Changes in lisdexamfetamine after PBS subsidy restriction change, A Research Letter

We will be examining time series between **01 Jan 2016 and 31 Jan 2022**. Our intervention date (change date) is **1 February 2021** when the subsidy listing for lisdexamfetamine was expanded to allow people who were diagnosed with ADHD in adulthood subsidised access, prior to this adults who were diagnosed in adulthood were not eligible for subsidised Lisdexamfetamine.

This file uses PBS 10% sample data, the Australian Government Services Australia External Request Evaluation Committee granted data access (Approval Numbers: MI7542/RMS1941).Direct access to the data and analytical files to other individuals or authorities is not permitted without the express permission of the approving human research ethics committees and data custodians. The New South Wales (NSW) Population and Health Services Research Ethics Committee approved this study (Approval Number: 2013/11/494).

Previous code, has cleaned the data and extracted all dispensing records for approved and subsidised ADHD medicines in Australia for people >18 years. We excluded dexamfetamine dispensing records which indicated use was for Narcolepsy. atc\_code == ‘C02AC02’, ‘Guanfacine’, atc\_code == ‘N06BA02’, ‘Dexamfetamine’, atc\_code == ‘N06BA04’, ‘Methylphenidate’, atc\_code == ‘N06BA09’, ‘Atomoxetine’, atc\_code == ‘N06BA12’, ‘Lisdexamfetamine’

It also uses publicly available data from Australian Bureau of Statistics, data builder (see comments)

This file has two parts. First, create tables for the incidence (18 month lookback period) and prevalence of ADHD medicine use among adults in Australia for 2021. Estimates were presented *per 1000 adults*.

Second, to conduct an interrupted time series analysis quantifying changes in monthly dispensing following the lisdexamfetamine listing change on 1 February 2021. This part creates multiple pdfs to examine monthly dispensing rates as well as exploring ARIMA models, statistics and residuals. Finally, it produces a pdf with the final figure (monthly predicted and observed dispensing rates) and the effect estimates (with 95% confidence intervals) for the change in step, ramp, and trend effects.

# run load, clean, libraries needed ext  
session\_info()

## ─ Session info ───────────────────────────────────────────────────────────────  
## setting value  
## version R version 4.2.1 (2022-06-23 ucrt)  
## os Windows 10 x64 (build 22621)  
## system x86\_64, mingw32  
## ui RTerm  
## language (EN)  
## collate English\_Australia.utf8  
## ctype English\_Australia.utf8  
## tz Australia/Sydney  
## date 2023-03-07  
## pandoc 2.19.2 @ C:/Program Files/RStudio/resources/app/bin/quarto/bin/tools/ (via rmarkdown)  
##   
## ─ Packages ───────────────────────────────────────────────────────────────────  
## package \* version date (UTC) lib source  
## anytime 0.3.9 2020-08-27 [1] CRAN (R 4.2.2)  
## askpass 1.1 2019-01-13 [1] CRAN (R 4.2.2)  
## audio 0.1-10 2021-11-25 [1] CRAN (R 4.2.0)  
## base64enc 0.1-3 2015-07-28 [1] CRAN (R 4.2.0)  
## beepr \* 1.3 2018-06-04 [1] CRAN (R 4.2.2)  
## cachem 1.0.6 2021-08-19 [1] CRAN (R 4.2.1)  
## cli 3.4.1 2022-09-23 [1] CRAN (R 4.2.1)  
## colorspace 2.0-3 2022-02-21 [1] CRAN (R 4.2.1)  
## crayon 1.5.2 2022-09-29 [1] CRAN (R 4.2.1)  
## crul 1.3 2022-09-03 [1] CRAN (R 4.2.2)  
## curl 5.0.0 2023-01-12 [1] CRAN (R 4.2.2)  
## data.table \* 1.14.2 2021-09-27 [1] CRAN (R 4.2.1)  
## DBI \* 1.1.3 2022-06-18 [1] CRAN (R 4.2.1)  
## digest 0.6.29 2021-12-01 [1] CRAN (R 4.2.1)  
## distributional 0.3.1 2022-09-02 [1] CRAN (R 4.2.2)  
## dplyr \* 1.0.10 2022-09-01 [1] CRAN (R 4.2.1)  
## ellipsis 0.3.2 2021-04-29 [1] CRAN (R 4.2.1)  
## evaluate 0.20 2023-01-17 [1] CRAN (R 4.2.1)  
## fable \* 0.3.2 2022-09-01 [1] CRAN (R 4.2.2)  
## fabletools \* 0.3.2 2021-11-29 [1] CRAN (R 4.2.2)  
## fansi 1.0.3 2022-03-24 [1] CRAN (R 4.2.1)  
## farver 2.1.1 2022-07-06 [1] CRAN (R 4.2.1)  
## fastmap 1.1.0 2021-01-25 [1] CRAN (R 4.2.1)  
## feasts \* 0.3.0 2022-09-01 [1] CRAN (R 4.2.2)  
## flextable \* 0.8.4 2023-01-21 [1] CRAN (R 4.2.2)  
## ftExtra \* 0.5.0 2023-01-20 [1] CRAN (R 4.2.2)  
## gdtools 0.3.0 2023-01-15 [1] CRAN (R 4.2.2)  
## generics 0.1.3 2022-07-05 [1] CRAN (R 4.2.1)  
## gfonts 0.2.0 2023-01-08 [1] CRAN (R 4.2.2)  
## ggplot2 \* 3.4.0 2022-11-04 [1] CRAN (R 4.2.2)  
## glue 1.6.2 2022-02-24 [1] CRAN (R 4.2.1)  
## gridExtra \* 2.3 2017-09-09 [1] CRAN (R 4.2.1)  
## gtable 0.3.1 2022-09-01 [1] CRAN (R 4.2.1)  
## hms 1.1.2 2022-08-19 [1] CRAN (R 4.2.1)  
## htmltools 0.5.4 2022-12-07 [1] CRAN (R 4.2.2)  
## httpcode 0.3.0 2020-04-10 [1] CRAN (R 4.2.2)  
## httpuv 1.6.8 2023-01-12 [1] CRAN (R 4.2.2)  
## jsonlite 1.8.4 2022-12-06 [1] CRAN (R 4.2.2)  
## knitr \* 1.41 2022-11-18 [1] CRAN (R 4.2.2)  
## later 1.3.0 2021-08-18 [1] CRAN (R 4.2.2)  
## lifecycle 1.0.3 2022-10-07 [1] CRAN (R 4.2.1)  
## lubridate \* 1.9.0 2022-11-06 [1] CRAN (R 4.2.2)  
## magrittr \* 2.0.3 2022-03-30 [1] CRAN (R 4.2.1)  
## memoise 2.0.1 2021-11-26 [1] CRAN (R 4.2.1)  
## mime 0.12 2021-09-28 [1] CRAN (R 4.2.0)  
## munsell 0.5.0 2018-06-12 [1] CRAN (R 4.2.1)  
## officer 0.5.1 2023-01-06 [1] CRAN (R 4.2.2)  
## openssl 2.0.5 2022-12-06 [1] CRAN (R 4.2.2)  
## pillar 1.8.1 2022-08-19 [1] CRAN (R 4.2.1)  
## pkgconfig 2.0.3 2019-09-22 [1] CRAN (R 4.2.1)  
## promises 1.2.0.1 2021-02-11 [1] CRAN (R 4.2.2)  
## purrr \* 1.0.1 2023-01-10 [1] CRAN (R 4.2.2)  
## R6 2.5.1 2021-08-19 [1] CRAN (R 4.2.1)  
## Rcpp 1.0.9 2022-07-08 [1] CRAN (R 4.2.1)  
## readr \* 2.1.3 2022-10-01 [1] CRAN (R 4.2.1)  
## rlang 1.0.6 2022-09-24 [1] CRAN (R 4.2.1)  
## rmarkdown 2.20 2023-01-19 [1] CRAN (R 4.2.2)  
## rstudioapi 0.14 2022-08-22 [1] CRAN (R 4.2.2)  
## scales 1.2.1 2022-08-20 [1] CRAN (R 4.2.1)  
## sessioninfo \* 1.2.2 2021-12-06 [1] CRAN (R 4.2.1)  
## shiny 1.7.4 2022-12-15 [1] CRAN (R 4.2.2)  
## stringi 1.7.12 2023-01-11 [1] CRAN (R 4.2.2)  
## stringr 1.5.0 2022-12-02 [1] CRAN (R 4.2.2)  
## systemfonts 1.0.4 2022-02-11 [1] CRAN (R 4.2.1)  
## tibble 3.1.8 2022-07-22 [1] CRAN (R 4.2.1)  
## tidyr \* 1.2.1 2022-09-08 [1] CRAN (R 4.2.1)  
## tidyselect 1.2.0 2022-10-10 [1] CRAN (R 4.2.1)  
## timechange \* 0.2.0 2023-01-11 [1] CRAN (R 4.2.2)  
## tsibble \* 1.1.3 2022-10-09 [1] CRAN (R 4.2.2)  
## tzdb 0.3.0 2022-03-28 [1] CRAN (R 4.2.1)  
## utf8 1.2.2 2021-07-24 [1] CRAN (R 4.2.1)  
## uuid 1.1-0 2022-04-19 [1] CRAN (R 4.2.0)  
## vctrs 0.5.1 2022-11-16 [1] CRAN (R 4.2.2)  
## withr 2.5.0 2022-03-03 [1] CRAN (R 4.2.1)  
## xfun 0.36 2022-12-21 [1] CRAN (R 4.2.2)  
## xml2 1.3.3 2021-11-30 [1] CRAN (R 4.2.1)  
## xtable 1.8-4 2019-04-21 [1] CRAN (R 4.2.1)  
## yaml 2.3.6 2022-10-18 [1] CRAN (R 4.2.2)  
## zip 2.2.2 2022-10-26 [1] CRAN (R 4.2.2)  
##   
## [1] C:/Users/z5022817/AppData/Local/R/win-library/4.2  
## [2] C:/Program Files/R/R-4.2.1/library  
##   
## ──────────────────────────────────────────────────────────────────────────────

options(scipen = 999)  
options(error = function() {  
 beep()  
})  
  
# Create output  
time\_start <- Sys.time()  
  
# log current git commit  
# system('git --no-pager log -1 --date=short --no-decorate --pretty=format:"%d @ %h %cd" HEAD')  
# message()  
  
# source("R\_folder/Lisdexamfetamine-load.R")  
source("R\_folder/Lisdexamfetamine-clean.R")

## [1] "Last updated 2023-03-07"

##

## [1] "Number of people with an ADHD dispensing between 2015-01-01 and 2022-12-14 is 57121"

# number of dispensings  
ADHD\_char <- ADHD[supp\_date %between% c(as.Date("2016-01-01"), as.Date("2022-01-31")) & age > 17, ]  
  
min(ADHD\_char$supp\_date)

## [1] "2016-01-01"

max(ADHD\_char$supp\_date)

## [1] "2022-01-31"

min(ADHD\_char$age)

## [1] 18

num <- ADHD\_char %>%  
 filter(supp\_date <= "2022-01-31" & supp\_date > "2016-01-01" & age > 17) %>%  
 summarise(cnt = sum(scripts)) %>%  
 mutate(cnt = formatC(cnt \* 10, format = "d", big.mark = ","))  
  
cat("number of scripts dispensed", pull(num), sep = " ")

## number of scripts dispensed 3,515,370

persons <- ADHD\_char %>%  
 group\_by(pat\_id) %>%  
 arrange(pat\_id) %>%  
 distinct(pat\_id, .keep\_all = TRUE) %>%  
 ungroup()  
  
num <- persons %>%  
 filter(supp\_date <= "2022-01-31" & supp\_date > "2016-01-01" & age > 17) %>%  
 summarise(cnt = n()) %>%  
 mutate(cnt = formatC(cnt \* 10, format = "d", big.mark = ","))  
cat("number of persons dispensed", pull(num), sep = " ")

## number of persons dispensed 198,730

persons %>%  
 group\_by(sex) %>%  
 summarise(n = n()) %>%  
 flextable() %>%  
 set\_caption("number of males and females")

number of males and females

| sex | n |
| --- | --- |
| Female | 8,550 |
| Male | 11,428 |

ABS\_ERP <- readr::read\_csv("data/ABS\_ERP\_22\_23.01.30.csv", show\_col\_types = FALSE)  
  
# data extracted on 30.01.22 from  
# https://explore.data.abs.gov.au/vis?tm=quarterly%20population&pg=0&df[ds]=ABS\_ABS\_TOPICS&df[id]=ERP\_Q&df[ag]=ABS&df[vs]=1.0.0&hc[Frequency]=Quarterly&pd=2015-Q1%2C2022-Q2&dq=1.2%2B1%2B3.85%2B86%2B87%2B88%2B89%2B90%2B91%2B92%2B100%2B99%2B98%2B97%2B96%2B95%2B94%2B93%2B84%2B83%2B82%2B81%2B80%2B79%2B78%2B77%2B76%2B75%2B74%2B73%2B72%2B71%2B70%2B69%2B68%2B67%2B66%2B65%2B64%2B63%2B62%2B61%2B60%2B59%2B58%2B57%2B56%2B55%2B49%2B48%2B47%2B46%2B45%2B44%2B43%2B42%2B41%2B40%2B39%2B38%2B37%2B36%2B35%2B34%2B33%2B32%2B31%2B30%2B29%2B28%2B27%2B26%2B25%2B24%2B23%2B22%2B21%2B20%2B54%2B53%2B52%2B51%2B50%2B19%2B18.AUS.Q&ly[cl]=TIME\_PERIOD  
# Filters:  
# Measures: "Estimated Resident Population"  
# Sex: "Males"; "Females"; "Persons"  
# Age: (83 items) -> individual age from 18 (inclusive) to 99 and 100 and older group  
# Region: "Australia"  
# Frequency: "Quarterly"  
# Time period: "Start: 2015-Q1"; "End: 2022-Q2"  
  
ERP <- ABS\_ERP %>%  
 select(`SEX: Sex`, `AGE: Age`, `TIME\_PERIOD: Time Period`, OBS\_VALUE) %>%  
 separate(`AGE: Age`, c("age"), sep = ":", extra = "drop", convert = "true") %>%  
 separate(`SEX: Sex`, c("sex", "sex\_chr"), sep = ":", convert = "true") %>%  
 rename(date = `TIME\_PERIOD: Time Period`) %>%  
 rename(value = OBS\_VALUE) %>%  
 mutate(date = zoo::as.yearqtr(date, format = "%Y-Q%q"), age\_group = agegrp(age), year = year(date)) %>%  
 arrange(sex, date, age)  
  
if (length(unique(ERP$age)) == 101) {  
 print("ERP data has all ages 0-100")  
}  
  
if (length(unique(ERP$age)) == 101 - 18) {  
 print("ERP data has all ages 18-100")  
}

## [1] "ERP data has all ages 18-100"

ERP\_prev <- ERP %>%  
 filter(grepl("Q2", date)) %>%  
 group\_by(year, sex, age\_group) %>%  
 summarise(value = sum(value)) %>%  
 mutate(sex = case\_when(  
 sex == 1 ~ "Male",  
 sex == 2 ~ "Female",  
 sex == 3 ~ "persons"  
 ))

## `summarise()` has grouped output by 'year', 'sex'. You can override using the  
## `.groups` argument.

ERP\_time <- ERP %>%  
 group\_by(date, sex) %>%  
 summarise(value = sum(value)) %>%  
 mutate(sex = case\_when(  
 sex == 1 ~ "Male",  
 sex == 2 ~ "Female",  
 sex == 3 ~ "persons"  
 ), date = zoo::as.Date(zoo::as.yearmon(date, format = "%Y-%m-%d")))

## `summarise()` has grouped output by 'date'. You can override using the  
## `.groups` argument.

erp <- tsibble(ERP\_time %>% mutate(sex = factor(sex)) %>%  
 mutate(date = yearmonth(date)), key = sex, index = date) %>%  
 full\_join(tibble(  
 date = rep(seq(yearmonth("2015-01-01"), yearmonth("2022-07-01"), by = 1), 3),  
 sex = c(rep("Male", 91), rep("Female", 91), rep("persons", 91))  
 )) %>%  
 fill\_gaps()

## Joining, by = c("date", "sex")

erpx <- erp %>%  
 split(list(.$sex)) %>%  
 map\_dfr(., ~ .x %>%  
 model(lm = TSLM(value ~ trend(  
 knots = yearmonth(c(  
 "2016-01-01", "2016-04-01", "2016-07-01", "2016-10-01",  
 "2017-01-01", "2017-04-01", "2017-07-01", "2017-10-01",  
 "2018-01-01", "2018-04-01", "2018-07-01", "2018-10-01",  
 "2019-01-01", "2019-04-01", "2019-07-01", "2019-10-01",  
 "2020-01-01", "2020-04-01", "2020-07-01", "2020-10-01",  
 "2021-01-01", "2021-04-01", "2021-07-01", "2021-10-01",  
 "2022-01-01", "2022-04-01", "2022-07-01"  
 ))  
 ))) %>%  
 interpolate(erp))  
  
glimpse(erpx)

## Rows: 273  
## Columns: 3  
## Key: sex [3]  
## $ sex <chr> "Female", "Female", "Female", "Female", "Female", "Female", "Fem…  
## $ date <mth> 2015 Jan, 2015 Feb, 2015 Mar, 2015 Apr, 2015 May, 2015 Jun, 2015…  
## $ value <dbl> 9346570, 9355089, 9368004, 9378993, 9393835, 9406751, 9416333, 9…

# Prevalence for overall use by year;  
ADHD\_1 <- ADHD %>%  
 # filter(age > 17) %>%  
 group\_by(pat\_id, year) %>%  
 mutate(first = row\_number() == 1) %>%  
 filter(age > 17) %>%  
 filter(first == TRUE) %>%  
 select(pat\_id, first, year, age, sex, supp\_date, atc\_code, medicine, age\_group\_2) %>%  
 rename(age\_group = age\_group\_2)  
  
# Prevalence for overall use by year;  
ADHD\_prev\_overall <- ADHD\_1 %>%  
 group\_by(year) %>%  
 summarise(n = n()) %>%  
 mutate(sex = "persons")  
ADHD\_prev\_sex <- ADHD\_1 %>%  
 group\_by(year, sex) %>%  
 summarise(n = n())

## `summarise()` has grouped output by 'year'. You can override using the  
## `.groups` argument.

ERP\_overall <- ERP\_prev %>%  
 group\_by(year, sex) %>%  
 summarise(value = sum(value))

## `summarise()` has grouped output by 'year'. You can override using the  
## `.groups` argument.

prevalence <- rbind(ADHD\_prev\_overall, ADHD\_prev\_sex) %>%  
 left\_join(ERP\_overall, by = c("year", "sex")) %>%  
 mutate(n = n \* 10, prev = round((n / value) \* 1000, digits = 1), n = paste0("(", formatC(n, format = "d", big.mark = ","), ")")) %>%  
 unite(rate, c("prev", "n"), sep = " ") %>%  
 select(year, rate, sex) %>%  
 filter(year == 2021 & sex == "persons") %>%  
 mutate(age\_group = "Overall")  
  
  
# prevalent user based on age group and year  
# Estimated resident populations - by age and sex and year  
# estimated prevalence by age group, sex and year  
  
ADHD\_prev\_age\_sex <- ADHD\_1 %>%  
 group\_by(age\_group, sex, year) %>%  
 summarise(n = n())

## `summarise()` has grouped output by 'age\_group', 'sex'. You can override using  
## the `.groups` argument.

ADHD\_prev\_age <- ADHD\_1 %>%  
 group\_by(age\_group, year) %>%  
 summarise(n = n()) %>%  
 mutate(sex = "persons")

## `summarise()` has grouped output by 'age\_group'. You can override using the  
## `.groups` argument.

prevalence\_age\_sex <- rbind(ADHD\_prev\_age\_sex, ADHD\_prev\_age) %>%  
 left\_join(ERP\_prev, by = c("age\_group", "sex", "year")) %>%  
 mutate(n = n \* 10, prev = round((n / value) \* 1000, digits = 1), n = paste0("(", formatC(n, format = "d", big.mark = ","), ")")) %>%  
 unite(rate, c("prev", "n"), sep = " ") %>%  
 select(year, rate, sex, age\_group) %>%  
 filter(year == 2021)  
  
prevalence\_age\_sex$sex <- factor(prevalence\_age\_sex$sex, levels = c("persons", "Male", "Female"))  
  
###############################################################################################################################################  
  
# prevalent user based on medicine and year  
  
## to restrict to age>17 before selecting first record or not?  
ADHD\_2 <- ADHD %>%  
 # filter(age > 17) %>%  
 group\_by(pat\_id, year, medicine) %>%  
 mutate(first = row\_number() == 1) %>%  
 filter(age > 17) %>%  
 filter(first == TRUE) %>%  
 select(pat\_id, first, year, age\_group\_2, sex, supp\_date, atc\_code, medicine, class) %>%  
 rename(age\_group = age\_group\_2) %>%  
 mutate(class = case\_when(  
 atc\_code == "C02AC02" ~ "Nonstimulant",  
 atc\_code == "N06BA02" ~ "Stimulant",  
 atc\_code == "N06BA04" ~ "Stimulant",  
 atc\_code == "N06BA09" ~ "Nonstimulant",  
 atc\_code == "N06BA12" ~ "Stimulant",  
 TRUE ~ "Other"  
 ))  
  
  
# estimated prevalence by medicine and year  
  
ADHD\_prev\_med <- ADHD\_2 %>%  
 group\_by(year, class, medicine) %>%  
 summarise(n = n())

## `summarise()` has grouped output by 'year', 'class'. You can override using the  
## `.groups` argument.

prevalence\_med <- ERP\_overall %>%  
 filter(sex == "persons") %>%  
 left\_join(ADHD\_prev\_med, by = c("year")) %>%  
 mutate(n = n \* 10, prev = round((n / value) \* 1000, digits = 1), n = paste0("(", formatC(n, format = "d", big.mark = ","), ")")) %>%  
 unite(rate, c("prev", "n"), sep = " ") %>%  
 select(year, rate, medicine) %>%  
 filter(year == 2021) %>%  
 mutate(age\_group = "Medicine") %>%  
 rename(sex = medicine)  
  
  
#################################################################################  
ADHD\_prev\_class <- ADHD\_2 %>%  
 group\_by(year, class) %>%  
 summarise(n = n())

## `summarise()` has grouped output by 'year'. You can override using the  
## `.groups` argument.

prevalence\_class <- ERP\_overall %>%  
 filter(sex == "persons") %>%  
 left\_join(ADHD\_prev\_class, by = c("year")) %>%  
 mutate(n = n \* 10, prev = round((n / value) \* 1000, digits = 1), n = paste0("(", formatC(n, format = "d", big.mark = ","), ")")) %>%  
 unite(rate, c("prev", "n"), sep = " ") %>%  
 select(year, rate, class) %>%  
 filter(year == 2021) %>%  
 mutate(age\_group = "Medicine") %>%  
 rename(sex = class)  
  
ERP\_2021 <- ERP\_overall %>%  
 filter(year == 2021 & sex == "persons") %>%  
 mutate(age\_group = "Population", sex = "Population") %>%  
 rename(rate = value) %>%  
 mutate(rate = formatC(rate, format = "d", big.mark = ","))  
  
  
sexratio\_18 <- prevalence\_age\_sex %>%  
 filter(sex != "persons" & age\_group == "18--24 years") %>%  
 separate(rate, "prev", sep = " ") %>%  
 select(prev, sex) %>%  
 mutate(prev = as.numeric(prev))

## Warning: Expected 1 pieces. Additional pieces discarded in 2 rows [1, 2].

## Adding missing grouping variables: `age\_group`

sexratio\_25 <- prevalence\_age\_sex %>%  
 filter(sex != "persons" & age\_group == "25+ years") %>%  
 separate(rate, "prev", sep = " ") %>%  
 select(prev, sex) %>%  
 mutate(prev = as.numeric(prev))

## Warning: Expected 1 pieces. Additional pieces discarded in 2 rows [1, 2].

## Adding missing grouping variables: `age\_group`

sex\_ratio <- data.frame(  
 rate = c(  
 round(pull(sexratio\_18[2, 2]) / pull(sexratio\_18[1, 2]), digits = 1),  
 round(pull(sexratio\_25[2, 2]) / pull(sexratio\_25[1, 2]), digits = 1)  
 ),  
 sex = c("Male to female sex ratio", "Male to female sex ratio"),  
 age\_group = c("18--24 years", "25+ years"),  
 year = c(2021, 2021)  
)  
  
  
prevalence\_table <- rbind(prevalence, prevalence\_age\_sex, prevalence\_med, prevalence\_class, ERP\_2021, sex\_ratio) %>%  
 select(age\_group, sex, rate) %>%  
 arrange(age\_group, sex)  
  
  
prevalence\_table$age\_group <- factor(prevalence\_table$age\_group, levels = c("Overall", "18--24 years", "25+ years", "Medicine"))  
prevalence\_table$sex <- factor(prevalence\_table$sex, levels = c("persons", "Male", "Female", "Male to female sex ratio", "Stimulant", "Dexamfetamine", "Lisdexamfetamine", "Methylphenidate", "Nonstimulant", "Atomoxetine", "Guanfacine", "Population"))  
  
  
  
prevalence\_table %>%  
 arrange(age\_group, sex) %>%  
 group\_by(age\_group) %>%  
 as\_flextable(hide\_grouplabel = TRUE) %>%  
 bold(i = ~ !is.na(age\_group), bold = TRUE, part = "body") %>%  
 bold(i = ~ sex == "persons" | sex == "Population") %>%  
 italic(i = ~ sex == "Stimulant" | sex == "Nonstimulant") %>% #  
 font(fontname = "sans", part = "all") %>%  
 bold(bold = TRUE, part = "header") %>%  
 set\_header\_labels(sex = "", rate = "Prevalence (n)") %>%  
 add\_footer\_lines("ADHD, attention-deficit/hyperactivity disorder") %>%  
 autofit()

|  | **Prevalence (n)** |
| --- | --- |
| **Overall** | |
| **persons** | **6.5 (130,820)** |
| **18--24 years** | |
| **persons** | **16.3 (35,990)** |
| Male | 17.1 (19,480) |
| Female | 15.4 (16,510) |
| Male to female sex ratio | 1.1 |
| **25+ years** | |
| **persons** | **5.3 (94,830)** |
| Male | 5.9 (51,490) |
| Female | 4.7 (43,340) |
| Male to female sex ratio | 1.3 |
| **Medicine** | |
| *Stimulant* | *7.4 (147,720)* |
| Dexamfetamine | 3 (60,180) |
| Lisdexamfetamine | 2.3 (45,730) |
| Methylphenidate | 2.1 (41,810) |
| *Nonstimulant* | *0.5 (9,570)* |
| Atomoxetine | 0.3 (6,200) |
| Guanfacine | 0.2 (3,370) |
|  |  |
| **Population** | **20,039,723** |
| ADHD, attention-deficit/hyperactivity disorder | |

# incidence for overall use by year;  
ADHD\_1 <- ADHD %>%  
 group\_by(pat\_id) %>%  
 arrange(pat\_id, supp\_date) %>%  
 mutate(first = row\_number() == 1) %>%  
 filter(first == TRUE) %>%  
 filter(supp\_date > as.Date(min(ADHD$supp\_date) + months(17))) %>% # 18-month period look back  
 filter(age > 17) %>%  
 select(pat\_id, first, year, age, sex, supp\_date, atc\_code, medicine, class, age\_group\_2) %>%  
 rename(age\_group = age\_group\_2) %>%  
 mutate(class = case\_when(  
 atc\_code == "C02AC02" ~ "Nonstimulant",  
 atc\_code == "N06BA02" ~ "Stimulant",  
 atc\_code == "N06BA04" ~ "Stimulant",  
 atc\_code == "N06BA09" ~ "Nonstimulant",  
 atc\_code == "N06BA12" ~ "Stimulant",  
 TRUE ~ "Other"  
 ))  
  
  
# incidence for overall use by year;  
ADHD\_inc\_overall <- ADHD\_1 %>%  
 group\_by(year) %>%  
 summarise(n = n()) %>%  
 mutate(sex = "persons")  
ADHD\_inc\_sex <- ADHD\_1 %>%  
 group\_by(year, sex) %>%  
 summarise(n = n())

## `summarise()` has grouped output by 'year'. You can override using the  
## `.groups` argument.

ERP\_overall <- ERP\_prev %>%  
 group\_by(year, sex) %>%  
 summarise(value = sum(value))

## `summarise()` has grouped output by 'year'. You can override using the  
## `.groups` argument.

incidence <- rbind(ADHD\_inc\_overall, ADHD\_inc\_sex) %>%  
 left\_join(ERP\_overall, by = c("year", "sex")) %>%  
 mutate(n = n \* 10, prev = round((n / value) \* 1000, digits = 1), n = paste0("(", formatC(n, format = "d", big.mark = ","), ")")) %>%  
 unite(rate, c("prev", "n"), sep = " ") %>%  
 select(year, rate, sex) %>%  
 filter(year == 2021 & sex == "persons") %>%  
 mutate(age\_group = "Overall")  
  
  
# Estimated resident populations - by age and sex and year  
# estimated incidence by age group, sex and year  
  
ADHD\_inc\_age\_sex <- ADHD\_1 %>%  
 group\_by(age\_group, sex, year) %>%  
 summarise(n = n())

## `summarise()` has grouped output by 'age\_group', 'sex'. You can override using  
## the `.groups` argument.

ADHD\_inc\_age <- ADHD\_1 %>%  
 group\_by(age\_group, year) %>%  
 summarise(n = n()) %>%  
 mutate(sex = "persons")

## `summarise()` has grouped output by 'age\_group'. You can override using the  
## `.groups` argument.

incidence\_age\_sex <- rbind(ADHD\_inc\_age\_sex, ADHD\_inc\_age) %>%  
 left\_join(ERP\_prev, by = c("age\_group", "sex", "year")) %>%  
 mutate(n = n \* 10, prev = round((n / value) \* 1000, digits = 1), n = paste0("(", formatC(n, format = "d", big.mark = ","), ")")) %>%  
 unite(rate, c("prev", "n"), sep = " ") %>%  
 select(year, rate, sex, age\_group) %>%  
 filter(year == 2021)  
  
incidence\_age\_sex$sex <- factor(incidence\_age\_sex$sex, levels = c("persons", "Male", "Female"))  
  
###################################################################################################################################################  
  
# estimated incidence by medicine and year  
### create a new incidence based on incident medicine use:  
ADHD\_2 <- ADHD %>%  
 group\_by(pat\_id, medicine) %>%  
 arrange(pat\_id, supp\_date) %>%  
 mutate(first = row\_number() == 1) %>%  
 filter(first == TRUE) %>%  
 filter(supp\_date > as.Date(min(ADHD$supp\_date) + months(17))) %>% # 18-month period look back  
 filter(age > 17) %>%  
 select(pat\_id, first, year, age, sex, supp\_date, atc\_code, medicine, class, age\_group\_2) %>%  
 rename(age\_group = age\_group\_2) %>%  
 mutate(class = case\_when(  
 atc\_code == "C02AC02" ~ "Nonstimulant",  
 atc\_code == "N06BA02" ~ "Stimulant",  
 atc\_code == "N06BA04" ~ "Stimulant",  
 atc\_code == "N06BA09" ~ "Nonstimulant",  
 atc\_code == "N06BA12" ~ "Stimulant",  
 TRUE ~ "Other"  
 ))  
  
# estimated incidence by medicine and year  
  
ADHD\_inc\_med <- ADHD\_2 %>%  
 group\_by(year, class, medicine) %>%  
 summarise(n = n())

## `summarise()` has grouped output by 'year', 'class'. You can override using the  
## `.groups` argument.

incidence\_med <- ERP\_overall %>%  
 filter(sex == "persons") %>%  
 left\_join(ADHD\_inc\_med, by = c("year")) %>%  
 mutate(n = n \* 10, prev = round((n / value) \* 1000, digits = 1), n = paste0("(", formatC(n, format = "d", big.mark = ","), ")")) %>%  
 unite(rate, c("prev", "n"), sep = " ") %>%  
 select(year, rate, medicine) %>%  
 filter(year == 2021) %>%  
 mutate(age\_group = "Medicine") %>%  
 rename(sex = medicine)  
  
  
#################################################################################  
ADHD\_inc\_class <- ADHD\_2 %>%  
 group\_by(year, class) %>%  
 summarise(n = n())

## `summarise()` has grouped output by 'year'. You can override using the  
## `.groups` argument.

incidence\_class <- ERP\_overall %>%  
 filter(sex == "persons") %>%  
 left\_join(ADHD\_inc\_class, by = c("year")) %>%  
 mutate(n = n \* 10, prev = round((n / value) \* 1000, digits = 1), n = paste0("(", formatC(n, format = "d", big.mark = ","), ")")) %>%  
 unite(rate, c("prev", "n"), sep = " ") %>%  
 select(year, rate, class) %>%  
 filter(year == 2021) %>%  
 mutate(age\_group = "Medicine") %>%  
 rename(sex = class)  
  
ERP\_2021 <- ERP\_overall %>%  
 filter(year == 2021 & sex == "persons") %>%  
 mutate(age\_group = "Population", sex = "Population") %>%  
 rename(rate = value) %>%  
 mutate(rate = formatC(rate, format = "d", big.mark = ","))  
  
sexratio\_18 <- incidence\_age\_sex %>%  
 filter(sex != "persons" & age\_group == "18--24 years") %>%  
 separate(rate, "prev", sep = " ") %>%  
 select(prev, sex) %>%  
 mutate(prev = as.numeric(prev))

## Warning: Expected 1 pieces. Additional pieces discarded in 2 rows [1, 2].

## Adding missing grouping variables: `age\_group`

sexratio\_25 <- incidence\_age\_sex %>%  
 filter(sex != "persons" & age\_group == "25+ years") %>%  
 separate(rate, "prev", sep = " ") %>%  
 select(prev, sex) %>%  
 mutate(prev = as.numeric(prev))

## Warning: Expected 1 pieces. Additional pieces discarded in 2 rows [1, 2].

## Adding missing grouping variables: `age\_group`

sex\_ratio <- data.frame(  
 rate = c(  
 round(pull(sexratio\_18[2, 2]) / pull(sexratio\_18[1, 2]), digits = 1),  
 round(pull(sexratio\_25[2, 2]) / pull(sexratio\_25[1, 2]), digits = 1)  
 ),  
 sex = c("Male to female sex ratio", "Male to female sex ratio"),  
 age\_group = c("18--24 years", "25+ years"),  
 year = c(2021, 2021)  
)  
  
  
incidence\_table <- rbind(incidence, incidence\_age\_sex, incidence\_med, incidence\_class, ERP\_2021, sex\_ratio) %>%  
 select(age\_group, sex, rate) %>%  
 arrange(age\_group, sex)  
  
  
incidence\_table$age\_group <- factor(incidence\_table$age\_group, levels = c("Overall", "18--24 years", "25+ years", "Medicine"))  
incidence\_table$sex <- factor(incidence\_table$sex, levels = c("persons", "Male", "Female", "Male to female sex ratio", "Stimulant", "Dexamfetamine", "Lisdexamfetamine", "Methylphenidate", "Nonstimulant", "Atomoxetine", "Guanfacine", "Population"))  
  
  
incidence\_table %>%  
 rename(rate\_inc = rate) %>%  
 left\_join(prevalence\_table) %>%  
 arrange(age\_group, sex) %>%  
 group\_by(age\_group) %>%  
 as\_flextable(hide\_grouplabel = TRUE) %>%  
 bold(i = ~ !is.na(age\_group), bold = TRUE, part = "body") %>%  
 bold(i = ~ sex == "persons" | sex == "Population") %>%  
 italic(i = ~ sex == "Stimulant" | sex == "Nonstimulant") %>% #  
 font(fontname = "sans", part = "all") %>%  
 bold(bold = TRUE, part = "header") %>%  
 set\_header\_labels(sex = "", rate\_inc = "Incidence (n)", rate = "Prevalence (n)") %>%  
 add\_footer\_lines("ADHD, attention-deficit/hyperactivity disorder") %>%  
 autofit()

## Joining, by = c("age\_group", "sex")

|  | **Incidence (n)** | **Prevalence (n)** |
| --- | --- | --- |
| **Overall** | | |
| **persons** | **2.2 (44,740)** | **6.5 (130,820)** |
| **18--24 years** | | |
| **persons** | **5.1 (11,300)** | **16.3 (35,990)** |
| Male | 3.9 (4,460) | 17.1 (19,480) |
| Female | 6.4 (6,840) | 15.4 (16,510) |
| Male to female sex ratio | 0.6 | 1.1 |
| **25+ years** | | |
| **persons** | **1.9 (33,440)** | **5.3 (94,830)** |
| Male | 1.8 (15,720) | 5.9 (51,490) |
| Female | 1.9 (17,720) | 4.7 (43,340) |
| Male to female sex ratio | 0.9 | 1.3 |
| **Medicine** | | |
| *Stimulant* | *3.3 (67,020)* | *7.4 (147,720)* |
| Dexamfetamine | 1 (19,610) | 3 (60,180) |
| Lisdexamfetamine | 1.6 (33,050) | 2.3 (45,730) |
| Methylphenidate | 0.7 (14,360) | 2.1 (41,810) |
| *Nonstimulant* | *0.3 (5,260)* | *0.5 (9,570)* |
| Atomoxetine | 0.2 (3,310) | 0.3 (6,200) |
| Guanfacine | 0.1 (1,950) | 0.2 (3,370) |
|  |  |  |
| **Population** | **20,039,723** | **20,039,723** |
| ADHD, attention-deficit/hyperactivity disorder | | |

# using tidyverts: Tsibbles:  
# Index is a variable with inherent ordering from past to present.  
# Key is a set of variables that define observational units over time.  
# Each observation should be uniquely identified by index and key.  
# Each observational unit should be measured at a common interval, if regularly spaced  
  
# Convert data to time series object - dispensings by supply date: for each age group and medicine  
# kept in supp\_date so can be flexible with tsibble  
# monthly aggregate dispensing by sex, age and medicine ()  
  
# update 08/02/23  
all <- ADHD %>%  
 filter(age\_group == "Over 18") %>%  
 mutate(medicine = class) %>%  
 # rename(date = mos) %>%  
 group\_by(medicine, mos) %>%  
 summarise(cnt = sum(scripts)) %>%  
 mutate(sex = "persons", date = yearmonth(mos))

## `summarise()` has grouped output by 'medicine'. You can override using the  
## `.groups` argument.

all\_sex <- ADHD %>%  
 filter(age\_group == "Over 18") %>%  
 mutate(medicine = class) %>% # updating plot individual stimulants and non-stimulants together  
 # rename(date = mos) %>%  
 group\_by(sex, medicine, mos) %>%  
 summarise(cnt = sum(scripts)) %>%  
 mutate(date = yearmonth(mos))

## `summarise()` has grouped output by 'sex', 'medicine'. You can override using  
## the `.groups` argument.

all <- rbind(all, all\_sex)  
  
### do analysis on data from 2016 to end of 2021  
#\*\*\*\* IS More post intervention data necessary can have data up until July 2022 - Possibly September.  
# Update 8/12 - run with data until July 2022  
  
a <- all %>% filter(date >= yearmonth("2016-02-01") &  
 date < yearmonth("2022-02-01"))  
# update to only include data from 12 months after intervention  
  
ts <- tsibble(a, key = c(sex, medicine), index = date) %>% fill\_gaps()  
  
### Pre-intervention not mapped - unused  
  
a2 <- all %>% filter(date >= yearmonth("2016-02-01") & date < yearmonth("2021-01-01"))  
  
ts2 <- tsibble(a2, key = c(sex, medicine), index = date) %>%  
 fill\_gaps() %>%  
 inner\_join(erpx) %>%  
 mutate(rate = (cnt \* 10000) / value, cnt = as.numeric(cnt \* 10))

## Joining, by = c("sex", "date")

# define change date  
change\_date <- "2021-02-01"  
  
ramp <- ts %>%  
 ungroup() %>%  
 filter(medicine == "Dexamfetamine" & sex == "Female") %>%  
 filter(as.Date(date) >= as.Date(change\_date)) %>%  
 select(date) %>%  
 mutate(ramp = 1:n())  
  
tss <- merge(ts, ramp, by = "date", all.x = TRUE) %>%  
 mutate(  
 step = ifelse(as.Date(date) >= as.Date(change\_date), 1, 0),  
 ramp = ifelse(is.na(ramp), 0, ramp)  
 ) %>%  
 as\_tsibble(key = c(sex, medicine), index = date)  
  
## update rate per 1000 population  
ts <- tss %>%  
 inner\_join(erpx) %>%  
 mutate(rate = (cnt \* 10000) / value, cnt = as.numeric(cnt \* 10))

## Joining, by = c("date", "sex")

### Remove data point for APRIL 2020, dispensing outlier due to covid 19 lockdowns.  
  
ts %<>% mutate(flag = ifelse(date == yearmonth("2020-04-01"), 1, 0)) %>%  
 filter(flag == 0) %>%  
 fill\_gaps()  
  
nest\_ts <- ts %>%  
 select(date, medicine, sex, rate, cnt, ramp, step) %>%  
 group\_by(sex, medicine) %>%  
 mutate(  
 # log data variables  
 log\_rate = log(rate)  
 ) %>%  
 group\_by(sex, medicine) %>%  
 nest()

Exploring first whether data should be transformed and which differences are needed (are the data autocorrelated and seasonal)

## plot SLT trend prior to intervention  
  
plot\_slt <- ts2 %>% model(STL = STL(log(rate)))  
  
  
plot\_all <- plot\_slt %>% mutate(  
 plot = pmap(list(STL, medicine, sex), ~ ..1 %>%  
 components() %>%  
 autoplot() + labs(title = paste0("Log rate STL plots for", ..2, ..3, sep = " ")))  
)  
  
  
pdf(file = paste0(Sys.Date(), "\_check plot STL.pdf"))  
  
for (i in 1:12) {  
 # plot.new()  
 print(plot\_all$plot[[i]])  
}  
dev.off()

## png   
## 2

plot\_fun <- function(nest\_data, y) {  
 nest\_data %>%  
 mutate(plot = pmap(list(data, medicine, sex), ~ gg\_tsdisplay(..1, y = .data[[y]], plot\_type = "season") +  
 labs(title = paste0(y, "plots for", ..2, ..3, sep = " "))))  
}  
  
  
  
  
uniroot\_fun <- function(nest\_data, y) {  
 nest\_data %>%  
 mutate(  
 map\_df(data, ~ .x %>% features(.data[[y]], unitroot\_ndiffs)),  
 map\_df(data, ~ .x %>% features(.data[[y]], unitroot\_nsdiffs))  
 ) %>%  
 select(-data)  
}  
  
  
plot\_vars <- c("log\_rate")  
  
for (vars in plot\_vars) {  
 plot\_all <- plot\_fun(nest\_data = nest\_ts, y = vars)  
 ft <- uniroot\_fun(nest\_ts, vars) %>%  
 flextable() %>%  
 ## create plot friendly font  
 font(fontname = "sans", part = "all") %>%  
 autofit() %>%  
 add\_header\_lines(paste0("Number of suggested seasonal and nonseasonal differences based on uniroot test for ", vars))  
  
  
 # create pdf files  
 pdf(file = paste0(Sys.Date(), "\_Raw plots (with season) for", vars, " by medicine and sex.pdf"))  
  
  
 for (i in 1:12) {  
 plot.new()  
 print(plot\_all[["plot"]][[i]])  
 }  
 plot.new()  
 # print flextable inside loop  
  
 plot(ft, method = "grob")  
 dev.off()  
}



## Warning: Removed 1 rows containing missing values (`geom\_point()`).  
## Removed 1 rows containing missing values (`geom\_point()`).  
## Removed 1 rows containing missing values (`geom\_point()`).  
## Removed 1 rows containing missing values (`geom\_point()`).  
## Removed 1 rows containing missing values (`geom\_point()`).  
## Removed 1 rows containing missing values (`geom\_point()`).  
## Removed 1 rows containing missing values (`geom\_point()`).  
## Removed 1 rows containing missing values (`geom\_point()`).  
## Removed 1 rows containing missing values (`geom\_point()`).  
## Removed 1 rows containing missing values (`geom\_point()`).  
## Removed 1 rows containing missing values (`geom\_point()`).  
## Removed 1 rows containing missing values (`geom\_point()`).



Step 3. Model selection: While automated algorithms in several statistical packages can identify candidate p and q parameters, they can sometimes be estimated based on the ACF/PACF plots. a. Determine differencing order to induce stationary: If there is seasonality, a seasonal difference is required and D =1. (Going with arma model instead) b. Plot the ACF/PACF of stationarity data to determine potential AR/MA orders (gg\_tsresidual) c. Estimate model and use information criteria to find the best model (AICc wont help with models based on differencing)

## create models: decision was made to only use seasonal arima models based on residual checks and for all series. Note cannot compare information criteras when models have different differencing.  
  
  
model\_ts <- ts %>%  
 model(  
 seasonal\_arma\_log = ARIMA(log(rate) ~ pdq(d = 0) + PDQ(D = 1) + step + ramp + trend(), stepwise = FALSE),  
 # arma\_log = ARIMA(log(rate)~pdq(d=0)+PDQ()+step+ramp+trend(), stepwise=FALSE),  
 # arima\_diff\_log = ARIMA(log(rate)~pdq(d=1)+PDQ()+step+ramp, stepwise=FALSE),  
 # arma = ARIMA(rate~pdq(d=0)+PDQ()+step+ramp+trend(), stepwise=FALSE),  
 # arima\_diff = ARIMA(rate~pdq(d=1)+PDQ()+step+ramp, stepwise=FALSE),  
  
 # auto\_trend\_log= ARIMA(log(rate)~ramp+step+trend()),  
 # auto\_log = ARIMA(log(rate)~ramp+step)  
 )  
  
################################################################################################################  
saveRDS(model\_ts, paste0("data/model\_ts\_", Sys.Date(), ".Rds"))  
  
#model\_ts<-readRDS(paste0("data/model\_ts\_", Sys.Date(), ".Rds"))  
  
  
## seasonal lag formula - based on https://robjhyndman.com/hyndsight/ljung-box-test/  
  
# For non-seasonal time series, use h=\min(10, T/5)h=min(10,T/5).  
# For seasonal time series, use h=\min(2m, T/5)h=min(2m,T/5).  
## number of dates -1 as excluded data from APR 2020  
# non\_seasonal = min(10, (nrow(nest\_ts$data[[1]])-1)/5)  
seasonal\_lag <- min(12 \* 2, (nrow(nest\_ts$data[[1]])) / 5)  
seasonal\_lag2 <- min(12 \* 2, (nrow(nest\_ts$data[[1]]) - 12) / 5)  
  
  
model\_ts %>%  
 glance() %>%  
 arrange(medicine, sex, AICc) %>%  
 as\_tibble() %>%  
 as\_grouped\_data(groups = c("medicine", "sex")) %>%  
 flextable() %>%  
 bold(j = 7)

| medicine | sex | .model | sigma2 | log\_lik | AIC | AICc | BIC | ar\_roots | ma\_roots |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Dexamfetamine |  |  |  |  |  |  |  | [[NULL]] | [[NULL]] |
|  | Female |  |  |  |  |  |  | [[NULL]] | [[NULL]] |
|  |  | seasonal\_arma\_log | 0.0014757859 | 106.77612 | -199.55224 | **-197.39839** | -184.891823 | [[complex]] | [[complex]] |
| Dexamfetamine |  |  |  |  |  |  |  | [[NULL]] | [[NULL]] |
|  | Male |  |  |  |  |  |  | [[NULL]] | [[NULL]] |
|  |  | seasonal\_arma\_log | 0.0006846959 | 128.22197 | -242.44395 | **-240.29010** | -227.783538 | [[complex]] | [[complex]] |
| Dexamfetamine |  |  |  |  |  |  |  | [[NULL]] | [[NULL]] |
|  | persons |  |  |  |  |  |  | [[NULL]] | [[NULL]] |
|  |  | seasonal\_arma\_log | 0.0005457453 | 135.66596 | -255.33191 | **-252.50838** | -238.577156 | [[complex]] | [[complex]] |
| Lisdexamfetamine |  |  |  |  |  |  |  | [[NULL]] | [[NULL]] |
|  | Female |  |  |  |  |  |  | [[NULL]] | [[NULL]] |
|  |  | seasonal\_arma\_log | 0.0029050872 | 80.85045 | -143.70090 | **-140.10090** | -124.851798 | [[complex]] | [[complex]] |
| Lisdexamfetamine |  |  |  |  |  |  |  | [[NULL]] | [[NULL]] |
|  | Male |  |  |  |  |  |  | [[NULL]] | [[NULL]] |
|  |  | seasonal\_arma\_log | 0.0052400562 | 67.43455 | -122.86911 | **-121.28420** | -110.303038 | [[complex]] | [[complex]] |
| Lisdexamfetamine |  |  |  |  |  |  |  | [[NULL]] | [[NULL]] |
|  | persons |  |  |  |  |  |  | [[NULL]] | [[NULL]] |
|  |  | seasonal\_arma\_log | 0.0029284721 | 81.34191 | -148.68381 | **-146.52997** | -134.023401 | [[complex]] | [[complex]] |
| Methylphenidate |  |  |  |  |  |  |  | [[NULL]] | [[NULL]] |
|  | Female |  |  |  |  |  |  | [[NULL]] | [[NULL]] |
|  |  | seasonal\_arma\_log | 0.0024685423 | 90.96455 | -167.92910 | **-165.77526** | -153.268691 | [[complex]] | [[complex]] |
| Methylphenidate |  |  |  |  |  |  |  | [[NULL]] | [[NULL]] |
|  | Male |  |  |  |  |  |  | [[NULL]] | [[NULL]] |
|  |  | seasonal\_arma\_log | 0.0013558488 | 111.22601 | -206.45203 | **-203.62850** | -189.697269 | [[complex]] | [[complex]] |
| Methylphenidate |  |  |  |  |  |  |  | [[NULL]] | [[NULL]] |
|  | persons |  |  |  |  |  |  | [[NULL]] | [[NULL]] |
|  |  | seasonal\_arma\_log | 0.0011628002 | 112.39897 | -210.79795 | **-208.64410** | -196.137537 | [[complex]] | [[complex]] |
| Nonstimulant |  |  |  |  |  |  |  | [[NULL]] | [[NULL]] |
|  | Female |  |  |  |  |  |  | [[NULL]] | [[NULL]] |
|  |  | seasonal\_arma\_log | 0.0279858038 | 23.09396 | -28.18793 | **-24.58793** | -9.338828 | [[complex]] | [[complex]] |
| Nonstimulant |  |  |  |  |  |  |  | [[NULL]] | [[NULL]] |
|  | Male |  |  |  |  |  |  | [[NULL]] | [[NULL]] |
|  |  | seasonal\_arma\_log | 0.0096945808 | 50.94831 | -87.89661 | **-85.74277** | -73.236201 | [[complex]] | [[complex]] |
| Nonstimulant |  |  |  |  |  |  |  | [[NULL]] | [[NULL]] |
|  | persons |  |  |  |  |  |  | [[NULL]] | [[NULL]] |
|  |  | seasonal\_arma\_log | 0.0085324246 | 55.09343 | -94.18687 | **-91.36334** | -77.432110 | [[complex]] | [[complex]] |

# we expect the residuals to fail this test when we have sufficient data  
map\_data <- nest\_ts %>%  
 select(medicine, sex) %>%  
 as.data.frame()  
  
# function to create a tibble containing information on one model for each stratification (sex and medicine)  
filter\_fun <- function(mable\_data, model\_var) {  
 tibble(  
 # including columns sex and medicine  
 medicine = map\_data[[1]],  
 sex = map\_data[[2]],  
 # extracting individual model and placing one mable in each row of tibble (essential for plotting residuals ect)  
 models = map2(map\_data[[1]], map\_data[[2]], ~ mable\_data %>%  
 filter(medicine == .x, sex == .y) %>%  
 select(one\_of(model\_var, vars = model\_var))),  
 ## tidying the models  
 tidy = map(models, tidy),  
 # store residual plots  
 gg\_residual = pmap(list(models, sex, medicine), ~ ..1 %>% gg\_tsresiduals(type = "innovation", lag\_max = 24) + labs(title = paste0("Residual plot for", model\_var, ..2, ..3, sep = " "))),  
 # report summary for model (will automatically print calling function)  
 report = walk(models, report),  
 # augment based on broom  
 augment = map(models, augment),  
 # caculate number of parameters to define degree of freedom  
 para = map(tidy, nrow),  
 # ljungbox test -need to unnest test when extracting these for use  
 lbjung = map2\_dfr(augment, para, ~ .x %>%  
 features(.innov, ljung\_box, lag = seasonal\_lag, dof = .y)),  
 test2 = map2\_dfr(augment, para, ~ .x %>%  
 features(.innov, ljung\_box, lag = seasonal\_lag2, dof = .y)),  
 # extract model specs  
 model\_name = map\_chr(report, ~ .x %>%  
 pull() %>%  
 as.character())  
 # ,components = map(model, ~ .x %>% components() %>% autoplot() )  
 )  
}

#######################################################################################  
## PLOT AND PRINT INFO ABOUT ARMA MODELS INTO PDF  
## DEFINE FUNCTIONS TO EXTRACT RESIDUAL PLOTS AND PRINT TABLE FOR TEST BY MEDICINE AND SEX  
#######################################################################################  
  
# plot residuals for models  
# gg\_tsresidual may only work on one single mable.  
  
  
models <- c("seasonal\_arma\_log")  
  
  
for (model in models) {  
 # plot function  
 p <- filter\_fun(model\_ts, model)  
  
 # create flextable object for residual test table  
 ft <- p %>%  
 select(medicine, sex, model\_name, contains("test2")) %>%  
 unnest(test2) %>%  
 arrange(sex) %>%  
 as\_grouped\_data(groups = c(".model", "sex")) %>%  
 flextable() %>%  
 bold(i = ~ lb\_pvalue < 0.05, 6) %>% #  
 font(fontname = "sans", part = "all")  
  
 pdf(file = paste0(Sys.Date(), "Residual plot for", model, "model by medicine and sex.pdf"))  
  
 for (i in 1:12) {  
 plot.new()  
 print(p[["gg\_residual"]][[i]])  
 }  
 plot(ft, method = "grob")  
 dev.off()  
}



## Series: rate   
## Model: LM w/ ARIMA(1,0,1)(0,1,1)[12] errors   
## Transformation: log(rate)   
##   
## Coefficients:  
## ar1 ma1 sma1 step ramp trend()  
## 0.9723 -0.4895 -0.7029 -0.0100 0.0099 0.0072  
## s.e. 0.0362 0.1236 0.2270 0.0356 0.0065 0.0022  
##   
## sigma^2 estimated as 0.001476: log likelihood=106.78  
## AIC=-199.55 AICc=-197.4 BIC=-184.89  
## Series: rate   
## Model: LM w/ ARIMA(1,0,1)(0,1,1)[12] errors   
## Transformation: log(rate)   
##   
## Coefficients:  
## ar1 ma1 sma1 step ramp trend()  
## 0.9571 -0.5816 -0.7903 -0.0129 0.0086 0.0053  
## s.e. 0.0673 0.1358 0.4782 0.0248 0.0037 0.0009  
##   
## sigma^2 estimated as 0.0006847: log likelihood=128.22  
## AIC=-242.44 AICc=-240.29 BIC=-227.78  
## Series: rate   
## Model: LM w/ ARIMA(1,0,1)(2,1,0)[12] errors   
## Transformation: log(rate)   
##   
## Coefficients:  
## ar1 ma1 sar1 sar2 step ramp trend()  
## 0.9735 -0.3548 -0.2854 -0.5383 -0.0293 0.0063 0.0062  
## s.e. 0.0358 0.1495 0.1597 0.1343 0.0255 0.0047 0.0025  
##   
## sigma^2 estimated as 0.0005457: log likelihood=135.67  
## AIC=-255.33 AICc=-252.51 BIC=-238.58  
## Series: rate   
## Model: LM w/ ARIMA(2,0,1)(1,1,1)[12] errors   
## Transformation: log(rate)   
##   
## Coefficients:  
## ar1 ar2 ma1 sar1 sma1 step ramp trend()  
## -0.4883 0.4951 0.7643 -0.6423 -0.7092 0.3595 0.0677 0.0335  
## s.e. 0.1827 0.1525 0.1291 0.1388 0.4988 0.0656 0.0082 0.0005  
##   
## sigma^2 estimated as 0.002905: log likelihood=80.85  
## AIC=-143.7 AICc=-140.1 BIC=-124.85  
## Series: rate   
## Model: LM w/ ARIMA(0,0,0)(2,1,0)[12] errors   
## Transformation: log(rate)   
##   
## Coefficients:  
## sar1 sar2 step ramp trend()  
## -0.7873 -0.5312 0.2540 0.0360 0.0293  
## s.e. 0.1426 0.1388 0.0459 0.0061 0.0005  
##   
## sigma^2 estimated as 0.00524: log likelihood=67.43  
## AIC=-122.87 AICc=-121.28 BIC=-110.3  
## Series: rate   
## Model: LM w/ ARIMA(1,0,0)(1,1,1)[12] errors   
## Transformation: log(rate)   
##   
## Coefficients:  
## ar1 sar1 sma1 step ramp trend()  
## 0.3018 -0.4142 -0.7753 0.3160 0.0484 0.0309  
## s.e. 0.1370 0.2173 0.5468 0.0572 0.0074 0.0005  
##   
## sigma^2 estimated as 0.002928: log likelihood=81.34  
## AIC=-148.68 AICc=-146.53 BIC=-134.02  
## Series: rate   
## Model: LM w/ ARIMA(2,0,0)(0,1,1)[12] errors   
## Transformation: log(rate)   
##   
## Coefficients:  
## ar1 ar2 sma1 step ramp trend()  
## 0.0794 0.3284 -0.7558 0.0429 0.0131 0.0091  
## s.e. 0.1241 0.1399 0.3205 0.0409 0.0051 0.0006  
##   
## sigma^2 estimated as 0.002469: log likelihood=90.96  
## AIC=-167.93 AICc=-165.78 BIC=-153.27  
## Series: rate   
## Model: LM w/ ARIMA(1,0,1)(2,1,0)[12] errors   
## Transformation: log(rate)   
##   
## Coefficients:  
## ar1 ma1 sar1 sar2 step ramp trend()  
## 0.9644 -0.6940 -0.6146 -0.3177 -0.0047 -0.0011 0.0082  
## s.e. 0.0415 0.0914 0.1484 0.1568 0.0305 0.0042 0.0014  
##   
## sigma^2 estimated as 0.001356: log likelihood=111.23  
## AIC=-206.45 AICc=-203.63 BIC=-189.7  
## Series: rate   
## Model: LM w/ ARIMA(2,0,0)(0,1,1)[12] errors   
## Transformation: log(rate)   
##   
## Coefficients:  
## ar1 ar2 sma1 step ramp trend()  
## 0.2751 0.2906 -0.7946 0.0163 0.0053 0.0085  
## s.e. 0.1306 0.1497 0.4040 0.0356 0.0041 0.0005  
##   
## sigma^2 estimated as 0.001163: log likelihood=112.4  
## AIC=-210.8 AICc=-208.64 BIC=-196.14  
## Series: rate   
## Model: LM w/ ARIMA(1,0,3)(0,1,1)[12] errors   
## Transformation: log(rate)   
##   
## Coefficients:  
## ar1 ma1 ma2 ma3 sma1 step ramp trend()  
## 0.7495 -0.6475 0.0642 0.3752 -0.4825 0.0013 0.0042 0.0270  
## s.e. 0.1269 0.1655 0.1408 0.1480 0.1570 0.1419 0.0190 0.0037  
##   
## sigma^2 estimated as 0.02799: log likelihood=23.09  
## AIC=-28.19 AICc=-24.59 BIC=-9.34  
## Series: rate   
## Model: LM w/ ARIMA(1,0,0)(2,1,0)[12] errors   
## Transformation: log(rate)   
##   
## Coefficients:  
## ar1 sar1 sar2 step ramp trend()  
## 0.5012 -0.7157 -0.4530 0.1322 -0.0106 0.0205  
## s.e. 0.1227 0.1440 0.1404 0.0947 0.0121 0.0013  
##   
## sigma^2 estimated as 0.009695: log likelihood=50.95  
## AIC=-87.9 AICc=-85.74 BIC=-73.24  
## Series: rate   
## Model: LM w/ ARIMA(1,0,1)(2,1,0)[12] errors   
## Transformation: log(rate)   
##   
## Coefficients:  
## ar1 ma1 sar1 sar2 step ramp trend()  
## 0.9532 -0.6194 -0.7904 -0.4366 0.0585 -0.007 0.0226  
## s.e. 0.0587 0.1289 0.1473 0.1544 0.0808 0.012 0.0033  
##   
## sigma^2 estimated as 0.008532: log likelihood=55.09  
## AIC=-94.19 AICc=-91.36 BIC=-77.43

## Warning: Removed 1 rows containing missing values (`geom\_point()`).

## Warning: Removed 1 rows containing non-finite values (`stat\_bin()`).

## Warning: Removed 1 rows containing missing values (`geom\_point()`).

## Warning: Removed 1 rows containing non-finite values (`stat\_bin()`).

## Warning: Removed 1 rows containing missing values (`geom\_point()`).

## Warning: Removed 1 rows containing non-finite values (`stat\_bin()`).

## Warning: Removed 1 rows containing missing values (`geom\_point()`).

## Warning: Removed 1 rows containing non-finite values (`stat\_bin()`).

## Warning: Removed 1 rows containing missing values (`geom\_point()`).

## Warning: Removed 1 rows containing non-finite values (`stat\_bin()`).

## Warning: Removed 1 rows containing missing values (`geom\_point()`).

## Warning: Removed 1 rows containing non-finite values (`stat\_bin()`).

## Warning: Removed 1 rows containing missing values (`geom\_point()`).

## Warning: Removed 1 rows containing non-finite values (`stat\_bin()`).

## Warning: Removed 1 rows containing missing values (`geom\_point()`).

## Warning: Removed 1 rows containing non-finite values (`stat\_bin()`).

## Warning: Removed 1 rows containing missing values (`geom\_point()`).

## Warning: Removed 1 rows containing non-finite values (`stat\_bin()`).

## Warning: Removed 1 rows containing missing values (`geom\_point()`).

## Warning: Removed 1 rows containing non-finite values (`stat\_bin()`).

## Warning: Removed 1 rows containing missing values (`geom\_point()`).

## Warning: Removed 1 rows containing non-finite values (`stat\_bin()`).

## Warning: Removed 1 rows containing missing values (`geom\_point()`).

## Warning: Removed 1 rows containing non-finite values (`stat\_bin()`).



ft

| .model | sex | medicine | model\_name | lb\_stat | lb\_pvalue |
| --- | --- | --- | --- | --- | --- |
| seasonal\_arma\_log |  |  |  |  |  |
|  | Female |  |  |  |  |
|  |  | Dexamfetamine | <LM w/ ARIMA(1,0,1)(0,1,1)[12] errors> | 6.487646 | 0.370833077 |
|  |  | Lisdexamfetamine | <LM w/ ARIMA(2,0,1)(1,1,1)[12] errors> | 14.052396 | **0.007129692** |
|  |  | Methylphenidate | <LM w/ ARIMA(2,0,0)(0,1,1)[12] errors> | 7.442301 | 0.281871141 |
|  |  | Nonstimulant | <LM w/ ARIMA(1,0,3)(0,1,1)[12] errors> | 6.031624 | 0.196799007 |
| seasonal\_arma\_log |  |  |  |  |  |
|  | Male |  |  |  |  |
|  |  | Dexamfetamine | <LM w/ ARIMA(1,0,1)(0,1,1)[12] errors> | 6.109250 | 0.411063857 |
|  |  | Lisdexamfetamine | <LM w/ ARIMA(0,0,0)(2,1,0)[12] errors> | 6.241277 | 0.511876292 |
|  |  | Methylphenidate | <LM w/ ARIMA(1,0,1)(2,1,0)[12] errors> | 7.800682 | 0.167567952 |
|  |  | Nonstimulant | <LM w/ ARIMA(1,0,0)(2,1,0)[12] errors> | 21.915640 | **0.001254238** |
| seasonal\_arma\_log |  |  |  |  |  |
|  | persons |  |  |  |  |
|  |  | Dexamfetamine | <LM w/ ARIMA(1,0,1)(2,1,0)[12] errors> | 6.115703 | 0.295122790 |
|  |  | Lisdexamfetamine | <LM w/ ARIMA(1,0,0)(1,1,1)[12] errors> | 5.192175 | 0.519412215 |
|  |  | Methylphenidate | <LM w/ ARIMA(2,0,0)(0,1,1)[12] errors> | 13.639287 | **0.033935596** |
|  |  | Nonstimulant | <LM w/ ARIMA(1,0,1)(2,1,0)[12] errors> | 8.867286 | 0.114476197 |

### remove comments with ## if want to loop and test multiple models, and replace . Some reason graphics card on this version of R is making the fonts play up and cause error  
  
# comment this out if want multiple models - note that plotting models depends on model specified.  
model <- "seasonal\_arma\_log"  
  
 models <- c("seasonal\_arma\_log")  
  
for (model in models) {  
  
  
# plot function  
p <- filter\_fun(model\_ts, model)  
  
# create flextable object for residual test table  
tidy\_estimates <- p %>%  
 select(medicine, sex, tidy, test2) %>%  
 unnest(c(tidy)) %>%  
 select(medicine, sex, term, estimate, std.error, .model) %>%  
 mutate(  
 change = ifelse(grepl("log", .model) == TRUE & (term == "step" | term == "ramp" | term == "trend()"),  
 # log model exponeniate if not just round:  
 round(100 \* (exp(estimate) - 1), digits = 2), round(estimate, digits = 2)  
 ),  
 lwr.ci = ifelse(grepl("log", .model) == TRUE & (term == "step" | term == "ramp" | term == "trend()"),  
 round(100 \* (exp(estimate - 1.96 \* std.error) - 1), digits = 2), round(estimate - 1.96 \* std.error, digits = 2)  
 ),  
 upr.ci = ifelse(grepl("log", .model) == TRUE & (term == "step" | term == "ramp" | term == "trend()"),  
 round(100 \* (exp(estimate + 1.96 \* std.error) - 1), digits = 2), round(estimate + 1.96 \* std.error, digits = 2)  
 )  
 ) %>%  
 filter(  
 term == "step" | term == "ramp" | term == "trend()"  
 # | term =="intercept"  
 ) %>%  
 mutate(  
 term = ifelse(term == "trend()", "trend", term)  
 ) %>%  
 select(-std.error)  
  
  
tidy\_estimates$medicine <- factor(tidy\_estimates$medicine, levels = c("Lisdexamfetamine", "Dexamfetamine", "Methylphenidate", "Nonstimulant"))  
tidy\_estimates$term <- factor(tidy\_estimates$term, levels = c("step", "ramp", "trend"))  
  
  
specs <- tidyr::build\_wider\_spec(tidy\_estimates,  
 names\_from = term,  
 values\_from = c(change, lwr.ci, upr.ci),  
 names\_glue = "{term}\_{.value}"  
) %>%  
 arrange(term, .value)  
  
  
ft <- pivot\_wider\_spec(tidy\_estimates %>% select(-estimate, -.model), specs) %>%  
 arrange(sex, medicine) %>%  
 as\_grouped\_data(groups = c("sex")) %>%  
 unite("step\_ci", c("step\_lwr.ci", "step\_upr.ci"), sep = " to ") %>%  
 unite("ramp\_ci", c("ramp\_lwr.ci", "ramp\_upr.ci"), sep = " to ") %>%  
 unite("trend\_ci", c("trend\_lwr.ci", "trend\_upr.ci"), sep = " to ") %>%  
 flextable() %>%  
 flextable::font(fontname = "sans", part = "all") %>%  
 set\_header\_labels(sex = "Sex", medicine = "Medicine") %>%  
 separate\_header(split = "[\_]", opts = c("center-hspan")) %>%  
 autofit()  
  
  
fit\_specs <- p %>%  
 select(medicine, sex, tidy, test2) %>%  
 unnest(c(tidy)) %>%  
 select(medicine, sex, term, estimate, .model) %>%  
 pivot\_wider(names\_from = term, values\_from = estimate) %>%  
 rename(r = ramp, t = `trend()`, s = step)  
  
df <- merge(ts, fit\_specs, by = c("medicine", "sex")) %>% mutate(time = date - yearmonth(min(date)))  
  
mean\_data <- nest\_ts %>%  
 mutate(  
 data = map(data, ~ .x %>%  
 as\_tibble() %>%  
 select(date, rate, cnt, ramp, step) %>%  
 mutate(time = date - yearmonth(min(date)))),  
  
 # baseline for seasonal differences  
 data\_mean = map\_dbl(data, ~ .x %>%  
 filter(time %in% c(0:11)) %>%  
 mutate(mean = mean(log(rate))) %>%  
 filter(time == 1) %>%  
 select(mean) %>%  
 pull())  
 ) %>%  
 select(data\_mean)  
  
  
df <- merge(df, mean\_data, by = c("medicine", "sex")) %>%  
 rename(ss = data\_mean) %>%  
 mutate(  
 step\_value = s \* step,  
 ramp\_value = r \* ramp,  
 # to account for seasonal differencing trend  
 trend\_value = t \* (time - 6)  
 ) %>%  
 mutate(  
 fit\_un = rowSums(select(., .dots = all\_of(c("step\_value", "ramp\_value", "trend\_value", "ss"))), na.rm = TRUE),  
 # exponeniating if needed i.e., model is log (all models?)  
 fit = ifelse(grepl("log", .model) == TRUE, exp(fit\_un), fit\_un),  
 class = factor(ifelse(medicine != "Lisdexamfetamine", "Other", "Lisdexamfetamine"))  
 ) %>%  
 mutate\_at(1:2, factor) %>%  
 ### creating a break in the trend line to improve readability of graph:  
 # based on https://stackoverflow.com/questions/14821064/line-break-when-no-data-in-ggplot2  
  
 mutate(fit = ifelse(date == yearmonth(change\_date), NA\_real\_, fit))  
  
  
plot <- df %>%  
 filter(sex != "persons") %>%  
 ggplot(aes(x = date, colour = medicine)) +  
 geom\_point(aes(x = date, y = rate), size = 1) +  
 geom\_line(aes(y = fit), alpha = .80, size = 1) +  
 facet\_grid(class ~ sex) +  
 scale\_x\_yearmonth(date\_labels = "%b %y", date\_breaks = "1 year") +  
 xlab("Date") +  
 ylab("Monthly PBS dispensings per 1000 adults") +  
 scale\_y\_continuous(limits = c(0, 2)) +  
 geom\_vline(xintercept = as.Date(change\_date), linetype = 4, size = 1.1, colour = "black") +  
 guides(colour = guide\_legend(title = NULL)) +  
 theme\_bw() +  
 theme(  
 # text = element\_text(family = "sans", size = 12),  
 #axis.title.x = element\_text(size = 12),  
 #axis.title.y = element\_text(size = 12),  
 # axis.text.x = element\_text(angle = 90, vjust = 0.5, hjust=1),  
 panel.grid.major.y = element\_line(colour = "grey90"),  
 panel.grid.major.x = element\_line(colour = "grey90"),  
 panel.grid.minor.y = element\_blank(),  
 panel.grid.minor.x = element\_blank(),  
 # plot.title = element\_text(  
 # size = 12, colour = "black",  
 # face = "plain", margin = unit(c(1, 1, 1, 1), "pt"),  
 # lineheight = 1, angle = NULL  
 # ),  
 legend.position = "bottom",  
 panel.border = element\_rect(colour = "darkgrey", fill = NA)  
 # ,line = element\_blank()  
 , axis.line = element\_line(color = "black")  
 )  
  
ggsave(  
 filename = paste0(Sys.Date(), "\_Figure 1. predicted and observed for", model, "model by medicine and sex.tiff"),  
 plot = plot,  
 device = NULL,  
 scale = 1,  
 # width = 4000,  
 # height = 3000,  
 units = "px",  
 dpi = 300,  
 limitsize = FALSE,  
)  
  
#pdf(file = paste0(Sys.Date(), "\_Figure 1. predicted and observed for", model, "model by medicine and sex.pdf"))  
  
plot.new()  
print(plot)  
  
 #plot(ft, method = "grob")  
 #dev.off()  
 }



## Series: rate   
## Model: LM w/ ARIMA(1,0,1)(0,1,1)[12] errors   
## Transformation: log(rate)   
##   
## Coefficients:  
## ar1 ma1 sma1 step ramp trend()  
## 0.9723 -0.4895 -0.7029 -0.0100 0.0099 0.0072  
## s.e. 0.0362 0.1236 0.2270 0.0356 0.0065 0.0022  
##   
## sigma^2 estimated as 0.001476: log likelihood=106.78  
## AIC=-199.55 AICc=-197.4 BIC=-184.89  
## Series: rate   
## Model: LM w/ ARIMA(1,0,1)(0,1,1)[12] errors   
## Transformation: log(rate)   
##   
## Coefficients:  
## ar1 ma1 sma1 step ramp trend()  
## 0.9571 -0.5816 -0.7903 -0.0129 0.0086 0.0053  
## s.e. 0.0673 0.1358 0.4782 0.0248 0.0037 0.0009  
##   
## sigma^2 estimated as 0.0006847: log likelihood=128.22  
## AIC=-242.44 AICc=-240.29 BIC=-227.78  
## Series: rate   
## Model: LM w/ ARIMA(1,0,1)(2,1,0)[12] errors   
## Transformation: log(rate)   
##   
## Coefficients:  
## ar1 ma1 sar1 sar2 step ramp trend()  
## 0.9735 -0.3548 -0.2854 -0.5383 -0.0293 0.0063 0.0062  
## s.e. 0.0358 0.1495 0.1597 0.1343 0.0255 0.0047 0.0025  
##   
## sigma^2 estimated as 0.0005457: log likelihood=135.67  
## AIC=-255.33 AICc=-252.51 BIC=-238.58  
## Series: rate   
## Model: LM w/ ARIMA(2,0,1)(1,1,1)[12] errors   
## Transformation: log(rate)   
##   
## Coefficients:  
## ar1 ar2 ma1 sar1 sma1 step ramp trend()  
## -0.4883 0.4951 0.7643 -0.6423 -0.7092 0.3595 0.0677 0.0335  
## s.e. 0.1827 0.1525 0.1291 0.1388 0.4988 0.0656 0.0082 0.0005  
##   
## sigma^2 estimated as 0.002905: log likelihood=80.85  
## AIC=-143.7 AICc=-140.1 BIC=-124.85  
## Series: rate   
## Model: LM w/ ARIMA(0,0,0)(2,1,0)[12] errors   
## Transformation: log(rate)   
##   
## Coefficients:  
## sar1 sar2 step ramp trend()  
## -0.7873 -0.5312 0.2540 0.0360 0.0293  
## s.e. 0.1426 0.1388 0.0459 0.0061 0.0005  
##   
## sigma^2 estimated as 0.00524: log likelihood=67.43  
## AIC=-122.87 AICc=-121.28 BIC=-110.3  
## Series: rate   
## Model: LM w/ ARIMA(1,0,0)(1,1,1)[12] errors   
## Transformation: log(rate)   
##   
## Coefficients:  
## ar1 sar1 sma1 step ramp trend()  
## 0.3018 -0.4142 -0.7753 0.3160 0.0484 0.0309  
## s.e. 0.1370 0.2173 0.5468 0.0572 0.0074 0.0005  
##   
## sigma^2 estimated as 0.002928: log likelihood=81.34  
## AIC=-148.68 AICc=-146.53 BIC=-134.02  
## Series: rate   
## Model: LM w/ ARIMA(2,0,0)(0,1,1)[12] errors   
## Transformation: log(rate)   
##   
## Coefficients:  
## ar1 ar2 sma1 step ramp trend()  
## 0.0794 0.3284 -0.7558 0.0429 0.0131 0.0091  
## s.e. 0.1241 0.1399 0.3205 0.0409 0.0051 0.0006  
##   
## sigma^2 estimated as 0.002469: log likelihood=90.96  
## AIC=-167.93 AICc=-165.78 BIC=-153.27  
## Series: rate   
## Model: LM w/ ARIMA(1,0,1)(2,1,0)[12] errors   
## Transformation: log(rate)   
##   
## Coefficients:  
## ar1 ma1 sar1 sar2 step ramp trend()  
## 0.9644 -0.6940 -0.6146 -0.3177 -0.0047 -0.0011 0.0082  
## s.e. 0.0415 0.0914 0.1484 0.1568 0.0305 0.0042 0.0014  
##   
## sigma^2 estimated as 0.001356: log likelihood=111.23  
## AIC=-206.45 AICc=-203.63 BIC=-189.7  
## Series: rate   
## Model: LM w/ ARIMA(2,0,0)(0,1,1)[12] errors   
## Transformation: log(rate)   
##   
## Coefficients:  
## ar1 ar2 sma1 step ramp trend()  
## 0.2751 0.2906 -0.7946 0.0163 0.0053 0.0085  
## s.e. 0.1306 0.1497 0.4040 0.0356 0.0041 0.0005  
##   
## sigma^2 estimated as 0.001163: log likelihood=112.4  
## AIC=-210.8 AICc=-208.64 BIC=-196.14  
## Series: rate   
## Model: LM w/ ARIMA(1,0,3)(0,1,1)[12] errors   
## Transformation: log(rate)   
##   
## Coefficients:  
## ar1 ma1 ma2 ma3 sma1 step ramp trend()  
## 0.7495 -0.6475 0.0642 0.3752 -0.4825 0.0013 0.0042 0.0270  
## s.e. 0.1269 0.1655 0.1408 0.1480 0.1570 0.1419 0.0190 0.0037  
##   
## sigma^2 estimated as 0.02799: log likelihood=23.09  
## AIC=-28.19 AICc=-24.59 BIC=-9.34  
## Series: rate   
## Model: LM w/ ARIMA(1,0,0)(2,1,0)[12] errors   
## Transformation: log(rate)   
##   
## Coefficients:  
## ar1 sar1 sar2 step ramp trend()  
## 0.5012 -0.7157 -0.4530 0.1322 -0.0106 0.0205  
## s.e. 0.1227 0.1440 0.1404 0.0947 0.0121 0.0013  
##   
## sigma^2 estimated as 0.009695: log likelihood=50.95  
## AIC=-87.9 AICc=-85.74 BIC=-73.24  
## Series: rate   
## Model: LM w/ ARIMA(1,0,1)(2,1,0)[12] errors   
## Transformation: log(rate)   
##   
## Coefficients:  
## ar1 ma1 sar1 sar2 step ramp trend()  
## 0.9532 -0.6194 -0.7904 -0.4366 0.0585 -0.007 0.0226  
## s.e. 0.0587 0.1289 0.1473 0.1544 0.0808 0.012 0.0033  
##   
## sigma^2 estimated as 0.008532: log likelihood=55.09  
## AIC=-94.19 AICc=-91.36 BIC=-77.43

## Adding missing grouping variables: `sex`, `medicine`

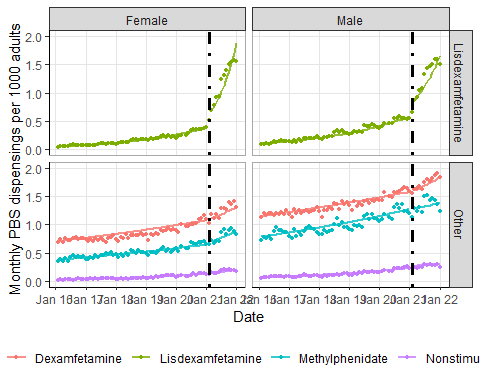
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.  
## ℹ Please use `linewidth` instead.

## Saving 1500 x 1200 px image

## Warning: Removed 8 rows containing missing values (`geom\_point()`).



## Warning: Removed 8 rows containing missing values (`geom\_point()`).



ft

| sex | medicine | step | step | ramp | ramp | trend | trend |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | change | ci | change | ci | change | ci |
| Female |  |  | NA to NA |  | NA to NA |  | NA to NA |
|  | Lisdexamfetamine | 43.26 | 25.99 to 62.91 | 7.00 | 5.3 to 8.73 | 3.41 | 3.3 to 3.52 |
|  | Dexamfetamine | -0.99 | -7.67 to 6.17 | 1.00 | -0.29 to 2.3 | 0.72 | 0.29 to 1.16 |
|  | Methylphenidate | 4.38 | -3.66 to 13.09 | 1.32 | 0.31 to 2.34 | 0.91 | 0.8 to 1.03 |
|  | Nonstimulant | 0.13 | -24.19 to 32.24 | 0.42 | -3.26 to 4.24 | 2.74 | 1.99 to 3.5 |
| Male |  |  | NA to NA |  | NA to NA |  | NA to NA |
|  | Lisdexamfetamine | 28.92 | 17.83 to 41.05 | 3.66 | 2.43 to 4.91 | 2.97 | 2.87 to 3.07 |
|  | Dexamfetamine | -1.29 | -5.97 to 3.64 | 0.86 | 0.14 to 1.59 | 0.53 | 0.35 to 0.71 |
|  | Methylphenidate | -0.47 | -6.25 to 5.67 | -0.11 | -0.93 to 0.72 | 0.82 | 0.54 to 1.1 |
|  | Nonstimulant | 14.14 | -5.2 to 37.42 | -1.06 | -3.38 to 1.33 | 2.07 | 1.82 to 2.33 |
| persons |  |  | NA to NA |  | NA to NA |  | NA to NA |
|  | Lisdexamfetamine | 37.17 | 22.62 to 53.45 | 4.95 | 3.43 to 6.5 | 3.14 | 3.04 to 3.23 |
|  | Dexamfetamine | -2.89 | -7.62 to 2.08 | 0.64 | -0.28 to 1.56 | 0.62 | 0.14 to 1.11 |
|  | Methylphenidate | 1.65 | -5.21 to 8.99 | 0.53 | -0.27 to 1.34 | 0.85 | 0.75 to 0.96 |
|  | Nonstimulant | 6.03 | -9.5 to 24.22 | -0.70 | -3.01 to 1.66 | 2.29 | 1.64 to 2.94 |

#based on data including April 2020 (removed from time series)  
mean\_data <- tss %>%  
 inner\_join(erpx) %>%  
 mutate(rate = (cnt \* 10000) / value, cnt = as.numeric(cnt \* 10)) %>%  
 select(date, medicine, sex, rate, cnt, ramp, step) %>%  
 group\_by(sex, medicine) %>%  
 nest() %>%   
 mutate(  
 data = map(data, ~ .x %>%  
 as\_tibble() %>%  
 select(date, rate, cnt, ramp, step) %>%  
 mutate(time = date - yearmonth(min(date)))),  
 # baseline for seasonal differences  
 mean\_before = map\_dbl(data, ~ .x %>%  
 filter(between(as.Date(date), as.Date("2020-02-01"), as.Date("2021-01-31"))) %>%  
 mutate(mean\_before = mean(rate)) %>%  
 filter(date == as.Date("2020-02-01")) %>%  
 select(mean\_before) %>%  
 pull()),  
 mean\_after = map\_dbl(data, ~ .x %>%  
 filter(between(as.Date(date), as.Date("2021-02-01"), as.Date("2022-01-31"))) %>%  
 mutate(mean\_after = mean(rate)) %>%  
 filter(date == as.Date("2021-02-01")) %>%  
 select(mean\_after) %>%  
 pull())  
 )

## Joining, by = c("date", "sex")

## Warning in mask$eval\_all\_filter(dots, env\_filter): Incompatible methods  
## ("==.vctrs\_vctr", "==.Date") for "=="  
  
## Warning in mask$eval\_all\_filter(dots, env\_filter): Incompatible methods  
## ("==.vctrs\_vctr", "==.Date") for "=="  
  
## Warning in mask$eval\_all\_filter(dots, env\_filter): Incompatible methods  
## ("==.vctrs\_vctr", "==.Date") for "=="  
  
## Warning in mask$eval\_all\_filter(dots, env\_filter): Incompatible methods  
## ("==.vctrs\_vctr", "==.Date") for "=="  
  
## Warning in mask$eval\_all\_filter(dots, env\_filter): Incompatible methods  
## ("==.vctrs\_vctr", "==.Date") for "=="  
  
## Warning in mask$eval\_all\_filter(dots, env\_filter): Incompatible methods  
## ("==.vctrs\_vctr", "==.Date") for "=="  
  
## Warning in mask$eval\_all\_filter(dots, env\_filter): Incompatible methods  
## ("==.vctrs\_vctr", "==.Date") for "=="  
  
## Warning in mask$eval\_all\_filter(dots, env\_filter): Incompatible methods  
## ("==.vctrs\_vctr", "==.Date") for "=="  
  
## Warning in mask$eval\_all\_filter(dots, env\_filter): Incompatible methods  
## ("==.vctrs\_vctr", "==.Date") for "=="  
  
## Warning in mask$eval\_all\_filter(dots, env\_filter): Incompatible methods  
## ("==.vctrs\_vctr", "==.Date") for "=="  
  
## Warning in mask$eval\_all\_filter(dots, env\_filter): Incompatible methods  
## ("==.vctrs\_vctr", "==.Date") for "=="  
  
## Warning in mask$eval\_all\_filter(dots, env\_filter): Incompatible methods  
## ("==.vctrs\_vctr", "==.Date") for "=="  
  
## Warning in mask$eval\_all\_filter(dots, env\_filter): Incompatible methods  
## ("==.vctrs\_vctr", "==.Date") for "=="  
  
## Warning in mask$eval\_all\_filter(dots, env\_filter): Incompatible methods  
## ("==.vctrs\_vctr", "==.Date") for "=="  
  
## Warning in mask$eval\_all\_filter(dots, env\_filter): Incompatible methods  
## ("==.vctrs\_vctr", "==.Date") for "=="  
  
## Warning in mask$eval\_all\_filter(dots, env\_filter): Incompatible methods  
## ("==.vctrs\_vctr", "==.Date") for "=="  
  
## Warning in mask$eval\_all\_filter(dots, env\_filter): Incompatible methods  
## ("==.vctrs\_vctr", "==.Date") for "=="  
  
## Warning in mask$eval\_all\_filter(dots, env\_filter): Incompatible methods  
## ("==.vctrs\_vctr", "==.Date") for "=="  
  
## Warning in mask$eval\_all\_filter(dots, env\_filter): Incompatible methods  
## ("==.vctrs\_vctr", "==.Date") for "=="  
  
## Warning in mask$eval\_all\_filter(dots, env\_filter): Incompatible methods  
## ("==.vctrs\_vctr", "==.Date") for "=="  
  
## Warning in mask$eval\_all\_filter(dots, env\_filter): Incompatible methods  
## ("==.vctrs\_vctr", "==.Date") for "=="  
  
## Warning in mask$eval\_all\_filter(dots, env\_filter): Incompatible methods  
## ("==.vctrs\_vctr", "==.Date") for "=="  
  
## Warning in mask$eval\_all\_filter(dots, env\_filter): Incompatible methods  
## ("==.vctrs\_vctr", "==.Date") for "=="  
  
## Warning in mask$eval\_all\_filter(dots, env\_filter): Incompatible methods  
## ("==.vctrs\_vctr", "==.Date") for "=="

mean\_data %>%  
 select(-data) %>%  
 flextable()

| medicine | sex | mean\_before | mean\_after |
| --- | --- | --- | --- |
| Dexamfetamine | Female | 1.0210154 | 1.2479840 |
| Lisdexamfetamine | Female | 0.3212176 | 1.1657456 |
| Methylphenidate | Female | 0.6380165 | 0.8020832 |
| Nonstimulant | Female | 0.1157204 | 0.1759380 |
| Dexamfetamine | Male | 1.5602236 | 1.7430624 |
| Lisdexamfetamine | Male | 0.5114558 | 1.2550287 |
| Methylphenidate | Male | 1.2138973 | 1.3478297 |
| Nonstimulant | Male | 0.2135375 | 0.2697974 |
| Dexamfetamine | persons | 1.2858220 | 1.4911494 |
| Lisdexamfetamine | persons | 0.4146441 | 1.2095984 |
| Methylphenidate | persons | 0.9208327 | 1.0701354 |
| Nonstimulant | persons | 0.1637587 | 0.2220385 |

beep()