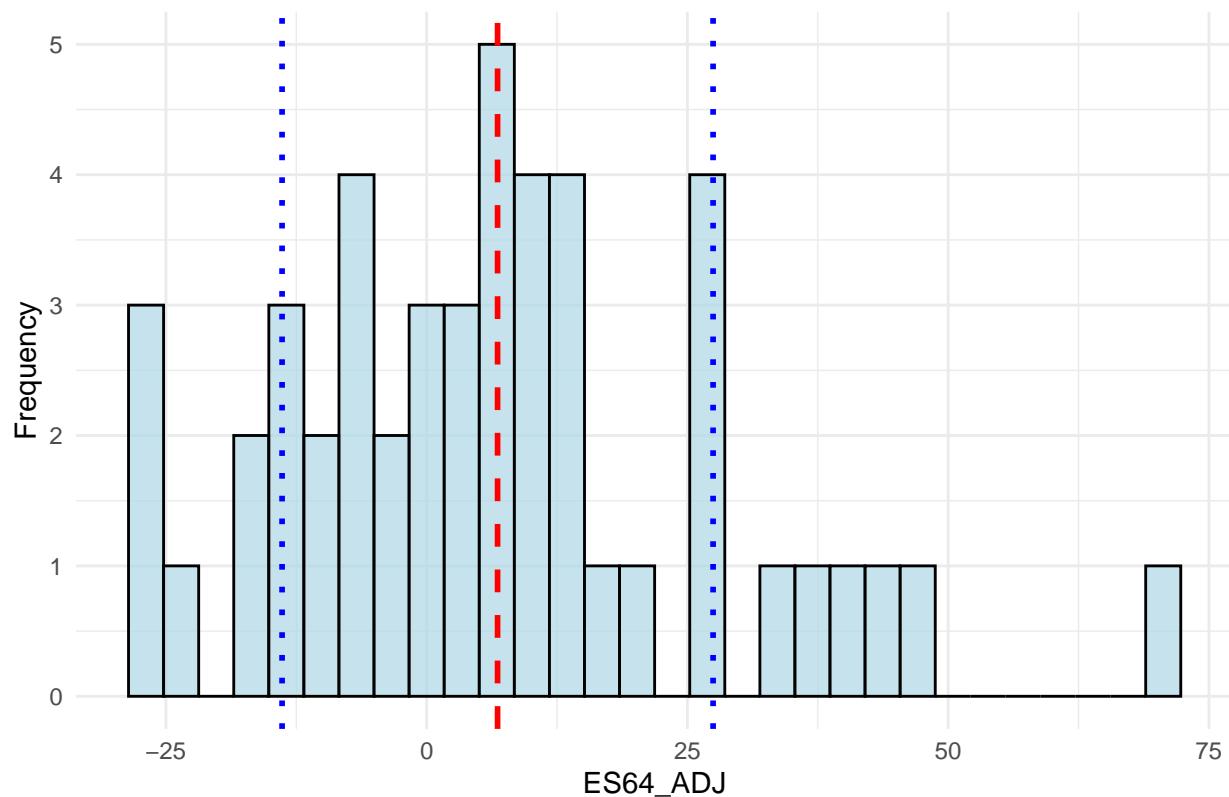
Histogram of AUTO\_ADJ target week 3



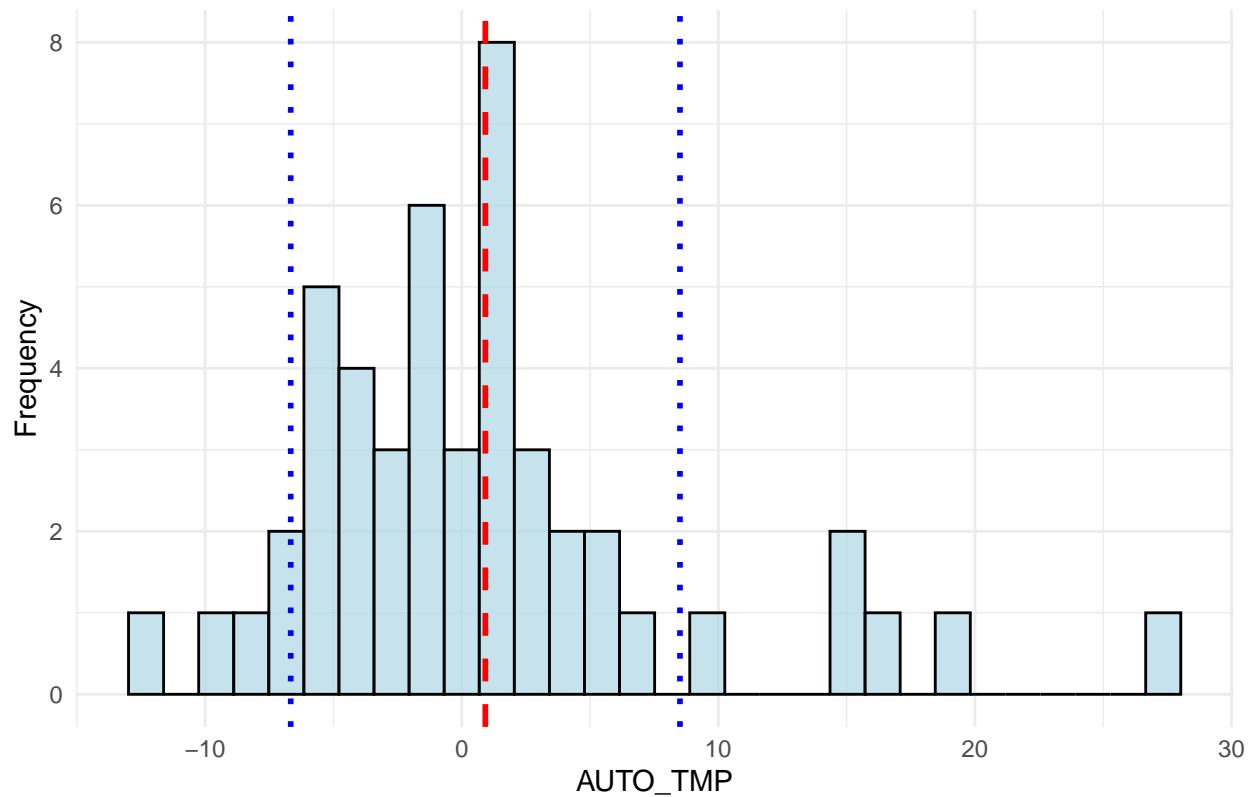
Histogram of ES27\_ADJ target week 3



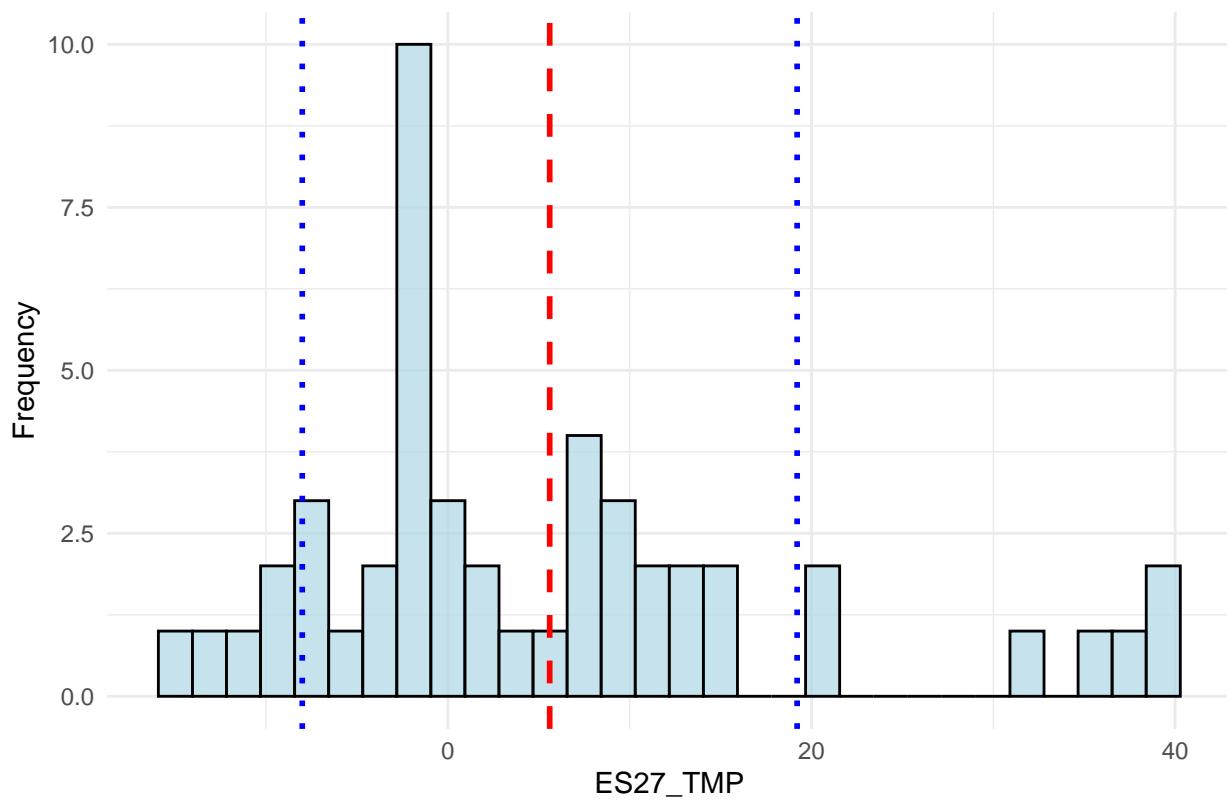
Histogram of ES64\_ADJ target week 3



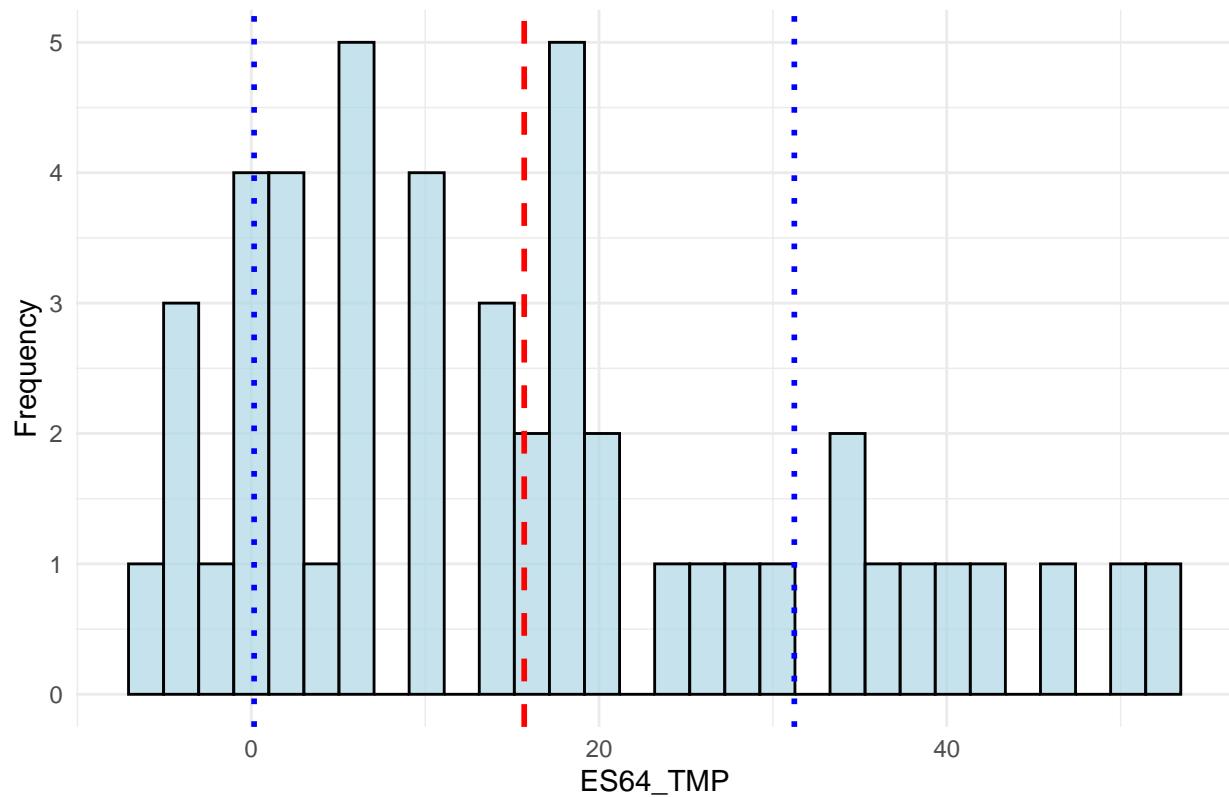
Histogram of AUTO\_TMP target week 3



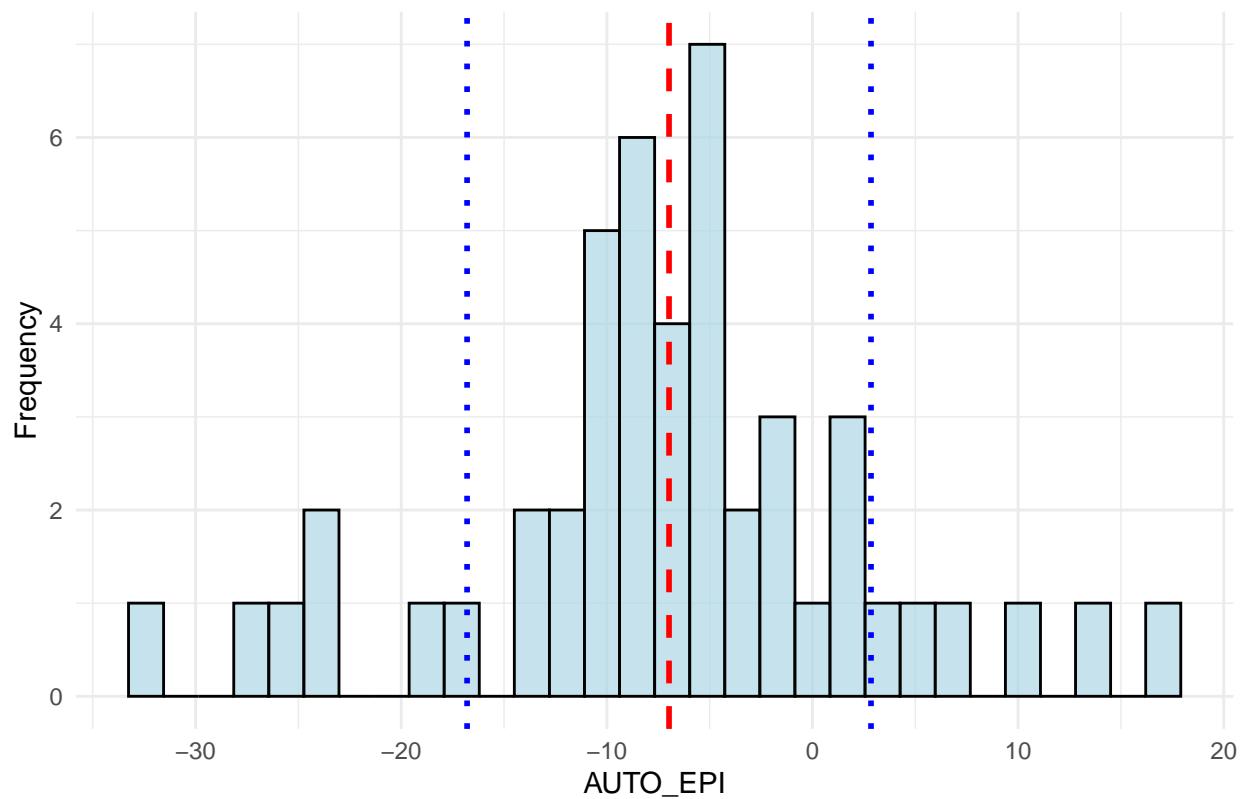
Histogram of ES27\_TMP target week 3



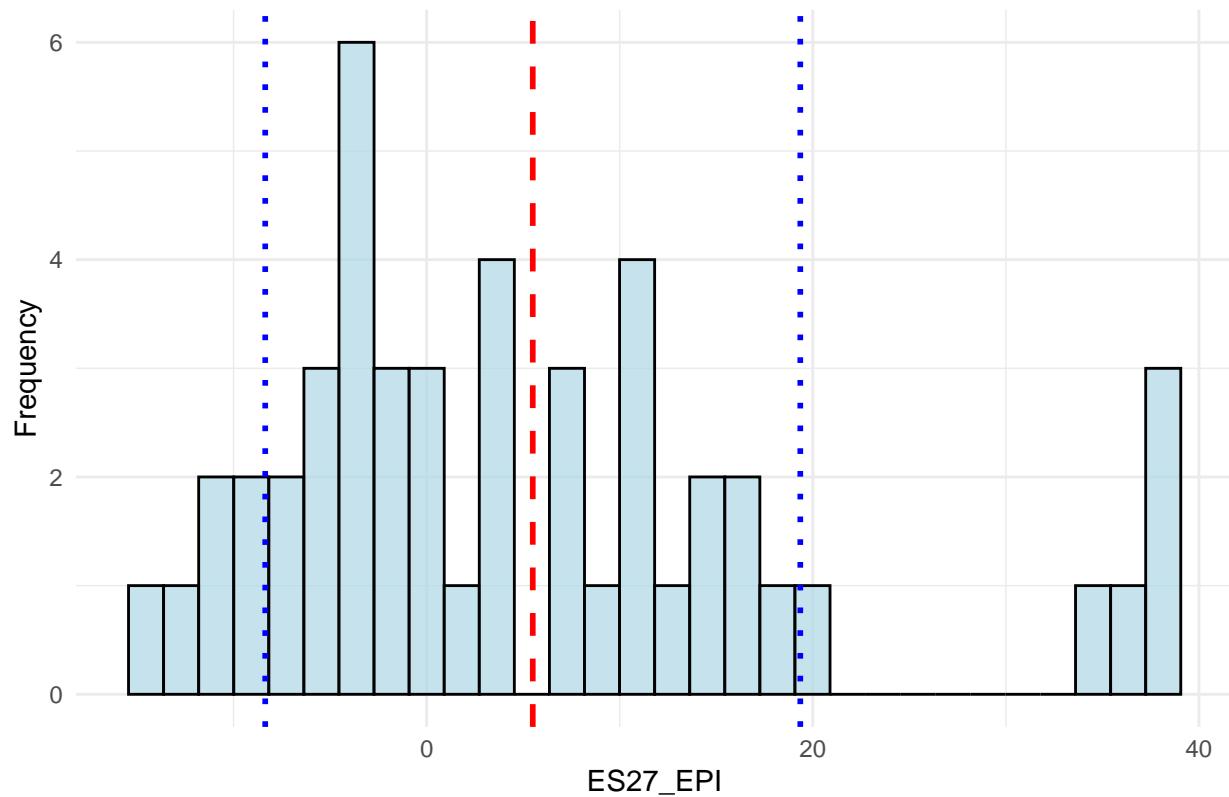
Histogram of ES64\_TMP target week 3



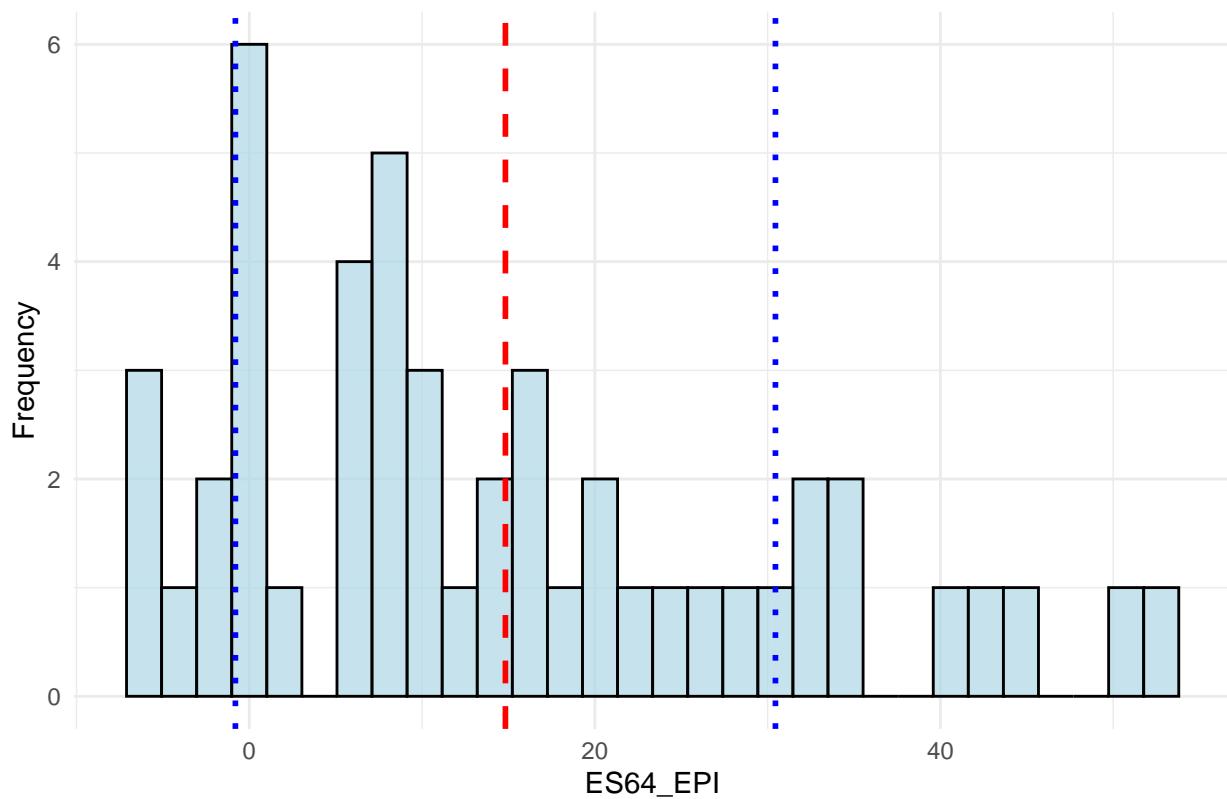
Histogram of AUTO\_EPI target week 3



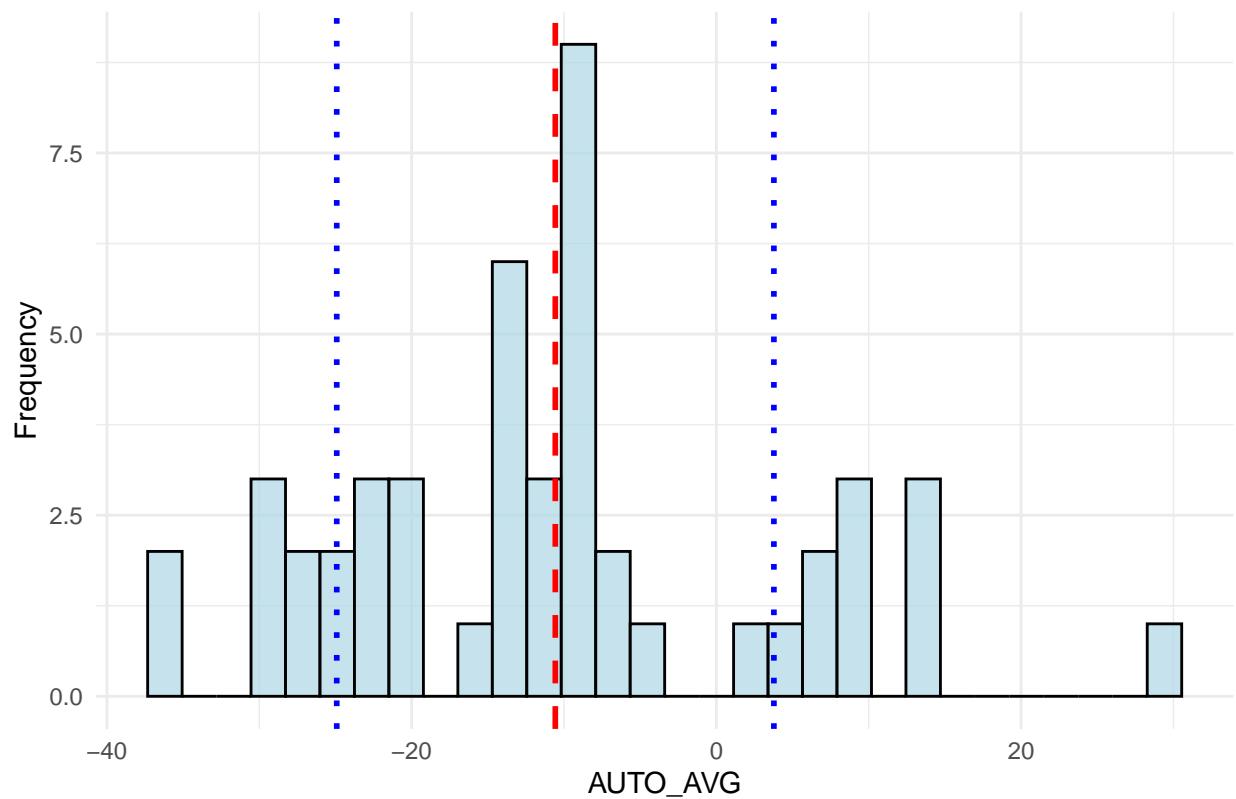
Histogram of ES27\_EPI target week 3



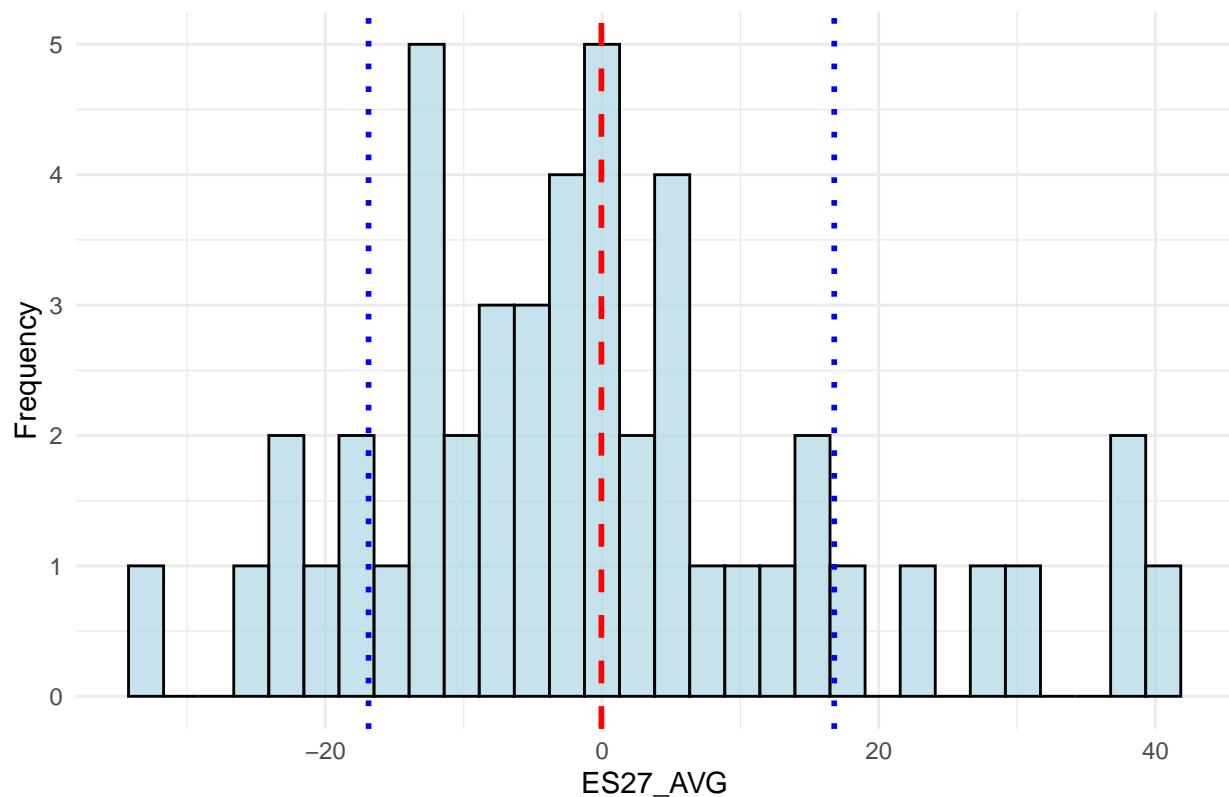
Histogram of ES64\_EPI target week 3



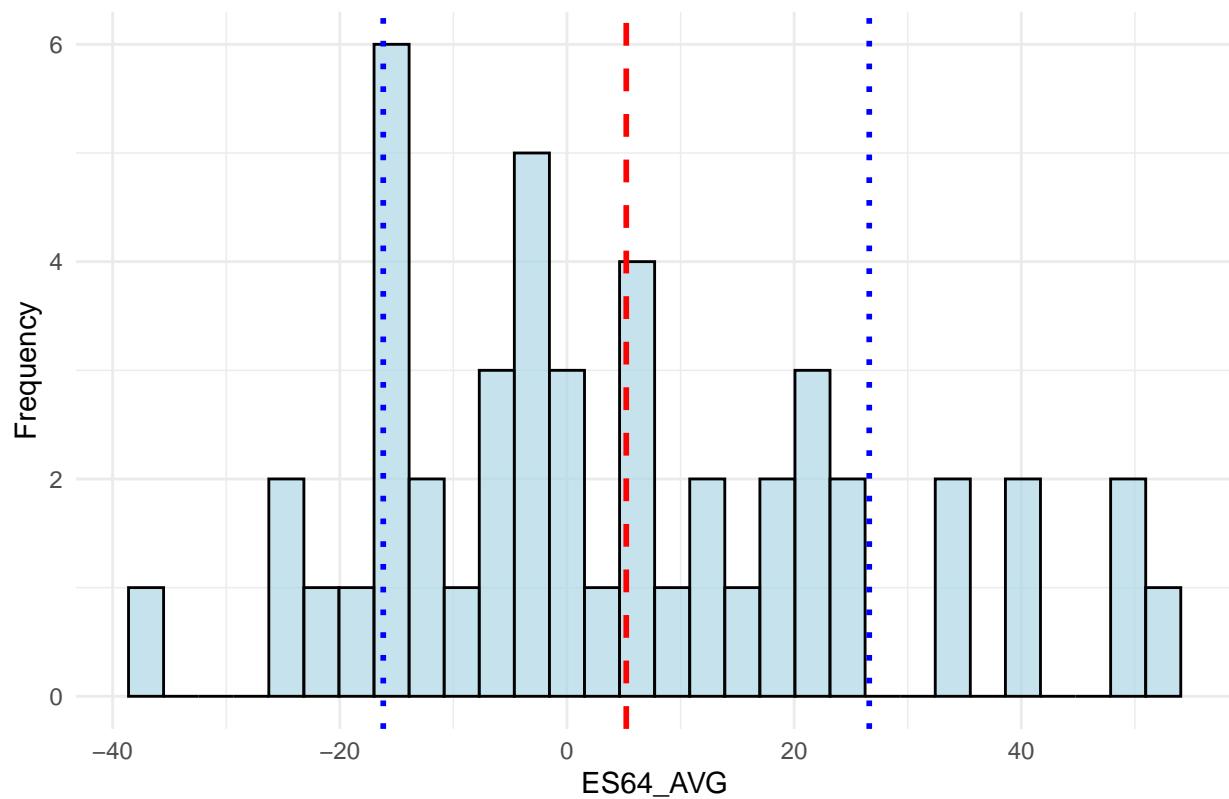
Histogram of AUTO\_AVG target week 3



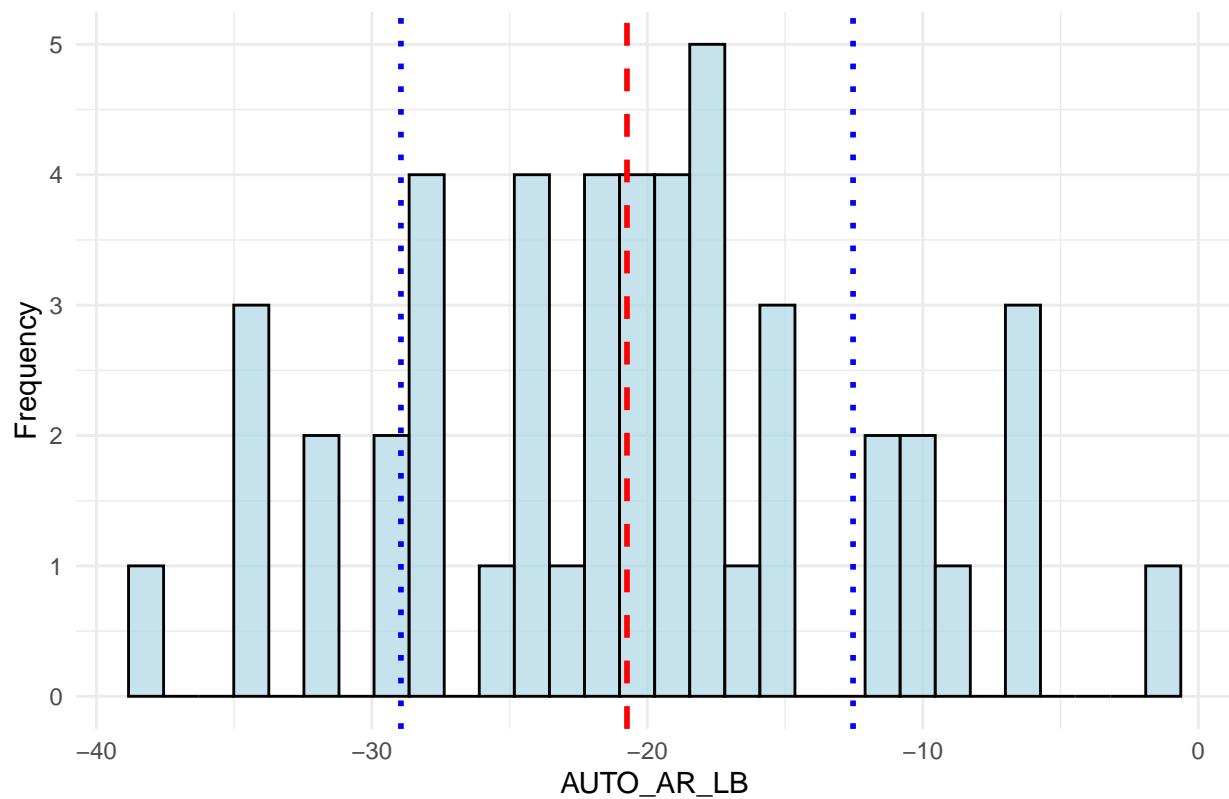
Histogram of ES27\_AVG target week 3



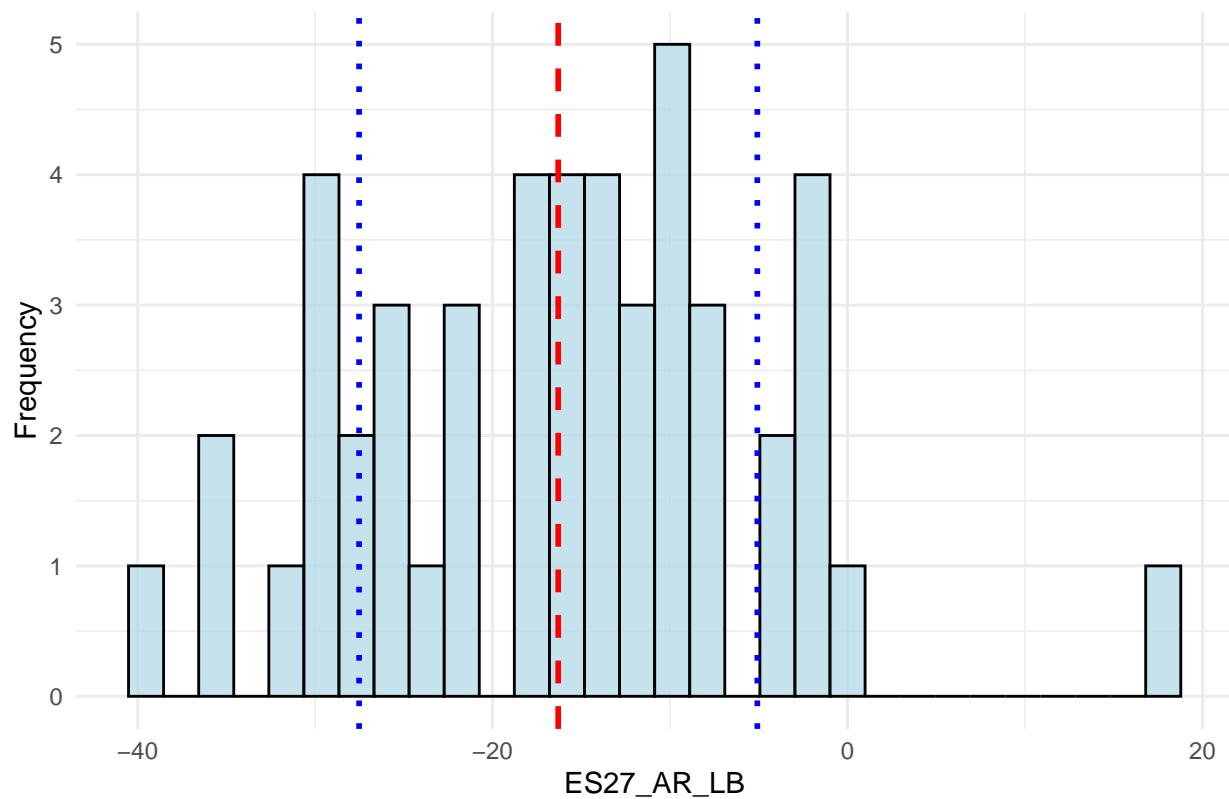
Histogram of ES64\_AVG target week 3



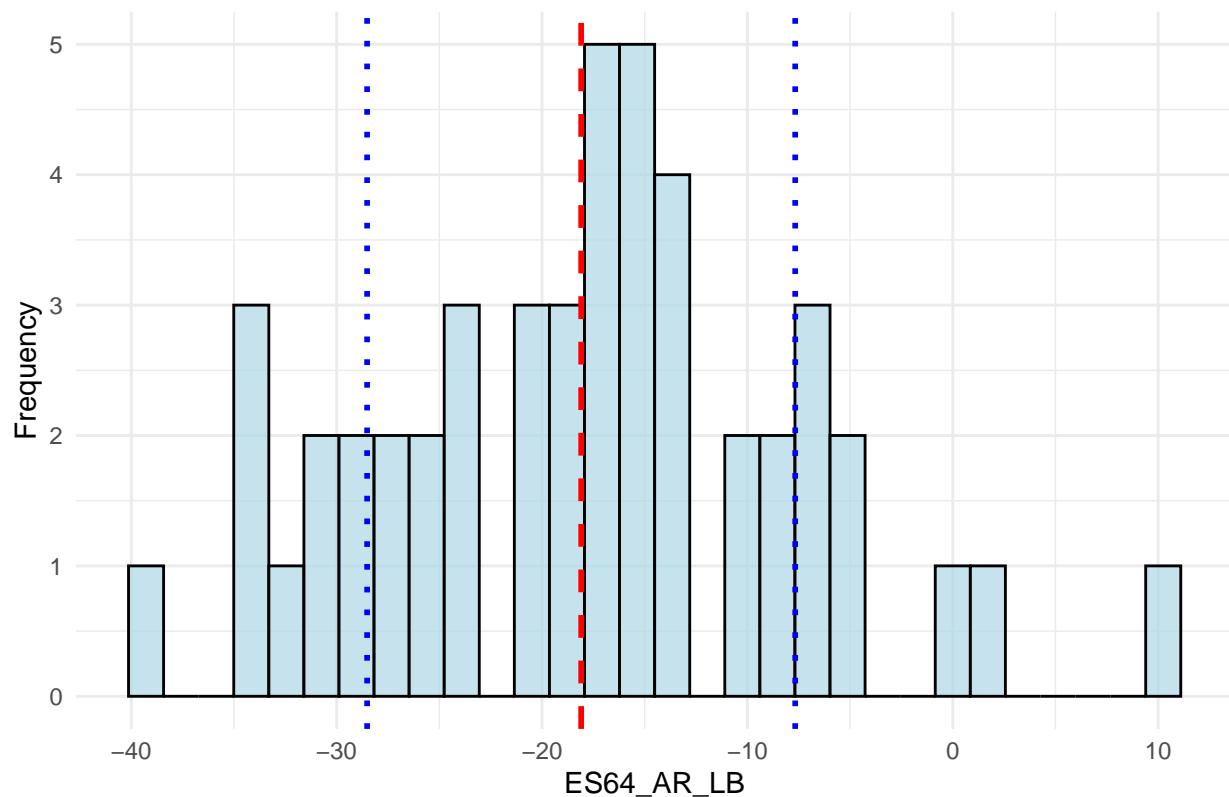
Histogram of AUTO\_AR\_LB target week 3



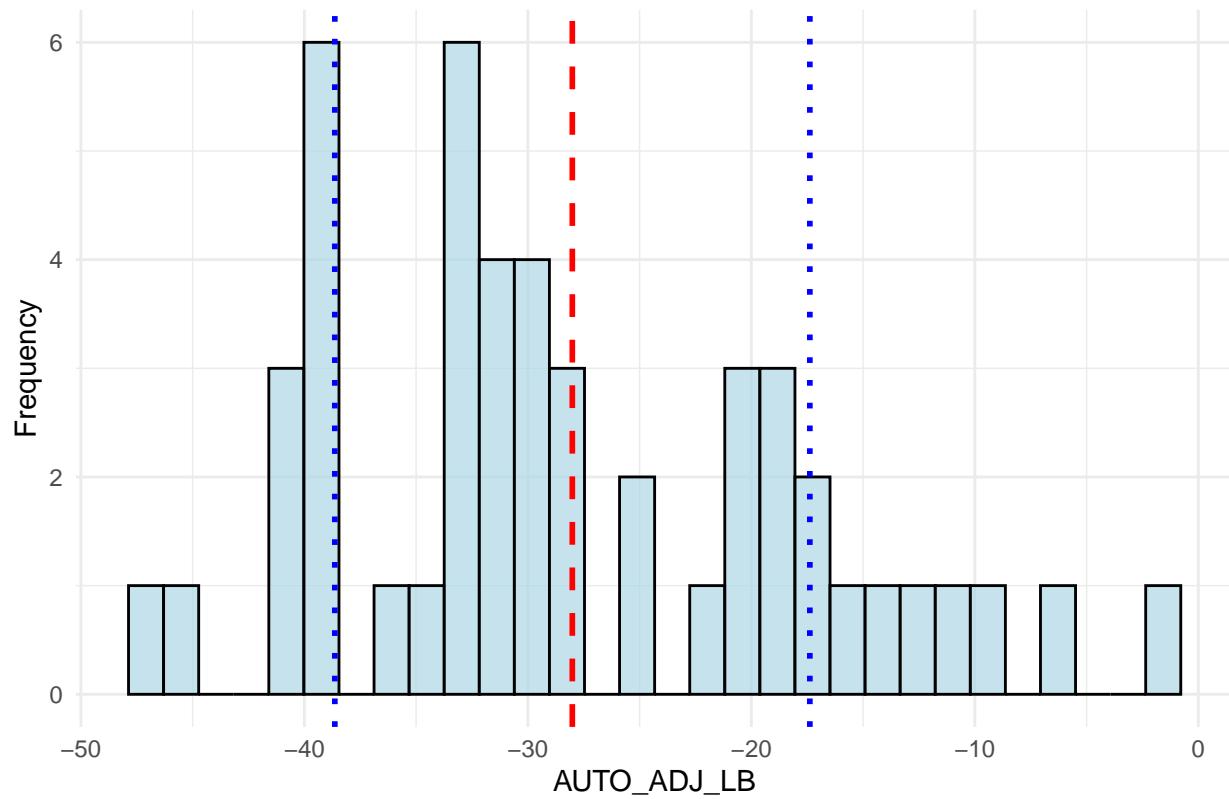
Histogram of ES27\_AR\_LB target week 3



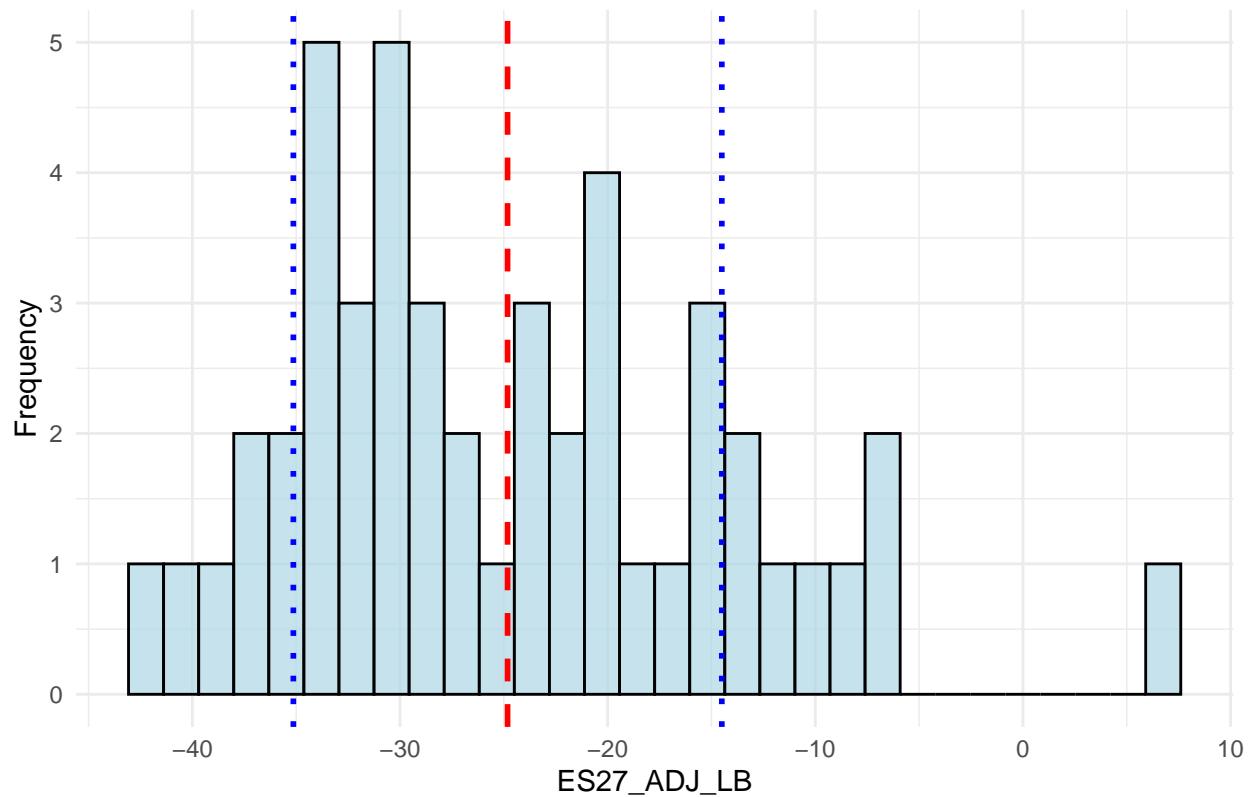
Histogram of ES64\_AR\_LB target week 3



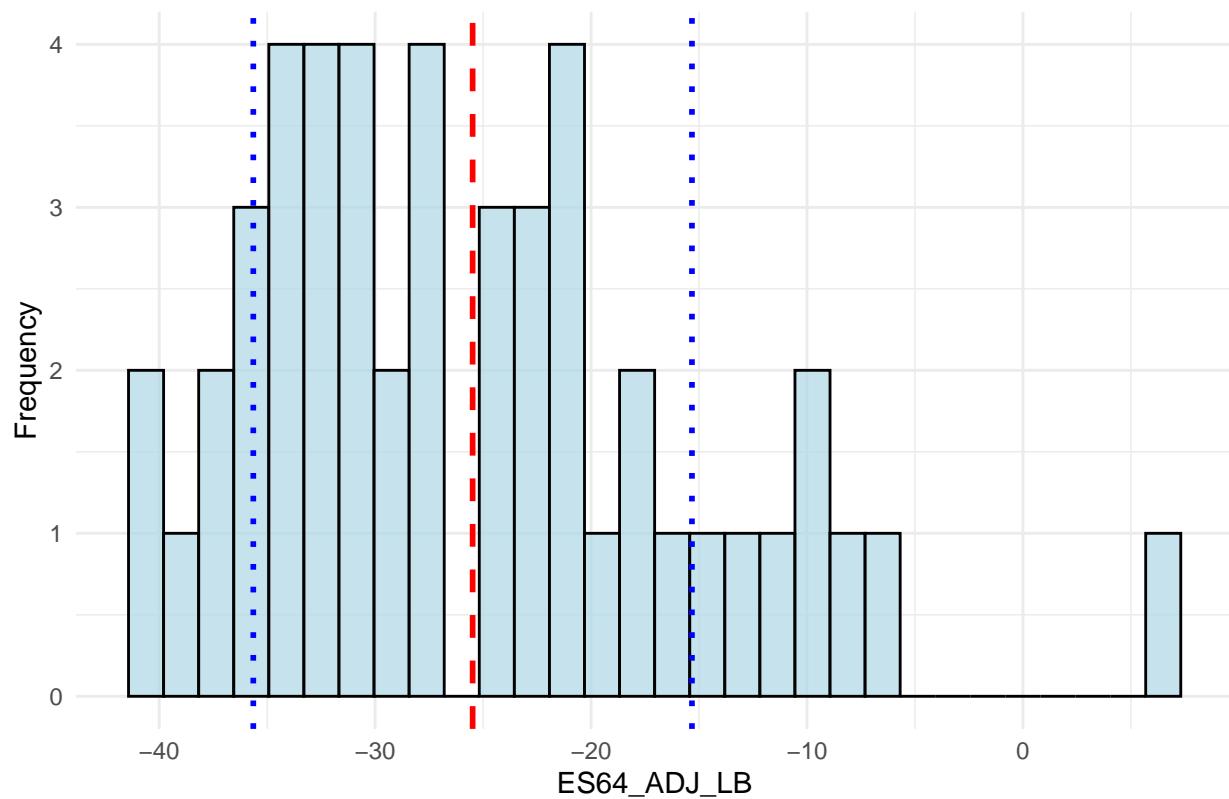
Histogram of AUTO\_ADJ\_LB target week 3



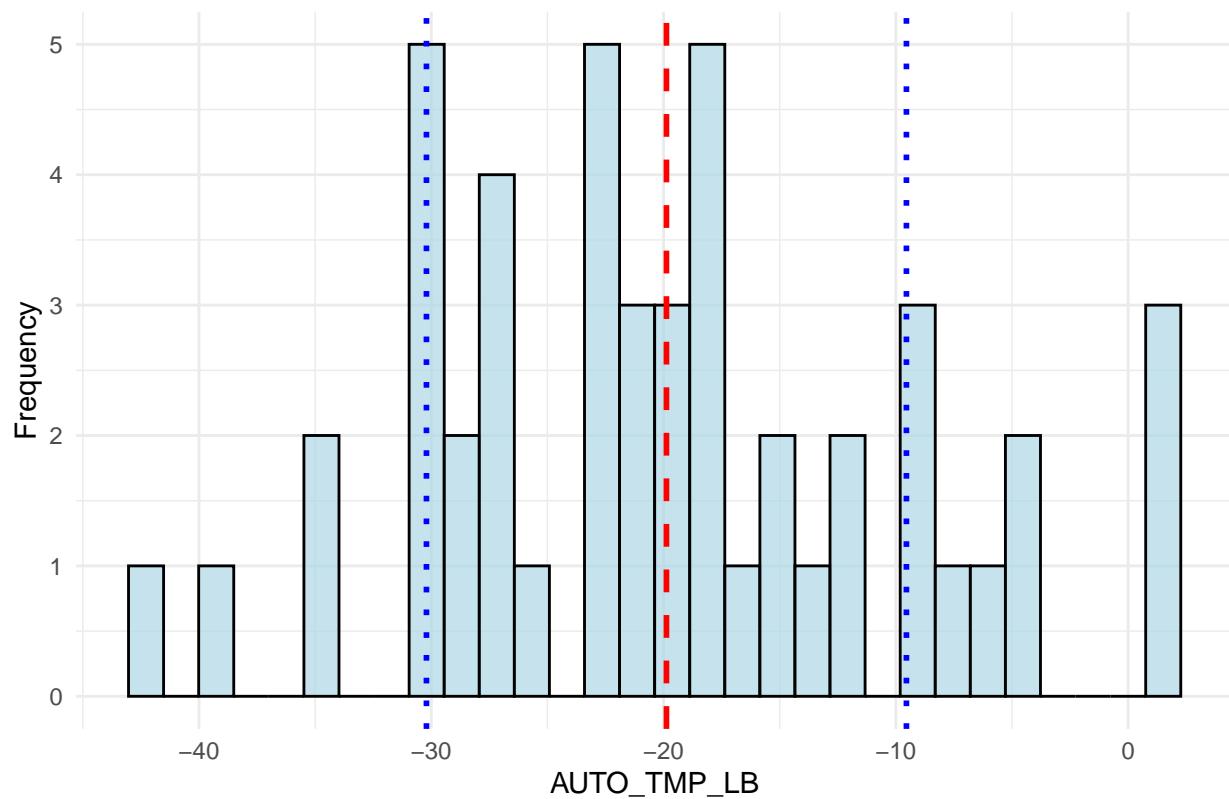
Histogram of ES27\_ADJ\_LB target week 3



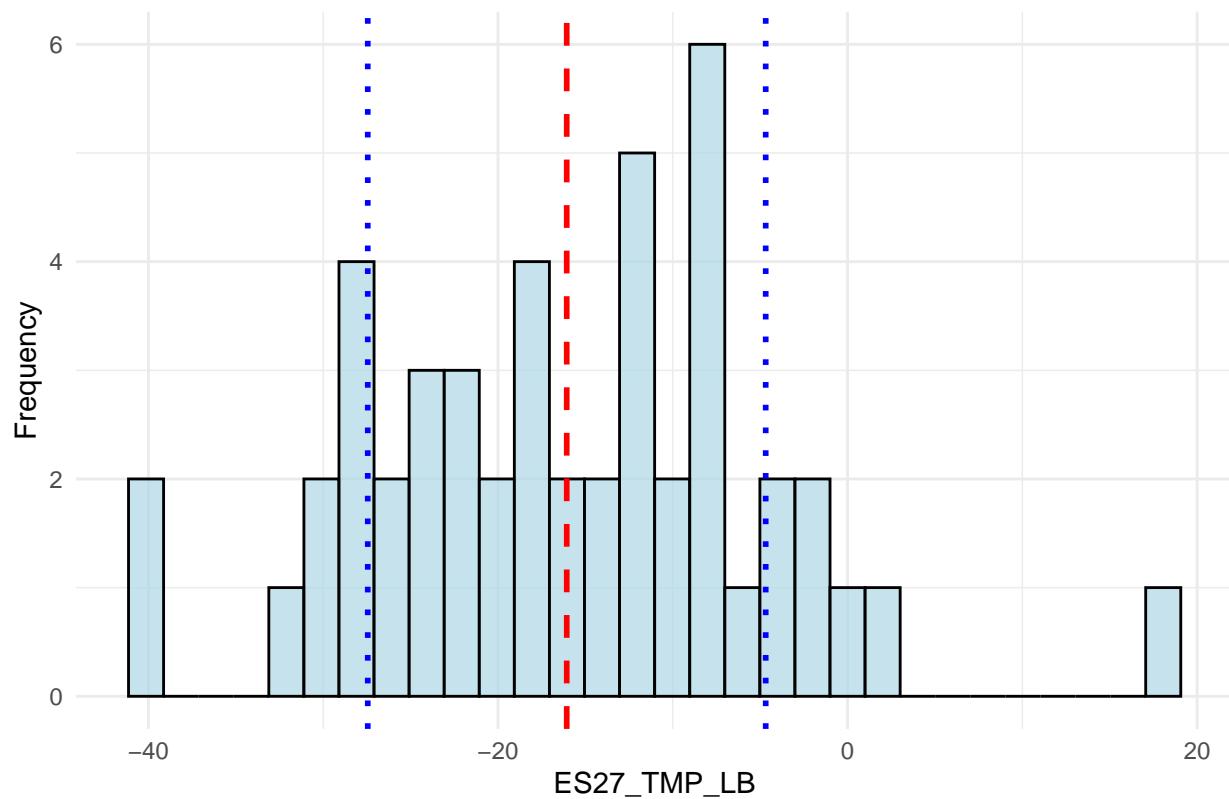
Histogram of ES64\_ADJ\_LB target week 3



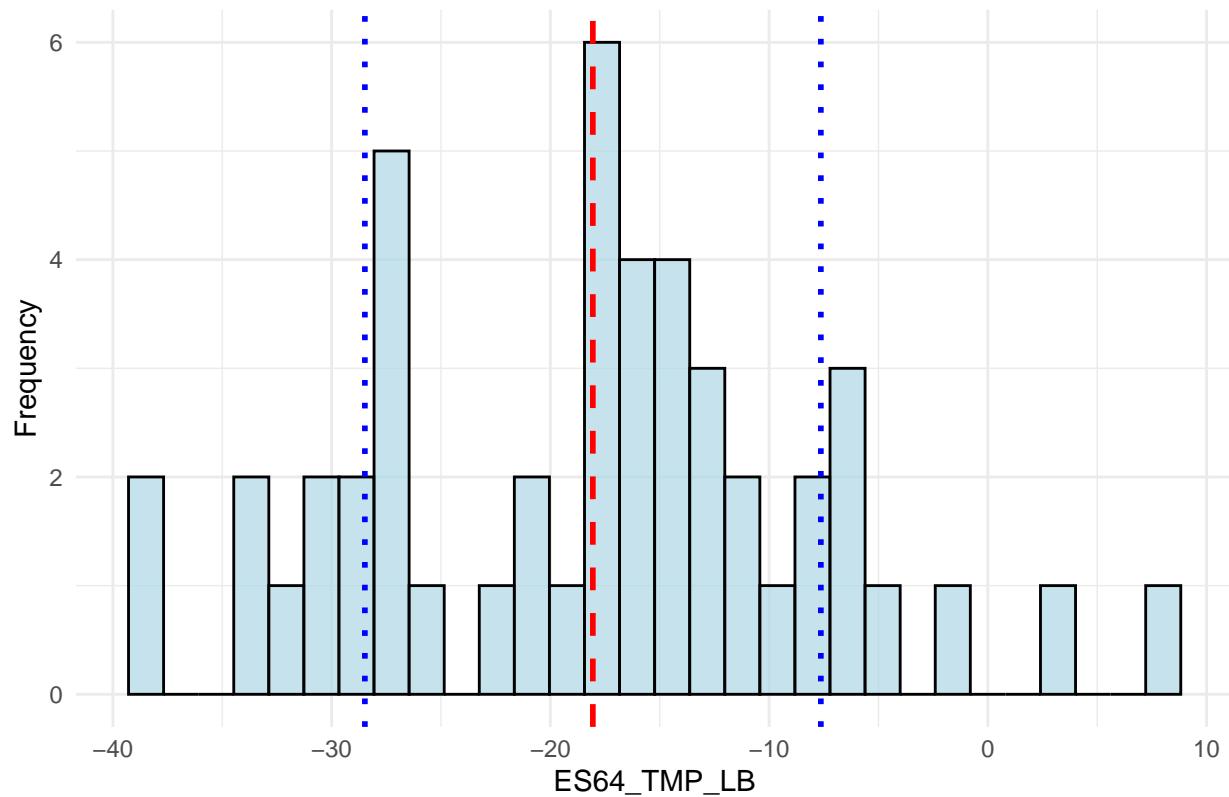
Histogram of AUTO\_TMP\_LB target week 3



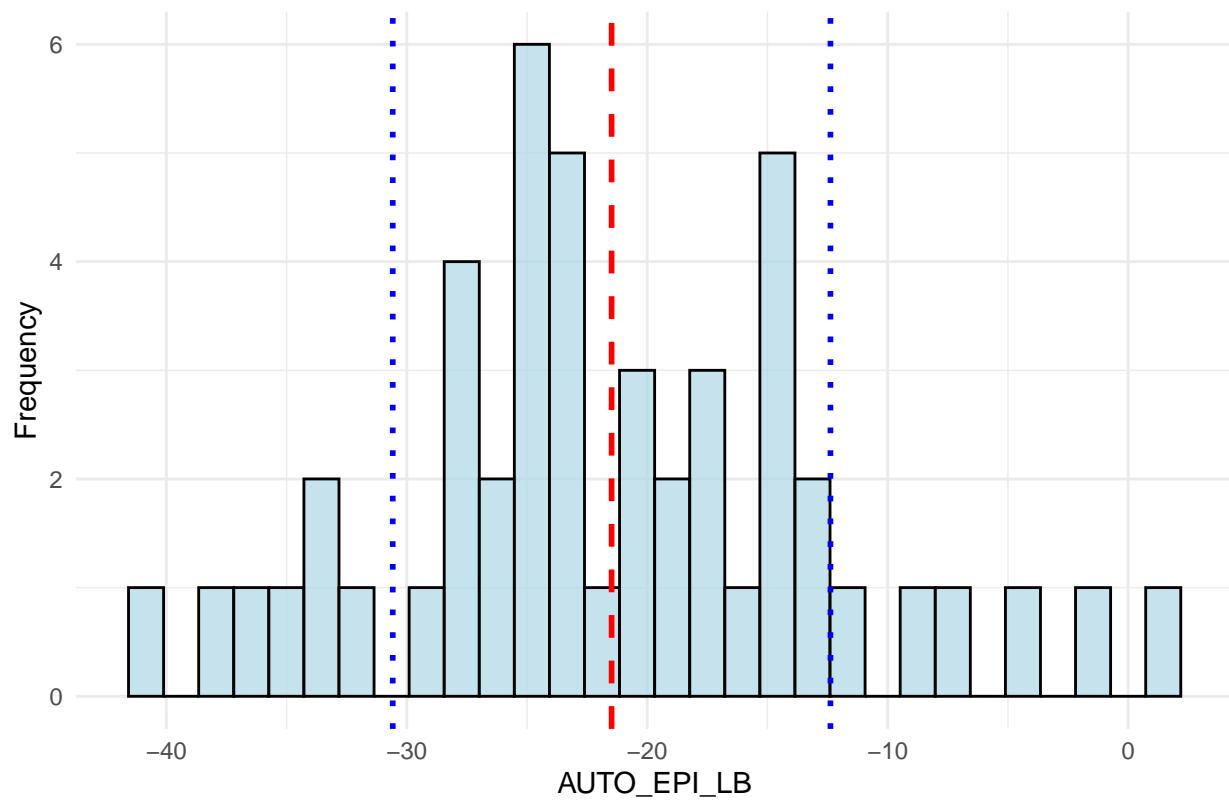
Histogram of ES27\_TMP\_LB target week 3



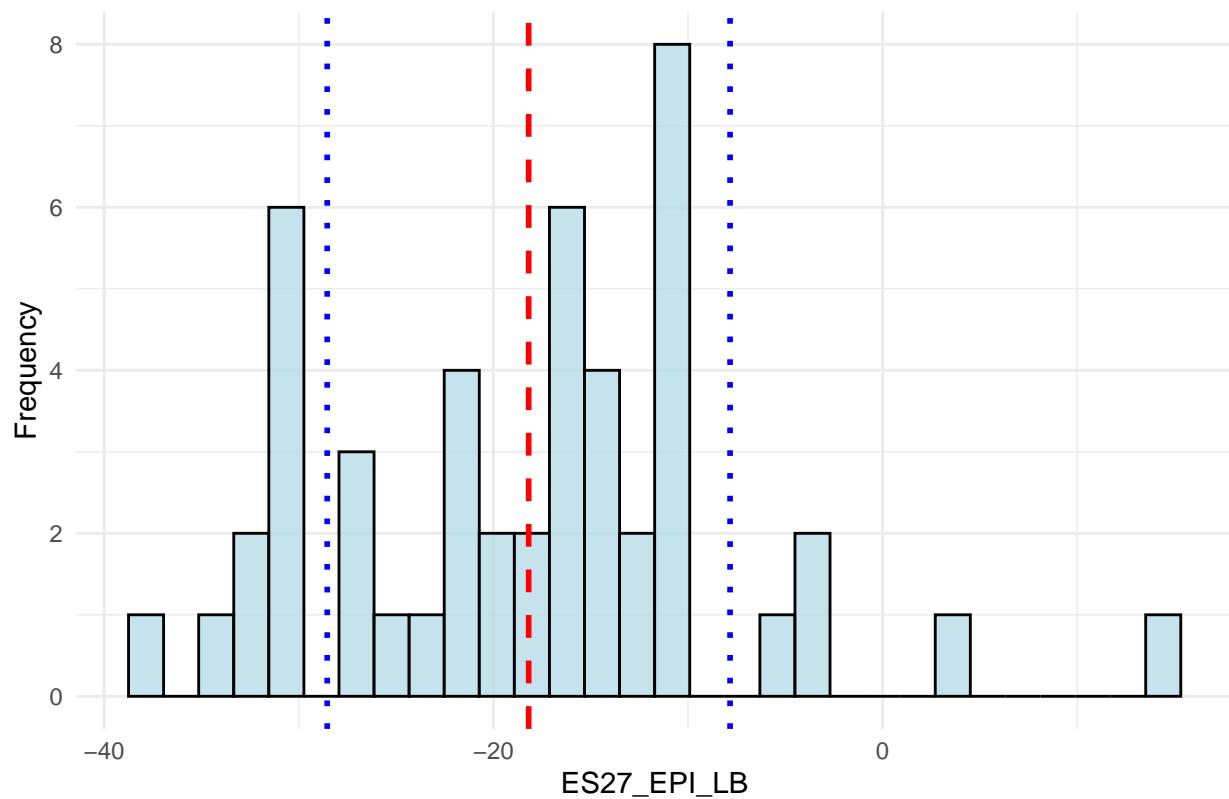
Histogram of ES64\_TMP\_LB target week 3



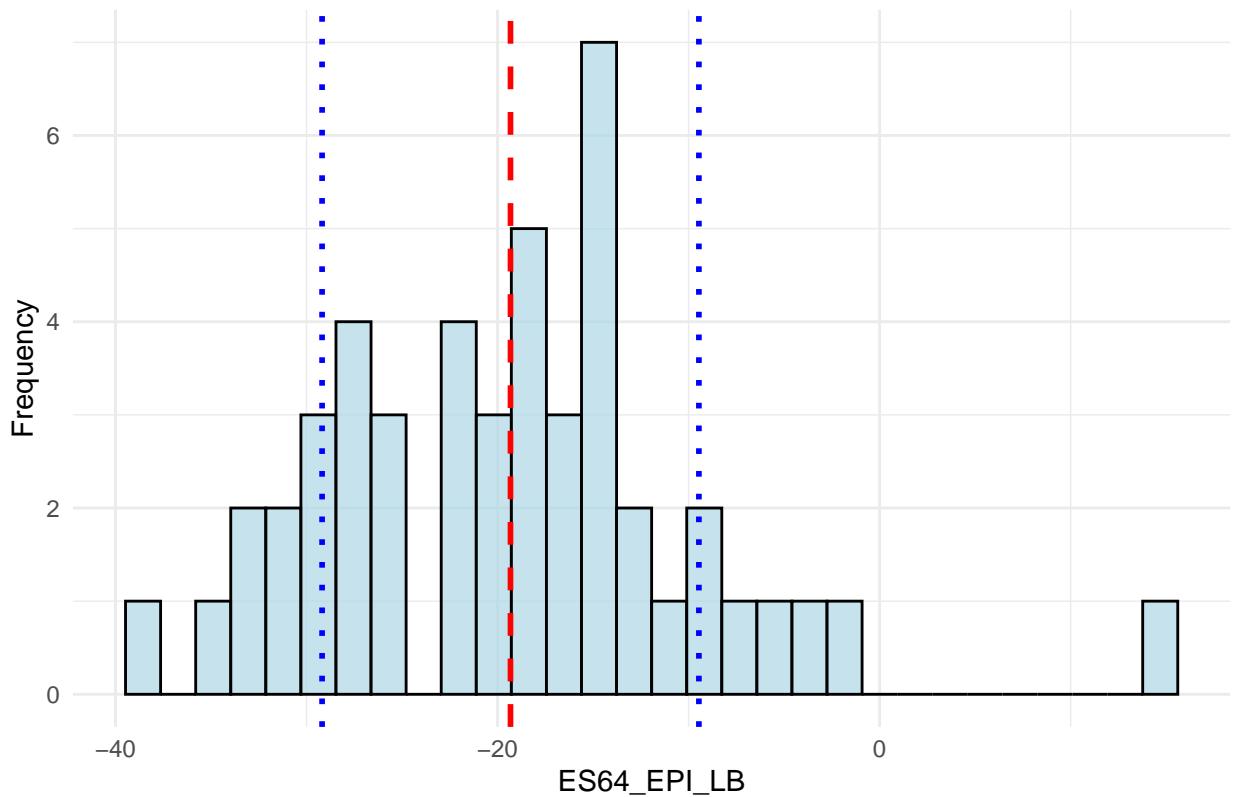
Histogram of AUTO\_EPI\_LB target week 3



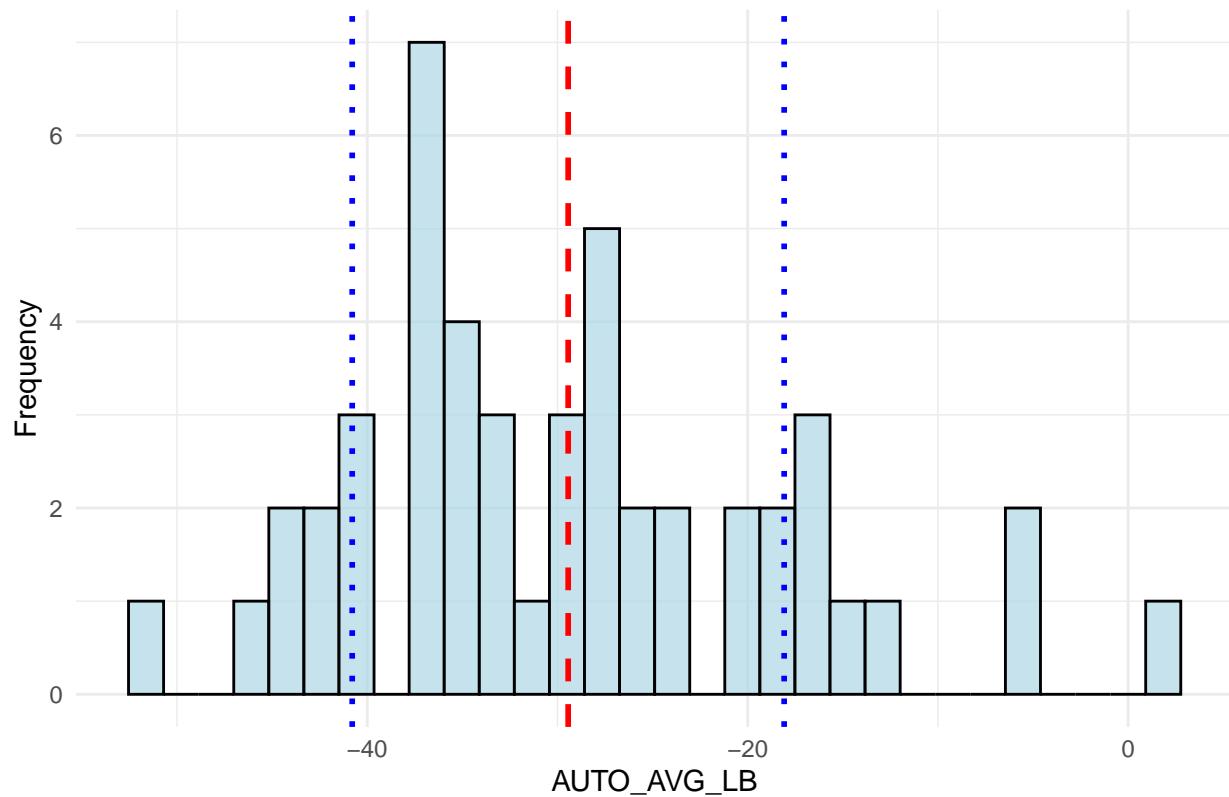
Histogram of ES27\_EPI\_LB target week 3



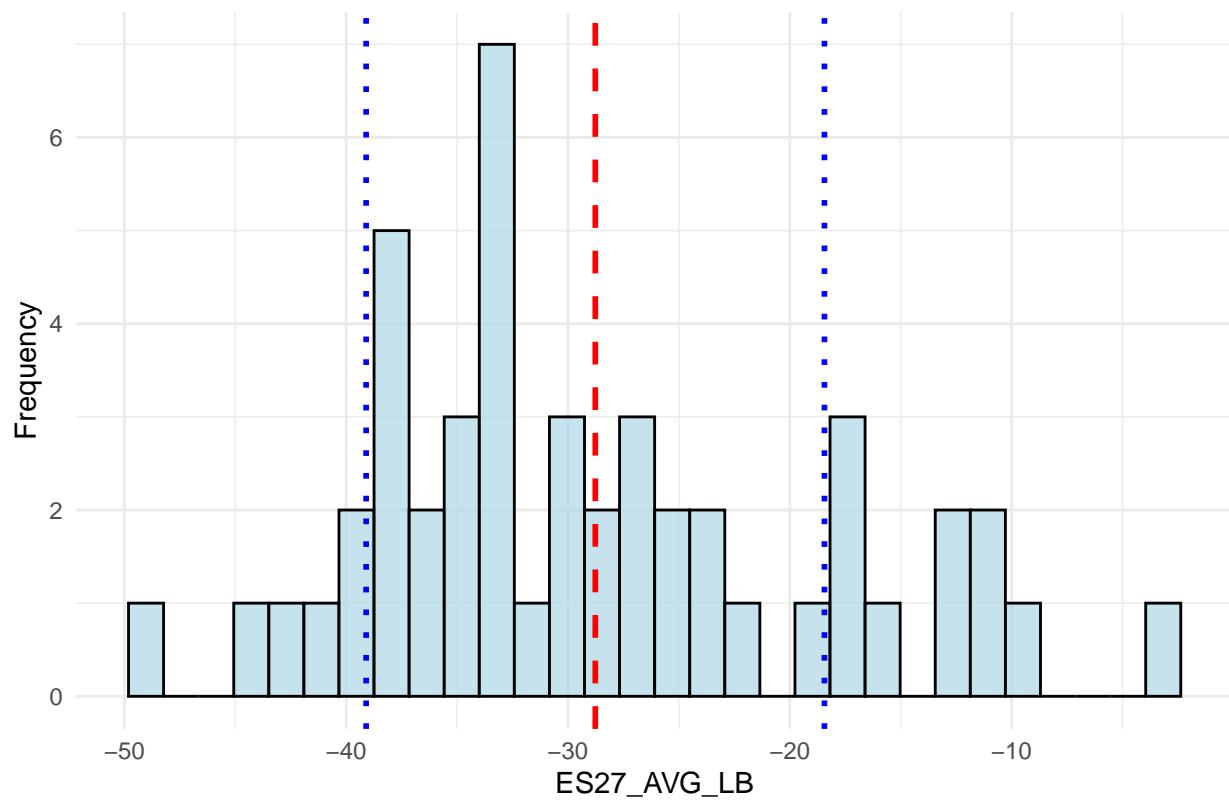
Histogram of ES64\_EPI\_LB target week 3



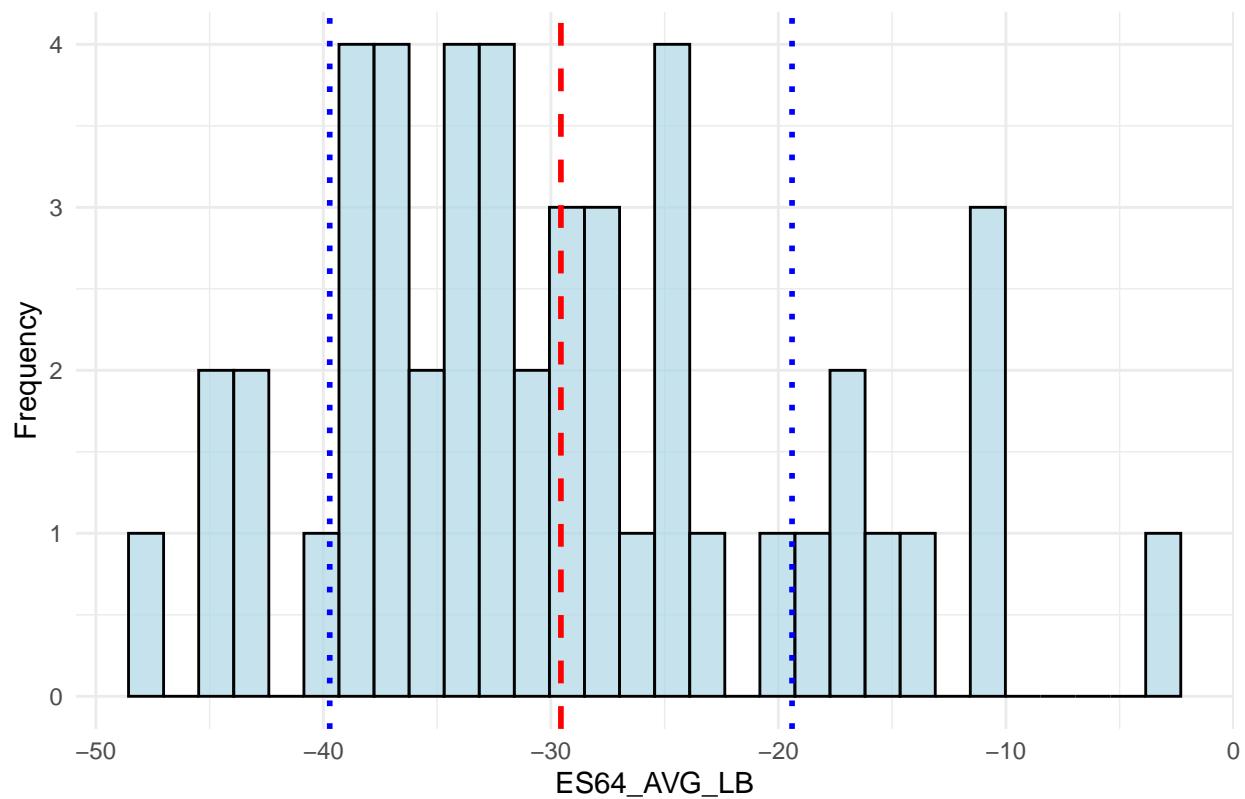
Histogram of AUTO\_AVG\_LB target week 3



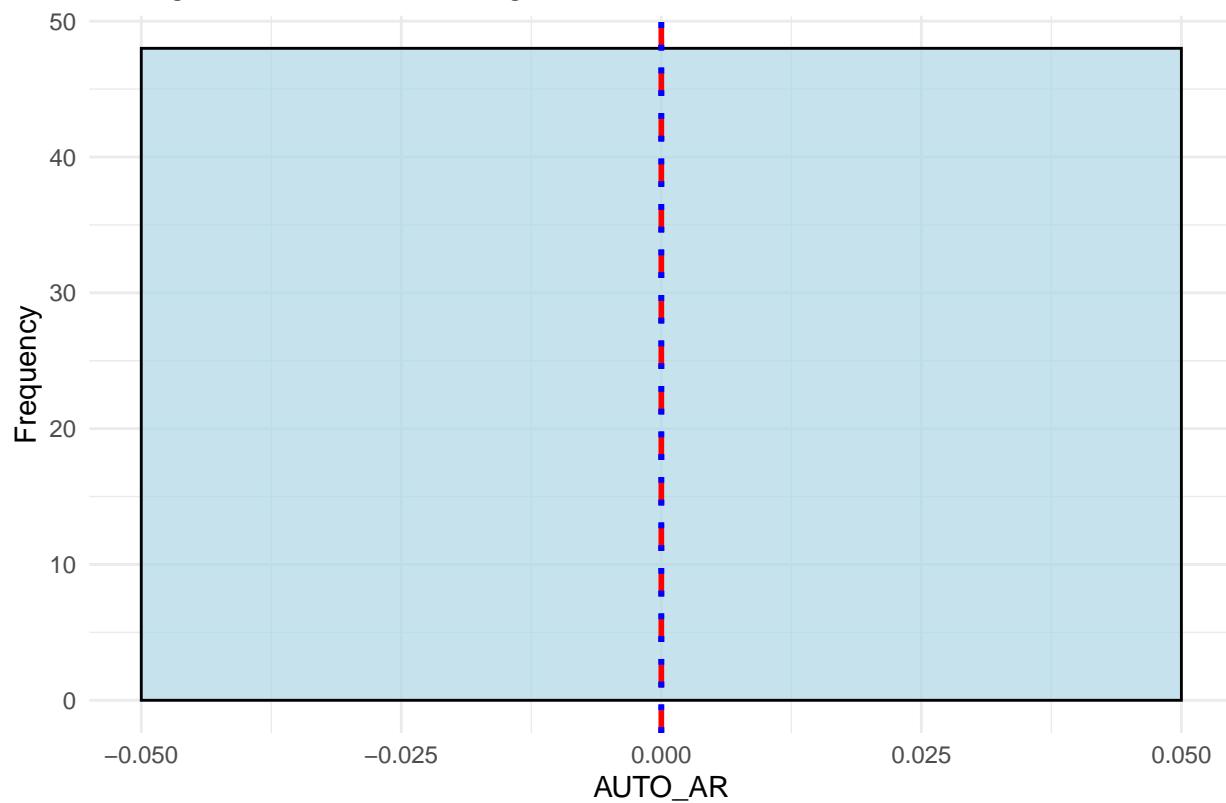
Histogram of ES27\_AVG\_LB target week 3



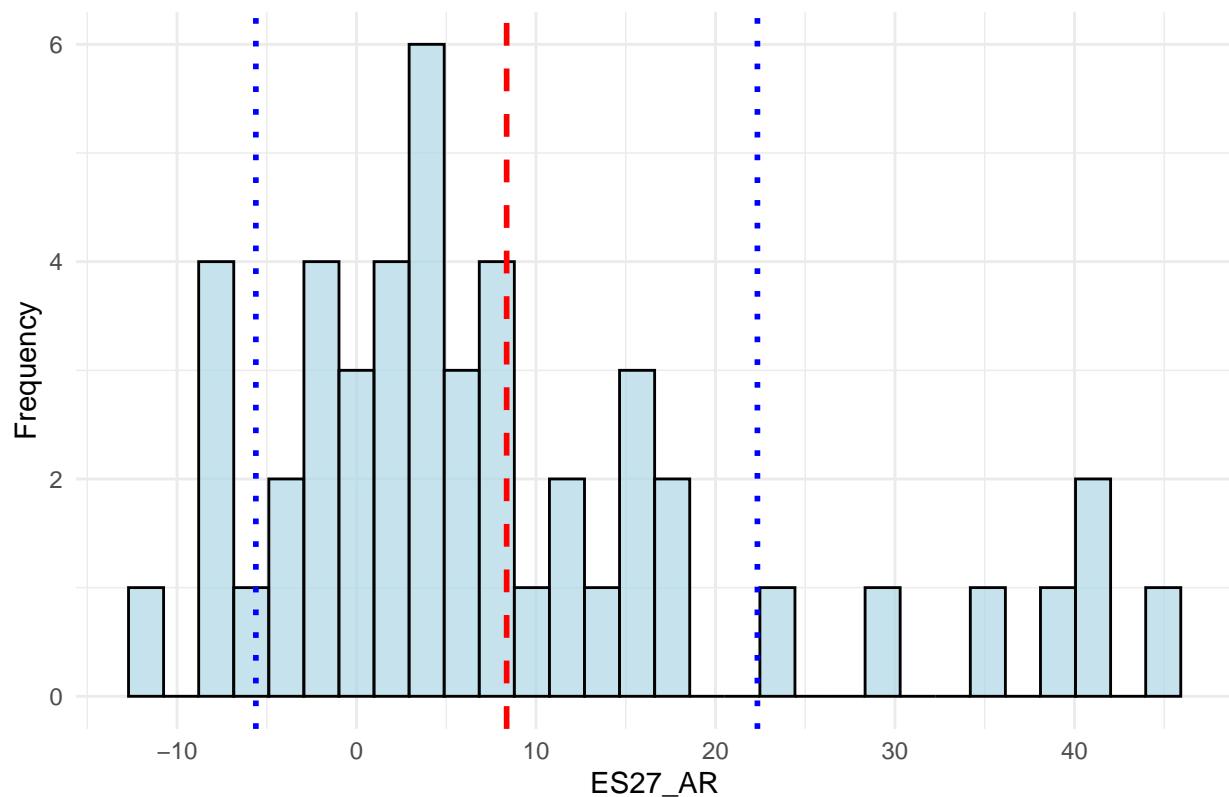
Histogram of ES64\_AVG\_LB target week 3



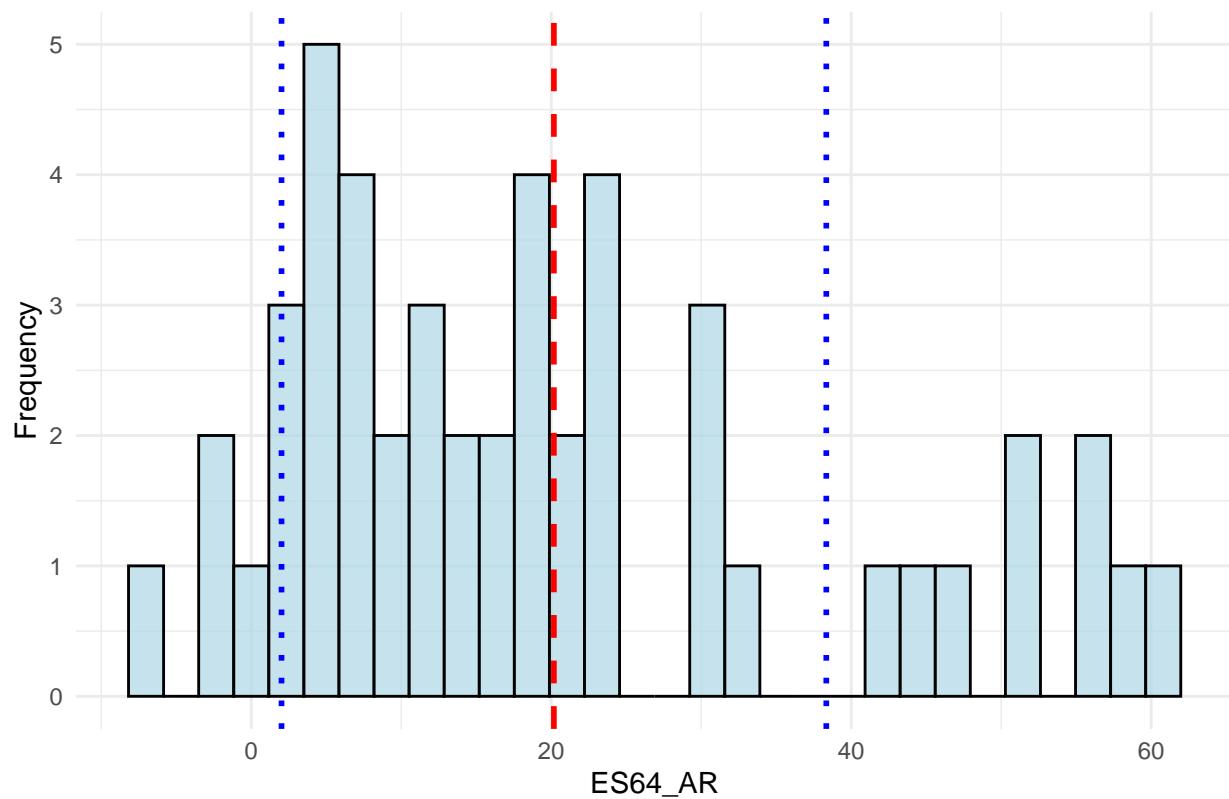
Histogram of AUTO\_AR target week 4



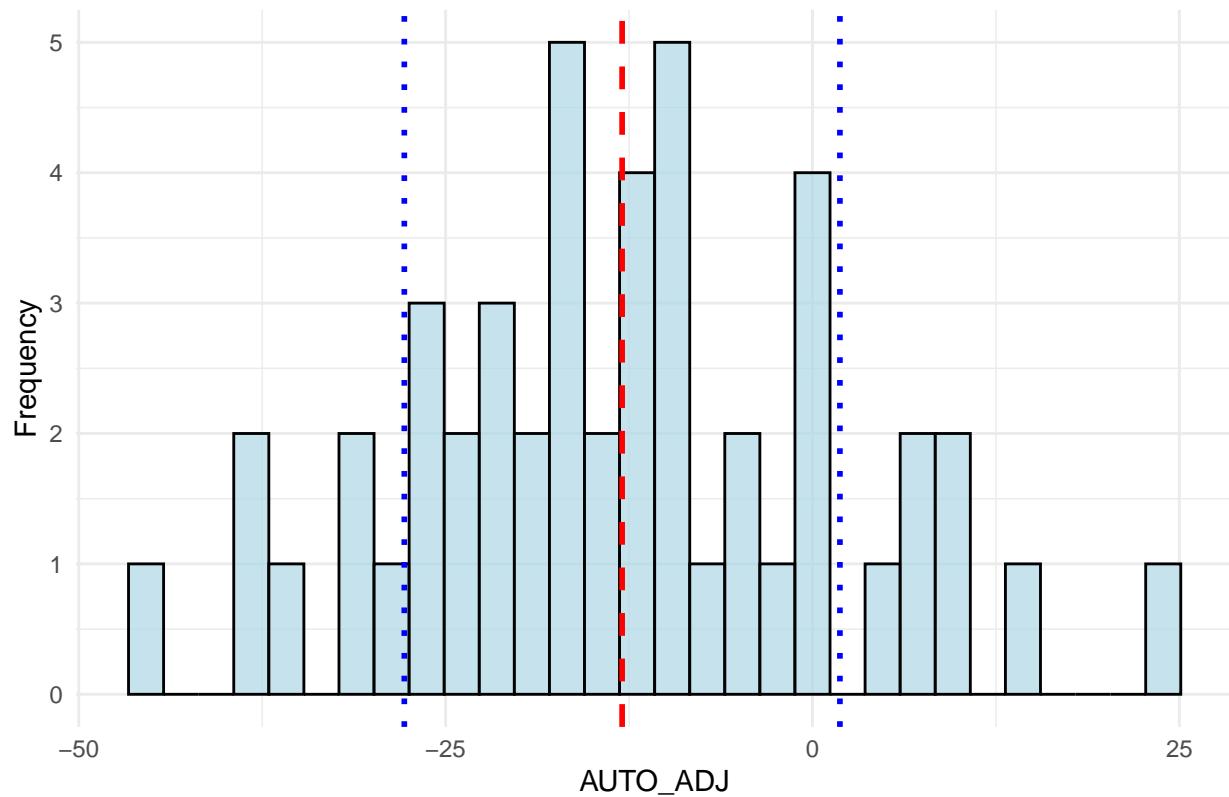
Histogram of ES27\_AR target week 4



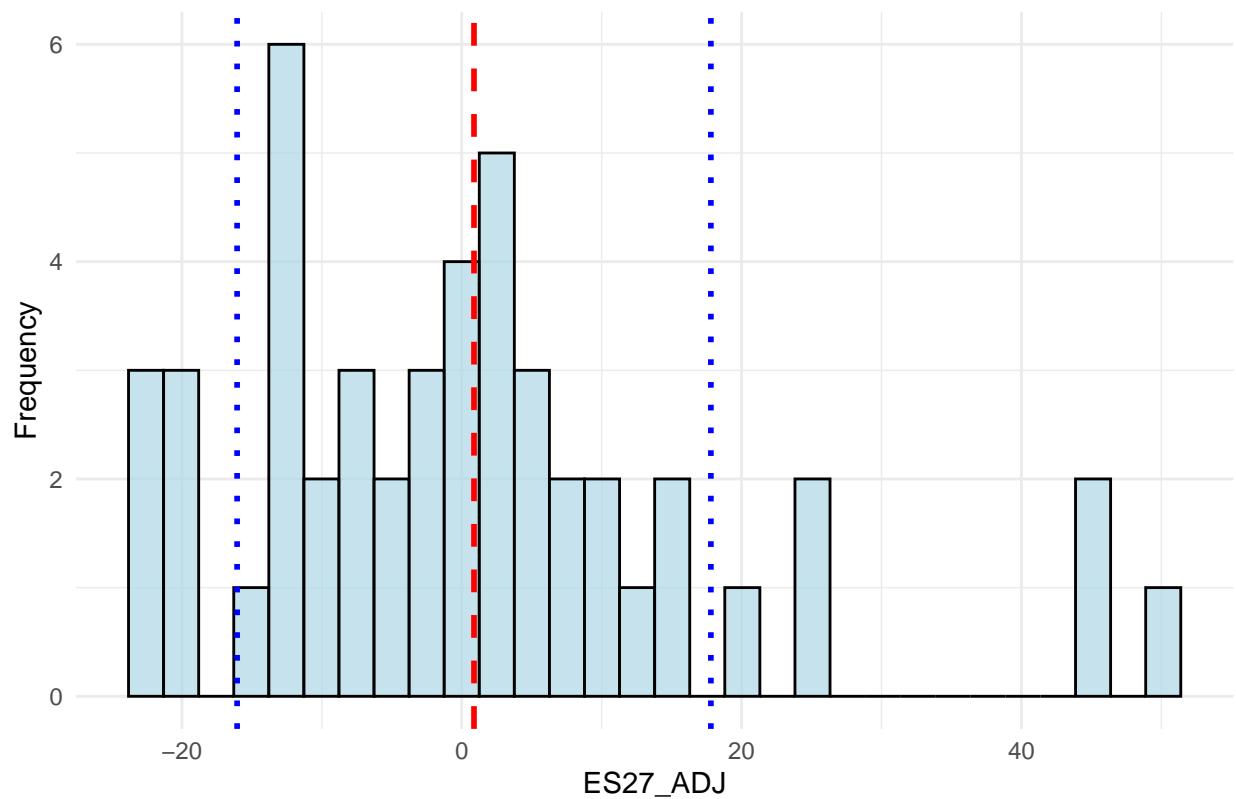
Histogram of ES64\_AR target week 4



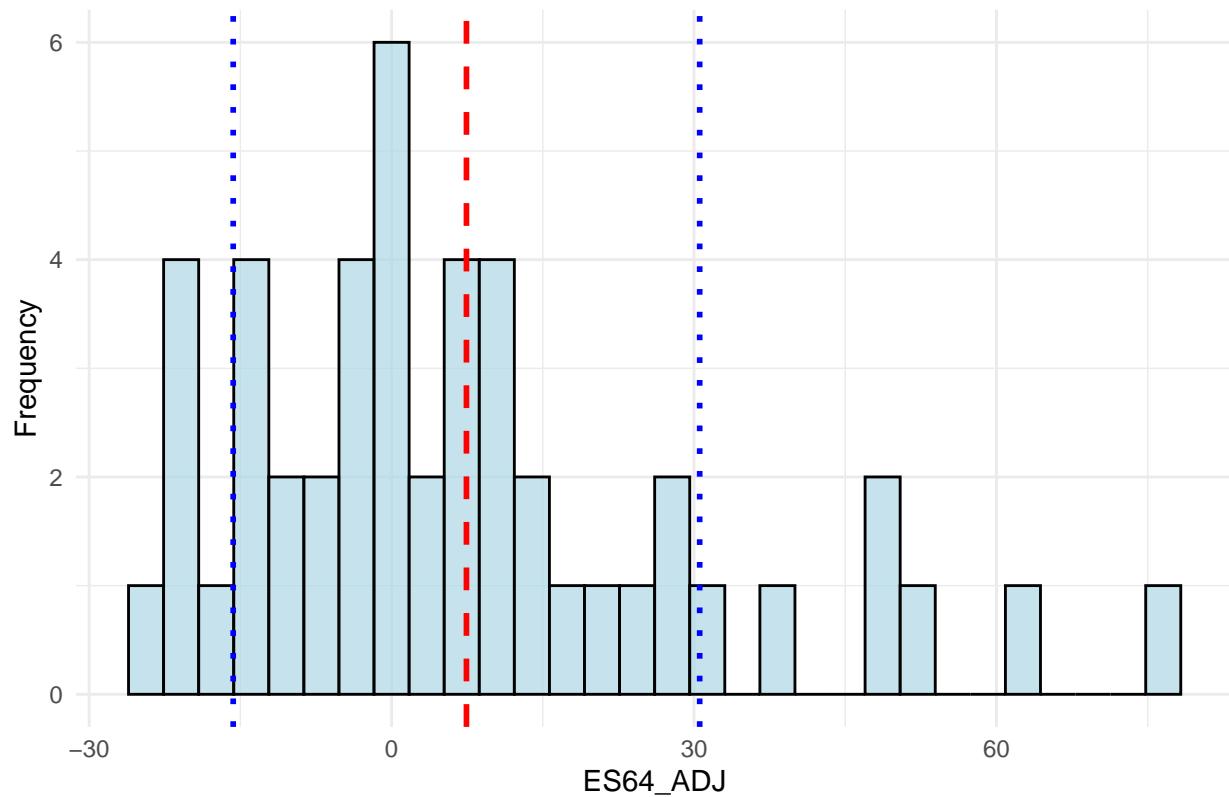
Histogram of AUTO\_ADJ target week 4



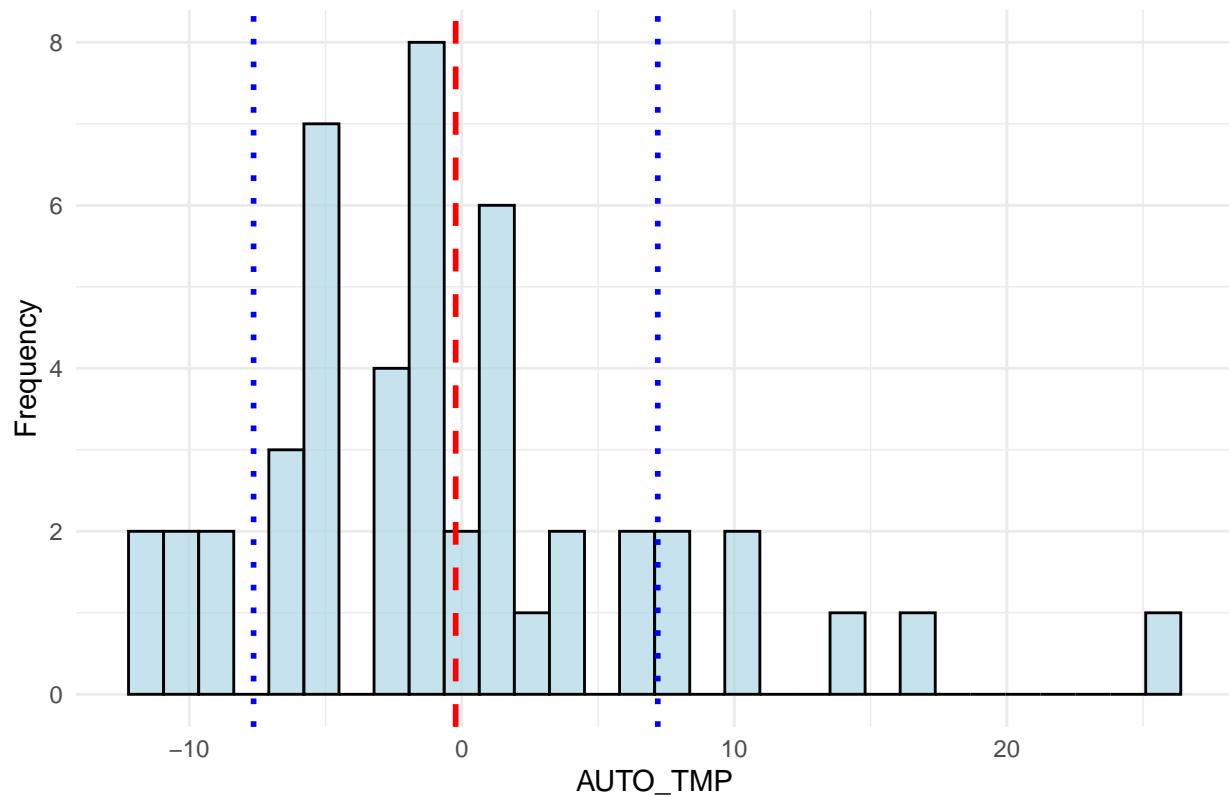
Histogram of ES27\_ADJ target week 4



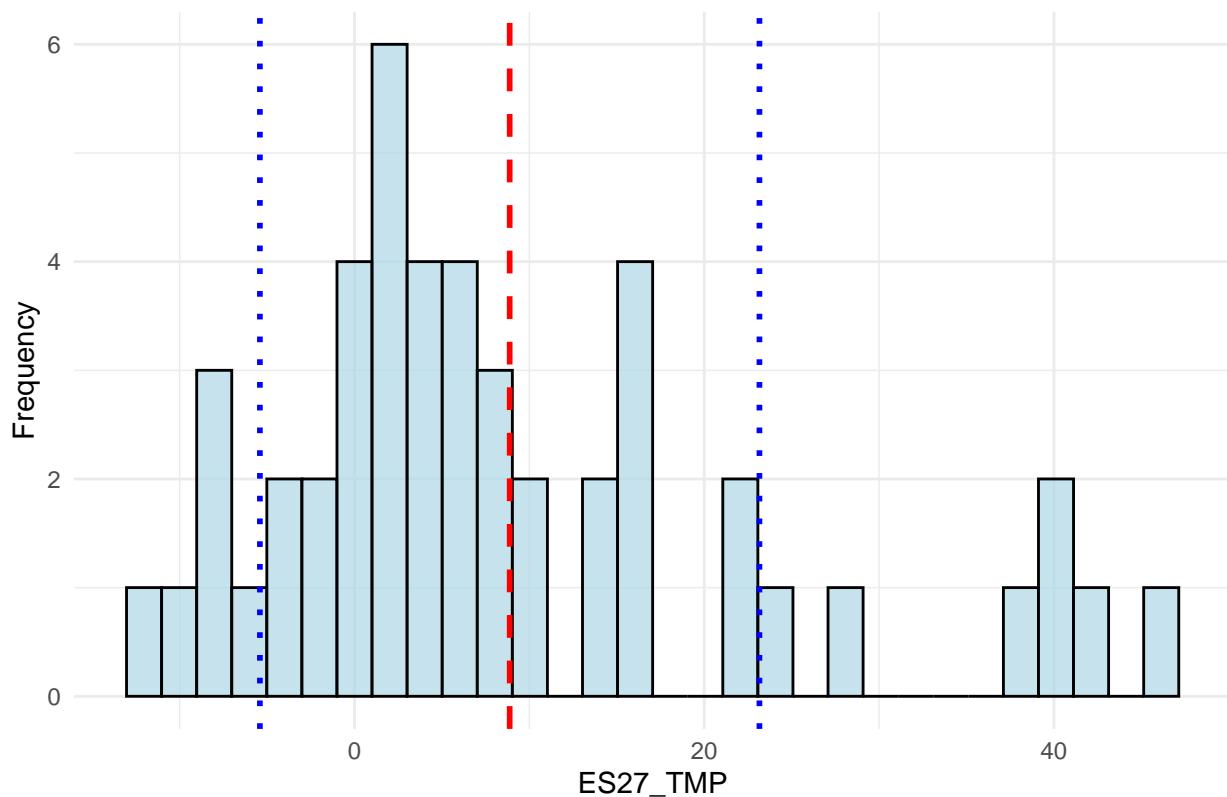
Histogram of ES64\_ADJ target week 4



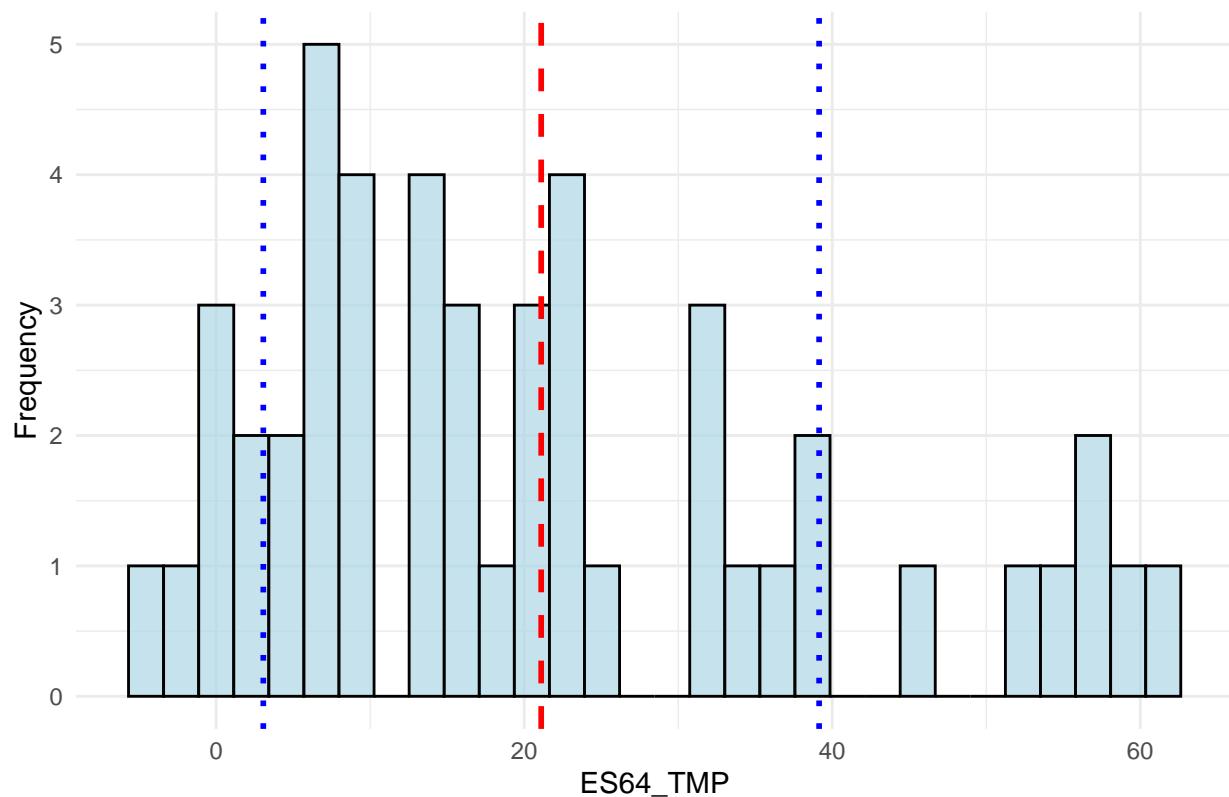
Histogram of AUTO\_TMP target week 4



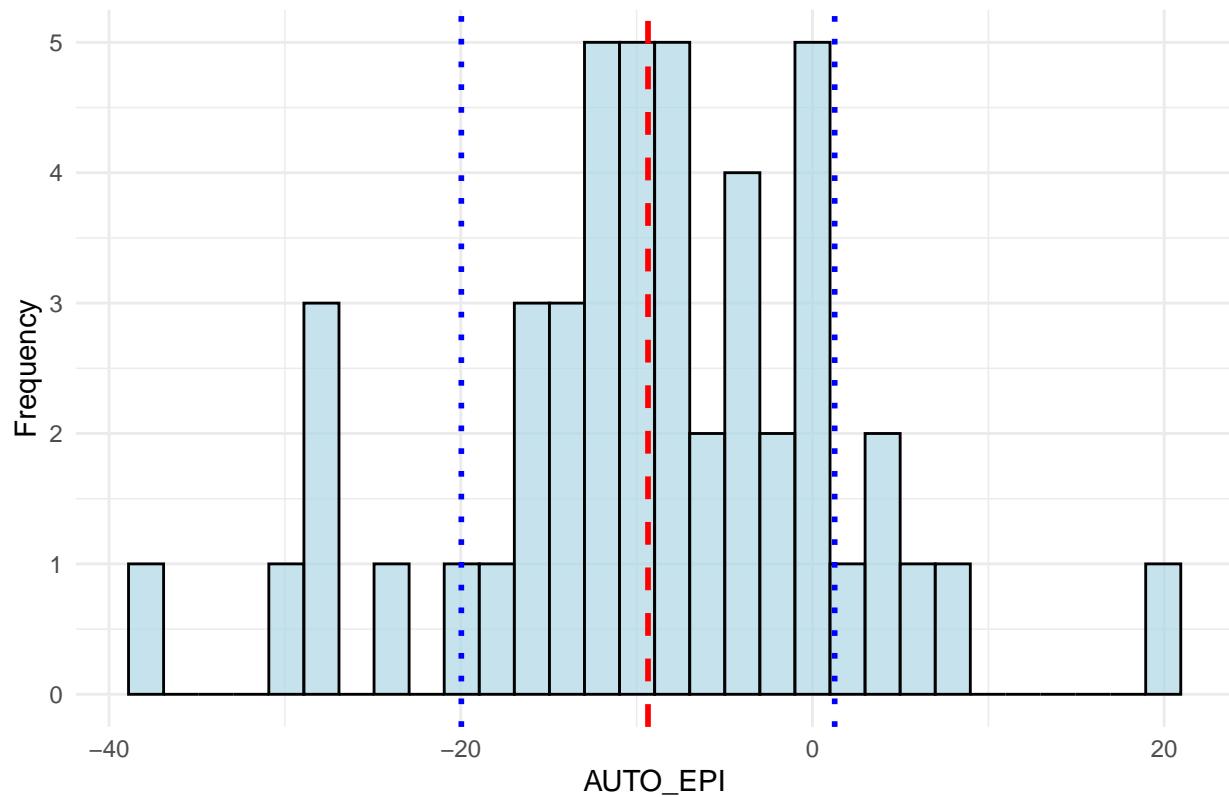
Histogram of ES27\_TMP target week 4



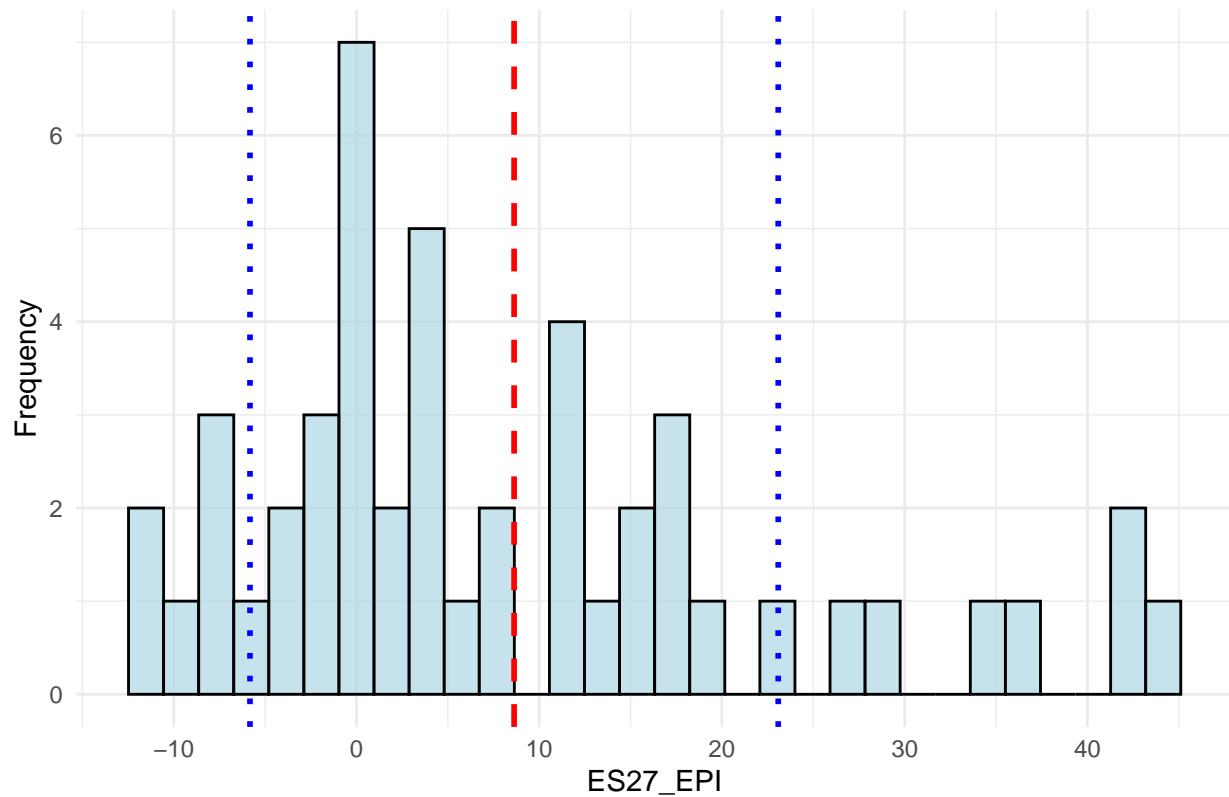
Histogram of ES64\_TMP target week 4



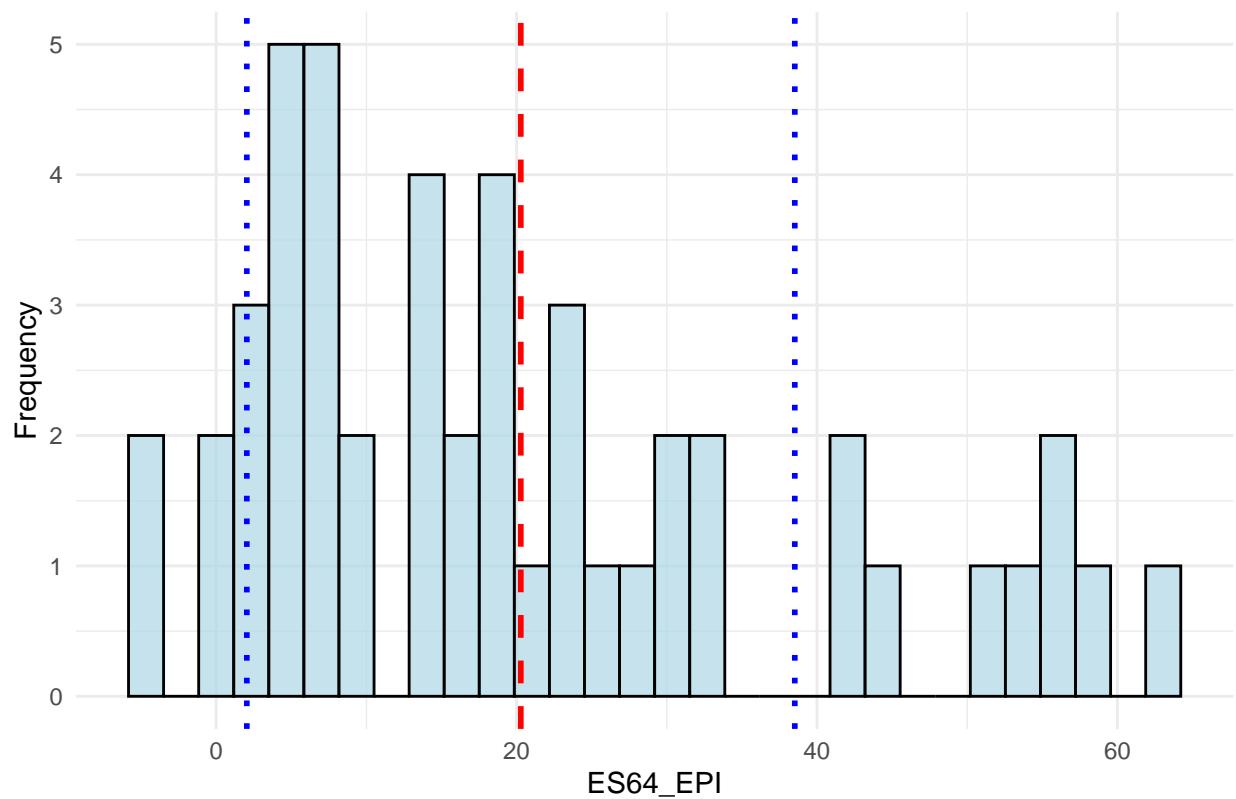
Histogram of AUTO\_EPI target week 4



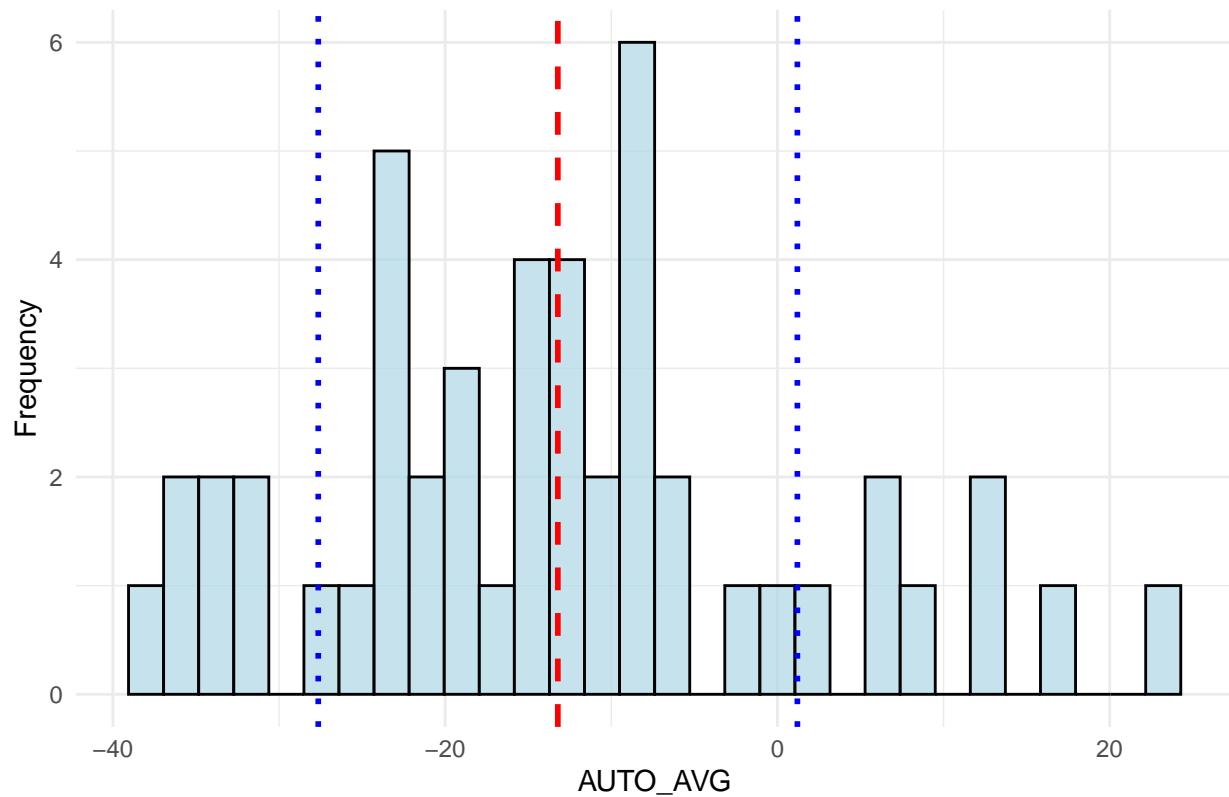
Histogram of ES27\_EPI target week 4



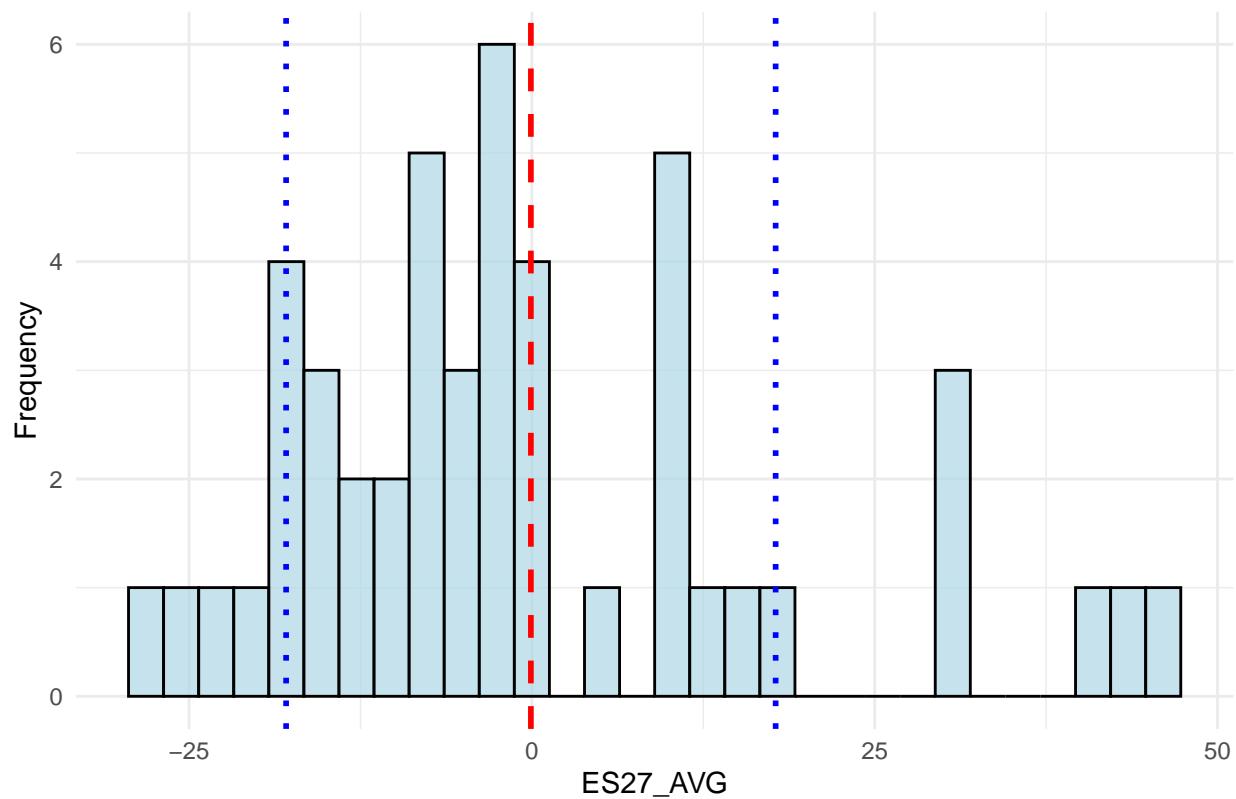
Histogram of ES64\_EPI target week 4



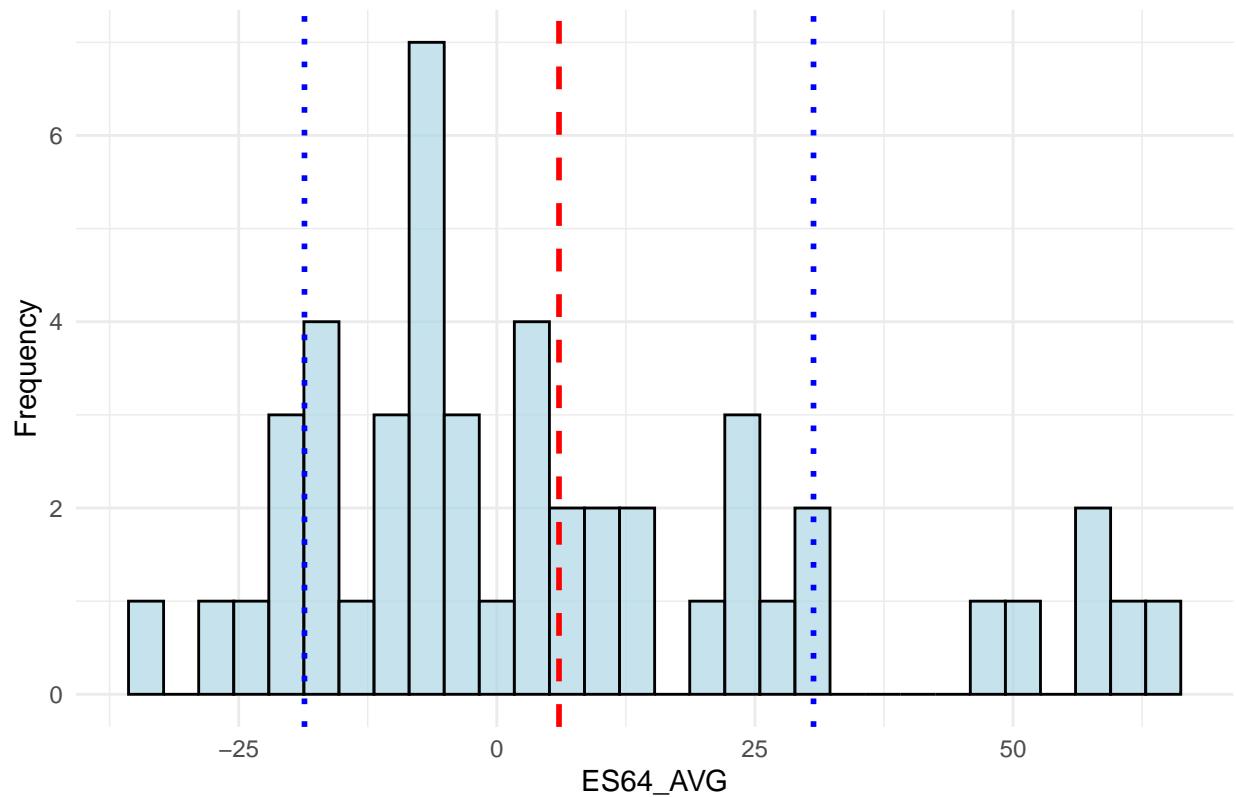
Histogram of AUTO\_AVG target week 4



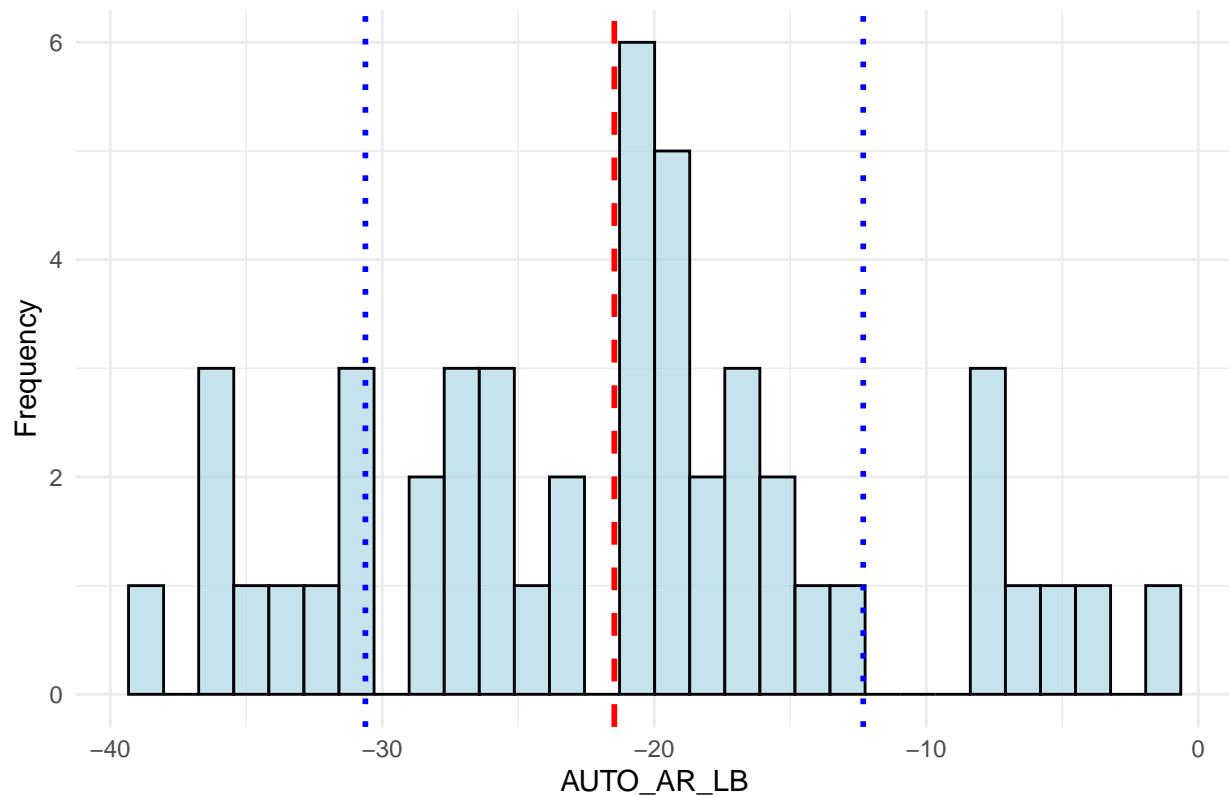
Histogram of ES27\_AVG target week 4



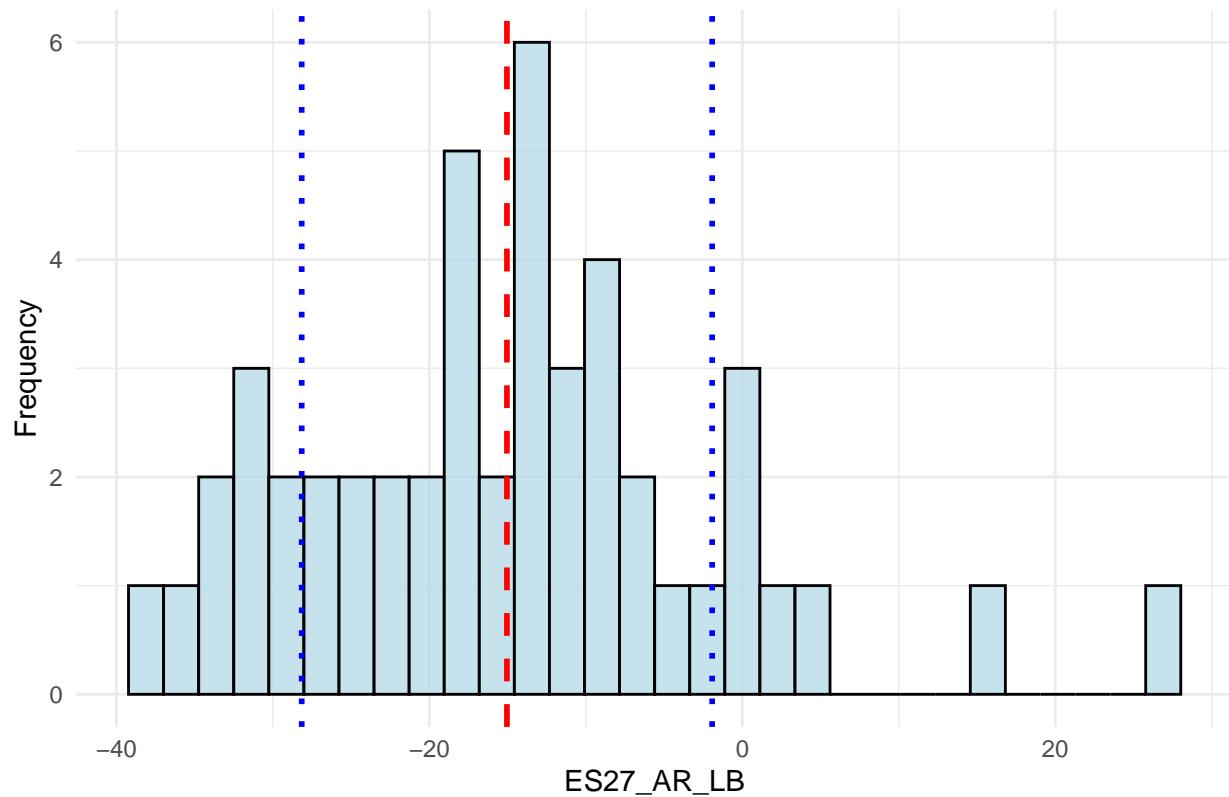
Histogram of ES64\_AVG target week 4



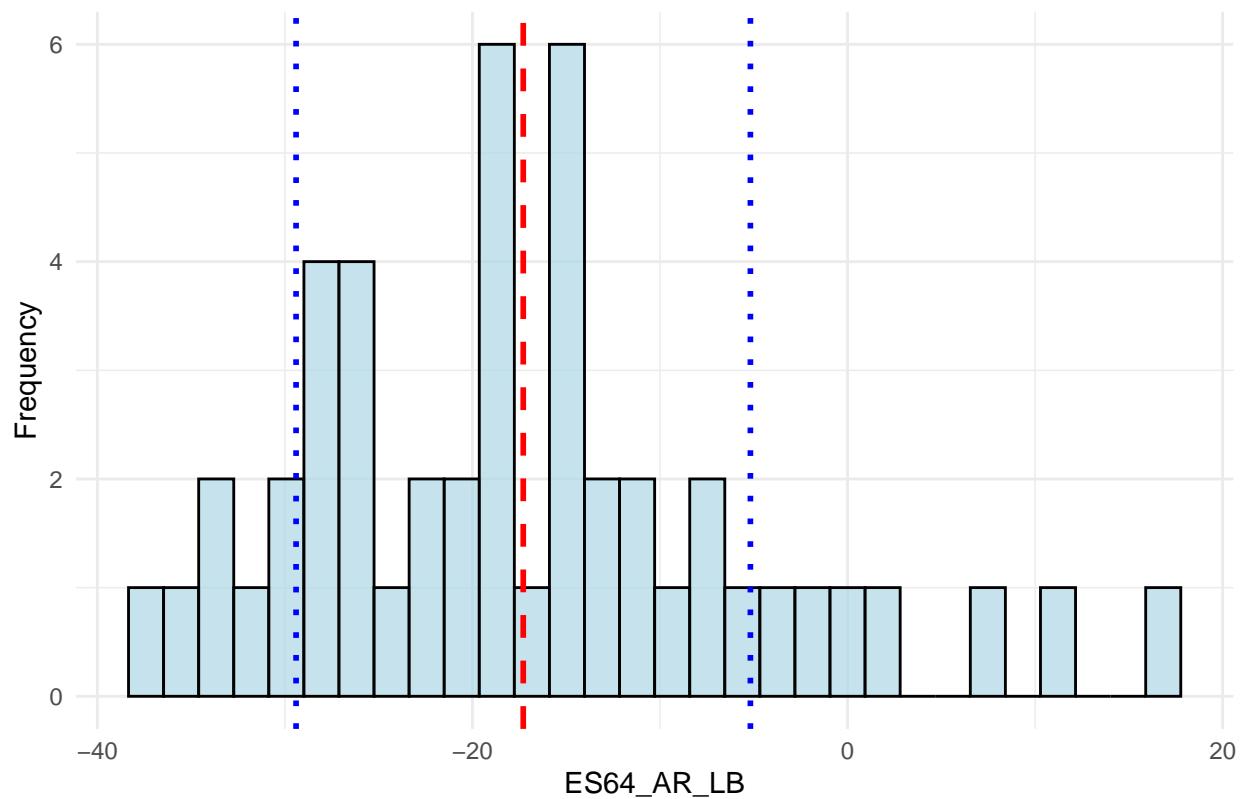
Histogram of AUTO\_AR\_LB target week 4



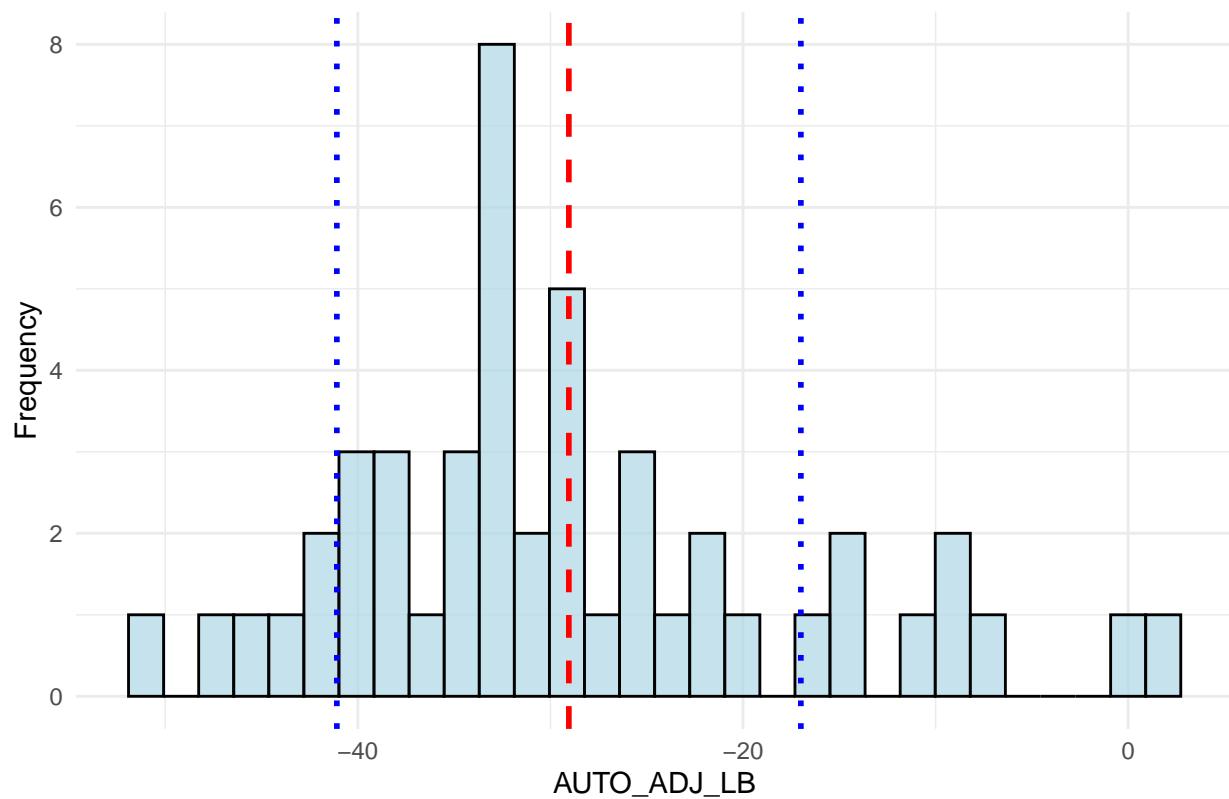
Histogram of ES27\_AR\_LB target week 4



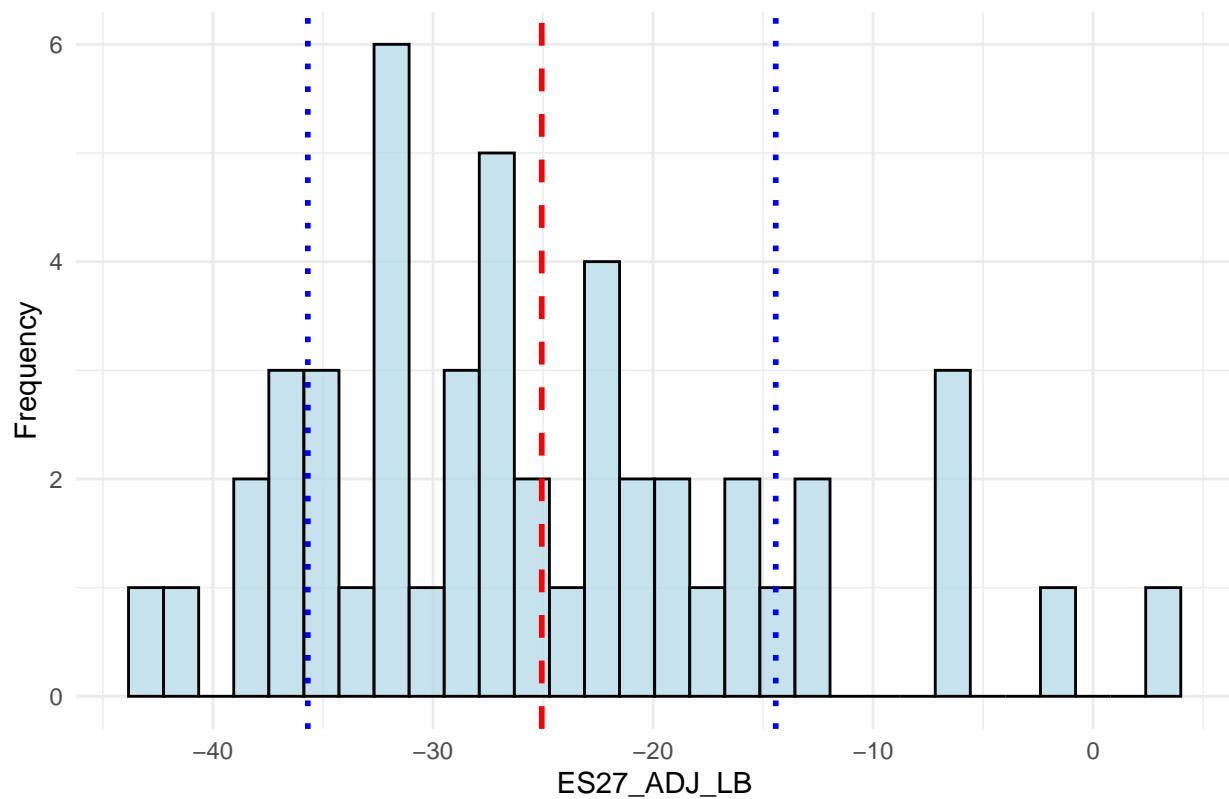
Histogram of ES64\_AR\_LB target week 4



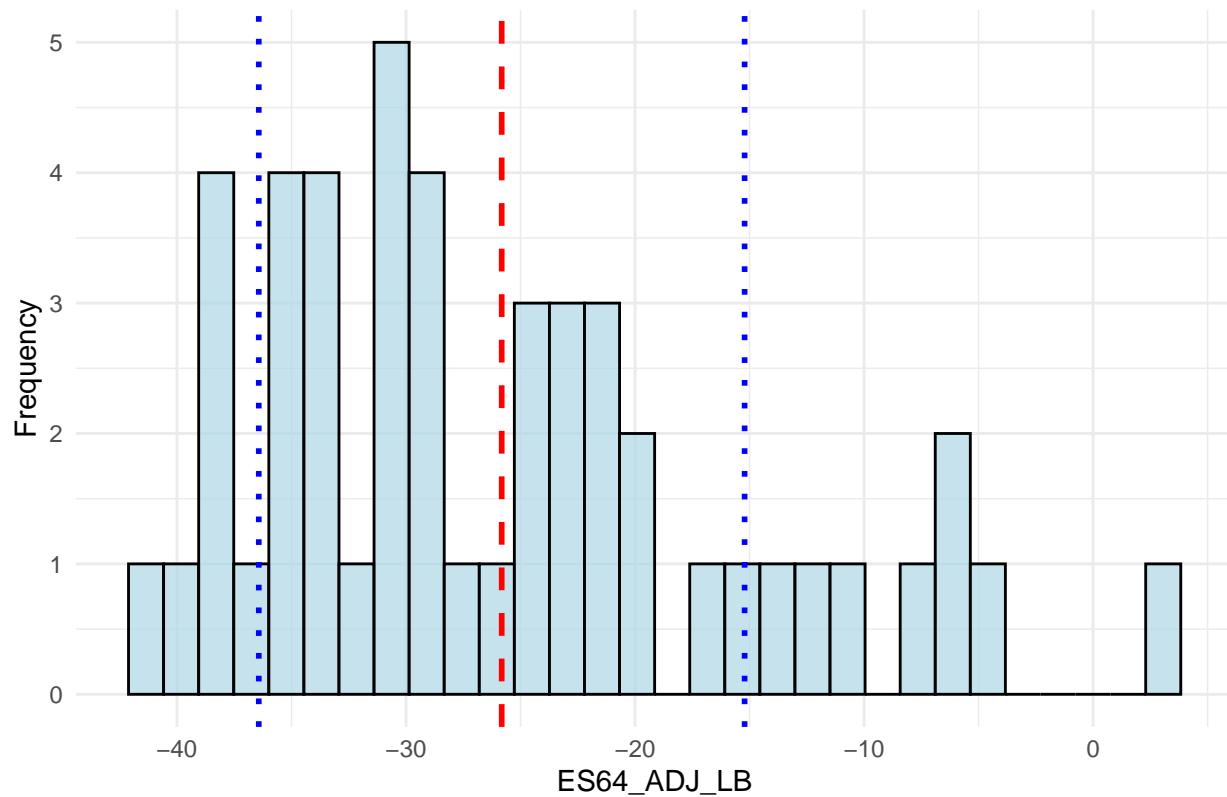
Histogram of AUTO\_ADJ\_LB target week 4



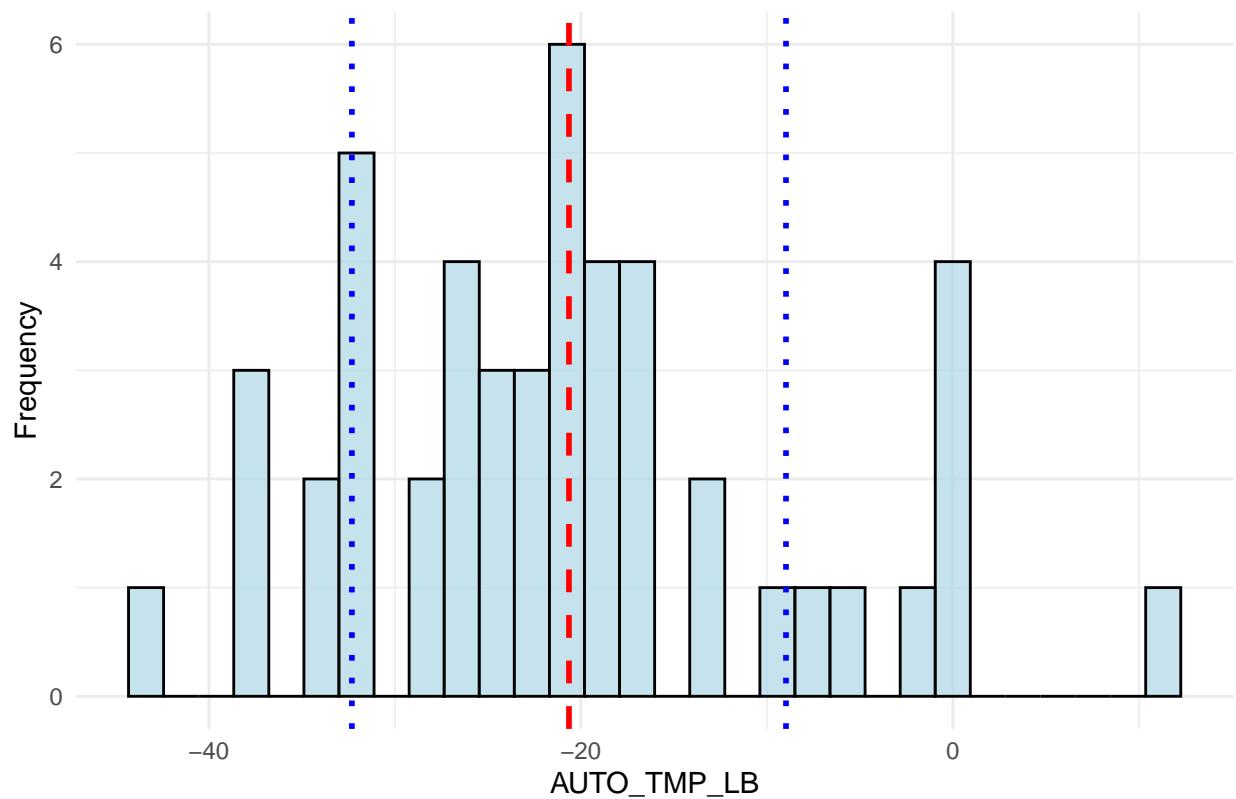
Histogram of ES27\_ADJ\_LB target week 4



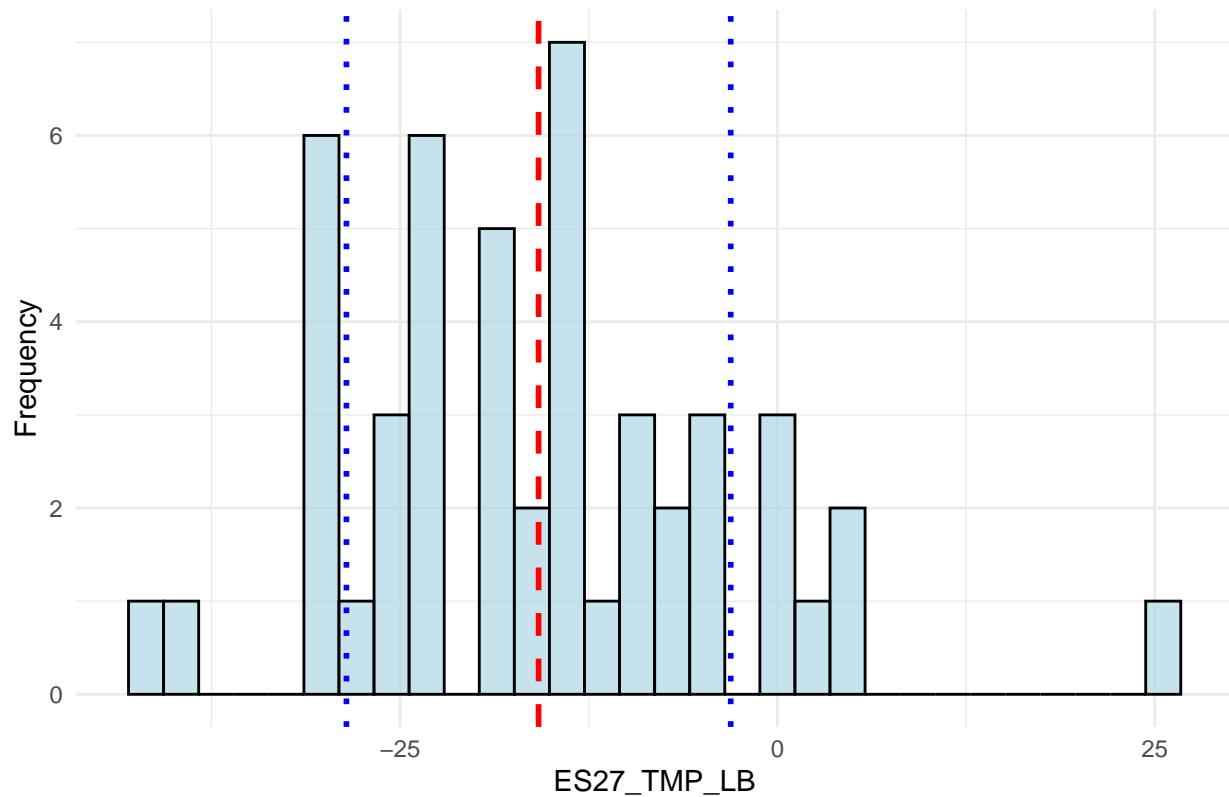
Histogram of ES64\_ADJ\_LB target week 4



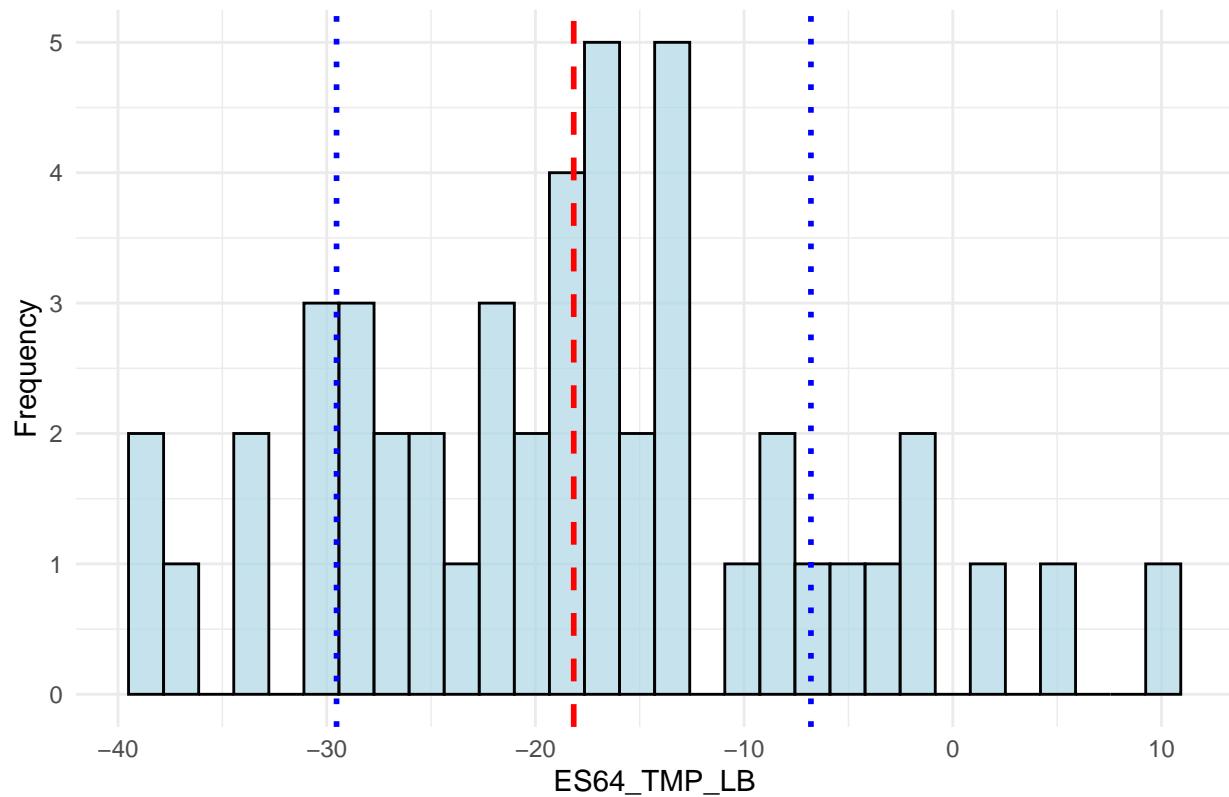
Histogram of AUTO\_TMP\_LB target week 4



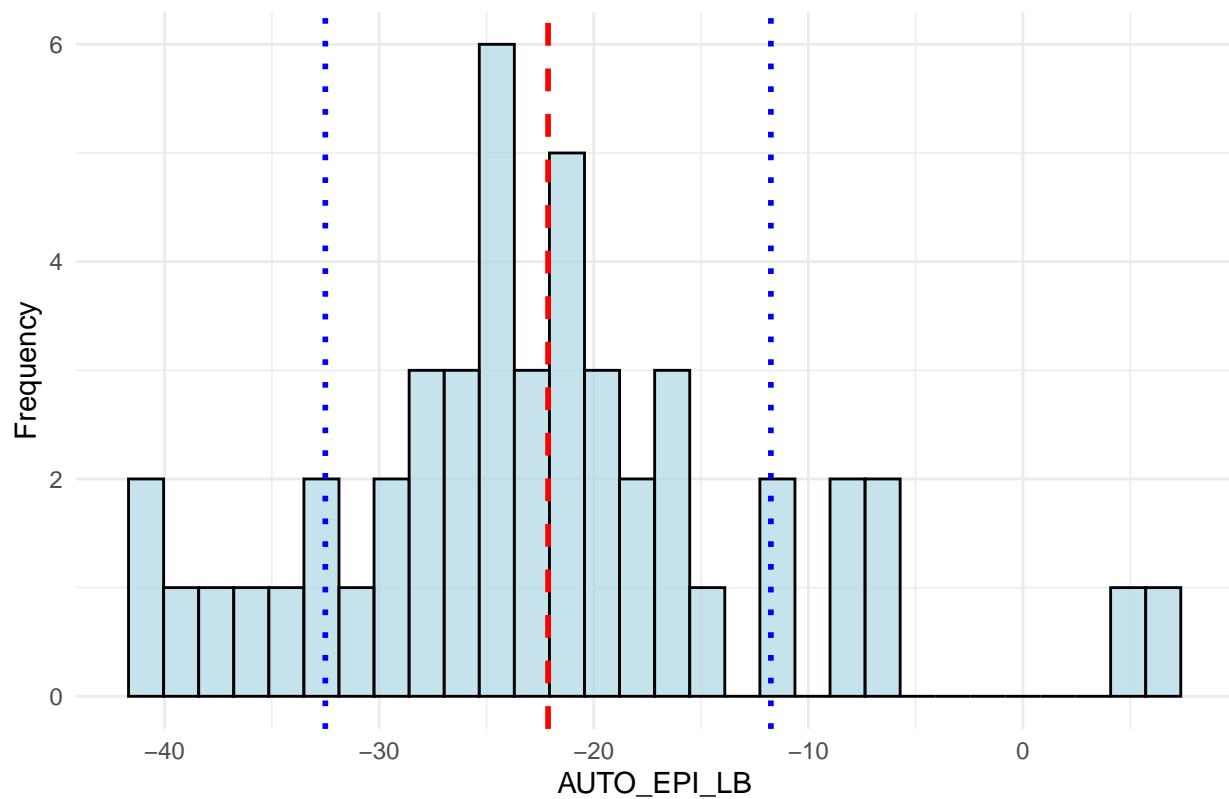
Histogram of ES27\_TMP\_LB target week 4



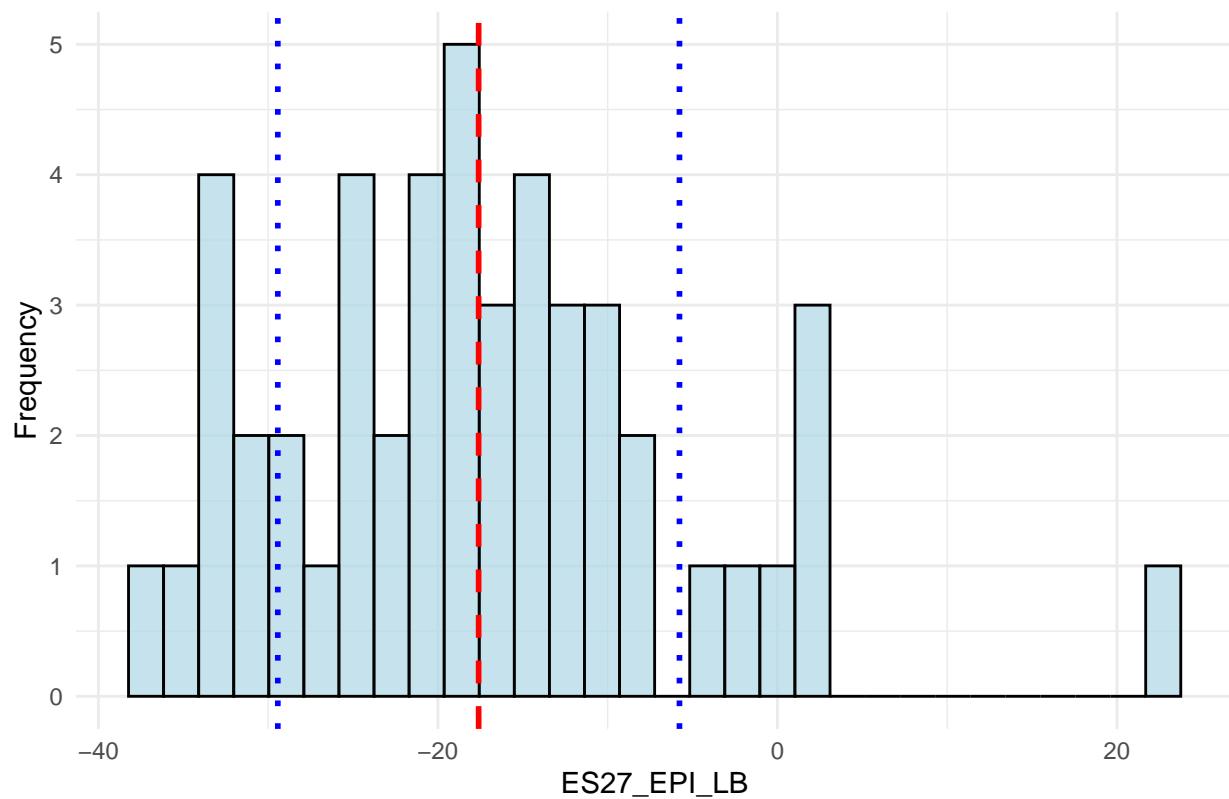
Histogram of ES64\_TMP\_LB target week 4



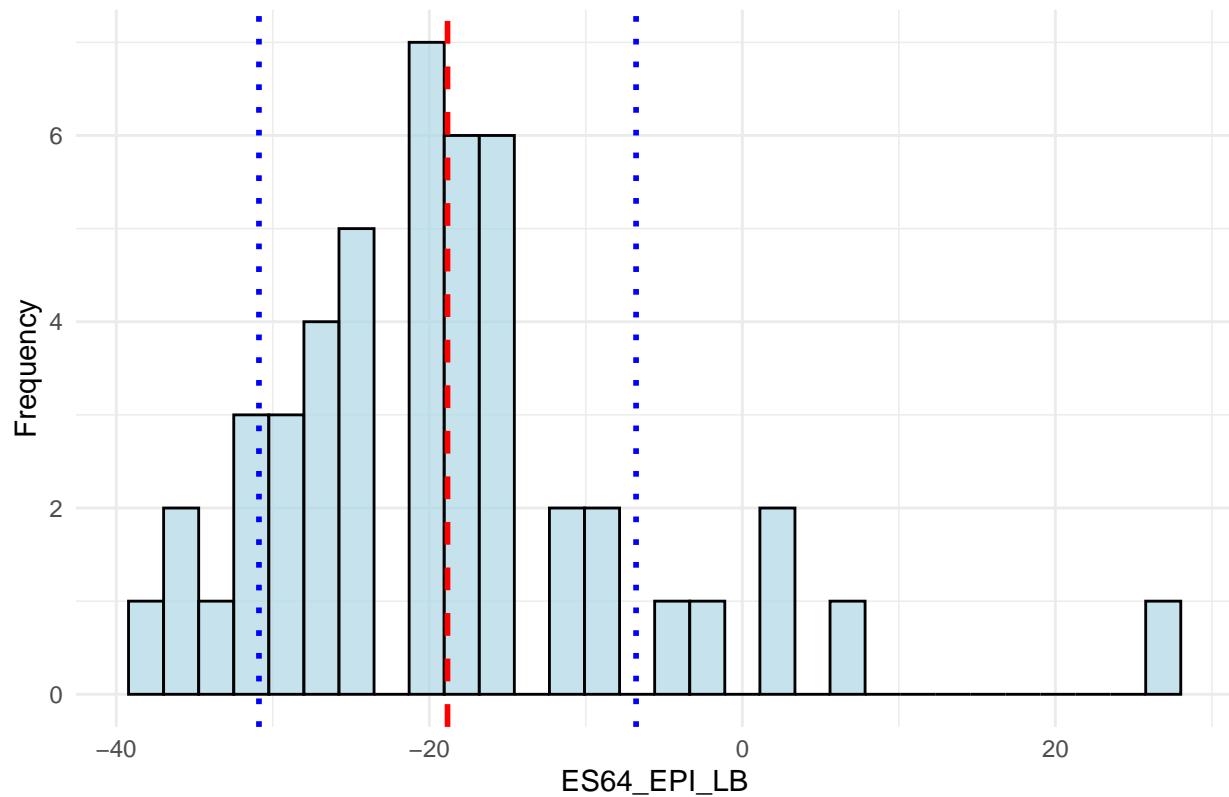
Histogram of AUTO\_EPI\_LB target week 4



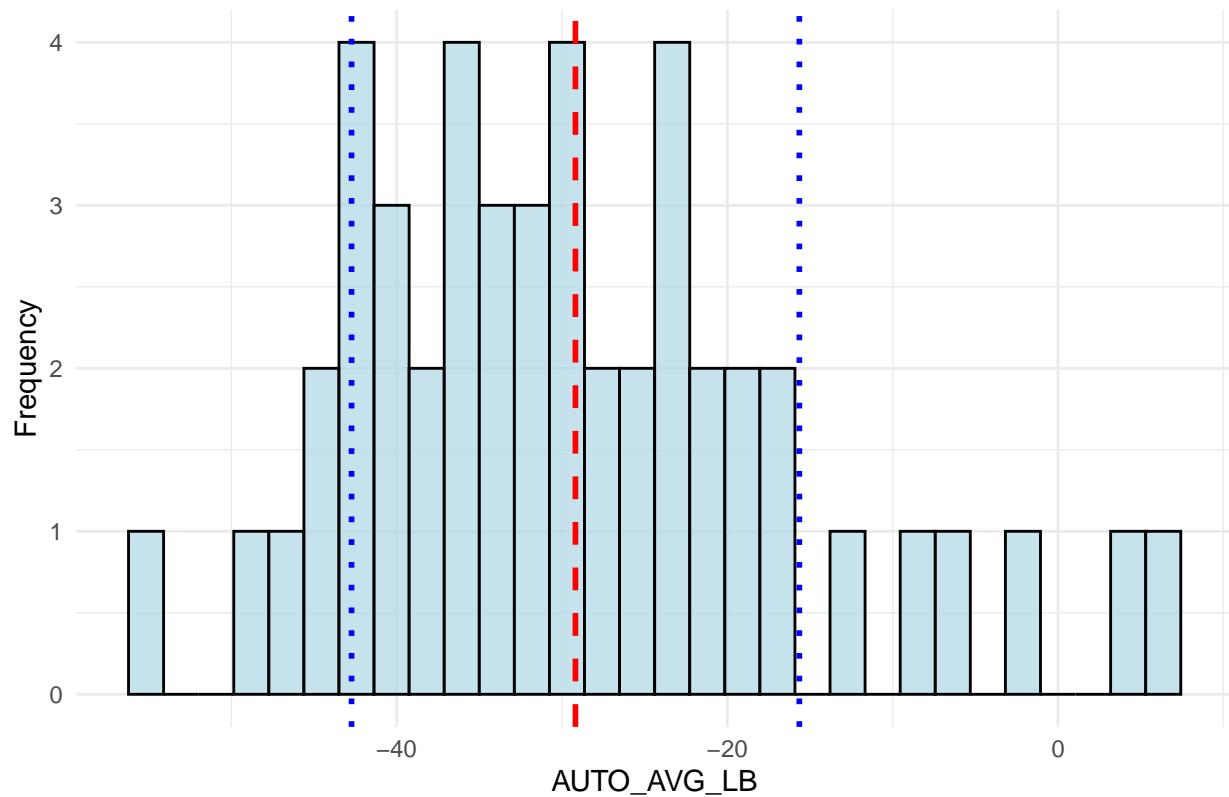
Histogram of ES27\_EPI\_LB target week 4



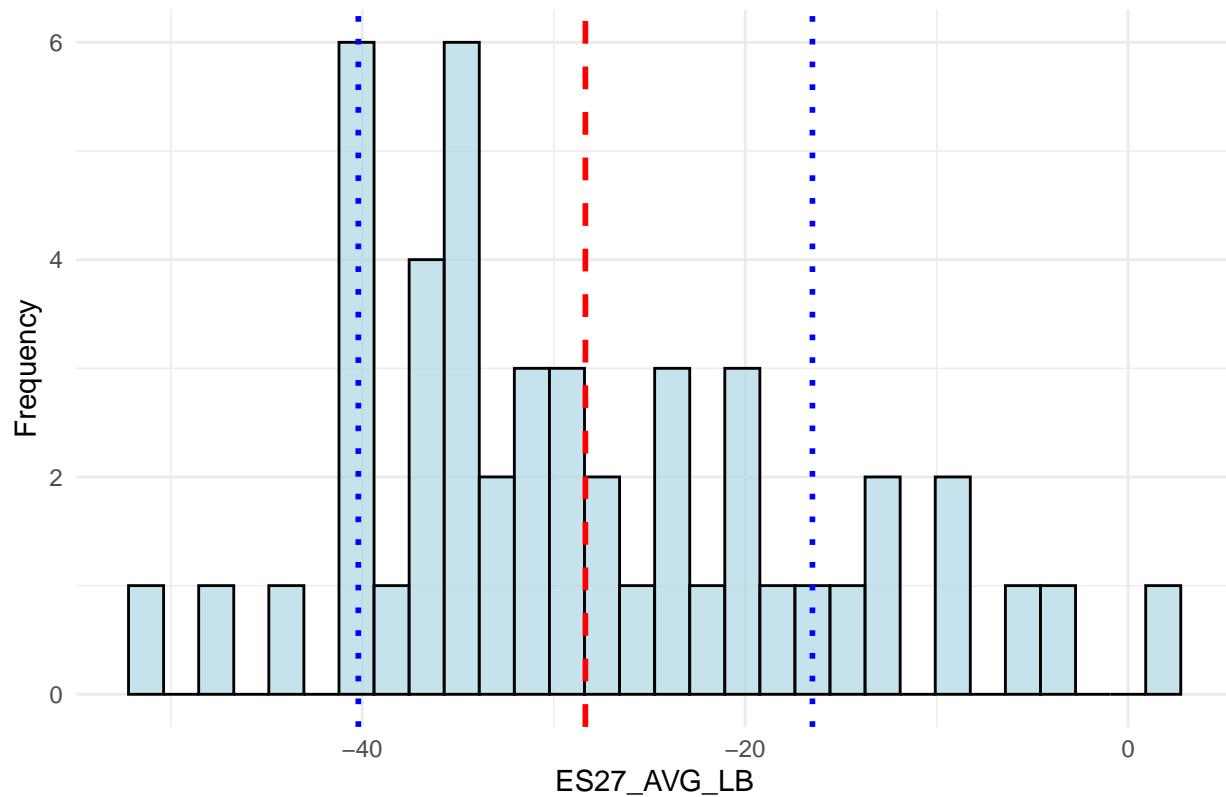
Histogram of ES64\_EPI\_LB target week 4



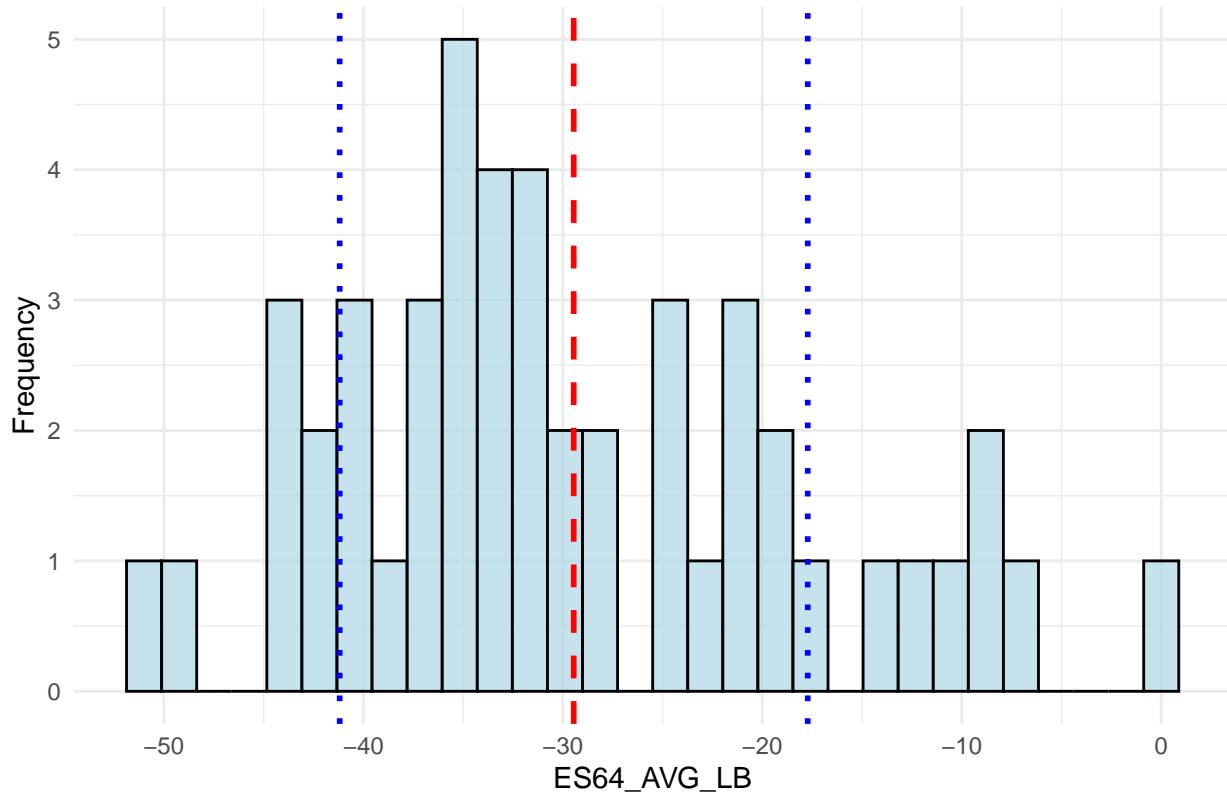
Histogram of AUTO\_AVG\_LB target week 4



Histogram of ES27\_AVG\_LB target week 4



Histogram of ES64\_AVG\_LB target week 4



```
summary_impr$WeekAhead<-as.numeric(summary_impr$WeekAhead)
summary_impr
```

```
##      WeekAhead     Model        m        sd
## 1          1 AUTO_AR 0.00000000 0.000000
## 2          1 ES27_AR -1.25289182 7.686815
## 3          1 ES64_AR  4.04487386 7.551136
## 4          1 AUTO_ADJ -2.95237348 13.168850
## 5          1 ES27_ADJ -1.05148636 11.231060
## 6          1 ES64_ADJ  3.55987148 12.822590
## 7          1 AUTO_TMP  1.59729403 6.681868
## 8          1 ES27_TMP -0.02225404 7.778999
## 9          1 ES64_TMP  6.55526387 7.773443
## 10         1 AUTO_EPI -2.53480697 5.638888
## 11         1 ES27_EPI -0.45731152 8.101599
## 12         1 ES64_EPI  5.04888295 7.947109
## 13         1 AUTO_AVG -4.98018019 13.247177
## 14         1 ES27_AVG -3.72915399 12.314509
## 15         1 ES64_AVG  0.46291910 13.403416
## 16         1 AUTO_AR_LB -10.01950671 10.725925
## 17         1 ES27_AR_LB -8.35136326 11.899393
## 18         1 ES64_AR_LB -8.50637189 11.516636
## 19         1 AUTO_ADJ_LB -19.13598422 11.910906
## 20         1 ES27_ADJ_LB -16.67625266 12.697133
```

```

## 21      1 ES64_ADJ_LB -16.30423259 12.640241
## 22      1 AUTO_TMP_LB -7.16885353 11.894120
## 23      1 ES27_TMP_LB -5.45364982 12.085733
## 24      1 ES64_TMP_LB -5.93976305 11.564033
## 25      1 AUTO_EPI_LB -11.13446060 10.691037
## 26      1 ES27_EPI_LB -9.62820229 11.692673
## 27      1 ES64_EPI_LB -9.47430384 11.521870
## 28      1 AUTO_AVG_LB -21.17299172 14.540335
## 29      1 ES27_AVG_LB -20.83876107 14.475799
## 30      1 ES64_AVG_LB -20.52385817 14.289050
## 31      2 AUTO_AR    0.00000000 0.000000
## 32      2 ES27_AR    1.67643229 11.256876
## 33      2 ES64_AR    9.75266415 11.496636
## 34      2 AUTO_ADJ   -8.22307279 13.843350
## 35      2 ES27_ADJ   -0.99220196 13.841665
## 36      2 ES64_ADJ   4.70814186 17.119378
## 37      2 AUTO_TMP   1.15405870 7.205287
## 38      2 ES27_TMP   2.29145066 11.220501
## 39      2 ES64_TMP   11.36709704 11.592635
## 40      2 AUTO_EPI   -4.93463720 7.614725
## 41      2 ES27_EPI   1.99082885 11.871120
## 42      2 ES64_EPI   10.12708040 11.631949
## 43      2 AUTO_AVG   -9.37011068 13.404967
## 44      2 ES27_AVG   -2.34874441 14.701157
## 45      2 ES64_AVG   2.48861184 17.236884
## 46      2 AUTO_AR_LB -18.20813865 8.327450
## 47      2 ES27_AR_LB -15.19589854 11.235234
## 48      2 ES64_AR_LB -16.17758393 10.406671
## 49      2 AUTO_ADJ_LB -25.99262777 10.651220
## 50      2 ES27_ADJ_LB -23.37052359 11.149179
## 51      2 ES64_ADJ_LB -23.62041465 10.856071
## 52      2 AUTO_TMP_LB -17.41415725 10.014965
## 53      2 ES27_TMP_LB -14.32599596 11.318442
## 54      2 ES64_TMP_LB -15.49762294 10.519694
## 55      2 AUTO_EPI_LB -19.38308019 8.519981
## 56      2 ES27_EPI_LB -16.88270860 10.235228
## 57      2 ES64_EPI_LB -17.51196474 9.722878
## 58      2 AUTO_AVG_LB -28.54266531 9.827794
## 59      2 ES27_AVG_LB -27.87476919 9.786796
## 60      2 ES64_AVG_LB -28.35481664 9.290224
## 61      3 AUTO_AR    0.00000000 0.000000
## 62      3 ES27_AR    5.07251585 13.506917
## 63      3 ES64_AR    14.53158910 15.519272
## 64      3 AUTO_ADJ   -9.76028919 14.582624
## 65      3 ES27_ADJ   0.76005814 15.956042
## 66      3 ES64_ADJ   6.79855818 20.662059
## 67      3 AUTO_TMP   0.91961249 7.585415
## 68      3 ES27_TMP   5.59820273 13.609330
## 69      3 ES64_TMP   15.69700224 15.535054
## 70      3 AUTO_EPI   -6.97630213 9.822077
## 71      3 ES27_EPI   5.49493342 13.857454
## 72      3 ES64_EPI   14.82399110 15.628778
## 73      3 AUTO_AVG   -10.57033304 14.345661
## 74      3 ES27_AVG   -0.04701205 16.831774

```

```

## 75      3   ES64_AVG    5.21960283 21.394604
## 76      3   AUTO_AR_LB -20.73905807  8.206188
## 77      3   ES27_AR_LB -16.30420745 11.219725
## 78      3   ES64_AR_LB -18.09433811 10.421837
## 79      3   AUTO_ADJ_LB -28.01006808 10.627341
## 80      3   ES27_ADJ_LB -24.81921007 10.316992
## 81      3   ES64_ADJ_LB -25.48603934 10.157255
## 82      3   AUTO_TMP_LB -19.87625943 10.331117
## 83      3   ES27_TMP_LB -16.06779327 11.383865
## 84      3   ES64_TMP_LB -18.05722737 10.422076
## 85      3   AUTO_EPI_LB -21.48320517  9.101863
## 86      3   ES27_EPI_LB -18.18885821 10.350102
## 87      3   ES64_EPI_LB -19.32324155  9.861832
## 88      3   AUTO_AVG_LB -29.43786276 11.353391
## 89      3   ES27_AVG_LB -28.76897217 10.331972
## 90      3   ES64_AVG_LB -29.55956206 10.160406
## 91      4   AUTO_AR     0.00000000  0.000000
## 92      4   ES27_AR     8.36417537 13.971694
## 93      4   ES64_AR     20.17564625 18.160920
## 94      4   AUTO_ADJ    -12.97263637 14.841453
## 95      4   ES27_ADJ    0.87529024 16.928595
## 96      4   ES64_ADJ    7.42462321 23.135790
## 97      4   AUTO_TMP    -0.22265325  7.415175
## 98      4   ES27_TMP    8.87287433 14.284722
## 99      4   ES64_TMP    21.10117478 18.044385
## 100     4   AUTO_EPI    -9.35303037 10.614739
## 101     4   ES27_EPI    8.62458564 14.454656
## 102     4   ES64_EPI    20.27888418 18.242002
## 103     4   AUTO_AVG    -13.21804363 14.420478
## 104     4   ES27_AVG    -0.07404029 17.852160
## 105     4   ES64_AVG    6.02253724 24.651413
## 106     4   AUTO_AR_LB -21.46996745  9.151643
## 107     4   ES27_AR_LB -15.04382283 13.114651
## 108     4   ES64_AR_LB -17.29252906 12.112060
## 109     4   AUTO_ADJ_LB -29.04380105 12.047413
## 110     4   ES27_ADJ_LB -25.05508454 10.632225
## 111     4   ES64_ADJ_LB -25.82185401 10.606441
## 112     4   AUTO_TMP_LB -20.63946959 11.670973
## 113     4   ES27_TMP_LB -15.82566080 12.731328
## 114     4   ES64_TMP_LB -18.16260295 11.365151
## 115     4   AUTO_EPI_LB -22.12955035 10.382041
## 116     4   ES27_EPI_LB -17.60081277 11.830589
## 117     4   ES64_EPI_LB -18.83848590 12.048246
## 118     4   AUTO_AVG_LB -29.18522972 13.541902
## 119     4   ES27_AVG_LB -28.34774624 11.857516
## 120     4   ES64_AVG_LB -29.45518522 11.732842

```

Let's create a combined dataset with mean differences, standard deviations and Wilcoxon Holm adjusted p-values

```

#
all_p_values<-rbind(p_values_wk1,p_values_wk2,p_values_wk3,p_values_wk4)

```

```

p_values_and_impr <- merge(summary_impr, all_p_values, by = c("WeekAhead", "Model"))

# Create a new column "Model_Type" based on model with and without log-back transformation
p_values_and_impr$Model_Type <- ifelse(grepl("_LB$", p_values_and_impr$Model), "LB", "no_LB")

# View the updated dataframe
head(p_values_and_impr)

```

	WeekAhead	Model	m	sd	p_values	Model_Type
## 1	1	AUTO_ADJ	-2.952373	13.168850	1.000000e+00	no_LB
## 2	1	AUTO_ADJ_LB	-19.135984	11.910906	3.482370e-11	LB
## 3	1	AUTO_AR_LB	-10.019507	10.725925	9.231208e-07	LB
## 4	1	AUTO_AVG	-4.980180	13.247177	1.161461e-01	no_LB
## 5	1	AUTO_AVG_LB	-21.172992	14.540335	1.809289e-08	LB
## 6	1	AUTO_EPI	-2.534807	5.638888	1.532974e-02	no_LB

p\_values\_and\_impr

	WeekAhead	Model	m	sd	p_values	Model_Type
## 1	1	AUTO_ADJ	-2.95237348	13.168850	1.000000e+00	no_LB
## 2	1	AUTO_ADJ_LB	-19.13598422	11.910906	3.482370e-11	LB
## 3	1	AUTO_AR_LB	-10.01950671	10.725925	9.231208e-07	LB
## 4	1	AUTO_AVG	-4.98018019	13.247177	1.161461e-01	no_LB
## 5	1	AUTO_AVG_LB	-21.17299172	14.540335	1.809289e-08	LB
## 6	1	AUTO_EPI	-2.53480697	5.638888	1.532974e-02	no_LB
## 7	1	AUTO_EPI_LB	-11.13446060	10.691037	2.567875e-07	LB
## 8	1	AUTO_TMP	1.59729403	6.681868	1.000000e+00	no_LB
## 9	1	AUTO_TMP_LB	-7.16885353	11.894120	2.507425e-04	LB
## 10	1	ES27_ADJ	-1.05148636	11.231060	1.000000e+00	no_LB
## 11	1	ES27_ADJ_LB	-16.67625266	12.697133	2.257359e-08	LB
## 12	1	ES27_AR	-1.25289182	7.686815	6.422568e-01	no_LB
## 13	1	ES27_AR_LB	-8.35136326	11.899393	1.044917e-04	LB
## 14	1	ES27_AVG	-3.72915399	12.314509	1.000000e+00	no_LB
## 15	1	ES27_AVG_LB	-20.83876107	14.475799	2.101837e-08	LB
## 16	1	ES27_EPI	-0.45731152	8.101599	1.000000e+00	no_LB
## 17	1	ES27_EPI_LB	-9.62820229	11.692673	4.817238e-05	LB
## 18	1	ES27_TMP	-0.02225404	7.778999	1.000000e+00	no_LB
## 19	1	ES27_TMP_LB	-5.45364982	12.085733	1.804422e-02	LB
## 20	1	ES64_ADJ	3.55987148	12.822590	1.161461e-01	no_LB
## 21	1	ES64_ADJ_LB	-16.30423259	12.640241	5.091761e-08	LB
## 22	1	ES64_AR	4.04487386	7.551136	1.161461e-01	no_LB
## 23	1	ES64_AR_LB	-8.50637189	11.516636	6.763713e-05	LB
## 24	1	ES64_AVG	0.46291910	13.403416	1.000000e+00	no_LB
## 25	1	ES64_AVG_LB	-20.52385817	14.289050	1.952345e-08	LB
## 26	1	ES64_EPI	5.04888295	7.947109	3.663236e-02	no_LB
## 27	1	ES64_EPI_LB	-9.47430384	11.521870	5.546505e-05	LB
## 28	1	ES64_TMP	6.55526387	7.773443	4.741434e-05	no_LB
## 29	1	ES64_TMP_LB	-5.93976305	11.564033	6.011928e-03	LB
## 30	2	AUTO_ADJ	-8.22307279	13.843350	5.094992e-03	no_LB
## 31	2	AUTO_ADJ_LB	-25.99262777	10.651220	5.542233e-13	LB
## 32	2	AUTO_AR_LB	-18.20813865	8.327450	4.263256e-12	LB
## 33	2	AUTO_AVG	-9.37011068	13.404967	2.629954e-04	no_LB

## 34	2	AUTO_AVG_LB	-28.54266531	9.827794	2.060574e-13	LB
## 35	2	AUTO_EPI	-4.93463720	7.614725	5.430881e-04	no_LB
## 36	2	AUTO_EPI_LB	-19.38308019	8.519981	5.542233e-13	LB
## 37	2	AUTO_TMP	1.15405870	7.205287	1.000000e+00	no_LB
## 38	2	AUTO_TMP_LB	-17.41415725	10.014965	2.594831e-10	LB
## 39	2	ES27_ADJ	-0.99220196	13.841665	1.000000e+00	no_LB
## 40	2	ES27_ADJ_LB	-23.37052359	11.149179	1.045919e-10	LB
## 41	2	ES27_AR	1.67643229	11.256876	1.000000e+00	no_LB
## 42	2	ES27_AR_LB	-15.19589854	11.235234	7.335643e-09	LB
## 43	2	ES27_AVG	-2.34874441	14.701157	1.000000e+00	no_LB
## 44	2	ES27_AVG_LB	-27.87476919	9.786796	2.060574e-13	LB
## 45	2	ES27_EPI	1.99082885	11.871120	1.000000e+00	no_LB
## 46	2	ES27_EPI_LB	-16.88270860	10.235228	2.586795e-09	LB
## 47	2	ES27_TMP	2.29145066	11.220501	1.000000e+00	no_LB
## 48	2	ES27_TMP_LB	-14.32599596	11.318442	7.335643e-09	LB
## 49	2	ES64_ADJ	4.70814186	17.119378	4.419932e-01	no_LB
## 50	2	ES64_ADJ_LB	-23.62041465	10.856071	1.045919e-10	LB
## 51	2	ES64_AR	9.75266415	11.496636	2.629954e-04	no_LB
## 52	2	ES64_AR_LB	-16.17758393	10.406671	3.571415e-09	LB
## 53	2	ES64_AVG	2.48861184	17.236884	1.000000e+00	no_LB
## 54	2	ES64_AVG_LB	-28.35481664	9.290224	2.060574e-13	LB
## 55	2	ES64_EPI	10.12708040	11.631949	1.363122e-04	no_LB
## 56	2	ES64_EPI_LB	-17.51196474	9.722878	2.891909e-10	LB
## 57	2	ES64_TMP	11.36709704	11.592635	2.131640e-05	no_LB
## 58	2	ES64_TMP_LB	-15.49762294	10.519694	3.814620e-09	LB
## 59	3	AUTO_ADJ	-9.76028919	14.582624	1.318634e-03	no_LB
## 60	3	AUTO_ADJ_LB	-28.01006808	10.627341	2.060574e-13	LB
## 61	3	AUTO_AR_LB	-20.73905807	8.206188	2.060574e-13	LB
## 62	3	AUTO_AVG	-10.57033304	14.345661	8.570441e-04	no_LB
## 63	3	AUTO_AVG_LB	-29.43786276	11.353391	5.115908e-13	LB
## 64	3	AUTO_EPI	-6.97630213	9.822077	1.453191e-04	no_LB
## 65	3	AUTO_EPI_LB	-21.48320517	9.101863	3.552714e-13	LB
## 66	3	AUTO_TMP	0.91961249	7.585415	1.000000e+00	no_LB
## 67	3	AUTO_TMP_LB	-19.87625943	10.331117	5.393019e-12	LB
## 68	3	ES27_ADJ	0.76005814	15.956042	1.000000e+00	no_LB
## 69	3	ES27_ADJ_LB	-24.81921007	10.316992	2.141576e-11	LB
## 70	3	ES27_AR	5.07251585	13.506917	1.000000e+00	no_LB
## 71	3	ES27_AR_LB	-16.30420745	11.219725	2.458123e-10	LB
## 72	3	ES27_AVG	-0.04701205	16.831774	1.000000e+00	no_LB
## 73	3	ES27_AVG_LB	-28.76897217	10.331972	2.060574e-13	LB
## 74	3	ES27_EPI	5.49493342	13.857454	1.000000e+00	no_LB
## 75	3	ES27_EPI_LB	-18.18885821	10.350102	7.505605e-10	LB
## 76	3	ES27_TMP	5.59820273	13.609330	1.000000e+00	no_LB
## 77	3	ES27_TMP_LB	-16.06779327	11.383865	8.083134e-10	LB
## 78	3	ES64_ADJ	6.79855818	20.662059	1.626837e-01	no_LB
## 79	3	ES64_ADJ_LB	-25.48603934	10.157255	2.141576e-11	LB
## 80	3	ES64_AR	14.53158910	15.519272	1.145431e-06	no_LB
## 81	3	ES64_AR_LB	-18.09433811	10.421837	2.224141e-10	LB
## 82	3	ES64_AVG	5.21960283	21.394604	1.000000e+00	no_LB
## 83	3	ES64_AVG_LB	-29.55956206	10.160406	2.060574e-13	LB
## 84	3	ES64_EPI	14.82399110	15.628778	1.145242e-06	no_LB
## 85	3	ES64_EPI_LB	-19.32324155	9.861832	1.082867e-10	LB
## 86	3	ES64_TMP	15.69700224	15.535054	9.779156e-08	no_LB
## 87	3	ES64_TMP_LB	-18.05722737	10.422076	1.702389e-10	LB

```

## 88      4 AUTO_ADJ -12.97263637 14.841453 2.799617e-05 no_LB
## 89      4 AUTO_ADJ_LB -29.04380105 12.047413 1.293188e-12 LB
## 90      4 AUTO_AR_LB -21.46996745 9.151643 2.060574e-13 LB
## 91      4 AUTO_AVG -13.21804363 14.420478 1.982266e-05 no_LB
## 92      4 AUTO_AVG_LB -29.18522972 13.541902 8.185452e-11 LB
## 93      4 AUTO_EPI -9.35303037 10.614739 9.690508e-07 no_LB
## 94      4 AUTO_EPI_LB -22.12955035 10.382041 1.797673e-11 LB
## 95      4 AUTO_TMP -0.22265325 7.415175 5.914043e-01 no_LB
## 96      4 AUTO_TMP_LB -20.63946959 11.670973 7.617018e-11 LB
## 97      4 ES27_ADJ 0.87529024 16.928595 1.000000e+00 no_LB
## 98      4 ES27_ADJ_LB -25.05508454 10.632225 4.263256e-12 LB
## 99      4 ES27_AR 8.36417537 13.971694 2.645923e-02 no_LB
## 100     4 ES27_AR_LB -15.04382283 13.114651 1.012994e-07 LB
## 101     4 ES27_AVG -0.07404029 17.852160 1.000000e+00 no_LB
## 102     4 ES27_AVG_LB -28.34774624 11.857516 9.592327e-13 LB
## 103     4 ES27_EPI 8.62458564 14.454656 4.762452e-02 no_LB
## 104     4 ES27_EPI_LB -17.60081277 11.830589 3.365841e-09 LB
## 105     4 ES27_TMP 8.87287433 14.284722 7.764381e-03 no_LB
## 106     4 ES27_TMP_LB -15.82566080 12.731328 2.970194e-08 LB
## 107     4 ES64_ADJ 7.42462321 23.135790 3.263910e-01 no_LB
## 108     4 ES64_ADJ_LB -25.82185401 10.606441 1.776357e-12 LB
## 109     4 ES64_AR 20.17564625 18.160920 7.617018e-11 no_LB
## 110     4 ES64_AR_LB -17.29252906 12.112060 3.387709e-08 LB
## 111     4 ES64_AVG 6.02253724 24.651413 1.000000e+00 no_LB
## 112     4 ES64_AVG_LB -29.45518522 11.732842 3.979039e-13 LB
## 113     4 ES64_EPI 20.27888418 18.242002 6.669865e-11 no_LB
## 114     4 ES64_EPI_LB -18.83848590 12.048246 3.174591e-09 LB
## 115     4 ES64_TMP 21.10117478 18.044385 2.141576e-11 no_LB
## 116     4 ES64_TMP_LB -18.16260295 11.365151 7.974705e-10 LB

```

Here we are plotting the mean differences and standard deviation

```

# Get model order based based on lower values on WeekAhead == 1
model_order <- p_values_and_impr %>%
  filter(WeekAhead == 1) %>%
  arrange(desc(m)) %>%
  pull(Model)

# Prepare data with significance level, ordering, and model name
p_df <- p_values_and_impr %>%
  mutate(Significance = ifelse(p_values < 0.05 , "Significant", "Not Significant"),
         Model_Type = ifelse(grepl("_LB$", Model), "With log-back transformation", "Without log-back transformation"),
         Model = factor(Model, levels = model_order))

# Heatmap plot
plot_m_heatmap <- ggplot(p_df, aes(x = factor(WeekAhead), y = Model, fill = m)) +
  geom_tile(aes(fill = ifelse(Significance == "Significant", m, NA)), color = "black") + # Fill only significant cells
  geom_tile(data = p_df %>% filter(Significance == "Not Significant"),
            aes(x = factor(WeekAhead), y = Model, fill = "gray80", color = "black")) + # Gray for non-significant cells
  geom_text(aes(label = paste0("m:", round(m, 1), ", sd:", round(sd, 1))),
            color = "black", size = 4) + # Add text labels
  scale_fill_gradient2(
    low = "cyan",
    mid = "white",
    high = "red")

```

```

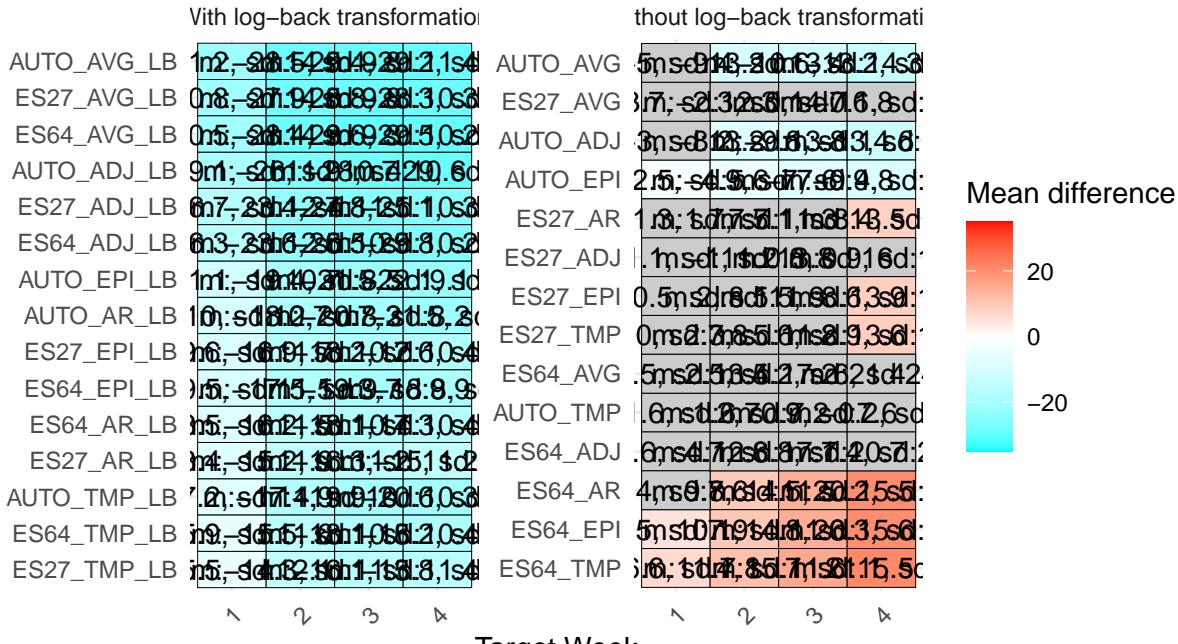
high = "red",
midpoint = 0,      # <-- Sets 0 as the midpoint
na.value = "gray80",
name = "Mean difference",
limits = c(-35, 35)
) +
  labs(title = "Mean differences between models' WIS and the AUTO_AR baseline across 48 states",
       subtitle = "Gray boxes indicate models that are not significantly different from the baseline (p > 0.05)",
       caption = "m = mean difference, sd = standard deviation",
       x = "Target Week",
       y = "") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1),
        plot.caption = element_text(size = 16),
        plot.title = element_text(size = 18, face = "bold"),
        plot.subtitle = element_text(size = 14),) +
  facet_wrap(~ Model_Type, scales = "free_y")

# Print plot
print(plot_m_heatmap)

```

## Mean differences between models' WIS and the AUTO\_AR baseline across 48 states

Gray boxes indicate models that are not significantly different from the baseline (p > 0.05)



```
ggsave("Fig5.jpg", plot_m_heatmap, width = 14, height = 7 )
```

Now let's use a regression model to evaluate effect size of each modelling approach.

```

# get model approaches and mean improvement
model_approaches <- summary_impr %>%
  mutate(
    LogBack = grepl("_LB$", Model),
    Type = sub(".*", "", Model), # extract the model type
    Family = sub("^.*?(AR|ADJ|EPI|AVG|TMP)(_LB)?$", "\\\\$1", Model)
  )

# rename "AR" because the intercept is in alphabetic order and I want to use the baseline
model_approaches <- model_approaches %>%
  mutate(Family = ifelse(Family == "AR", "AAR", Family))

# run the regression model
model <- lm(m ~ Type + Family + LogBack, data = model_approaches)

# extracting models coefficients
tidy_model <- broom::tidy(model)

# filter out intercept
tidy_model_no_intercept <- tidy_model %>% filter(term != "(Intercept)")

# Reordering for plotting
tidy_model_no_intercept <- tidy_model_no_intercept %>%
  mutate(term = forcats::fct_reorder(term, estimate, .desc = TRUE))

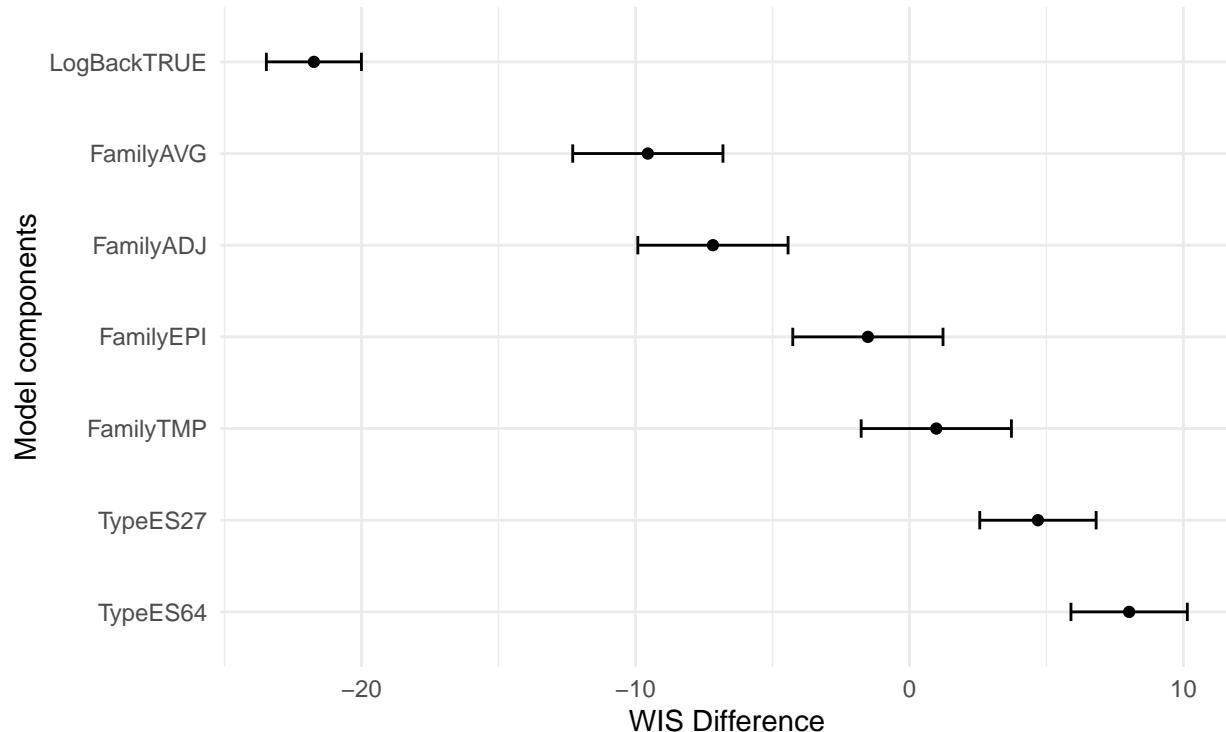
# Plot
effect_size<- ggplot(tidy_model_no_intercept, aes(x = term, y = estimate)) +
  geom_point() +
  geom_errorbar(aes(ymin = estimate - std.error * 1.96,
                     ymax = estimate + std.error * 1.96),
                width = 0.2) +
  coord_flip() +
  labs(
    title = "Effect size of model type, family and transformation on WIS difference",
    subtitle = "WIS Difference relative to the AUTO type, ARIMA family and LogBack False (intercept)",
    y = "WIS Difference",
    x = "Model components"
  ) +
  theme_minimal() + theme(plot.title = element_text(size = 16, face = "bold"),
                          plot.subtitle = element_text(size = 14))

effect_size

```

## Effect size of model type, family and transformation

WIS Difference relative to the AUTO type, ARIMA family and LogBackTRUE transformation.



```
ggsave("Fig6.jpg", effect_size, width =10, height =5)
```

Now let's organize the data to plot some maps with the best models.

```
# MAP 1 week ahead
W1_map <- data.frame(
  STATE = W1$STATE,
  percentage_improvement = W1_percentage_of_improvement$AUTO_AVG_LB,
  AUTO_AVG_LB=W1$AUTO_AVG_LB
)

# Include geometry
W1_map <- states %>%
  left_join(W1_map, by = c("STATE")) %>%
  drop_na()

# MAP 2 weeks a ahead
W2_map <- data.frame(
  STATE = W2$STATE,
  percentage_improvement = W2_percentage_of_improvement$AUTO_AVG_LB,
  AUTO_AVG_LB=W2$AUTO_AVG_LB
)

# Include geometry
W2_map <- states %>%
```

```

left_join(W2_map, by = c("STATE"))%>%
drop_na()

# MAP 3 weeks a ahead
W3_map <- data.frame(
  STATE = W3$STATE,
  percentage_improvement = W3_percentage_of_improvement$AUTO_AVG_LB,
  AUTO_AVG_LB=W3$AUTO_AVG_LB
)

# Include geometry
W3_map <- states %>%
  left_join(W3_map, by = c("STATE"))%>%
  drop_na()

# MAP 4 weeks ahead
W4_map <- data.frame(
  STATE = W4$STATE,
  percentage_improvement = W4_percentage_of_improvement$AUTO_AVG_LB,
  AUTO_AVG_LB=W4$AUTO_AVG_LB
)

# Include geometry
W4_map <- states %>%
  left_join(W4_map, by = c("STATE"))%>%
  drop_na()

```

Let's plot some maps with mean WIS for each state.

1 week ahead percentage of improvement

```

ES_1WEEK <- ggplot(W1_map) +
  geom_sf(aes(fill = AUTO_AVG_LB)) + # Fill based on the best model
  scale_fill_gradient2(low = "skyblue", mid = "lightyellow2", high = "red", midpoint = 100, limits = c(0, 100)) +
  theme_light() +
  theme(legend.position = "right",
        plot.title = element_text(hjust = 0.5)) +
  geom_sf_text(data = W1_map, aes(label = round(AUTO_AVG_LB,0)), # Round to whole numbers
               size = 3,
               color = "black",
               check_overlap = TRUE, fontface = "bold") +
  labs(
    fill = "mean WIS",
    x = "",
    y = ""
  )

x_limits <- c(-125, -67) # Set the desired longitude range
y_limits <- c(25, 50)     # Set the desired latitude range

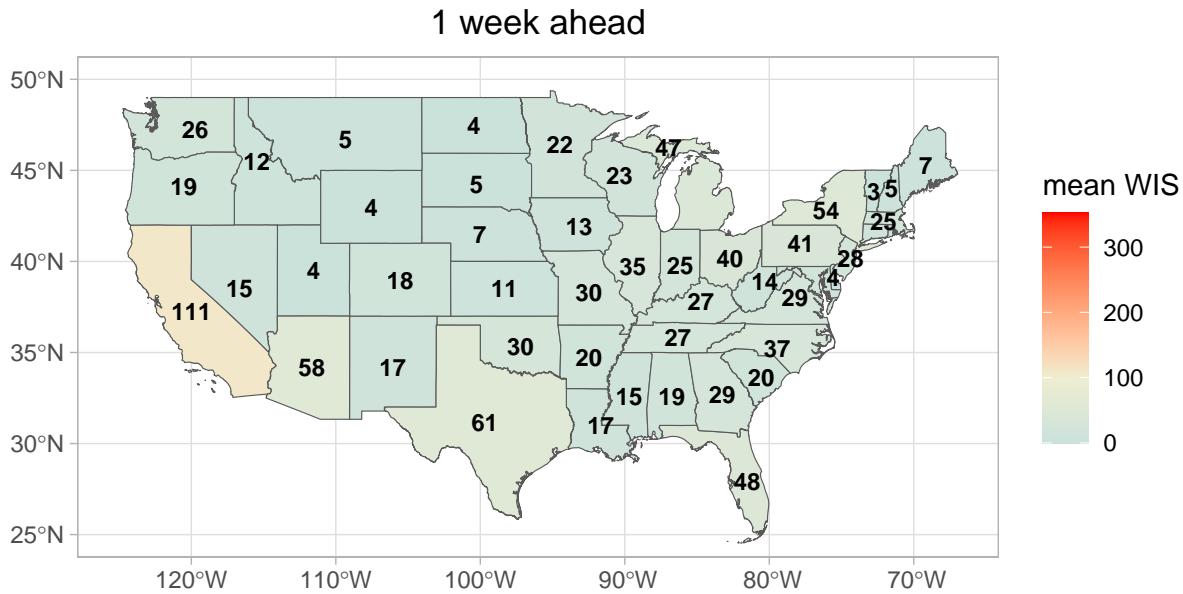
```

```

ES_1WEEK<- ES_1WEEK + coord_sf(xlim = x_limits, ylim = y_limits)
ES_1WEEK

```

```
## Warning in st_point_on_surface.sfc(sf::st_zm(x)): st_point_on_surface may not
## give correct results for longitude/latitude data
```



```
#ggsave(ES_1WEEK, height = 4.5, width = 6.5, filename="Fig4A.jpg")
```

2 weeks ahead

```
ES_2WEEK <- ggplot(W2_map) +
  geom_sf(aes(fill = AUTO_AVG_LB)) + # Fill based on the best model
  scale_fill_gradient2(low = "skyblue", mid = "lightyellow2", high = "red", midpoint = 100, limits = c(0,300))
  ggtitle("2 weeks ahead") +
  theme_light() +
  theme(legend.position = "right",
        plot.title = element_text(hjust = 0.5)) +
  geom_sf_text(data = W2_map, aes(label = round(AUTO_AVG_LB,0))), # Round to whole numbers
  size = 3,
  color = "black",
  check_overlap = TRUE, fontface = "bold") + # Display percentage improvement in each stat
  labs(
    fill = "mean WIS",
    x = "",
    y = ""
  )

x_limits <- c(-125, -67) # Set the desired longitude range
```

```

y_limits <- c(25, 50)      # Set the desired latitude range

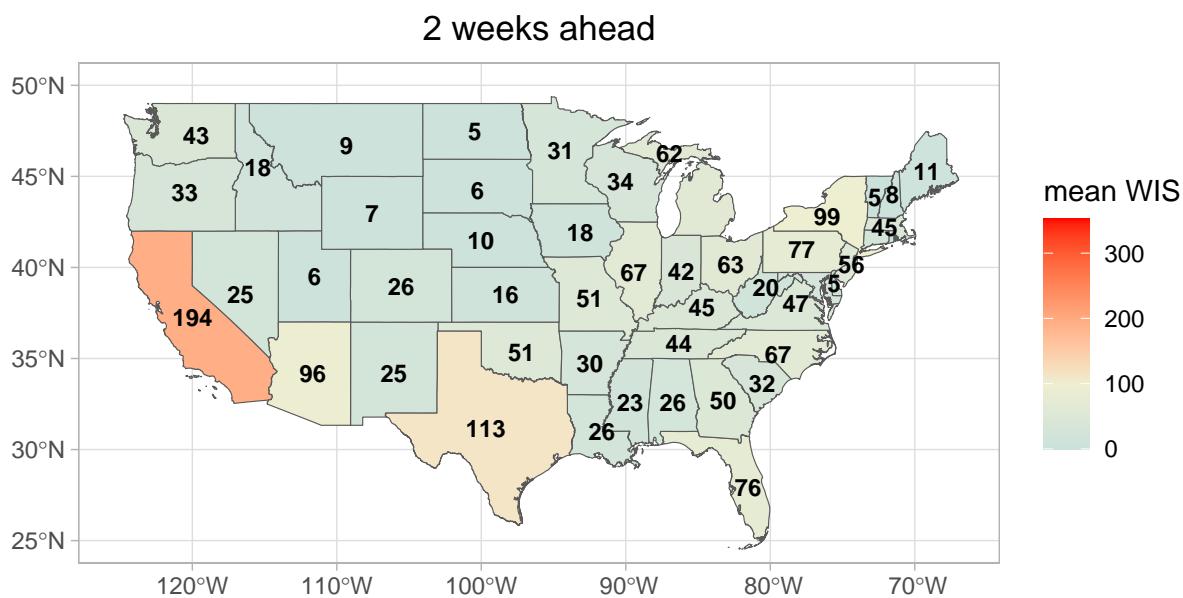
ES_2WEEK <-ES_2WEEK + coord_sf(xlim = x_limits, ylim = y_limits)
ES_2WEEK

```

```

## Warning in st_point_on_surface.sfc(sf::st_zm(x)): st_point_on_surface may not
## give correct results for longitude/latitude data

```



```
#ggsave(ES_2WEEK, height = 4.5, width = 6.5, filename="Fig4B.jpg")
```

3 weeks ahead

```

ES_3WEEK <- ggplot(W3_map) +
  geom_sf(aes(fill = AUTO_AVG_LB)) + # Fill based on the best model
  scale_fill_gradient2(low = "skyblue", mid = "lightyellow2", high = "red", midpoint = 100, limits = c(
    0, 300))
  ggtitle("3 weeks ahead") +
  theme_light() +
  theme(legend.position = "right",
        plot.title = element_text(hjust = 0.5)) +
  geom_sf_text(data = W3_map, aes(label = round(AUTO_AVG_LB,0)), # Round to whole numbers
               size = 3,
               color = "black",
               check_overlap = TRUE, fontface = "bold") +
  labs(

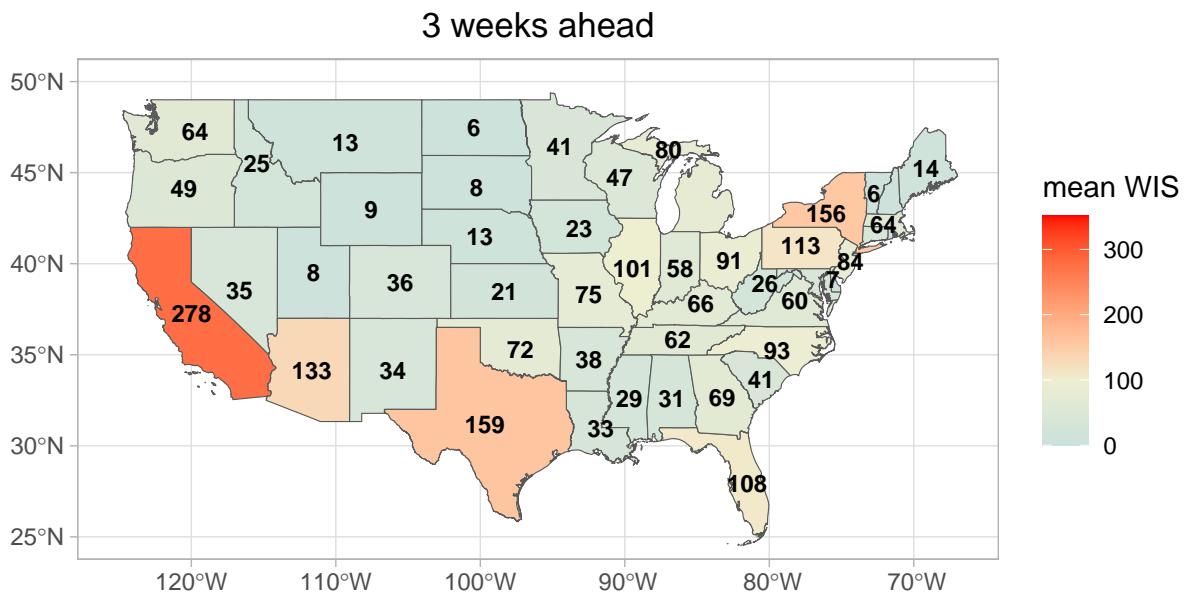
```

```
    fill = "mean WIS",
    x = "",
    y = ""
)

x_limits <- c(-125, -67) # Set the desired longitude range
y_limits <- c(25, 50)     # Set the desired latitude range

ES_3WEEK<- ES_3WEEK + coord_sf(xlim = x_limits, ylim = y_limits)
ES_3WEEK
```

```
## Warning in st_point_on_surface.sfc(sf::st_zm(x)): st_point_on_surface may not  
## give correct results for longitude/latitude data
```



```
#ggsave(ES_3WEEK, height = 4.5, width = 6.5, filename="Fig4C.jpg")
```

4 weeks ahead

```
ES_4WEEK <- ggplot(W4_map) +
  geom_sf(aes(fill = AUTO_AVG_LB)) +
  scale_fill_gradient2(low = "skyblue", mid = "lightyellow2", high = "red",
                       midpoint = 100, limits = c(0, 352)) +
  ggtitle("4 weeks ahead") +
  theme_light()
```

```

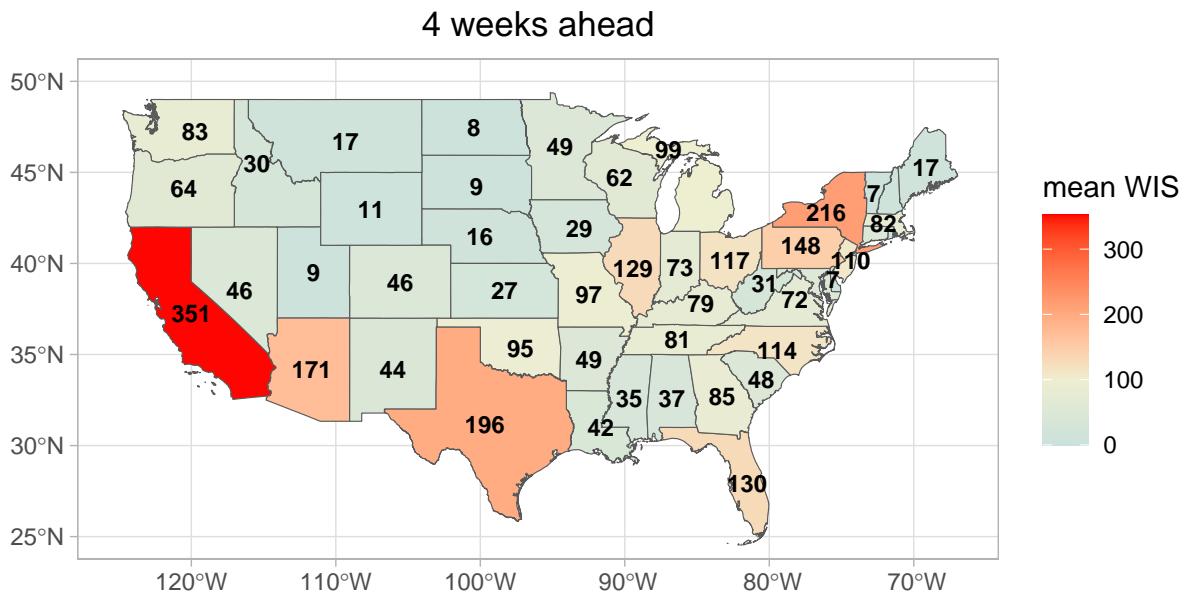
theme(legend.position = "right",
      plot.title = element_text(hjust = 0.5)) +
geom_sf_text(data = W4_map, aes(label = round(AUTO_AVG_LB,0)),
             size = 3,
             color = "black",
             check_overlap = TRUE, fontface = "bold") +
labs(
  fill = "mean WIS",
  x = "",
  y = ""
)

x_limits <- c(-125, -67)
y_limits <- c(25, 50)

ES_4WEEK<-ES_4WEEK + coord_sf(xlim = x_limits, ylim = y_limits)
ES_4WEEK

```

## Warning in st\_point\_on\_surface.sfc(sf::st\_zm(x)): st\_point\_on\_surface may not  
## give correct results for longitude/latitude data



```
#ggsave(ES_4WEEK, height = 4.5, width = 6.5, filename="Fig4D.jpg")
```

Combining maps in a 2x2 grid

```

# Combine in a 2x2 grid
combined_plot <- (ES_1WEEK | ES_2WEEK) /
  (ES_3WEEK | ES_4WEEK) +
  plot_annotation(
    title = "",
    subtitle = "",
    theme = theme(plot.title = element_text(size = 16, face = "bold"),
                  plot.subtitle = element_text(size = 14)))
)
# Print
print(combined_plot)

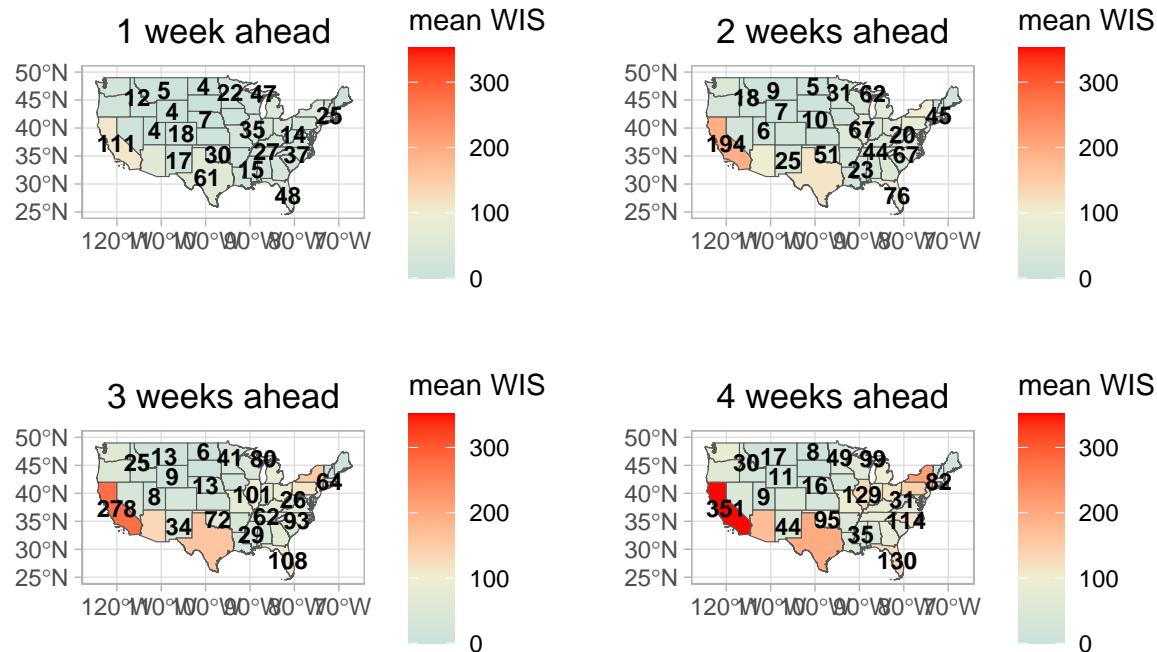
## Warning in st_point_on_surface.sfc(sf::st_zm(x)): st_point_on_surface may not
## give correct results for longitude/latitude data

## Warning in st_point_on_surface.sfc(sf::st_zm(x)): st_point_on_surface may not
## give correct results for longitude/latitude data

## Warning in st_point_on_surface.sfc(sf::st_zm(x)): st_point_on_surface may not
## give correct results for longitude/latitude data

## Warning in st_point_on_surface.sfc(sf::st_zm(x)): st_point_on_surface may not
## give correct results for longitude/latitude data

```



```

# Save to file
ggsave("Fig7.jpg", combined_plot, width =12, height =7)

## Warning in st_point_on_surface.sfc(sf::st_zm(x)): st_point_on_surface may not
## give correct results for longitude/latitude data

## Warning in st_point_on_surface.sfc(sf::st_zm(x)): st_point_on_surface may not
## give correct results for longitude/latitude data

## Warning in st_point_on_surface.sfc(sf::st_zm(x)): st_point_on_surface may not
## give correct results for longitude/latitude data

## Warning in st_point_on_surface.sfc(sf::st_zm(x)): st_point_on_surface may not
## give correct results for longitude/latitude data

```

Loading datasets for regression models analysis

```

pop_data <- read.csv("models_with_logback/regression_features/population_data.csv") # resident population
sovi_data <- read.csv("models_with_logback/regression_features/sovi_2010.2014.csv") # SOVI index
bric_data<-read.csv("models_with_logback/regression_features/bric2015.csv") # BRIC index
humidity_data<-read.csv("models_with_logback/regression_features/humidity_climatology_1990_2020.csv") #
temperature_data<-read.csv("models_with_logback/regression_features/temperature_climatology_1990_2020.csv")

```

## REGRESSION MODELS

Now we run regressions models for evaluating if the best model mean WIS in each state is related to given independent variables.

```

# List of data frames
WIS_dataframes <- list(W1 = W1_map, W2 = W2_map, W3 = W3_map, W4=W4_map)
regression_models<-data.frame()

```

AUTO\_ADJ\_LB WIS x RESIDENT POPULATION 2020

```

for(i in c(1,2,3,4)){

  # Combining data for the same states
  WIS_pop_data <- inner_join(WIS_dataframes[[i]], pop_data, by = "STATE")
  # Fitting the regression model
  model <- lm((WIS_pop_data$AUTO_AVG_LB) ~ (WIS_pop_data$Resident_population_2020))
  # View the model summary
  model_summary <- summary(model)
  # Getting the R and p values for the plot
  r_squared <- round(model_summary$r.squared, 3)
  p_value <- signif(model_summary$coefficients[2, 4], 3)
  # Saving the results in a dataframe
  regression_models2 <- data.frame(
    independent_variable = "Resident Population (2020)",
    Week_Ahead = i,
    r_squared = r_squared,
    p_value = p_value
  )
}

```

```

# Append to the main results dataframe
regression_models <- rbind(regression_models, regression_models2)
}

regression_models

##           independent_variable Week_Ahead r_squared p_value
## 1 Resident Population (2020)      1     0.853 9.18e-21
## 2 Resident Population (2020)      2     0.875 2.24e-22
## 3 Resident Population (2020)      3     0.874 2.50e-22
## 4 Resident Population (2020)      4     0.858 4.32e-21

best regression plot

plot_list <- list() # store plots

for(i in 1:4) {

  # Combine WIS data and population
  WIS_pop_data <- inner_join(WIS_dataframes[[i]], pop_data, by = "STATE")

  # Fit linear model
  model <- lm(AUTO_AVG_LB ~ Resident_population_2020, data = WIS_pop_data)
  model_summary <- summary(model)

  # Save R2 and p-values
  r_squared <- round(model_summary$r.squared, 3)
  p_value <- signif(model_summary$coefficients[2, 4], 3)

  # Create plot
  p <- ggplot(WIS_pop_data, aes(x = Resident_population_2020, y = AUTO_AVG_LB)) +
    geom_point(color = "steelblue", size = 2) +
    geom_smooth(method = "lm", se = FALSE, color = "darkred", linetype = "dashed") +
    labs(
      title = paste("Regression", "Target Week", i),
      subtitle = paste("R2 =", r_squared, "| p =", p_value),
      x = "Resident Population (2020)",
      y = "AUTO_AVG"
    ) +
    theme_minimal()

  # Store the plot
  plot_list[[i]] <- p
}

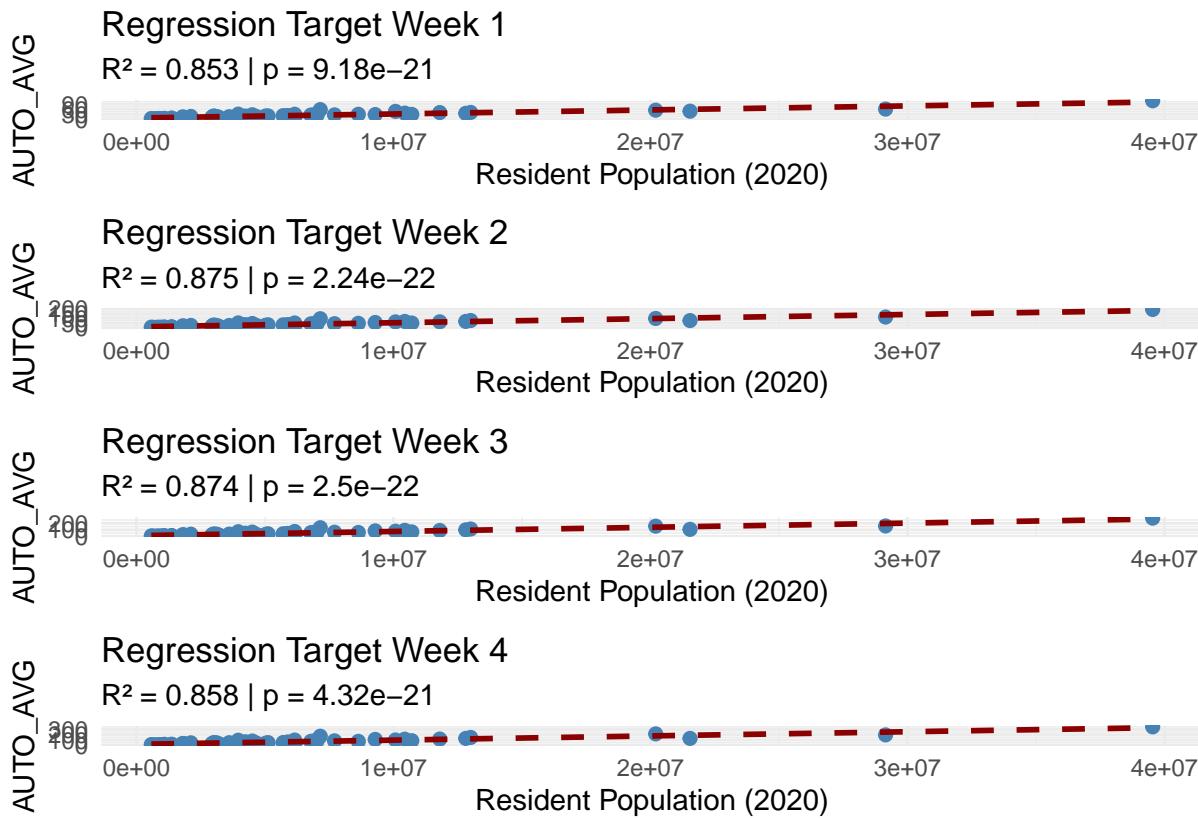
# Combine the plots vertically
combined_regressions <- wrap_plots(plot_list, ncol = 1)

# Display (optional)
combined_regressions

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

```

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



```
# Save the combined plot
#ggsave("Figx.png", combined_plot, width = 3, height = 5, dpi = 600)
```

AUTO\_ADJ WIS x POPULATION DENSITY

```
for(i in c(1,2,3,4)){

  WIS_pop_data <- inner_join(WIS_dataframes[[i]], pop_data, by = "STATE")
  # Fit the regression model
  model <- lm((WIS_pop_data$AUTO_AVG_LB) ~ (WIS_pop_data$Population_density_2020))
  # View the model summary
  model_summary <- summary(model)
  # Extract R-squared and p-value
  r_squared <- round(model_summary$r.squared, 3)
  p_value <- signif(model_summary$coefficients[2, 4], 3)
  # saving the results in a dataframe
  regression_models2<-data.frame("independent_variable"="Population Density (2020)","Week_Ahead"=i, "r_s
  # Append to the main results dataframe
  regression_models<-rbind(regression_models,regression_models2)
}
```

Here we will weight the Social Vulnerability Index (SoVI) and Baseline Resilience Indicators for Communities (BRIC) county indexes which for each states based on the population size in each county.

```
#####
# SOVI BY STATE
# weighted by population size in each county
sovi_by_state <- sovi_data %>%
  filter(!is.nan(sovi)) %>% # Exclude rows where 'sovi' is NaN
  group_by(state.name) %>%
  summarize(weighted_mean = weighted.mean(sovi, w = population.2020, na.rm = TRUE))

colnames(sovi_by_state)[1] <- "STATE"

#####
# BRIC BY STATE
# weighted by population size in each county

bric_by_state <- bric_data %>%
  group_by(state.name) %>%
  summarize(across(16:22, ~ weighted.mean(.x, w = population.2020, na.rm = TRUE), .names = "weighted_mean"))
colnames(bric_by_state)[1] <- "STATE"
```

AUTO\_AVG WIS x SOVI

```
for(i in c(1,2,3,4)){

  WIS_sovi<-NULL
  WIS_sovi <- inner_join(WIS_dataframes[[i]], sovi_by_state, by = "STATE")
  # Fit the regression model
  model <- lm((WIS_sovi$AUTO_AVG_LB) ~ (WIS_sovi$weighted_mean))
  # View the model summary
  model_summary <- summary(model)
  # Extract R-squared and p-value
  r_squared <- round(model_summary$r.squared, 3)
  p_value <- signif(model_summary$coefficients[2, 4], 3)
  # saving the results in a dataframe
  regression_models2<-data.frame("independent_variable"="Social Vulnerability Index (SoVI)","Week_Ahead"
  # appending the results
  regression_models<-rbind(regression_models,regression_models2)
}
```

AUTO\_AVG WIS x BRIC SOCIAL

```
for(i in 1:4){
  WIS_bric <- inner_join(WIS_dataframes[[i]], bric_by_state, by = "STATE")
  # Fit the regression model
  model <- lm((WIS_bric$AUTO_AVG_LB) ~ (WIS_bric$weighted_mean_z_bric.social))
  # View the model summary
  model_summary <- summary(model)
  # Extract R-squared and p-value
  r_squared <- round(model_summary$r.squared, 3)
  p_value <- signif(model_summary$coefficients[2, 4], 3)
  # saving the results in a dataframe
```

```

regression_models2<-data.frame("independent_variable"="BRIC Social (2015)", "Week_Ahead"=i, "r_squared"
# appending the results
regression_models<-rbind(regression_models, regression_models2)
}

```

AUTO\_AVG WIS x BRIC ECONOMIC

```

for(i in 1:4){
  WIS_bric <- inner_join(WIS_dataframes[[i]], bric_by_state, by = "STATE")
  # Fit the regression model
  model <- lm((WIS_bric$AUTO_AVG_LB) ~ (WIS_bric$weighted_mean_z_bric.economic))
  # View the model summary
  model_summary <- summary(model)
  # Extract R-squared and p-value
  r_squared <- round(model_summary$r.squared, 3)
  p_value <- signif(model_summary$coefficients[2, 4], 3)
  # saving the results in a dataframe
  regression_models2<-data.frame("independent_variable"="BRIC Economic (2015)", "Week_Ahead"=i, "r_squared"
    regression_models<-rbind(regression_models, regression_models2)
}

```

AUTO\_AVG WIS x BRIC Infrastructure

```

for (i in 1:4){
  WIS_bric <- inner_join(WIS_dataframes[[i]], bric_by_state, by = "STATE")
  # Fit the regression model
  model <- lm((WIS_bric$AUTO_AVG_LB) ~ (WIS_bric$weighted_mean_z_bric.infrastructure))
  # View the model summary
  model_summary <- summary(model)
  # Extract R-squared and p-value
  r_squared <- round(model_summary$r.squared, 3)
  p_value <- signif(model_summary$coefficients[2, 4], 3)
  # saving the results in a dataframe
  regression_models2<-data.frame("independent_variable"="BRIC Infrastructure (2015)", "Week_Ahead"=i, "r_squared"
    # appending results
    regression_models<-rbind(regression_models, regression_models2)
}

```

AUTO\_AVG WIS x BRIC institutional

```

for (i in 1:4){
  WIS_bric <- inner_join(WIS_dataframes[[i]], bric_by_state, by = "STATE")
  # Fit the regression model
  model <- lm((WIS_bric$AUTO_AVG_LB) ~ (WIS_bric$weighted_mean_z_bric.institutional))
  # View the model summary
  model_summary <- summary(model)
  # Extract R-squared and p-value
  r_squared <- round(model_summary$r.squared, 3)
  p_value <- signif(model_summary$coefficients[2, 4], 3)
  # saving the results in a dataframe
  regression_models2<-data.frame("independent_variable"="BRIC Institutional (2015)", "Week_Ahead"=i, "r_squared"
    # appending results
    regression_models<-rbind(regression_models, regression_models2)
}

```

AUTO\_AVG WIS x BRIC community

```
for(i in 1:4){  
  WIS_bric <- inner_join(WIS_dataframes[[1]], bric_by_state, by = "STATE")  
  # Fit the regression model  
  model <- lm((WIS_bric$AUTO_AVG_LB) ~ (WIS_bric$weighted_mean_z_bric.community))  
  # View the model summary  
  model_summary <- summary(model)  
  # Extract R-squared and p-value  
  r_squared <- round(model_summary$r.squared, 3)  
  p_value <- signif(model_summary$coefficients[2, 4], 3)  
  # saving the results in a dataframe  
  regression_models2<-data.frame("independent_variable"="BRIC Community (2015)","Week_Ahead"=i, "r_squared"  
  regression_models<-rbind(regression_models,regression_models2)  
}
```

AUTO\_AVG WIS x BRIC environment

```
for(i in 1:4){  
  WIS_bric <- inner_join(WIS_dataframes[[i]], bric_by_state, by = "STATE")  
  # Fit the regression model  
  model <- lm((WIS_bric$AUTO_AVG_LB) ~ (WIS_bric$weighted_mean_z_bric.environment))  
  # View the model summary  
  model_summary <- summary(model)  
  # Extract R-squared and p-value  
  r_squared <- round(model_summary$r.squared, 3)  
  p_value <- signif(model_summary$coefficients[2, 4], 3)  
  # saving the results in a dataframe  
  regression_models2<-data.frame("independent_variable"="BRIC Environment (2015)","Week_Ahead"=i, "r_squared"  
  regression_models<-rbind(regression_models,regression_models2)  
}
```

AUTO\_AVG WIS x BRIC total

```
for(i in 1:4){  
  # BRIC INDEX  
  WIS_bric <- inner_join(WIS_dataframes[[i]], bric_by_state, by = "STATE")  
  # Fit the regression model  
  model <- lm((WIS_bric$AUTO_AVG_LB) ~ (WIS_bric$weighted_mean_z_bric.total))  
  # View the model summary  
  model_summary <- summary(model)  
  # Extract R-squared and p-value  
  r_squared <- round(model_summary$r.squared, 3)  
  p_value <- signif(model_summary$coefficients[2, 4], 3)  
  # saving the results in a dataframe  
  regression_models2<-data.frame("independent_variable"="BRIC total (2015)","Week_Ahead"=i, "r_squared"  
  regression_models<-rbind(regression_models,regression_models2)  
}
```

TEMPERATURE - ERA5

Now we will look at regression models that uses mean temperature and specific humidity.

AUTO\_AVG WIS x Temperature ERA5 Data

```
for(i in 1:4){
  # TEMPERATURE DATA from ERA5
  WIS_temp <- inner_join(WIS_dataframes[[i]], temperature_data, by = "STATE")
  # Fit the regression model
  model <- lm((WIS_temp$AUTO_AVG_LB) ~ (WIS_temp$mean))
  # View the model summary
  model_summary <- summary(model)
  # Extract R-squared and p-value
  r_squared <- round(model_summary$r.squared, 3)
  p_value <- signif(model_summary$coefficients[2, 4], 3)
  # saving the results in a dataframe
  regression_models2<-data.frame("independent_variable"="Mean Temperature (1990-2020)","Week_Ahead"=i,
  # appending results
  regression_models<-rbind(regression_models,regression_models2)
}
```

AUTO\_AVG WIS x Specific Humidity ERA5 Data

```
# regression results
print(regression_models)
```

	independent_variable	Week_Ahead	r_squared	p_value
## 1	Resident Population (2020)	1	0.853	9.18e-21
## 2	Resident Population (2020)	2	0.875	2.24e-22
## 3	Resident Population (2020)	3	0.874	2.50e-22
## 4	Resident Population (2020)	4	0.858	4.32e-21
## 5	Population Density (2020)	1	0.010	5.06e-01
## 6	Population Density (2020)	2	0.017	3.75e-01
## 7	Population Density (2020)	3	0.022	3.10e-01
## 8	Population Density (2020)	4	0.025	2.85e-01
## 9	Social Vulnerability Index (SoVI)	1	0.116	1.78e-02
## 10	Social Vulnerability Index (SoVI)	2	0.113	1.94e-02
## 11	Social Vulnerability Index (SoVI)	3	0.120	1.61e-02
## 12	Social Vulnerability Index (SoVI)	4	0.125	1.39e-02
## 13	BRIC Social (2015)	1	0.234	4.96e-04
## 14	BRIC Social (2015)	2	0.224	6.84e-04
## 15	BRIC Social (2015)	3	0.215	9.01e-04
## 16	BRIC Social (2015)	4	0.207	1.16e-03
## 17	BRIC Economic (2015)	1	0.046	1.43e-01
## 18	BRIC Economic (2015)	2	0.047	1.40e-01
## 19	BRIC Economic (2015)	3	0.047	1.37e-01
## 20	BRIC Economic (2015)	4	0.048	1.35e-01
## 21	BRIC Infrastructure (2015)	1	0.008	5.55e-01
## 22	BRIC Infrastructure (2015)	2	0.015	4.15e-01
## 23	BRIC Infrastructure (2015)	3	0.025	2.86e-01
## 24	BRIC Infrastructure (2015)	4	0.033	2.20e-01
## 25	BRIC Institutional (2015)	1	0.066	7.70e-02
## 26	BRIC Institutional (2015)	2	0.067	7.60e-02
## 27	BRIC Institutional (2015)	3	0.067	7.59e-02
## 28	BRIC Institutional (2015)	4	0.067	7.65e-02
## 29	BRIC Community (2015)	1	0.108	2.28e-02

```

## 30 BRIC Community (2015) 2 0.108 2.28e-02
## 31 BRIC Community (2015) 3 0.108 2.28e-02
## 32 BRIC Community (2015) 4 0.108 2.28e-02
## 33 BRIC Environment (2015) 1 0.048 1.36e-01
## 34 BRIC Environment (2015) 2 0.053 1.16e-01
## 35 BRIC Environment (2015) 3 0.054 1.14e-01
## 36 BRIC Environment (2015) 4 0.057 1.01e-01
## 37 BRIC total (2015) 1 0.135 1.01e-02
## 38 BRIC total (2015) 2 0.135 1.02e-02
## 39 BRIC total (2015) 3 0.128 1.24e-02
## 40 BRIC total (2015) 4 0.124 1.42e-02
## 41 Mean Temperature (1990-2020) 1 0.069 7.19e-02
## 42 Mean Temperature (1990-2020) 2 0.067 7.57e-02
## 43 Mean Temperature (1990-2020) 3 0.056 1.04e-01
## 44 Mean Temperature (1990-2020) 4 0.050 1.28e-01

for(i in 1:4){
  # HUMIDITY DATA from ERA5
  WIS_humidity <- inner_join(WIS_dataframes[[i]], humidity_data, by = "STATE")
  # Fit the regression model
  model <- lm((WIS_humidity$AUTO_AVG_LB) ~ (WIS_humidity$mean))
  # View the model summary
  model_summary <- summary(model)
  # Extract R-squared and p-value
  r_squared <- round(model_summary$r.squared, 3)
  p_value <- signif(model_summary$coefficients[2, 4], 3)
  # saving the results in a dataframe
  regression_models2<-data.frame("independent_variable"="Mean Specific Humidity (1990-2020)","Week_Ahead"
  # appending results
  regression_models<-rbind(regression_models,regression_models2)
}

# Plot the table
regression_models

## independent_variable Week_Ahead r_squared p_value
## 1 Resident Population (2020) 1 0.853 9.18e-21
## 2 Resident Population (2020) 2 0.875 2.24e-22
## 3 Resident Population (2020) 3 0.874 2.50e-22
## 4 Resident Population (2020) 4 0.858 4.32e-21
## 5 Population Density (2020) 1 0.010 5.06e-01
## 6 Population Density (2020) 2 0.017 3.75e-01
## 7 Population Density (2020) 3 0.022 3.10e-01
## 8 Population Density (2020) 4 0.025 2.85e-01
## 9 Social Vulnerability Index (SoVI) 1 0.116 1.78e-02
## 10 Social Vulnerability Index (SoVI) 2 0.113 1.94e-02
## 11 Social Vulnerability Index (SoVI) 3 0.120 1.61e-02
## 12 Social Vulnerability Index (SoVI) 4 0.125 1.39e-02
## 13 BRIC Social (2015) 1 0.234 4.96e-04
## 14 BRIC Social (2015) 2 0.224 6.84e-04
## 15 BRIC Social (2015) 3 0.215 9.01e-04
## 16 BRIC Social (2015) 4 0.207 1.16e-03
## 17 BRIC Economic (2015) 1 0.046 1.43e-01
## 18 BRIC Economic (2015) 2 0.047 1.40e-01

```

## 19	BRIC Economic (2015)	3	0.047	1.37e-01
## 20	BRIC Economic (2015)	4	0.048	1.35e-01
## 21	BRIC Infrastructure (2015)	1	0.008	5.55e-01
## 22	BRIC Infrastructure (2015)	2	0.015	4.15e-01
## 23	BRIC Infrastructure (2015)	3	0.025	2.86e-01
## 24	BRIC Infrastructure (2015)	4	0.033	2.20e-01
## 25	BRIC Institutional (2015)	1	0.066	7.70e-02
## 26	BRIC Institutional (2015)	2	0.067	7.60e-02
## 27	BRIC Institutional (2015)	3	0.067	7.59e-02
## 28	BRIC Institutional (2015)	4	0.067	7.65e-02
## 29	BRIC Community (2015)	1	0.108	2.28e-02
## 30	BRIC Community (2015)	2	0.108	2.28e-02
## 31	BRIC Community (2015)	3	0.108	2.28e-02
## 32	BRIC Community (2015)	4	0.108	2.28e-02
## 33	BRIC Environment (2015)	1	0.048	1.36e-01
## 34	BRIC Environment (2015)	2	0.053	1.16e-01
## 35	BRIC Environment (2015)	3	0.054	1.14e-01
## 36	BRIC Environment (2015)	4	0.057	1.01e-01
## 37	BRIC total (2015)	1	0.135	1.01e-02
## 38	BRIC total (2015)	2	0.135	1.02e-02
## 39	BRIC total (2015)	3	0.128	1.24e-02
## 40	BRIC total (2015)	4	0.124	1.42e-02
## 41	Mean Temperature (1990–2020)	1	0.069	7.19e-02
## 42	Mean Temperature (1990–2020)	2	0.067	7.57e-02
## 43	Mean Temperature (1990–2020)	3	0.056	1.04e-01
## 44	Mean Temperature (1990–2020)	4	0.050	1.28e-01
## 45	Mean Specific Humidity (1990–2020)	1	0.037	1.91e-01
## 46	Mean Specific Humidity (1990–2020)	2	0.034	2.07e-01
## 47	Mean Specific Humidity (1990–2020)	3	0.027	2.62e-01
## 48	Mean Specific Humidity (1990–2020)	4	0.021	3.24e-01

## SUMMARY OF RESULTS

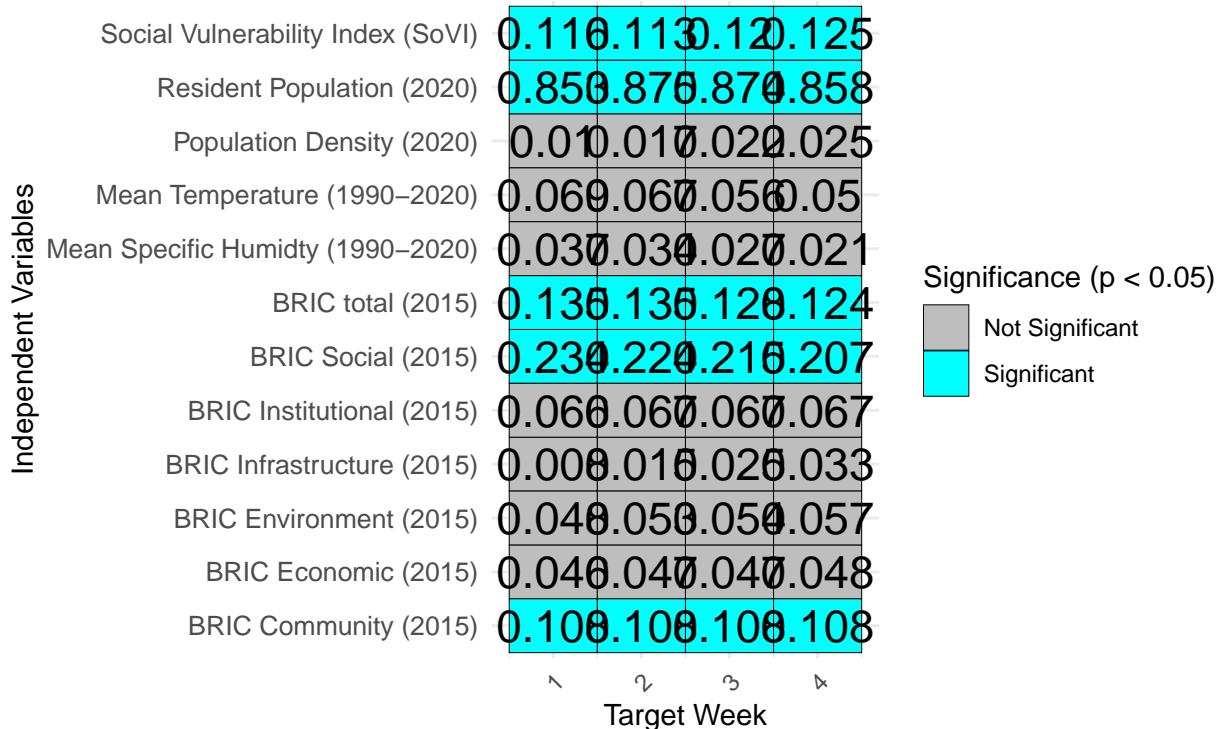
```
# Define a significance threshold (p < 0.05)
regression_models <- regression_models %>%
  mutate(significance = ifelse(p_value < 0.05, "Significant", "Not Significant"))

# Plot
reg2<-ggplot(regression_models, aes(x = Week_Ahead, y = independent_variable, fill = significance)) +
  geom_tile(color = "black") + # Add black borders to squares
  geom_text(aes(label = round(r_squared, 3)), color = "black", size = 6) + # Text inside boxes
  scale_fill_manual(values = c("Significant" = "cyan", "Not Significant" = "gray")) +
  labs(title = "Regression models R2",
       subtitle = "Mean WIS as dependent variable",
       x = "Target Week",
       y = "Independent Variables",
       fill = "Significance (p < 0.05)") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1),
        axis.text.y = element_text(size = 10),
        plot.title = element_text(size = 16, face = "bold"),
        plot.subtitle = element_text(size = 14))

reg2
```

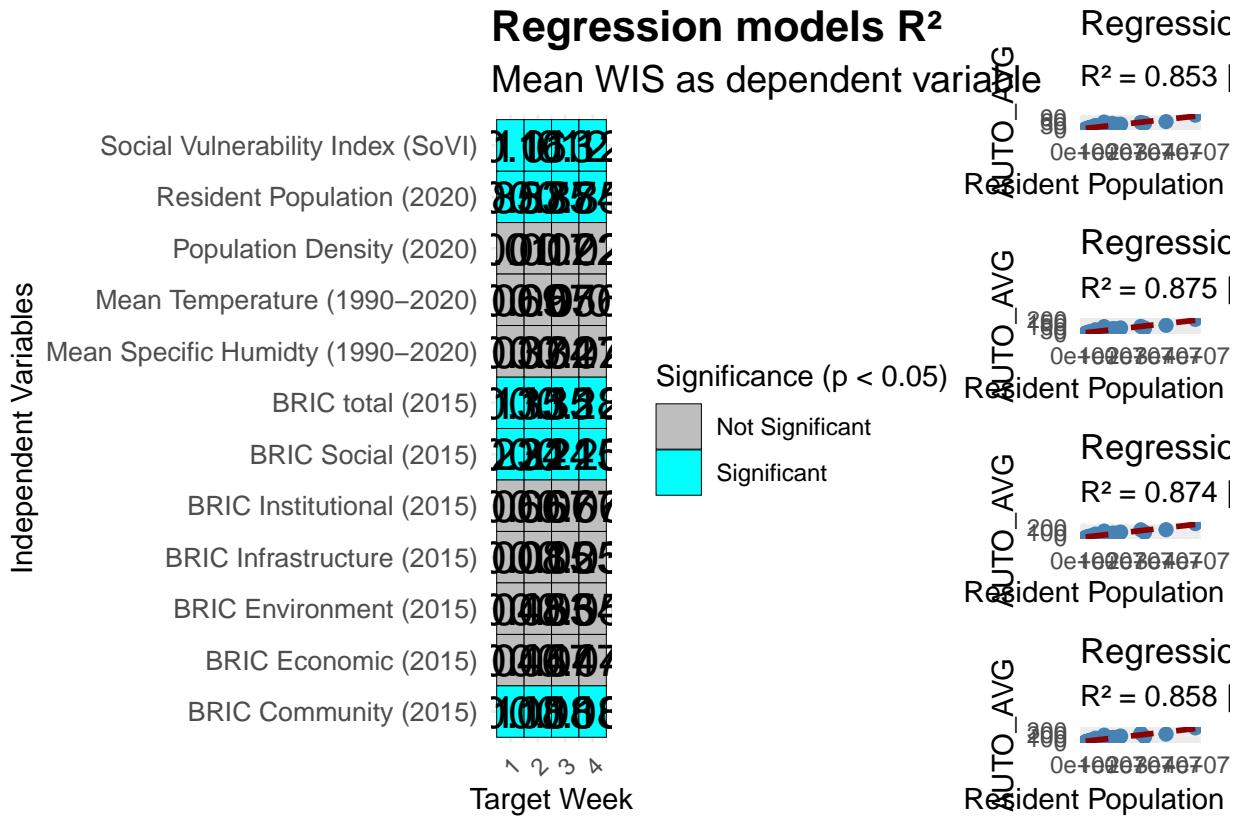
## Regression models R<sup>2</sup>

### Mean WIS as dependent variable



```
# Combine the plots vertically
reg2_scatter <- reg2 | combined_regressions
reg2_scatter
```

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



```
ggsave(reg2_scatter, height = 6, width = 12, filename="Fig8.jpg")
```

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

```
#ggsave(reg2, filename="reg2.jpg",height = 8, width = 10)
```

Let's plot some maps with the mean difference (%) between the AUTO\_AVG\_LB and the AUTO ARIMA models for the same state.

1 week ahead percentage of improvement

```
ES_1WEEK <- ggplot(W1_map) +
  geom_sf(aes(fill = percentage_improvement)) +
  scale_fill_gradient2(low = "skyblue", mid = "lightyellow", high = "darkred", midpoint = 0, limits = c
    ggttitle("1 weeks ahead") +
    theme_light() +
    theme(legend.position = "right",
      plot.title = element_text(hjust = 0.5)) +
    geom_sf_text(data = W1_map, aes(label = round(percentage_improvement,0)),
      size = 3,
      color = "black",
      check_overlap = TRUE, fontface = "bold") +
```

```

  labs(
    fill = "Mean difference",
    x = "",
    y = ""
  )

x_limits <- c(-125, -67) # Set the desired longitude range
y_limits <- c(25, 50)     # Set the desired latitude range

ES_1WEEK<-ES_1WEEK + coord_sf(xlim = x_limits, ylim = y_limits)

```

2 weeks ahead

```

ES_2WEEK <- ggplot(W2_map) +
  geom_sf(aes(fill = percentage_improvement)) +
  scale_fill_gradient2(low = "skyblue" , mid = "lightyellow", high = "darkred", midpoint = 0, limits = c
  ggttitle("2 weeks ahead") +
  theme_light() +
  theme(legend.position = "right",
        plot.title = element_text(hjust = 0.5)) +
  geom_sf_text(data = W2_map, aes(label = round(percentage_improvement,0)),
               size = 3,
               color = "black",
               check_overlap = TRUE, fontface = "bold") +
  labs(
    fill = "Mean difference",
    x = "",
    y = ""
  ) # Adding a subtitle

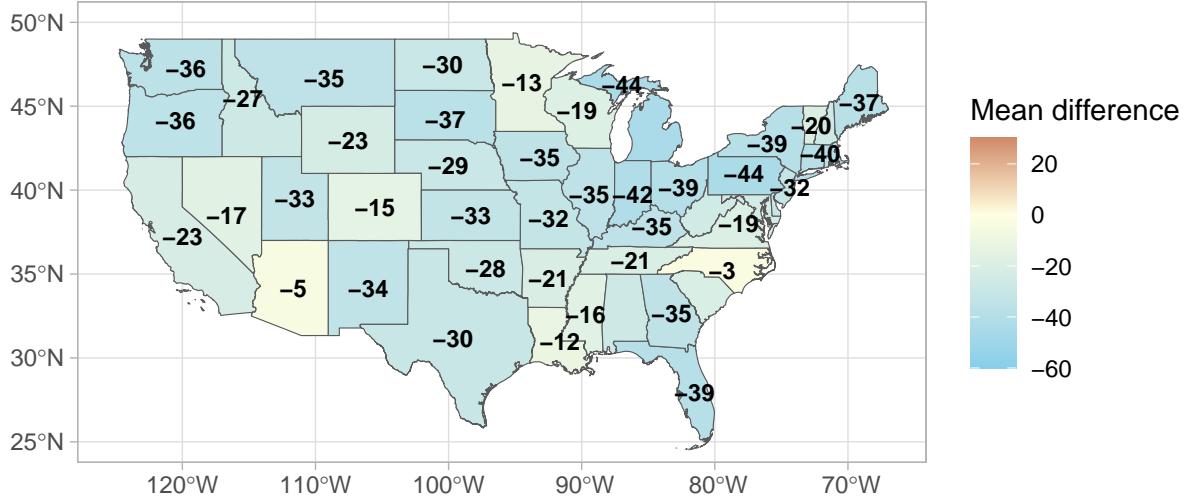
x_limits <- c(-125, -67) # Set the desired longitude range
y_limits <- c(25, 50)     # Set the desired latitude range

ES_2WEEK + coord_sf(xlim = x_limits, ylim = y_limits)

## Warning in st_point_on_surface.sfc(sf::st_zm(x)): st_point_on_surface may not
## give correct results for longitude/latitude data

```

## 2 weeks ahead



## 3 weeks ahead

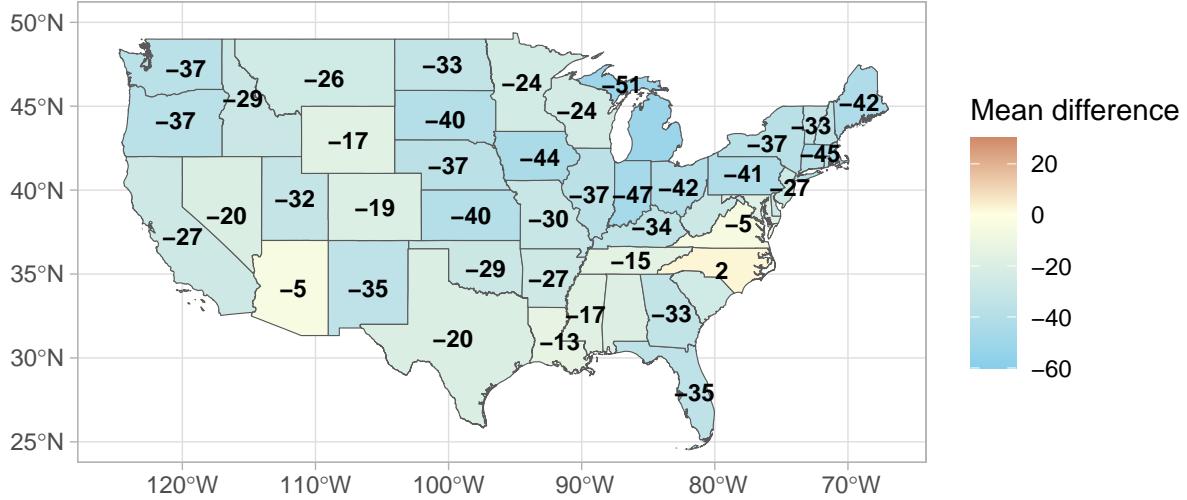
```
ES_3WEEK <- ggplot(W3_map) +
  geom_sf(aes(fill = percentage_improvement)) + # Fill based on the best model
  scale_fill_gradient2(low = "skyblue" , mid = "lightyellow", high = "darkred", midpoint = 0, limits = c(-67, 20))
  ggtitle("3 weeks ahead") +
  theme_light() +
  theme(legend.position = "right",
        plot.title = element_text(hjust = 0.5)) +
  geom_sf_text(data = W3_map, aes(label = round(percentage_improvement,0)),
               size = 3,
               color = "black",
               check_overlap = TRUE, fontface = "bold") + # Display percentage improvement in each state
  labs(
    fill = "Mean difference", # Label for the legend
    x = "",
    y = ""
  ) # Adding a subtitle

x_limits <- c(-125, -67) # Set the desired longitude range
y_limits <- c(25, 50)      # Set the desired latitude range

ES_3WEEK + coord_sf(xlim = x_limits, ylim = y_limits)

## Warning in st_point_on_surface.sfc(sf::st_zm(x)): st_point_on_surface may not
## give correct results for longitude/latitude data
```

### 3 weeks ahead



### 4 weeks ahead

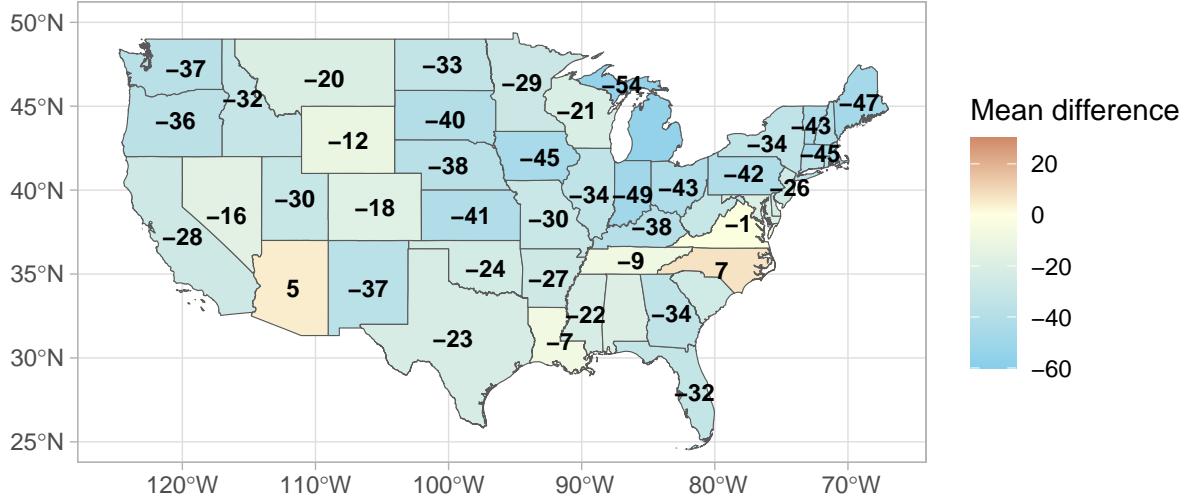
```
ES_4WEEK <- ggplot(W4_map) +
  geom_sf(aes(fill = percentage_improvement)) + # Fill based on the best model
  scale_fill_gradient2(low = "skyblue", mid = "lightyellow", high = "darkred", midpoint = 0, limits = c(-67, 20))
  ggtitle("4 weeks ahead") +
  theme_light() +
  theme(legend.position = "right",
        plot.title = element_text(hjust = 0.5)) +
  geom_sf_text(data = W4_map, aes(label = round(percentage_improvement,0)),
               size = 3,
               color = "black",
               check_overlap = TRUE, fontface = "bold") + # Display percentage improvement in each state
  labs(
    fill = "Mean difference", # Label for the legend
    x = "",
    y = ""
  ) # Adding a subtitle

x_limits <- c(-125, -67) # Set the desired longitude range
y_limits <- c(25, 50)     # Set the desired latitude range

ES_4WEEK + coord_sf(xlim = x_limits, ylim = y_limits)

## Warning in st_point_on_surface.sfc(sf::st_zm(x)): st_point_on_surface may not
## give correct results for longitude/latitude data
```

4 weeks ahead



Combining plot on a 2x2 grid.

```
# Ensure each plot has coord_sf applied
ES_1WEEK <- ES_1WEEK + coord_sf(xlim = x_limits, ylim = y_limits)

## Coordinate system already present. Adding new coordinate system, which will
## replace the existing one.

ES_2WEEK <- ES_2WEEK + coord_sf(xlim = x_limits, ylim = y_limits)
ES_3WEEK <- ES_3WEEK + coord_sf(xlim = x_limits, ylim = y_limits)
ES_4WEEK <- ES_4WEEK + coord_sf(xlim = x_limits, ylim = y_limits)

# Combine in a 2x2 grid
combined_plot <- (ES_1WEEK | ES_2WEEK) /
  (ES_3WEEK | ES_4WEEK) +
  plot_annotation(
    title = "",
    subtitle = "",
    theme = theme(plot.title = element_text(size = 16, face = "bold"),
                  plot.subtitle = element_text(size = 14)))
  )

# Print
print(combined_plot)

## Warning in st_point_on_surface.sfc(sf::st_zm(x)): st_point_on_surface may not
```

```

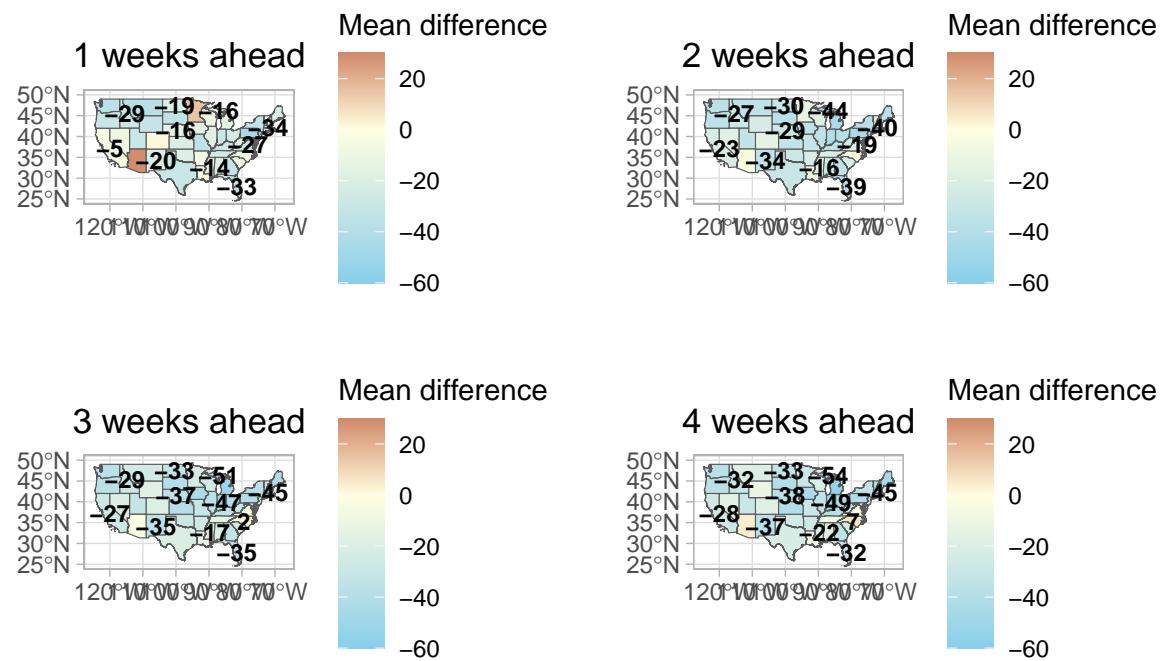
## give correct results for longitude/latitude data

## Warning in st_point_on_surface.sfc(sf::st_zm(x)): st_point_on_surface may not
## give correct results for longitude/latitude data

## Warning in st_point_on_surface.sfc(sf::st_zm(x)): st_point_on_surface may not
## give correct results for longitude/latitude data

## Warning in st_point_on_surface.sfc(sf::st_zm(x)): st_point_on_surface may not
## give correct results for longitude/latitude data

```



```

# Save to file
ggsave("Fig9.jpg", combined_plot, width =12, height =7)

```

```

## Warning in st_point_on_surface.sfc(sf::st_zm(x)): st_point_on_surface may not
## give correct results for longitude/latitude data

## Warning in st_point_on_surface.sfc(sf::st_zm(x)): st_point_on_surface may not
## give correct results for longitude/latitude data

## Warning in st_point_on_surface.sfc(sf::st_zm(x)): st_point_on_surface may not
## give correct results for longitude/latitude data

## Warning in st_point_on_surface.sfc(sf::st_zm(x)): st_point_on_surface may not
## give correct results for longitude/latitude data

```

## REGRESSION MODELS

Now we will run the regression models for evaluating if the best model percentage of improvement in each state is related to given independent variables.

```
# List of data frames
WIS_dataframes <- list(W1 = W1_map, W2 = W2_map, W3 = W3_map, W4=W4_map)
regression_models<-data.frame()
```

Percentage of improvement regression analysis

AUTO\_AVG\_LB WIS x RESIDENT POPULATION 2020

```
for(i in c(1,2,3,4){

  # Combining data for the same states
  WIS_pop_data <- inner_join(WIS_dataframes[[i]], pop_data, by = "STATE")
  # Fitting the regression model
  model <- lm((WIS_pop_data$percentage_improvement) ~ (WIS_pop_data$Resident_population_2020))
  # View the model summary
  model_summary <- summary(model)
  # Getting the R and p values for the plot
  r_squared <- round(model_summary$r.squared, 3)
  p_value <- signif(model_summary$coefficients[2, 4], 3)
  # Saving the results in a dataframe
  regression_models2 <- data.frame(
    independent_variable = "Resident Population (2020)",
    Week_Ahead = i,
    r_squared = r_squared,
    p_value = p_value
  )
  # Append to the main results dataframe
  regression_models <- rbind(regression_models, regression_models2)
}

regression_models
```

	independent_variable	Week_Ahead	r_squared	p_value
## 1	Resident Population (2020)	1	0.000	0.885
## 2	Resident Population (2020)	2	0.006	0.591
## 3	Resident Population (2020)	3	0.002	0.756
## 4	Resident Population (2020)	4	0.004	0.686

AUTO\_AVG\_LB WIS x POPULATION DENSITY

```
for(i in c(1,2,3,4){

  WIS_pop_data <- inner_join(WIS_dataframes[[i]], pop_data, by = "STATE")
  # Fit the regression model
  model <- lm((WIS_pop_data$percentage_improvement) ~ (WIS_pop_data$Population_density_2020))
  # View the model summary
  model_summary <- summary(model)
  # Extract R-squared and p-value
  r_squared <- round(model_summary$r.squared, 3)
```

```

p_value <- signif(model_summary$coefficients[2, 4], 3)
# saving the results in a dataframe
regression_models2<-data.frame("independent_variable"="Population Density (2020)","Week_Ahead"=i, "r_s"
# Append to the main results dataframe
regression_models<-rbind(regression_models,regression_models2)
}

```

Here we will weight the Social Vulnerability Index (SoVI) and Baseline Resilience Indicators for Communities (BRIC) county indexes which for each states based on the population size in each county.

```

#####
# SOVI BY STATE
# weighted by population size in each county
sovi_by_state <- sovi_data %>%
  filter(!is.nan(sovi)) %>% # Exclude rows where 'sovi' is NaN
  group_by(state.name) %>%
  summarize(weighted_mean = weighted.mean(sovi, w = population.2020, na.rm = TRUE))

colnames(sovi_by_state)[1] <- "STATE"

#####
# BRIC BY STATE
# weighted by population size in each county

bric_by_state <- bric_data %>%
  group_by(state.name) %>%
  summarize(across(16:22, ~ weighted.mean(.x, w = population.2020, na.rm = TRUE), .names = "weighted_mean"))
colnames(bric_by_state)[1] <- "STATE"

```

AUTO\_AVG\_LB WIS x SOVI

```

for(i in c(1,2,3,4)){

  WIS_sovi<-NULL
  WIS_sovi <- inner_join(WIS_dataframes[[i]], sovi_by_state, by = "STATE")
  # Fit the regression model
  model <- lm((WIS_sovi$percentage_improvement) ~ (WIS_sovi$weighted_mean))
  # View the model summary
  model_summary <- summary(model)
  # Extract R-squared and p-value
  r_squared <- round(model_summary$r.squared, 3)
  p_value <- signif(model_summary$coefficients[2, 4], 3)
  # saving the results in a dataframe
  regression_models2<-data.frame("independent_variable"="Social Vulnerability Index (SoVI)","Week_Ahead"
  # appending the results
  regression_models<-rbind(regression_models,regression_models2)
}

```

AUTO\_AVG\_LB WIS x BRIC SOCIAL

```

for(i in 1:4){
  WIS_bric <- inner_join(WIS_dataframes[[i]], bric_by_state, by = "STATE")

```

```

# Fit the regression model
model <- lm((WIS_bric$percentage_improvement) ~ (WIS_bric$weighted_mean_z_bric.social))
# View the model summary
model_summary <- summary(model)
# Extract R-squared and p-value
r_squared <- round(model_summary$r.squared, 3)
p_value <- signif(model_summary$coefficients[2, 4], 3)
# saving the results in a dataframe
regression_models2<-data.frame("independent_variable"="BRIC Social (2015)","Week_Ahead"=i, "r_squared"
# appending the results
regression_models<-rbind(regression_models,regression_models2)
}

```

AUTO\_AVG\_LB WIS x BRIC ECONOMIC

```

for(i in 1:4){
  WIS_bric <- inner_join(WIS_dataframes[[i]], bric_by_state, by = "STATE")
  # Fit the regression model
  model <- lm((WIS_bric$percentage_improvement) ~ (WIS_bric$weighted_mean_z_bric.economic))
  # View the model summary
  model_summary <- summary(model)
  # Extract R-squared and p-value
  r_squared <- round(model_summary$r.squared, 3)
  p_value <- signif(model_summary$coefficients[2, 4], 3)
  # saving the results in a dataframe
  regression_models2<-data.frame("independent_variable"="BRIC Economic (2015)","Week_Ahead"=i, "r_squared"
  regression_models<-rbind(regression_models,regression_models2)
}

```

AUTO\_AVG\_LB WIS x BRIC Infrastructure

```

for (i in 1:4){
  WIS_bric <- inner_join(WIS_dataframes[[i]], bric_by_state, by = "STATE")
  # Fit the regression model
  model <- lm((WIS_bric$percentage_improvement) ~ (WIS_bric$weighted_mean_z_bric.infrastructure))
  # View the model summary
  model_summary <- summary(model)
  # Extract R-squared and p-value
  r_squared <- round(model_summary$r.squared, 3)
  p_value <- signif(model_summary$coefficients[2, 4], 3)
  # saving the results in a dataframe
  regression_models2<-data.frame("independent_variable"="BRIC Infrastructure (2015)","Week_Ahead"=i, "r_squared"
  # appending results
  regression_models<-rbind(regression_models,regression_models2)
}

```

Plotting best regression model

```

# Prepare a list of plots
bric_plots <- list()

for (i in 1:4) {
  # Join the WIS and BRIC data

```

```

WIS_bric <- inner_join(
  WIS_dataframes[[i]],
  bric_by_state,
  by = "STATE"
)

# Fit the linear model
model <- lm(
  percentage_improvement ~ weighted_mean_z_bric.infrastructure,
  data = WIS_bric
)
ms <- summary(model)

# Extract R-squared and p-value
r2 <- round(ms$r.squared, 3)
p <- signif(ms$coefficients[2, 4], 3)

# Build the ggplot
p <- ggplot(WIS_bric,
             aes(x = weighted_mean_z_bric.infrastructure,
                  y = percentage_improvement)) +
  geom_point(color = "steelblue", size = 2) +
  geom_smooth(method = "lm", se = FALSE, color = "darkred") +
  labs(
    title      = paste("Regression", "Target Week", i),
    subtitle   = paste0("R2 = ", r2, " p = ", p),
    x          = "BRIC infrastructure (2015)",
    y          = "Difference"
  ) +
  theme_minimal()

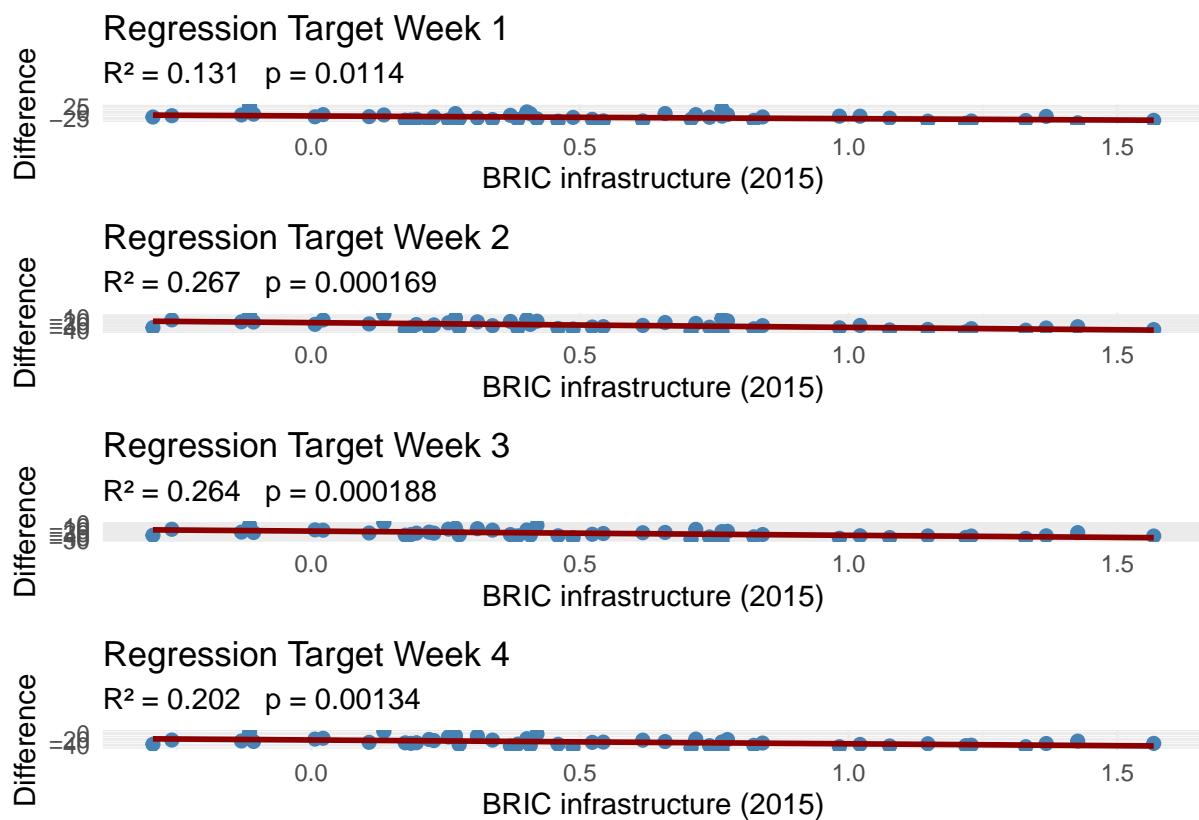
# Store it
bric_plots[[i]] <- p
}

# Combine the plots vertically
combined_regressions <- wrap_plots(bric_plots, ncol = 1)

# Display
combined_regressions

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

```



AUTO\_AVG\_LB WIS x BRIC institutional

```
for (i in 1:4){
  WIS_bric <- inner_join(WIS_dataframes[[i]], bric_by_state, by = "STATE")
  # Fit the regression model
  model <- lm((WIS_bric$percentage_improvement) ~ (WIS_bric$weighted_mean_z_bric.institutional))
  # View the model summary
  model_summary <- summary(model)
  # Extract R-squared and p-value
  r_squared <- round(model_summary$r.squared, 3)
  p_value <- signif(model_summary$coefficients[2, 4], 3)
  # saving the results in a dataframe
  regression_models2<-data.frame("independent_variable"="BRIC Institutional (2015)", "Week_Ahead"=i, "r_squ
  # appending results
  regression_models<-rbind(regression_models, regression_models2)
}
```

AUTO\_AVG\_LB WIS x BRIC community

```
for(i in 1:4){
  WIS_bric <- inner_join(WIS_dataframes[[1]], bric_by_state, by = "STATE")
  # Fit the regression model
  model <- lm((WIS_bric$percentage_improvement) ~ (WIS_bric$weighted_mean_z_bric.community))
  # View the model summary
  model_summary <- summary(model)
  # Extract R-squared and p-value
```

```

r_squared <- round(model_summary$r.squared, 3)
p_value <- signif(model_summary$coefficients[2, 4], 3)
# saving the results in a dataframe
regression_models2<-data.frame("independent_variable"="BRIC Community (2015)","Week_Ahead"=i, "r_squared"=r_squared)
regression_models<-rbind(regression_models,regression_models2)
}

```

AUTO\_AVG\_LB WIS x BRIC environment

```

for(i in 1:4){
  WIS_bric <- inner_join(WIS_dataframes[[i]], bric_by_state, by = "STATE")
  # Fit the regression model
  model <- lm((WIS_bric$percentage_improvement) ~ (WIS_bric$weighted_mean_z_bric.environment))
  # View the model summary
  model_summary <- summary(model)
  # Extract R-squared and p-value
  r_squared <- round(model_summary$r.squared, 3)
  p_value <- signif(model_summary$coefficients[2, 4], 3)
  # saving the results in a dataframe
  regression_models2<-data.frame("independent_variable"="BRIC Environment (2015)","Week_Ahead"=i, "r_squared"=r_squared)
  regression_models<-rbind(regression_models,regression_models2)
}

```

AUTO\_AVG\_LB WIS x BRIC total

```

for(i in 1:4){
  # BRIC INDEX
  WIS_bric <- inner_join(WIS_dataframes[[i]], bric_by_state, by = "STATE")
  # Fit the regression model
  model <- lm((WIS_bric$percentage_improvement) ~ (WIS_bric$weighted_mean_z_bric.total))
  # View the model summary
  model_summary <- summary(model)
  # Extract R-squared and p-value
  r_squared <- round(model_summary$r.squared, 3)
  p_value <- signif(model_summary$coefficients[2, 4], 3)
  # saving the results in a dataframe
  regression_models2<-data.frame("independent_variable"="BRIC total (2015)","Week_Ahead"=i, "r_squared"=r_squared)
  regression_models<-rbind(regression_models,regression_models2)
}

```

TEMPERATURE - ERA5

Now we will look at regression models that uses mean temperature and specific humidity.

AUTO\_AVG\_LB WIS x Temperature ERA5 Data

```

for(i in 1:4){
  # TEMPERATURE DATA from ERA5
  WIS_temp <- inner_join(WIS_dataframes[[i]], temperature_data, by = "STATE")
  # Fit the regression model
}

```

```

model <- lm((WIS_temp$percentage_improvement) ~ (WIS_temp$mean))
# View the model summary
model_summary <- summary(model)
# Extract R-squared and p-value
r_squared <- round(model_summary$r.squared, 3)
p_value <- signif(model_summary$coefficients[2, 4], 3)
# saving the results in a dataframe
regression_models2<-data.frame("independent_variable"="Mean Temperature (1990-2020)","Week_Ahead"=i,
# appending results
regression_models<-rbind(regression_models,regression_models2)
}

```

AUTO\_AVG\_LB WIS x Specific Humidity ERA5 Data

```

# regression results
print(regression_models)

```

	independent_variable	Week_Ahead	r_squared	p_value
## 1	Resident Population (2020)	1	0.000	0.885000
## 2	Resident Population (2020)	2	0.006	0.591000
## 3	Resident Population (2020)	3	0.002	0.756000
## 4	Resident Population (2020)	4	0.004	0.686000
## 5	Population Density (2020)	1	0.142	0.008300
## 6	Population Density (2020)	2	0.076	0.058700
## 7	Population Density (2020)	3	0.018	0.362000
## 8	Population Density (2020)	4	0.011	0.475000
## 9	Social Vulnerability Index (SoVI)	1	0.000	0.969000
## 10	Social Vulnerability Index (SoVI)	2	0.010	0.494000
## 11	Social Vulnerability Index (SoVI)	3	0.006	0.593000
## 12	Social Vulnerability Index (SoVI)	4	0.001	0.861000
## 13	BRIC Social (2015)	1	0.005	0.634000
## 14	BRIC Social (2015)	2	0.024	0.295000
## 15	BRIC Social (2015)	3	0.098	0.030200
## 16	BRIC Social (2015)	4	0.109	0.022000
## 17	BRIC Economic (2015)	1	0.003	0.697000
## 18	BRIC Economic (2015)	2	0.008	0.558000
## 19	BRIC Economic (2015)	3	0.037	0.192000
## 20	BRIC Economic (2015)	4	0.066	0.078600
## 21	BRIC Infrastructure (2015)	1	0.131	0.011400
## 22	BRIC Infrastructure (2015)	2	0.267	0.000169
## 23	BRIC Infrastructure (2015)	3	0.264	0.000188
## 24	BRIC Infrastructure (2015)	4	0.202	0.001340
## 25	BRIC Institutional (2015)	1	0.008	0.546000
## 26	BRIC Institutional (2015)	2	0.002	0.778000
## 27	BRIC Institutional (2015)	3	0.005	0.632000
## 28	BRIC Institutional (2015)	4	0.002	0.744000
## 29	BRIC Community (2015)	1	0.009	0.533000
## 30	BRIC Community (2015)	2	0.009	0.533000
## 31	BRIC Community (2015)	3	0.009	0.533000
## 32	BRIC Community (2015)	4	0.009	0.533000
## 33	BRIC Environment (2015)	1	0.000	0.954000
## 34	BRIC Environment (2015)	2	0.005	0.628000
## 35	BRIC Environment (2015)	3	0.028	0.259000

```

## 36 BRIC Environment (2015) 4 0.045 0.147000
## 37 BRIC total (2015) 1 0.016 0.397000
## 38 BRIC total (2015) 2 0.077 0.056300
## 39 BRIC total (2015) 3 0.176 0.003010
## 40 BRIC total (2015) 4 0.191 0.001870
## 41 Mean Temperature (1990–2020) 1 0.047 0.139000
## 42 Mean Temperature (1990–2020) 2 0.139 0.008930
## 43 Mean Temperature (1990–2020) 3 0.246 0.000335
## 44 Mean Temperature (1990–2020) 4 0.279 0.000114

for(i in 1:4){
  # HUMIDITY DATA from ERA5
  WIS_humidity <- inner_join(WIS_dataframes[[i]], humidity_data, by = "STATE")
  # Fit the regression model
  model <- lm((WIS_humidity$percentage_improvement) ~ (WIS_humidity$mean))
  # View the model summary
  model_summary <- summary(model)
  # Extract R-squared and p-value
  r_squared <- round(model_summary$r.squared, 3)
  p_value <- signif(model_summary$coefficients[2, 4], 3)
  # saving the results in a dataframe
  regression_models2<-data.frame("independent_variable"="Mean Specific Humidity (1990–2020)", "Week_Ahead"
  # appending results
  regression_models<-rbind(regression_models, regression_models2)
}

# Plot the table
regression_models

## independent_variable Week_Ahead r_squared p_value
## 1 Resident Population (2020) 1 0.000 0.885000
## 2 Resident Population (2020) 2 0.006 0.591000
## 3 Resident Population (2020) 3 0.002 0.756000
## 4 Resident Population (2020) 4 0.004 0.686000
## 5 Population Density (2020) 1 0.142 0.008300
## 6 Population Density (2020) 2 0.076 0.058700
## 7 Population Density (2020) 3 0.018 0.362000
## 8 Population Density (2020) 4 0.011 0.475000
## 9 Social Vulnerability Index (SoVI) 1 0.000 0.969000
## 10 Social Vulnerability Index (SoVI) 2 0.010 0.494000
## 11 Social Vulnerability Index (SoVI) 3 0.006 0.593000
## 12 Social Vulnerability Index (SoVI) 4 0.001 0.861000
## 13 BRIC Social (2015) 1 0.005 0.634000
## 14 BRIC Social (2015) 2 0.024 0.295000
## 15 BRIC Social (2015) 3 0.098 0.030200
## 16 BRIC Social (2015) 4 0.109 0.022000
## 17 BRIC Economic (2015) 1 0.003 0.697000
## 18 BRIC Economic (2015) 2 0.008 0.558000
## 19 BRIC Economic (2015) 3 0.037 0.192000
## 20 BRIC Economic (2015) 4 0.066 0.078600
## 21 BRIC Infrastructure (2015) 1 0.131 0.011400
## 22 BRIC Infrastructure (2015) 2 0.267 0.000169
## 23 BRIC Infrastructure (2015) 3 0.264 0.000188
## 24 BRIC Infrastructure (2015) 4 0.202 0.001340

```

```

## 25      BRIC Institutional (2015)      1      0.008 0.546000
## 26      BRIC Institutional (2015)      2      0.002 0.778000
## 27      BRIC Institutional (2015)      3      0.005 0.632000
## 28      BRIC Institutional (2015)      4      0.002 0.744000
## 29      BRIC Community (2015)        1      0.009 0.533000
## 30      BRIC Community (2015)        2      0.009 0.533000
## 31      BRIC Community (2015)        3      0.009 0.533000
## 32      BRIC Community (2015)        4      0.009 0.533000
## 33      BRIC Environment (2015)       1      0.000 0.954000
## 34      BRIC Environment (2015)       2      0.005 0.628000
## 35      BRIC Environment (2015)       3      0.028 0.259000
## 36      BRIC Environment (2015)       4      0.045 0.147000
## 37      BRIC total (2015)            1      0.016 0.397000
## 38      BRIC total (2015)            2      0.077 0.056300
## 39      BRIC total (2015)            3      0.176 0.003010
## 40      BRIC total (2015)            4      0.191 0.001870
## 41      Mean Temperature (1990–2020)  1      0.047 0.139000
## 42      Mean Temperature (1990–2020)  2      0.139 0.008930
## 43      Mean Temperature (1990–2020)  3      0.246 0.000335
## 44      Mean Temperature (1990–2020)  4      0.279 0.000114
## 45 Mean Specific Humidity (1990–2020) 1      0.004 0.659000
## 46 Mean Specific Humidity (1990–2020) 2      0.006 0.601000
## 47 Mean Specific Humidity (1990–2020) 3      0.041 0.170000
## 48 Mean Specific Humidity (1990–2020) 4      0.034 0.208000

```

## SUMMARY OF RESULTS

```

# Define a significance threshold (p < 0.05)
regression_models <- regression_models %>%
  mutate(significance = ifelse(p_value < 0.05, "Significant", "Not Significant"))

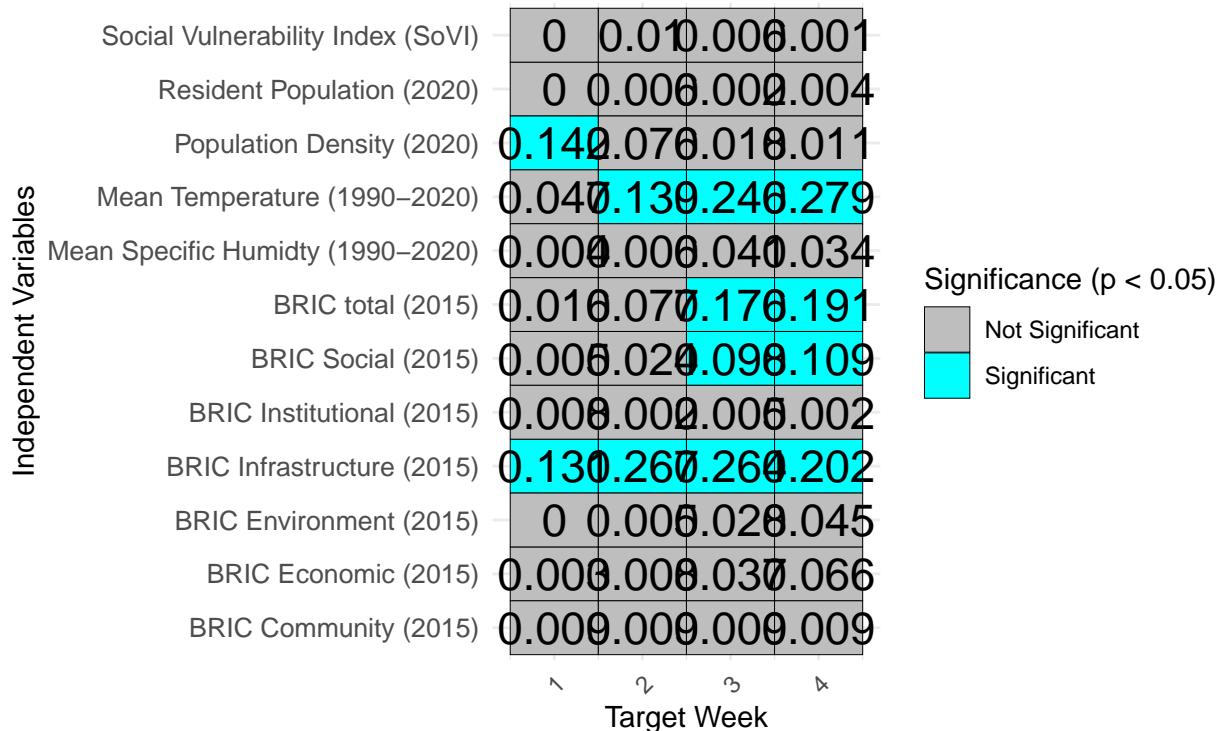
# Plot
reg2<-ggplot(regression_models, aes(x = Week_Ahead, y = independent_variable, fill = significance)) +
  geom_tile(color = "black") + # Add black borders to squares
  geom_text(aes(label = round(r_squared, 3)), color = "black", size = 6) + # Text inside boxes
  scale_fill_manual(values = c("Significant" = "cyan", "Not Significant" = "gray")) +
  labs(title = "Regression models R2",
       subtitle = "Mean WIS differences as dependent variable",
       x = "Target Week",
       y = "Independent Variables",
       fill = "Significance (p < 0.05)") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1),
        axis.text.y = element_text(size = 10),
        plot.title = element_text(size = 16, face = "bold"),
        plot.subtitle = element_text(size = 14))

reg2

```

## Regression models R<sup>2</sup>

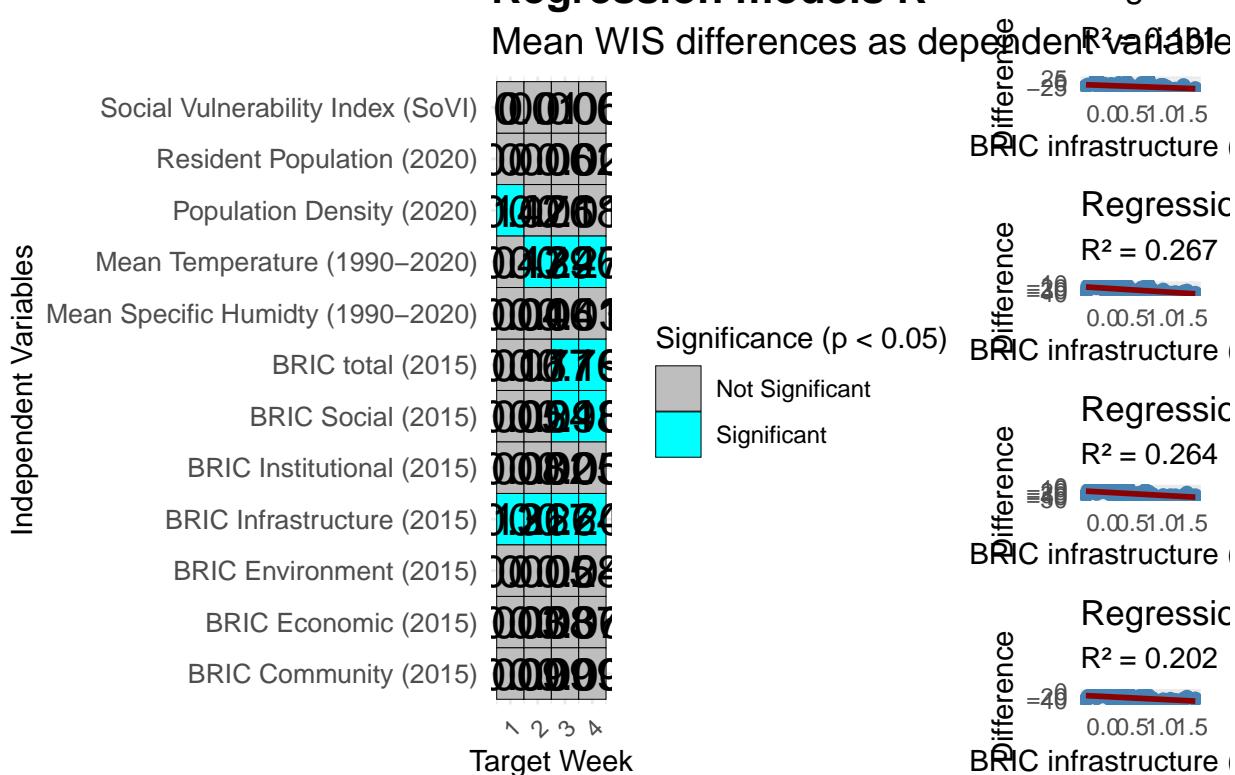
Mean WIS differences as dependent variable



```
reg2_scatter <- reg2 | combined_regressions
reg2_scatter
```

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

# Regression models R<sup>2</sup>



```
ggsave(reg2_scatter, height = 6, width = 12, filename="Fig10.jpg")
```

```
## `geom_smooth()` using formula = 'y ~ x'  
## `geom_smooth()` using formula = 'y ~ x'  
## `geom_smooth()` using formula = 'y ~ x'  
## `geom smooth()` using formula = 'y ~ x'
```

```
#qgsave(reg2, filename="Fig9.jpg", height = 6, width = 8)
```

Here we will be computing and plotting mean WIS improvement by epidemiological weeks comparing the best model and the baseline model.

```

# Create a list of data frames for the four weeks
weeks <- 1:4
results_by_epiweek_list <- list()

for (week in weeks) {
  # Merge the data frames for the current week
  combined_dataframes_ <- merge(get(paste0("filtered_df_W", week, "_NoLg")),
                                 get(paste0("filtered_df_W", week)))
  
  # Sum values by Julian date across all states
  mean_improvement_ <- combined_dataframes_ %>%
    group_by(Julian date) %>%

```

```

    summarize(
      AUTO_AR = mean(AUTO_AR, na.rm = TRUE), # Mean across all states
      AUTO_AVG_LB = mean(AUTO_AVG_LB, na.rm = TRUE) # Mean across all states
    ) %>%
    # Calculate the difference ratio in percentage
    mutate(Difference = ((AUTO_AVG_LB / AUTO_AR)-1)*100,
           Week = paste0("Week ", week))

    # Store the mean improvement by julian date
    results_by_epiweek_list[[week]] <- mean_improvement_
  }

  # Combine all weeks into a single data frame
  results_by_epiweek_all <- bind_rows(results_by_epiweek_list)

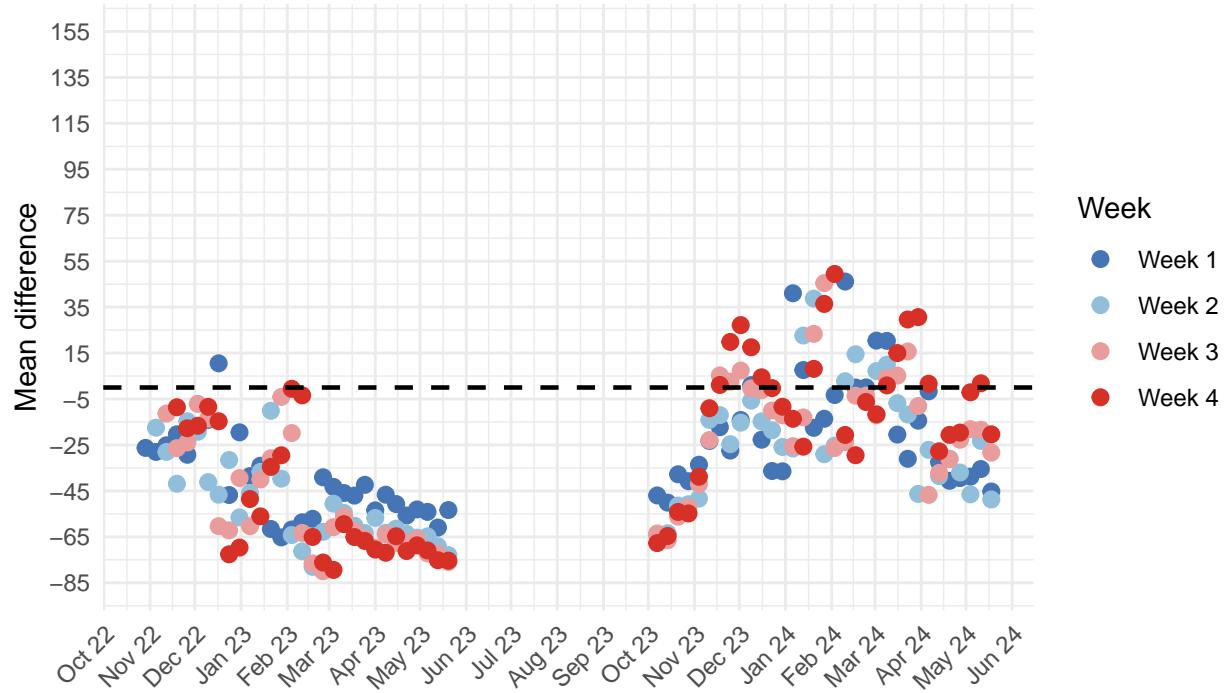
  # Plot the results
  model_improv<-ggplot(results_by_epiweek_all, aes(x = Julian_date, y = (Difference), color = Week)) +
    geom_point(size = 2.5) +
    geom_hline(yintercept = 0, color = "black", linetype = "dashed", size = 0.8) +
    theme_minimal() +
    labs(
      title = "A) Weekly mean WIS differences across all states",
      subtitle = "AUTO_AVG_LB compared to the AUTO_AR baseline",
      x = "",
      y = "Mean difference"
    ) +
    scale_y_continuous(
      limits = c(-85, 155),
      breaks = seq(-85, 155, by = 20)
    ) +
    scale_x_date(
      date_breaks = "1 month",
      date_labels = "%b %y"
    ) +
    scale_color_manual(
      values = c(
        "Week 1" = "#4575B4", # Blue
        "Week 2" = "#91BFDB", # Light blue
        "Week 3" = "#E89C9C", # Light red
        "Week 4" = "#D73027" # Red
      )
    ) +
    theme(
      plot.title = element_text(size = 18, face = "bold"),
      plot.subtitle = element_text(size = 14),
      axis.text.x = element_text(angle = 45, hjust = 1)
    )
  )

model_improv

```

## A) Weekly mean WIS differences across all states

AUTO\_AVG\_LB compared to the AUTO\_AR baseline



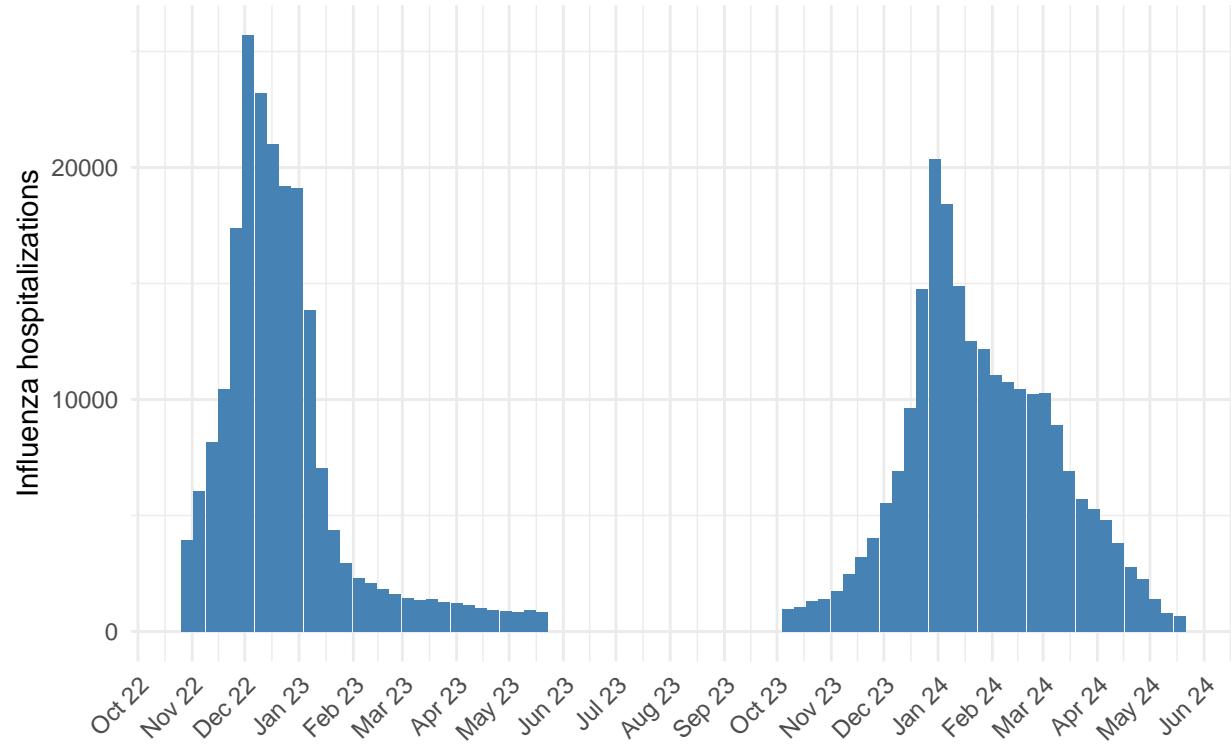
Total hospitalizations

```
# get total hospitalizations for the 48 states
total_hospitalizations_by_week <- AUTO_ADJACENT_WEEK1 %>%
  filter(epiweek >= 40 | epiweek <= 20) %>%
  group_by(target_end_date) %>%
  summarise(mean_cases = sum(cases, na.rm = TRUE))

# plotting results
hospital_plot <- ggplot(total_hospitalizations_by_week, aes(x = target_end_date, y = mean_cases)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  labs(title = "B) Total influenza hospitalizations in the 48 states",
       x = "",
       y = "Influenza hospitalizations") +
  scale_x_date(
    date_breaks = "1 month",
    date_labels = "%b %y"
  ) +
  theme_minimal() +
  theme(plot.title = element_text(size = 18, face = "bold"),
        plot.subtitle = element_text(size = 14),
        axis.text.x = element_text(angle = 45, hjust = 1))

hospital_plot
```

## B) Total influenza hospitalizations in the 48 states

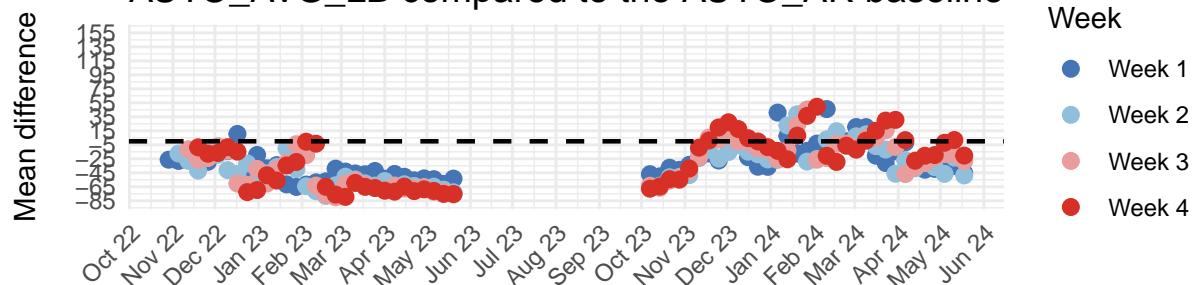


Combining model improvement and total hospitalizations

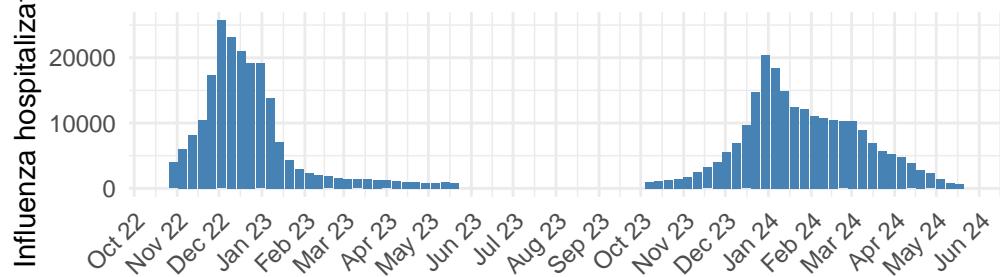
```
# Combining results in a single plot
combined_plot <- model_improv /hospital_plot
combined_plot
```

## A) Weekly mean WIS differences across all states

AUTO\_AVG\_LB compared to the AUTO\_AR baseline



## B) Total influenza hospitalizations in the 48 state



```
# Save the combined plot  
ggsave("Fig11.jpg", combined_plot, width = 7, height = 7, dpi = 600)
```