# **MANUAL** | AMR data analysis

#### EUCIC Course, 11-12 September 2023, Groningen, the Netherlands

*Instructors: Dennis Souverein (infectious disease epidemiologist), Tjibbe Donker (mathematical modeller), Gerolf de Boer (mathematical modeller), Matthijs Berends (infectious disease epidemiologist)*

## Introduction

Welcome to the practical on Antimicrobial Resistance (AMR) data analysis in R! In this exciting and informative practical, we will delve into the world of microbiological data from two regional, non-profit, clinical microbiological laboratories in the Netherlands ([Certe](https://www.certe.nl/) and [Streeklab Haarlem](https://www.streeklabhaarlem.nl/)), that cover the diagnostics of over 2 million Dutch inhabitants, over 10% of the total Dutch population. Our journey will take us through hundreds of thousands of test results from both primary and secondary care settings.

AMR is a global public health concern, and its rise poses a significant threat to the effective treatment of infectious diseases. To combat this challenge, it is crucial to understand the regional trends of AMR species groups, antibiotic groups, and demographic patterns. This practical is designed to equip you with the skills and tools to gain deep insights into these trends, enabling you to contribute to the fight against AMR by being able to apply data analytical principles at your own institution after this course. If you are passionate about healthcare, epidemiology, data analysis, or public health, this is the perfect opportunity to combine your interests and make a real impact on an urgent global health issue.

The goals for this practical are:

1. **Learn Data Analysis in R**: R is a powerful and widely used language for data analysis and visualisation. Through hands-on exercises, we will guide you through the process of working with real-world AMR data and boosting your data analysis skills. You will be provided R syntaxes in this manual to help you getting started.
2. **Uncover Regional AMR Trends**: The data from different regional clinical microbiological laboratories offer a unique perspective on how AMR varies across locations. You will be able to identify hotspots, track emerging patterns, and understand how AMR affects different communities.
3. **Explore Antibiotic Resistance Patterns**: By analysing antibiotic groups and their susceptibility trends, you will gain insights into the effectiveness of different antibiotics and the prevalence of resistance.
4. **Study Demographic Influences**: Understanding how AMR affects different demographic groups is essential for targeted interventions and policies. We will explore demographic trends, allowing you to contribute to more effective public health strategies.
5. **Contribute to Public Health**: Your participation in this EUCIC course will empower you to contribute meaningfully to the global fight against AMR. Your newfound skills will be invaluable for healthcare professionals, policymakers, and researchers working to combat this pressing health crisis.

## Preparation

## Follow these simple steps to get your computer ready.

## Log on to the computer:

## Please log on using credentials provided to you.

## If you encounter any issues with logging on, don't hesitate to ask the instructors for help.

## Start RStudio:

## Once you're logged in, look for the RStudio icon on the desktop or locate it through the applications menu.

## Click on the RStudio icon to launch the RStudio Integrated Development Environment (IDE).

## Load the required libraries:

* + In RStudio, you need to load specific libraries that will enable us to perform data analysis efficiently. Today, we'll be using the 'tidyverse' and 'AMR' libraries. To load the libraries, type the following commands in the RStudio console and press Enter/Return:

library(tidyverse)  
library(AMR)

The 'tidyverse' package is a collection of essential R packages for data manipulation and visualisation. Find more info about the tidyverse here: <https://www.tidyverse.org>. The ‘AMR’ package contains reference data about all antibiotics and the full microbiological taxonomy and contains functions to simplify AMR data analysis. Find more info about the AMR package here: <https://msberends.github.io/AMR/>.

1. Download the AMR data:
   * Before we proceed, let's ensure that you have the AMR data ready for analysis.
   * We have prepared a dataset that comprises the regional clinical microbiological laboratories' AMR data, which we will use throughout the practical. You can get the dataset **by clicking here:** [**https://bit.ly/eucic2023isolates**](https://bit.ly/eucic2023isolates). Please save it to a convenient location on your computer. The data are not sensitive as they are anonymised.
2. Set the working directory:
   * To ensure that R can locate the dataset, set your working directory to the location where you saved the downloaded data. You can do this by clicking on 'Session' in the top menu bar, then selecting 'Set Working Directory' and 'Choose Directory.' Navigate to the folder where you saved the dataset and select it.
3. Create a new file:
   * To create a new file to work in, choose from the menu ‘File’, then ‘New File’, then ‘R Script’.
   * You should now have an interface similar to this:

A screenshot of a computer

Description automatically generated

Source Pane

Object Overview

File Overview

Console

Great! You are now all set to embark on your AMR data analysis journey. Our instructors will guide you through each step, explaining concepts, and helping you gain valuable insights from the data. If you encounter any issues or have any questions during the practical, feel free to ask for assistance. Happy analysing!

NOTE: files will remain available (for a while at least), so this practical can be finished or redone at your own institute.

## Assignments

Let's introduce the assignments for our AMR data analysis practical. We will provide guidance and explain the objectives of each assignment to ensure everyone is on the right track. **Remember**, the practical is designed to be hands-on and engaging. Don't hesitate to ask questions and seek guidance from our instructors if you encounter any challenges.

### Assignment #1: Nitrofurantoin Resistance in Urine Culture Isolates from Primary Care Over Time

#### Objective

In this assignment, we will explore the trend of nitrofurantoin resistance in urine culture isolates obtained from patients visiting primary care clinics. We want to understand how the resistance to nitrofurantoin antibiotics has evolved over time and if there are any significant patterns or changes.

#### Guidance

The guidance on this first assignment will be rather extensive to get you started. Advanced R users are advised to take on the additional tasks as pointed out.

1. Load the AMR dataset into RStudio and familiarise yourself with its structure and some basic functions. Below example script can be pasted as a whole into the Source Pane:

# Load the AMR dataset into RStudio using the read\_rds function.

isolates <- read\_rds("isolates\_def.rds")

# To have a look at the data, simply type its name:

isolates

# Or, you can create a summary of the data to get an overview:

summary(isolates)

# Using select, we can choose specific columns of interest.

# For example, to select columns "gender" and "age" from the

# 'isolates' dataset:

select(isolates, gender, age)

# Or select columns where the data type is 'SIR', which are

# antibiotic test results:

select(isolates, where(is.sir))

# To create a summary out of that, you *could* do:

summary(select(isolates, where(is.sir)))

# But as you can see, this becomes quite cluttered and hard to read.

# We can use the pipe operator (|>) to make the code more readable.

# It allows us to pass the left-hand object to the right-hand side

# of the expression. Instead of using intermediate variables, we

# can chain commands together!

# This example below will do exactly the same thing as the previous

# command:

isolates |> select(where(is.sir)) |> summary()

# So, now you can read it from left to right:

# “take the isolates, select SIR columns, create a summary”

1. Identify the relevant columns that contain information about nitro resistance and urine culture isolates. Use select() to select columns, such as the date, specimen and nitrofurantoin results. Use filter() to filter out irrelevant rows. For filtering, use the mathematical operator == for “is”/“equal to”, and != for “is not”/“not equal to”.

# Replace the dots with the relevant value

isolates |>

filter(material == "...")

# The specimen group is not the only filter we require. Have a look

# at the assignment again, and add a second filter to the

# command above. More filter can be added by using the comma.

isolates |>

filter(material == "...",

... == "...")

1. Group the data by time intervals to create a time series. *Advanced* *R user? Then create intervals per quarter and summarise on the interval per specimen and type of care.* New variables can be made with mutate(), groups can be made with group\_by(), and year() will transform the data column to years. Use <- to save the code to a new object.

per\_year <- isolates |>

filter(material == "...",

... == "...") |>

# This will create a new column ‘year’:

mutate(year = year(date)) |>

# This will group the data on years:

group\_by(year)

# Now use select() to keep the relevant columns

per\_year <- per\_year |>

????

# After selecting, have a look at the new data:

per\_year

per\_year |> summary()

1. Calculate the proportion of nitro-resistant isolates in each time interval. We can use the summarise() function to calculate any metric per group. The ‘AMR’ function resistance() can be used to determine the resistance per group. It will take the nitrofurantoin column as input, similar to min() taking the date column as input below.

per\_year |>

summarise(lowest\_date = min(date),

highest\_date = max(date),

median\_date = median(date))

# Fill in the dots here

per\_year |>

summarise(nitro\_R = ...)

# Now run the whole command to retrieve the nitro AMR per group.

# Replace xxx with a function name and the dots with column names.

per\_year <- isolates |>

filter(..., ...) |>

group\_by(year = xxx(...), ... , ...) |>

summarise(nitro\_R = xxx(...))

# Observe the newly created data

per\_year

# Advanced users: also use sir\_confidence\_interval() to add

# the confidence intervals of nitro resistance. Read the AMR

# documentation on how to use it.

1. Visualise the trend using line plots or other suitable visualizations to understand the changes in nitro resistance over time. We will use the ‘ggplot2’ package (part of the tidyverse) to create appealing plots. *Advanced R user? Then try to create a plot with confidence intervals using error bars.*

# The ggplot() function will create a plot object, without

# any specifics:

ggplot(per\_year)

# Since we have the numeric data already, we can add those as

# so-called aesthetics using aes()

ggplot(per\_year, aes(y = nitro\_R))

# Add the years as x to the aesthetics.

# You will now notice that the plot object has dimensions. We

# now need to add another layer, called a ‘geom’, using +.

# Since we want a line plot, the geom of choice will be geom\_line().

ggplot(per\_year, aes(x = , y = )) +

geom\_line()

# You can create colours and even complete themes to fit the

# style of your institution. This will create a EUCIC plot!

ggplot(per\_year,

aes(x = as.character(year),

group = 1, # weird but needed

y = nitro\_R)) +

# set line colour and width

geom\_line(colour = "#4f8b8b", linewidth = 2) +

# add another layer with points

geom\_point(colour = "#a0cccd", size = 4) +

# set the y axis as percentages

scale\_y\_continuous(labels = scales::percent) +

# set labels with labs()

labs(title = "Nitrofurantoin Resistance Over The Years",

x = "Years",

y = "% Nitro Resistance",

caption = "In a EUCIC style!") +

# set theme with theme\_bw (black/white) and theme()

theme\_bw(base\_family = "Verdana") +

theme(plot.title = element\_text(hjust = 0.5),

plot.caption = element\_text(size = "11px",

colour = "#4f8b8b",

hjust = 1,

angle = 20,

face = "bold"))

A graph showing the growth of the stock market

Description automatically generated

Before applying themes, colours and labels. Notice how the y axis labels are decimal proportion instead of percentages, and how the x axis contains in-between lines on strange indices (i.e., years 2007.5 and 2012.5 are strange places for a vertical line).

A graph with blue lines and numbers

Description automatically generated

After applying themes, colours and labels.

1. Interpret the results and discuss any observations or trends you may discover. How do you now look at the AMR of nitrofurantoin? Can you create a single command to also do this for ciprofloxacine? Or trimethoprim?

### Assignment #2: Overestimation of Resistance Without Correcting for the First Isolate in a Heatmap

#### Objective

In this assignment, we will investigate the potential overestimation of antibiotic resistance when not accounting for the first isolate of a patient. To conduct epidemiological analyses on antimicrobial resistance data, only so-called first isolates should be included to prevent overestimation and underestimation of antimicrobial resistance, and a certain minimum number of isolates should be required when reporting AMR. Four different methods can be used to determine first isolates: isolate-based, patient-based, episode-based and phenotype-based (please refer to the course slides). In this assignment, you will determine the number of first isolates for all those four algorithms.

#### Guidance

1. Create the four new columns that displays the selection of isolates grouped by material and request group.

isolates\_2 <- isolates %>%

# grouping variables for which first isolates need to be calculated

group\_by(material, …) %>%

mutate(phenotype = first\_isolate(col\_date = "sampling\_date",

col\_patient\_id = "id",

method = "…")) %>%

mutate(episode = first\_isolate(col\_date = "sampling\_date",

col\_patient\_id = "id",

method = "…")) %>%

mutate(patient = first\_isolate(col\_date = "sampling\_date",

col\_patient\_id = "id",

method = "…")) %>%

mutate(isolate = first\_isolate(col\_date = "sampling\_date",

col\_patient\_id = "id",

method = "…")) %>%

# create variable to aggregate results per year

mutate(Year = year(sampling\_date)) %>%

arrange(id, sampling\_date)

1. Determine for each method the AMR percentages for E. coli isolated from urine requested by the general practitioner per year for “ciprofloxacin", “amoxicillin\_clavulanic\_acid", “nitrofurantoin”, “fosfomycin” and and “trimethoprim”. Build a heatmap visualizing resistance estimates.

**sum\_isolates\_2\_1 <- isolates\_2 %>%**

# filter the dataset to answer the question above

**filter(material == "…",**

**request\_group == "…",**

**genus\_species == "…") %>%**

# select the variables needed for analysis

**select(id, Year, material, request\_group, genus\_species,**

**ciprofloxacin, amoxicillin\_clavulanic\_acid,**

**nitrofurantoin, fosfomycin, trimethoprim, phenotype, episode,**

**patient, isolate)**

**sum\_isolates\_2\_2 <- sum\_isolates\_2\_1 %>%**

# make the dataset longer in order to calculate the resistance

# percentages per group in one go. Otherwise, you need to calculate

# the resistance rates per method 4 times

**pivot\_longer(phenotype:isolate,**

**names\_to = "method",**

**values\_to = "value") %>%**

**filter(value == TRUE) %>%**

**group\_by(Year, method) %>%**

**summarise(**

**amoxicillin\_clavulanic\_acid =**

**resistance(amoxicillin\_clavulanic\_acid),**

**ciprofloxacin = resistance(ciprofloxacin),**

**nitrofurantoin = resistance(nitrofurantoin),**

**fosfomycin = resistance(fosfomycin),**

**trimethoprim = resistance(trimethoprim))**

**sum\_isolates\_2\_3 <- sum\_isolates\_2\_2 %>%**

# for visualisation purposes we need tidy data. That means one column

# with the antibiotics and one column with al resistance percentages

**pivot\_longer(amoxicillin\_clavulanic\_acid:trimethoprim,**

**names\_to = "antibiotic",**

**values\_to = "resistance") %>%**

**mutate(resistance = round(resistance \* 100, 1))**

# creating a heatmap with the resistance percentages

**sum\_isolates\_2\_3 %>%**

**ggplot(aes(x = as.character(Year), y = method, fill= resistance)) +**

# geom\_tile creates the heatmap

**geom\_tile() +**

**scale\_fill\_viridis\_c(option = "A", direction = -1) +**

# make a different heatmap for each antibiotic

**facet\_grid(. ~ antibiotic) +**

# styling of the text and adding titles to the chart

**theme(axis.text.x = element\_text(angle = 45, hjust = 1)) +**

**labs(title = "Resistance estimates per method",**

**subtitle = "Material = Urine, species = E. coli",**

**caption = "Data source: EUCIC",**

**x = NULL, y = NULL, fill = "Resistance") +**

# fill each square with the resistance number

**geom\_text(aes(label = glue("{resistance}%")),**

**colour = "darkgreen", size = 2.5)**



1. Compare the results and calculate the percentage difference in resistance estimates to quantify the overestimation compared to the “isolate-based” method. Build a heatmap visualizing the differences.

# use the dataset of the previous question and calculate the

# difference to the isolate-based method

**sum\_isolates\_2\_4 <- sum\_isolates\_2\_3 %>%**

**group\_by(Year, antibiotic) %>%**

# adding a column with the resistance percentages from the isolate

# method in order to compare

**mutate(isolate\_res = case\_when(**

**method == "isolate" ~ resistance)) %>%**

**arrange(Year, antibiotic) %>%**

**fill(isolate\_res, .direction = c("downup")) %>%**

# calculating the actual difference

**mutate(diff = round(isolate\_res - resistance, 1)) %>%**

**filter(method != "isolate")**

# same visual as in the previous question but using diff instead of

# resistance

**sum\_isolates\_2\_4 %>%**

**ggplot(, aes(x = as.character(Year), y = method, fill= diff)) +**

# what was that geom heatmap function again??

**geom\_...() +**

**scale\_fill\_viridis\_c(option = "A", direction = -1) +**

**facet\_grid(. ~ antibiotic) +**

**theme(axis.text.x = element\_text(angle = 45, hjust = 1)) +**

**labs(title = "Difference in resistance estimates per method compared to the isolate method",**

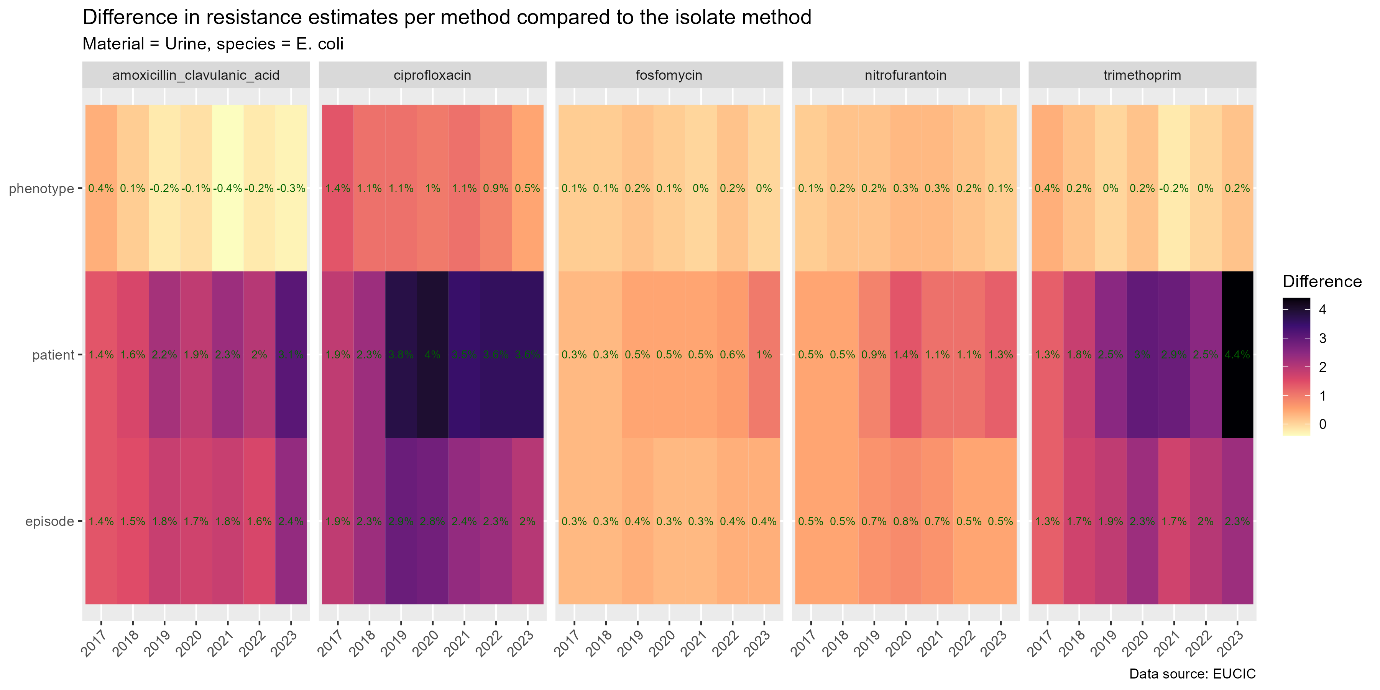
**subtitle = "Material = Urine, species = E. coli",**

**caption = "Data source: EUCIC",**

**x = NULL, y = NULL, fill = "Difference") +**

**geom\_text(aes(label = glue("{diff}%")),**

**color = "darkgreen", size = 2.5)**

****

1. Discuss the implications of not correcting for the first isolate and the importance of considering this factor in AMR data analysis.

### Assignment #3: Empirical Coverage of Ceftriaxone + Various Aminoglycosides in Septic Patients in 2022 – Cumulative Antibiograms

#### Objective

In this assignment, we will assess the empirical coverage of the antibiotic combination of ceftriaxone and different aminoglycosides for treating septic patients in the year 2022. Understanding the coverage of this combination is crucial for guiding treatment decisions. Also, determining empiric coverage is essential for AMS (antimicrobial stewardship) programs that are most often in place in hospitals. This assignment aims at providing you with the skills to determine this yourself, for your own institution.

We will apply the knowledge gained from the first two assignments.

#### Guidance

1. Filter the data to include only the patients including ceftriaxone and various aminoglycosides columns for sepsis treatment in 2022 in hospitals. Filter the data based on the first\_isolate function with the “episode\_based” method.

# for this question we only need to filter the data and selection the

# right columns based on the earlier created dataset

**isolates\_3\_1 <- isolates\_2 %>%**

**filter(material == "…",**

**request\_group == "…",**

**episode == TRUE,**

**Year == …) %>%**

**select(genus\_species, ceftriaxone, aminoglycosides())**

1. Analyse the susceptibility of bacterial isolates each antibiotic.

**antibiogram(isolates\_3\_1)**

1. Calculate the empirical coverage, which is the proportion of isolates susceptible to both ceftriaxone and the specific aminoglycoside(s).

**antibiogram(isolates\_3\_1,**

**antibiotics = c("ceftriaxone + amikacin",**

**"ceftriaxone + gentamicin",**

**"ceftriaxone + kanamycin",**

**"ceftriaxone + tobramycin"),**

**only\_all\_tested = TRUE)**

1. Interpret the coverage results, discuss the potential effectiveness of this antibiotic combination, and consider implications for clinical practice. If time left, discuss the use of only\_all\_tested and run the antibiogram()function again with only\_all\_tested = FALSE to see the differences.

### Assignment #4: Survival Analysis (Kaplan-Meier) of *Staphylococcus aureus* Bacteraemia Per Age Group

#### Objective

The objective of this assignment is to conduct a survival analysis using Kaplan-Meier survival curves to investigate the survival patterns of patients diagnosed with *Staphylococcus aureus* bacteraemia across different age groups. The analysis aims to explore how age impacts the time-to-event distribution, providing insights into the disease's progression and its association with age. By accomplishing this objective, the assignment aims to contribute to a better understanding of the disease's epidemiology and its implications for clinical management.

#### Guidance

1. Download the data set here: <https://bit.ly/eucic2023km> and load the dataset using the function below:

**KM\_aureus <- read\_rds("km\_merged.rds")**

1. Perform a descriptive analysis on the dataset. How many people survived (status = 0) and how many deceased (status = 1) the *Staphylococcus aureus* bacteraemia per age group?

**sum\_aureus\_bact <- KM\_aureus %>%**

**group\_by(…) %>%**

**count()**

**sum\_aureus\_bact**

1. Create a Kaplan Meijer curve for the *Staphylococcus aureus* bacteraemia survival per age group and perform a log-rank statistic in order to test for statistical differences.

**library(survminer)**

**library(survival)**

**library(ggsurvfit)**

# first create the fit

**fit\_aureus <- survfit(Surv(days, status) ~ age\_group, data = KM\_aureus)**

# inspect the fit

**fit\_aureus**

# plot it

**fit\_aureus %>%**

**ggsurvplot(**

# this must be the original data:

**data = KM\_aureus,**

# change line size (this might have become ‘linewidth’)

**size = 0.5,**

**ylab = "Proportion survival",**

# custom colour palettes, these are EUCIC colours

**palette = c("#**4f8b8b**", "#**a0cccd**"),**

# add confidence interval

**conf.int = TRUE,**

# add p-value

**pval = TRUE,**

# add risk table

**risk.table = TRUE,**

# risk table colour by group

**risk.table.col = "strata",**

# useful to change when you have multiple groups

**risk.table.height = 0.25,**

# this can be any ggplot2 theme, we choose ggplot2::theme\_bw()

**ggtheme = theme\_bw(),**

**cumevents = TRUE)**

A screenshot of a graph

Description automatically generated

1. Interpret the survival results. What can you say about the plot and about the ‘Number at risk’ table? Do the age groups diverge over time?

For this final assignment, the data had already been prepared. You can find the preparation script here for learning purposes. The ‘labdata’ data set is a common LIS data set summarised by lab number, containing just the columns: patient\_id, specimen\_date, mo, result, age, gender.

**km\_merged <- labdata |>**

**group\_by(patient\_id) |>**

# sorting required for episode determination

**arrange(specimen\_date) |>**

# flag the SABs (Staph aureus bacteraemias)

**mutate(STAU\_positive = (results == "positive" & mo == "S. aureus")) |>**

# determine episode for positive AND negative outcomes

**group\_by(patient\_id, STAU\_positive) |>**

**mutate(episode = get\_episode(specimen\_date, case\_free\_days = 5)) |>**

# back again to grouping on patient level

**group\_by(patient\_id) |>**

# create variabele to test whether negative was preceded by positive

**mutate(neg\_after\_pos = STAU\_positive == FALSE &**

**lag(STAU\_positive) == TRUE) |>**

# keep only positives from episode 1, or it was a negative after

# a positive result

**filter(episode == 1 & STAU\_positive == TRUE | neg\_after\_pos == TRUE) |>**

# create data set to keep

**transmute(**

**patient\_id,**

# 1: ends with negative, 0: ends with positive

**status = as.double(any(neg\_after\_pos, na.rm = TRUE)),**

# get the difference in days

**days = as.double(difftime(max(afname\_new[STAU\_positive == TRUE]),**

**min(afname\_new[STAU\_positive == TRUE]),**

**units = "days")),**

**age\_group = age\_groups(age, split\_at = 67),**

**gender) |>**

# remove groups at this point

**ungroup() |>**

# keep only distinct rows

**distinct()**