

Voice-driven Data Science: Real-Time Data Analysis with R

Phil Bowsher, Posit/RStudio PBC

ABSTRACT

ggbot2 is a new R package for voice assisted analysis with ggplot2. This paper will introduce it and show how statistical programmers can get up and running with it easily. By leveraging OpenAI's Realtime API and Shiny, users can generate ggplot2 visualizations through natural language commands. This is an experiment in the possibility of this tool in drug development as voice driven support takes shape.

INTRODUCTION

Voice-to-Code (V2C) allows for a "flow-state" where the programmer speaks and an active R session renders the logic. With ggbot2, this will happen via a Shiny interface.

GGBOT2 - SETUP

ggbot2 was released in 2025: <https://github.com/tidyverse/ggbot2>

ggbot2 requires an OpenAI API key for OpenAI's Realtime API. Users will need to set it as an environment variable named `OPENAI_API_KEY` as explained below in Positron using the R package `usethis`. This can be done with `usethis::edit_r_environ()` or via **Ctrl Shift P in Positron and File: Open File**. Add the following to the `.Renviron`, for example:

```
OPENAI_API_KEY=my-api-key-openai-xyz
```

Make sure to restart the R session.

Be sure to install ggbot2 as well with:

```
pak::pak("tidyverse/ggbot2")
```

USING GGBOT2

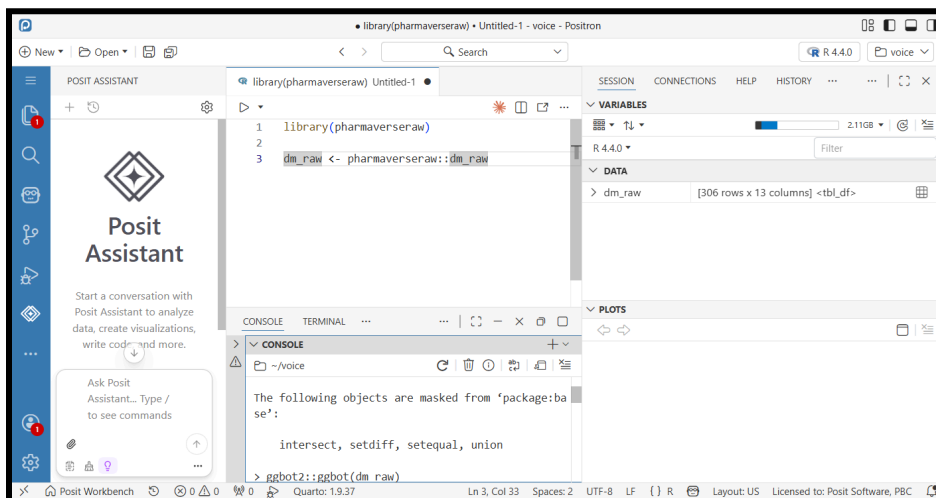
To start using ggbot2, we need to load some clinical data into Positron. Run this code:

```
# install.packages("pharmaverseraw")
```

```
library(pharmaverseraw)
```

```
dm_raw <- pharmaverseraw::dm_raw
```

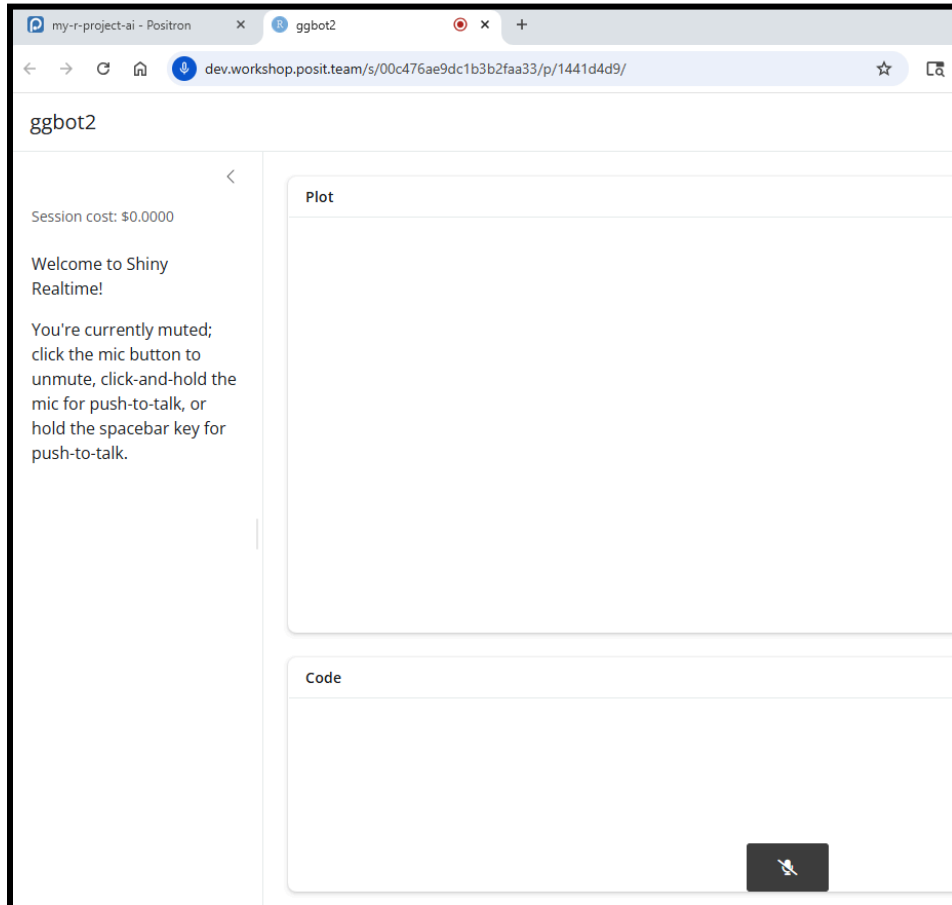
This will load the raw clinical data into the R session memory. It is often difficult for AI to work with in-memory data as they lack the context or connections to these capabilities which are required via tool calls or MCP. ggbot2 will do this automatically.



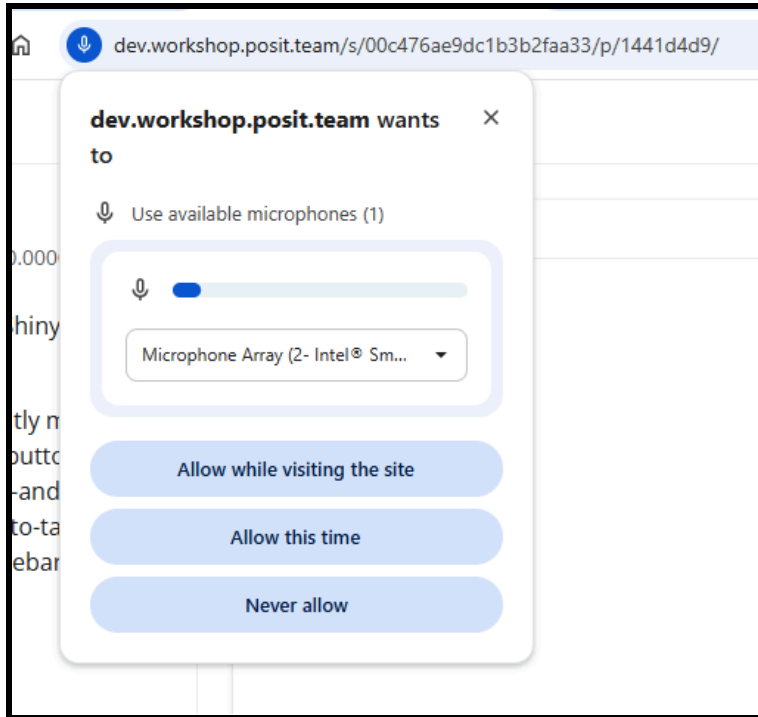
After the data is loaded, then run:

`ggbot2::ggbot(dm_raw)`

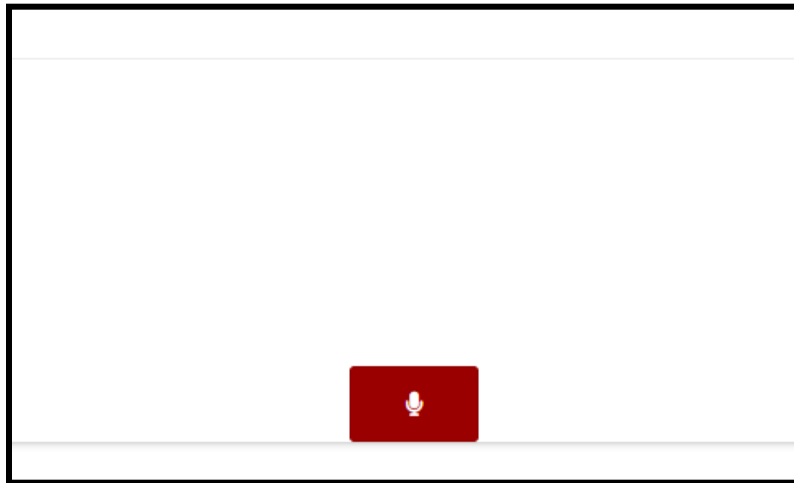
This will launch a Shiny app in a separate tab:



Next, users will need to permit a microphone to be used such as:



After the microphone is enabled, users will need to un-mute the microphone



The mic feature is implemented with the *shinyrealtime*: <https://github.com/posit-dev/shinyrealtime>

Say "Hello" to start the session with *ggbot2*. Users can use natural language and *ggbot2* will speak back and even suggest ideas or ask for next steps. Below are some visualizations made with *ggbot2* using raw DM data from the *pharmaverseraw* R package. Example spoken prompts used.

- "Using the DM data, create a bar chart of demographics"

ggbot2

Session cost: \$0.0546

You got it! Let's cook up a nice demographic bar chart. Since we've got variables like sex, ethnicity, race, and age, we can break them down visually. How about we start by showing the distribution of gender, ethnicity, and race in separate bars? Then we can add more if needed. Let's get it rolling!

Plot

Distribution of Gender

Code

```
library(ggplot2)

# Plotting the distribution of gender
gender_plot <- ggplot(dm_raw, aes(x = IT.SEX)) +
  geom_bar() +
  labs(title = 'Distribution of Gender', x = 'Gender', y = 'Count!')
```

- "Make the bars using best practices for colorblindness"

ggbot2

Session cost: \$0.0857

Plot

Distribution of Gender

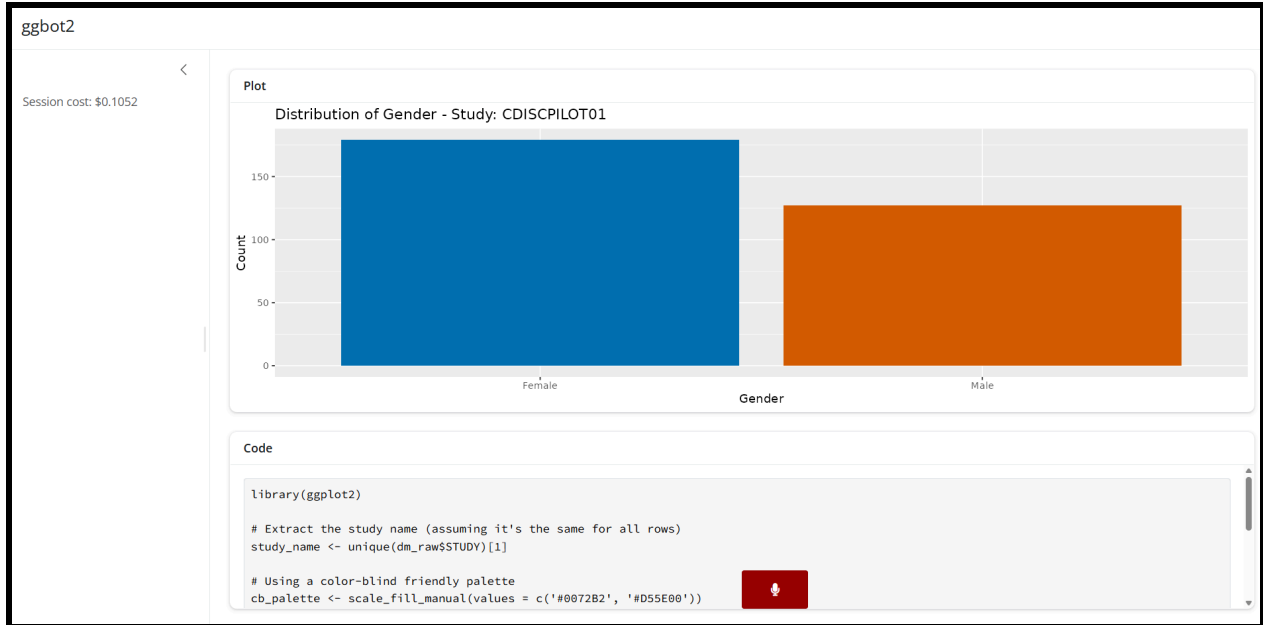
Code

```
library(ggplot2)

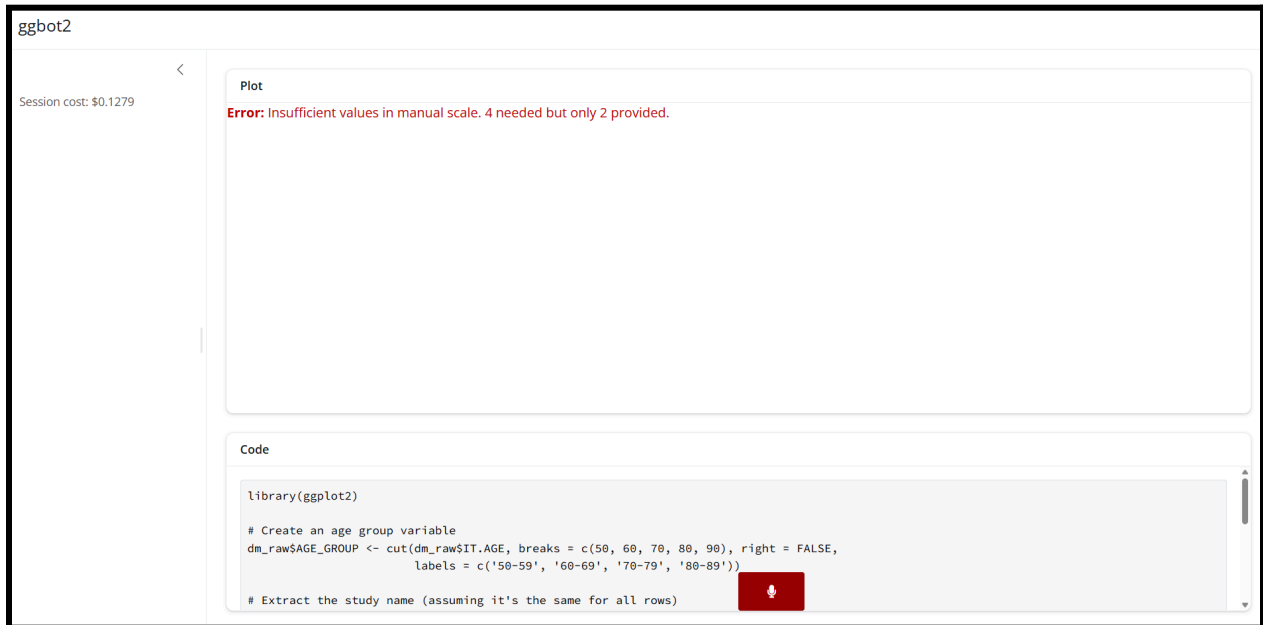
# Using a color-blind friendly palette
cb_palette <- scale_fill_manual(values = c('#0072B2', '#D55E00'))

# Plotting the distribution of gender with color-blind friendly colors
gender_plot <- ggplot(dm_raw, aes(x = IT.SEX, fill = IT.SEX)) +
```

- "Add the study name to the title to the bar chart"



- "Let's add age groups"



This generated an error and the user can then use "Go back and add another dimension to the student data"

ggbot2

Session cost: \$0.1536

Plot

Distribution of Gender by Age Group - Study: CDISCIPILOT01

Gender	50-59	60-69	70-79	80-89
Female	~15	~20	~80	~60
Male	~5	~25	~50	~45

Code

```
library(ggplot2)

# Create an age group variable using a suitable color-blind friendly palette for multiple categories
# We'll use a broader palette for age groups
age_group_palette <- scale_fill_manual(values = c('#0072B2', '#D55E00', '#009E73', '#F0E442'))

# Create an age group variable
```

“add another layer”

ggbot2

Session cost: \$0.3315

Plot

Distribution of Gender by Age Group and Planned Arm - Study: CDISCIPILOT01

Planned Arm	Gender	50-59	60-69	70-79	80-89
Placebo	Female	~5	~10	~15	~20
	Male	~5	~5	~10	~10
Screen Failure	Female	~5	~5	~10	~10
	Male	~5	~5	~5	~5
Xan High	Female	~5	~5	~10	~10
	Male	~5	~5	~10	~10
Xan Low	Female	~5	~5	~10	~10
	Male	~5	~5	~10	~10

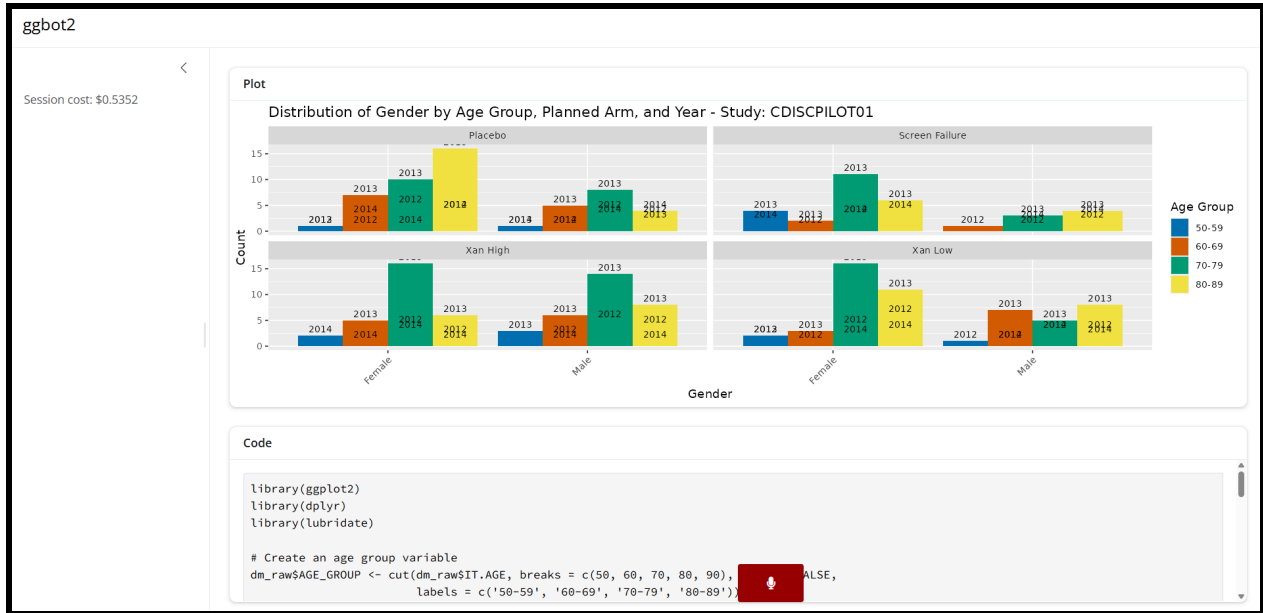
Code

```
library(ggplot2)

# Create an age group variable
dm_raw$AGE_GROUP <- cut(dm_raw$IT.AGE, breaks = c(50, 60, 70, 80, 90), right = FALSE,
  labels = c('50-59', '60-69', '70-79', '80-89'))

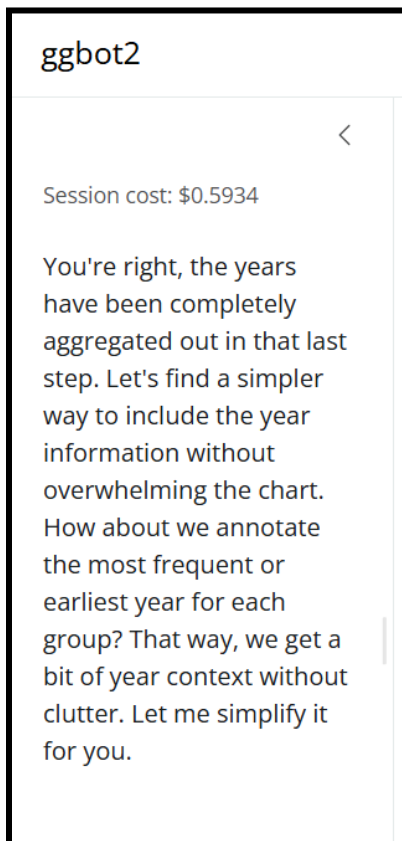
# Extract the study name (assuming it's the same for all rows)
```

- "Add an annotation that calls out the year"



Users can also keep with a current visualization and attempt to make improvements such as changing the display of years using comments like:

- "Make the plot/bar chart look less busy"
- "Let's go back to the last version"



ggbot2

Session cost: \$0.6773

Awesome! We've now got a simplified chart, and the annotations should give a nice hint of the earliest year for each group without making it too busy. Let me know if you'd like any more ●

Plot

Distribution of Gender by Age Group, Planned Arm, with Year Annotation - Study: CDISCILOT01

Code

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

# Create an age group variable
dm_raw$AGE_GROUP <- cut(dm_raw$IT.AGE, breaks = c(50, 60, 70, 80, 90),
  labels = c('50-59', '60-69', '70-79', '80-89'))
```

Users can also ask the tool to speak and write in other languages like Japanese:

ggbot2

Session cost: \$0.8260

はい、もちろんできます。
これから日本語でお話ししますので、何かご希望があれば遠慮なく教えてください。

Plot

Distribution of Gender by Age Group, Planned Arm, with Year Annotation - Study: CDISCILOT01

Code

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

# Create an age group variable
dm_raw$AGE_GROUP <- cut(dm_raw$IT.AGE, breaks = c(50, 60, 70, 80, 90),
  labels = c('50-59', '60-69', '70-79', '80-89'))
```

Or switch to German:

ggbot2

Session cost: \$0.9791

Sure, I can definitely switch to German for you. Lass uns jetzt auf Deutsch weitermachen. Wenn du irgendetwas Bestimmtes ändern möchtest oder neue Visualisierungen brauchst, sag einfach Bescheid!

Users can also ask to show the code in other programming languages like Python or SAS:

ggbot2

Session cost: \$1.1542

Ah, yes! The ggplot2 equivalent for Python is called `plotnine`. It's a fantastic package that brings the grammar of graphics to Python.

You can absolutely convert R ggplot2 code into `plotnine` syntax. The structure is quite similar, just with a few Pythonic tweaks. If you'd like, I can help you translate the current plot into Python's `plotnine` version.

Plot

Distribution of Gender by Age Group, Planned Arm, with Year Annotation - Study: CDISCPLOT01

Arm	Gender	50-59	60-69	70-79	80-89
Placebo	Female	2012	2012	2012	2012
	Male	2013	2012	2012	2012
Screen Failure	Female	2013	2012	2012	2012
	Male	2012	2012	2012	2012
Xan High	Female	2014	2013	2012	2012
	Male	2013	2012	2012	2012
Xan Low	Female	2012	2012	2012	2012
	Male	2012	2012	2012	2012

Code

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

# Create an age group variable
dm_raw$AGE_GROUP <- cut(dm_raw$IT.AGE, breaks = c(50, 60, 70, 80, 90),
  labels = c('50-59', '60-69', '70-79', '80-89'))
```

ggbot2

Session cost: \$1.2626

I currently can't run or display Python code directly or generate Python plots in this environment. My capabilities here are focused on generating and displaying R plots. However, I can definitely help you with the Python code translation so you can run it in your local Python environment using `plotnine`. Let me show you the equivalent Python code now.

ggbot2

Session cost: \$1.4761

You're absolutely right—let's go ahead and convert the last R plot into Python using `plotnine`. Here's the equivalent Python code:

```
# Import necessary libraries
from plotnine import *
import pandas as pd

# Assuming dm_raw is already loaded as a pandas DataFrame

# Create the age group variable in the DataFrame
dm_raw['AGE_GROUP'] = pd.cut(dm_raw['IT.AGE'], bins=[50, 60,

# Extract the study name (assuming it's the same for all rows)
study_name = dm_raw['STUDY'].unique()[0]

# Summarize to get the count per group
summary_df = dm_raw.groupby(['IT.SEX', 'AGE_GROUP', 'PLANNED.

# Define a color-blind friendly palette
cb_palette = ['#0072B2', '#D55E00', '#009E73', '#F0E442']

# Create the plot using plotnine
from plotnine import ggplot, aes, geom_bar, labs, facet_wrap

plot = (
```

Here is the Python code is generated:

Python

```
# Import necessary libraries
from plotnine import *
import pandas as pd

# Assuming dm_raw is already loaded as a pandas DataFrame

# Create the age group variable in the DataFrame
dm_raw['AGE_GROUP'] = pd.cut(dm_raw['IT.AGE'], bins=[50, 60, 70, 80, 90],
right=False, labels=['50-59', '60-69', '70-79', '80-89'])

# Extract the study name (assuming it's the same for all rows)
study_name = dm_raw['STUDY'].unique()[0]

# Summarize to get the count per group
summary_df = dm_raw.groupby(['IT.SEX', 'AGE_GROUP',
'PLANNED_ARM']).size().reset_index(name='Count')

# Define a color-blind friendly palette
cb_palette = ['#0072B2', '#D55E00', '#009E73', '#F0E442']

# Create the plot using plotnine
from plotnine import ggplot, aes, geom_bar, labs, facet_wrap, scale_fill_manual,
theme, element_text, position_dodge

plot = (
    ggplot(summary_df, aes(x='IT.SEX', y='Count', fill='AGE_GROUP')) +
    geom_bar(stat='identity', position=position_dodge()) +
    labs(title=f'Distribution of Gender by Age Group and Planned Arm - Study:
{study_name}',
        x='Gender', y='Count', fill='Age Group') +
    scale_fill_manual(values=cb_palette) +
    facet_wrap('~PLANNED_ARM') +
    theme(axis_text_x=element_text(angle=45, hjust=1))
)

# To display the plot, you would use: print(plot)
```

Users can also generate SAS code:

ggbot2

Session cost: \$1.6065

I currently can't execute or display SAS plots directly here. However, I can certainly help you outline the equivalent SAS code using PROC SGPLOT or other procedures. You'd need to run it in your own SAS environment.

Would you like me to generate the SAS code that corresponds to the plot? I'm happy to walk you through the key steps.

ggbot2

Session cost: \$2.0909

Understood! Here's how you can run the equivalent SAS code—in Viya, Base SAS, or in Positron with the SAS Viya Code extension.

Here's the SAS code:

```
/* Step 1: Create age groups */
DATA dm_raw_with_age_group;
  SET dm_raw;
  LENGTH AGE_GROUP $6;
  IF IT_AGE >= 50 AND IT_AGE < 60 THEN AGE_GROUP = '50-59';
  ELSE IF IT_AGE >= 60 AND IT_AGE < 70 THEN AGE_GROUP = '60-69';
  ELSE IF IT_AGE >= 70 AND IT_AGE < 80 THEN AGE_GROUP = '70-79';
  ELSE IF IT_AGE >= 80 AND IT_AGE < 90 THEN AGE_GROUP = '80-89';
RUN;

/* Step 2: Plot the data */
ODS GRAPHICS / RESET IMAGENAME = "gender_age_arm_plot";

PROC SGPLOT DATA=dm_raw_with_age_group;
  VBAR IT_SEX / RESPONSE=Count GROUP=AGE_GROUP GROUPDISPLAY=COUNT;
  FACET WRAP = PLANNED_ARM / COLUMNS=3;
  KEYLEGEND / TITLE = 'Age Group';
  XAXIS LABEL = 'Gender';
  YAXIS LABEL = 'Count';
  TITLE "Distribution of Gender by Age Group and Planned Arm";
RUN;
```

Here is the SAS code:

None

```
/* Step 1: Create age groups */
DATA dm_raw_with_age_group;
  SET dm_raw;
  LENGTH AGE_GROUP $6;
  IF IT_AGE >= 50 AND IT_AGE < 60 THEN AGE_GROUP = '50-59';
  ELSE IF IT_AGE >= 60 AND IT_AGE < 70 THEN AGE_GROUP = '60-69';
  ELSE IF IT_AGE >= 70 AND IT_AGE < 80 THEN AGE_GROUP = '70-79';
  ELSE IF IT_AGE >= 80 AND IT_AGE < 90 THEN AGE_GROUP = '80-89';
RUN;

/* Step 2: Plot the data */
ODS GRAPHICS / RESET IMAGENAME = "gender_age_arm_plot";

PROC SGPLOT DATA=dm_raw_with_age_group;
  VBAR IT_SEX / RESPONSE=Count GROUP=AGE_GROUP GROUPDISPLAY=CLUSTER STAT=FREQ
  TRANSPARENCY=0.1;
  FACET WRAP = PLANNED_ARM / COLUMNS=3;
  KEYLEGEND / TITLE = 'Age Group';
  XAXIS LABEL = 'Gender';
  YAXIS LABEL = 'Count';
  TITLE "Distribution of Gender by Age Group and Planned Arm - Study:
  &study_name";
RUN;

ODS GRAPHICS / RESET;
```

NOTES

Users need to install packages first before launching the Shiny app:

ggbot2

Session cost: \$2.4564

Great question! If we tried to use an R package that isn't installed in the current session, the code would throw an error because the package wouldn't be available.

We'd need to ensure that all necessary libraries are loaded and installed first. In practice, we'd either select from commonly available packages or explicitly include code to install those packages if needed.

So if you have a specific package in mind, just let me know, and we can add that step to ensure it runs smoothly.

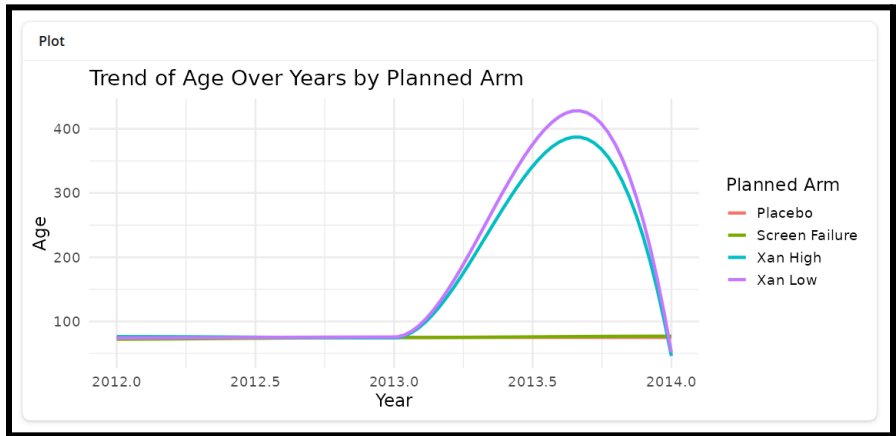
ggbot2 can't install packages:

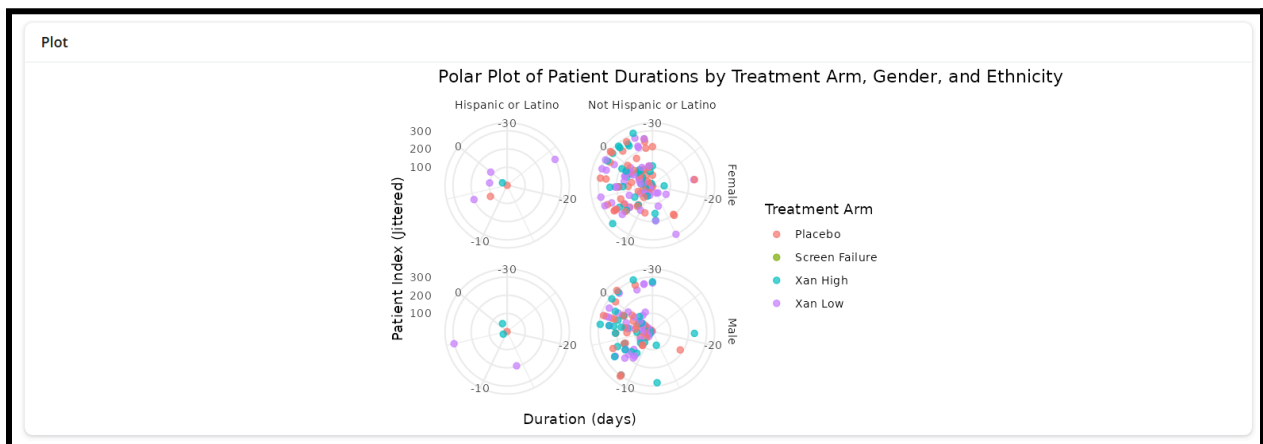
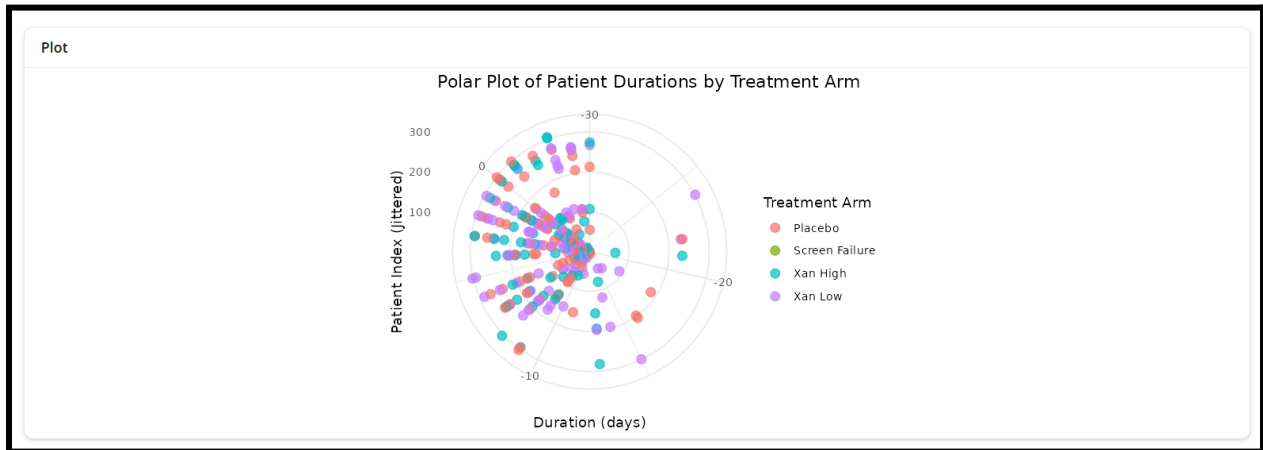
ggbot2

Session cost: \$2.5890

That's right. In this environment, I can't install packages for you or modify the system environment. I rely on the packages that are already available in the session. But I can guide you on which packages are needed and how to install them on your local machine if you run the code on your side.

But it can make some nice visualizations if you ask it to try something new:





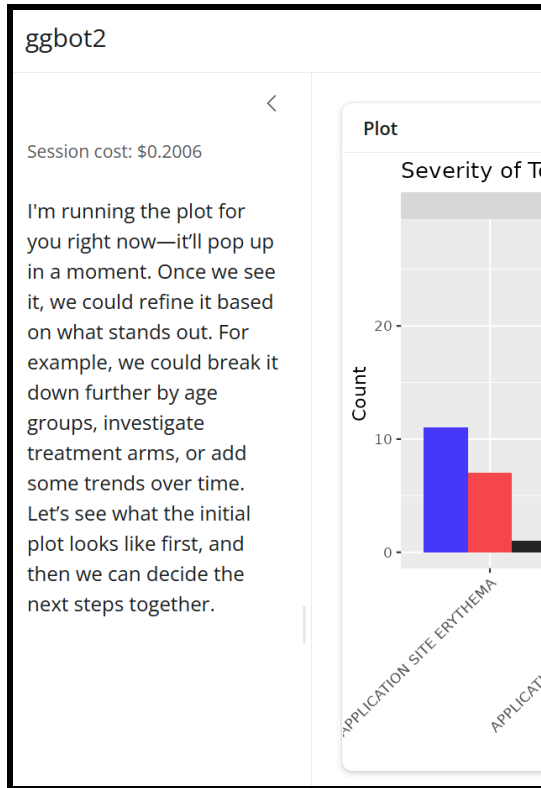
ggbot2 will not try to fix or resolve issues, so users will need to use “Go Back” or tell it about the error.

```

city-gini DROM
> ggbot2::ggbot(adae)
Warning: Error in count: Must group by variables found in `data`.
X Column `AGE_GROUP_10YR` is not found.
199: <Anonymous>
198: stop
197: value[[3L]]
196: tryCatchOne
195: tryCatchList
194: tryCatch

```

ggbot2 will display the cost of each session and will start at \$0.00 and use funds as the code is written.



CONCLUSION & CONTACT

ggbot2 highlights voice-driven interfaces in the R ecosystem. This tool helps bypass syntax jump into analysis and lowers the barrier to entry for reproducible data science.

Phil Bowsher

phil@posit.co

<https://github.com/philbowsher>