I want GLobal Baseline Estimate algorithm implementation in R for the following dataset. try to break down the code as much as possible and use the dataframe name df. The objective is to fill in the NA values with an estimated movie rating:

Name
<chr>
Superman
<dbl>
F1 The Movie
<dbl>
Mission Impossible
<dbl>
Jurassic World Rebirth
<dbl>
Sinners
<dbl>
Zootopia 2
<dbl>
Foizunnesa 3.5 4.2 4.5 3.0 NA NA
Jahid 4.5 4.0 4.1 3.0 NA 4.3
Mahmudul NA NA 4.8 3.5 NA NA
Sadman NA 4.9 4.0 NA 4.5 3.9
Shahjahan 3.6 NA 4.8 NA 4.0 NA
Readwanul NA 3.5 3.7 NA 4.1 NA
Tabassumul NA NA NA NA NA 4.6

Good

Got it! You want to implement the **Global Baseline Estimate (GBE)** algorithm in R for missing movie ratings. Let's break it down carefully and clearly. We'll use your dataframe `df` and fill in `NA` values with estimated ratings based on global average, user bias, and item bias. I'll provide the R code step by step with explanations.

---

## Step 1: Set up the data

First, let's create the dataframe `df` from your dataset:

r

```r
# Create the data frame
df <- data.frame(
  Name = c("Foizunnesa", "Jahid", "Mahmudul", "Sadman", "Shahjahan", "Readwanul",
  Superman = c(3.5, 4.5, NA, NA, 3.6, NA, NA),
  F1_The_Movie = c(4.2, 4.0, NA, 4.9, NA, 3.5, NA),
  Mission_Impossible = c(4.5, 4.1, 4.8, 4.0, 4.8, 3.7, NA),
  Jurassic_World_Rebirth = c(3.0, 3.0, 3.5, NA, NA, NA, NA),
  Sinners = c(NA, NA, NA, 4.5, 4.0, 4.1, NA),
  Zootopia_2 = c(NA, 4.3, NA, 3.9, NA, NA, 4.6)
)

# Check the structure
str(df)
df
```

Copy code

Good

---

## Step 2: Compute global mean rating

The **global mean** is the average of all non-NA ratings:

```r
# Compute global mean rating
global_mean <- mean(as.matrix(df[,-1]), na.rm = TRUE)
global_mean
```

Copy code

---

## Step 3: Compute user bias

User bias ( `b_u` ) is how each user deviates from the global mean.

For each user (row), calculate:

```r
# Compute user bias
user_bias <- rowMeans(df[,-1], na.rm = TRUE) - global_mean
user_bias
```

Copy code

Good

- `rowMeans(..., na.rm = TRUE)` calculates the mean of available ratings per user.
- Subtracting `global_mean` gives the bias for that user.

---

## Step 4: Compute item bias

Item bias ( `b_i` ) is how each movie deviates from the global mean, adjusted for user bias:

```r
```

```r
# Initialize item bias vector
item_bias <- numeric(ncol(df) - 1)
names(item_bias) <- colnames(df)[-1]

# Calculate item bias
for (i in 2:ncol(df)) {
  # Extract the column
  item_ratings <- df[[i]]
  # Consider only non-NA ratings
  not_na <- !is.na(item_ratings)
  # Compute item bias: mean(rating - global_mean - user_bias)
  item_bias[i-1] <- mean(item_ratings[not_na] - global_mean - user_b
}

item_bias
```

Copy code

---

## Step 5: Fill in missing values with Global Baseline Estimate

Good

The formula for GBE:

$$\hat{r}_{ui} = \text{global_mean} + b_u + b_i$$

Where:

- `b_u` = user bias
- `b_i` = item bias

r

```r
# Create a copy of df to fill in NA
df_filled <- df

# Fill missing values
for (i in 2:ncol(df_filled)) {  # for each movie
  for (j in 1:nrow(df_filled)) {  # for each user
    if (is.na(df_filled[j, i])) {
      df_filled[j, i] <- global_mean + user_bias[j] + item_bias[i-1]
    }
  }
}

df_filled
```

Copy code

---

## ✅ Step 6: Result

`df_filled` now contains all original ratings, with missing values estimated using the Global Baseline Estimate.

Good

- You can round the ratings to 1 decimal for presentation:

r

```r
df_filled[,-1] <- round(df_filled[,-1], 1)
df_filled
```

Copy code

---

This approach ensures:

1. **Global trend** ( `global_mean` ) is considered.
2. **User-specific tendency** ( `user_bias` ) is incorporated.
3. **Movie-specific tendency** ( `item_bias` ) is incorporated.

The missing ratings are now **personalized estimates** instead of just filling with the global mean.

---

If you want, I can also provide a **vectorized version** that avoids `for` loops entirely and is much faster for large datasets, fully using `tidyverse` .

Do you want me to do that?

Here is the chunk of all codes:
{r}

```
library(tidyverse)

url <- "https://raw.githubusercontent.com/Nafee

df <- read_csv(
  file = url,
  show_col_types = FALSE,
  progress = FALSE
)
```

Good

```
df
```

```
{r}
df <- df %>%
  mutate(user_avg = rowMeans(select(., -Name), r
  ) %>%
  mutate(user_avg_mean_movie = user_avg - mean(u

df
```

```
{r}
movie_avg_values <- df %>%
  select(-Name, -user_avg, -user_avg_mean_movie)
  colMeans(na.rm = TRUE)

movie_avg <- data.frame(
  Name = "movie_avg",
  t(movie_avg_values),
  `user_avg` = NA,
  `user_avg_mean_movie` = NA,
  check.names = FALSE
)

df <- bind_rows(df, movie_avg)

overall_user_avg <- mean(df$user_avg, na.rm = TR
```

Good

```
movie_avg_mean_movie_values <- movie_avg_values
movie_avg_mean_movie <- data.frame(
  Name = "movie_avg_mean_movie",
  t(movie_avg_mean_movie_values),
  user_avg = NA,
  user_avg_mean_movie = NA,
  check.names = FALSE
)

df <- bind_rows(df, movie_avg_mean_movie)

df
```

Now write a code, that uses GBE formula to fill in the gaps.

Perfect! You already have **user bias** ( `user_avg_mean_movie` ) and **item bias** ( `movie_avg_mean_movie_values` ) calculated. Now we can apply the **Global Baseline Estimate formula** to fill in the missing values:

Good

$$\hat{r}_{ui} = \text{global mean} + b_u + b_i$$

Where:

- `global_mean` = mean of all ratings
- `b_u` = user bias ( `user_avg_mean_movie` )
- `b_i` = item bias ( `movie_avg_mean_movie_values` )

Here's the R code to fill the gaps using your current `df` :

r

```r
# Step 1: Compute global mean (excluding NAs)
global_mean <- df %>%
  filter(!Name %in% c("movie_avg", "movie_avg_mean_movie")) %>%
  select(-Name, -user_avg, -user_avg_mean_movie) %>%
  as.matrix() %>%
  mean(na.rm = TRUE)

global_mean

# Step 2: Extract user biases
user_bias <- df$user_avg_mean_movie[1:(nrow(df)-2)]  # excluding the last two row

# Step 3: Extract item biases
item_bias <- as.numeric(df[nrow(df), -c(1, ncol(df)-1, ncol(df))])  # last row =
names(item_bias) <- colnames(df)[2:(ncol(df)-2)]  # movie names

# Step 4: Fill in missing ratings
df_filled <- df[1:(nrow(df)-2), ]  # only keep real users

for (movie in names(item_bias)) {
  for (i in 1:nrow(df_filled)) {
    if (is.na(df_filled[i, movie])) {
      df_filled[i, movie] <- global_mean + user_bias[i] + item_bias[movie]
    }
  }
}

# Optional: Round the ratings for readability
df_filled[ , 2:(ncol(df_filled)-2)] <- round(df_filled[ , 2:(ncol(df_filled)-2)],
```

Good

Copy code

```
df_filled
```

---

### ✅ What this does:

1. Computes the **global mean** of all available ratings.
2. Uses your previously computed **user bias** ( `user_avg_mean_movie` ).
3. Uses the **item bias** ( `movie_avg_mean_movie` ).
4. Iterates over each missing cell and fills it with the **GBE estimate**.

---

If you want, I can also **rewrite this in a fully vectorized `dplyr` version**, so you don't need any nested `for` loops. It will be much faster and cleaner for larger datasets.

Do you want me to do that?

Good

I tried doing this:

```
{r}
# Identify numeric rating columns
num_cols <- names(df)[!names(df) %in% c("Name",

# Step 1: Compute overall user average
overall_user_avg <- mean(df$user_avg, na.rm = TI
```

```
# Step 2: Extract movie_avg_mean_movie row as a
movie_avg_mean_movie_row <- df %>%
  filter(Name == "movie_avg_mean_movie") %>%
  select(all_of(num_cols)) %>%
  unlist() %>%
  as.numeric()

# Step 3: Fill NAs for **actual users** using tl
df_filled <- df %>%
  rowwise() %>%
  mutate(across(all_of(num_cols), ~
                ifelse(
                  is.na(.) & !(Name %in% c("mo
                  overall_user_avg - movie_avg_
                  .
                )
  )) %>%
  ungroup()
df_filled
```

Good

But I see not change in NAs:

Name
<chr>
Superman
<dbl>
F1 The Movie
<dbl>
Mission Impossible
<dbl>

Jurassic World Rebirth
<dbl>
Sinners
<dbl>
Zootopia 2
<dbl>
user_avg
<dbl>
user_avg_mean_movie
<dbl>
Foizunnesa 3.5000000 4.20000000 4.5000000 3.0000000 NA NA 3.800000 -0.30785714
Jahid 4.5000000 4.00000000 4.1000000 3.0000000 NA 4.3000000 3.980000 -0.12785714
Mahmudul NA NA 4.8000000 3.5000000 NA NA 4.150000 0.04214286
Sadman NA 4.90000000 4.0000000 NA 4.50000000 3.9000000 4.325000 0.21714286
Shahjahan 3.6000000 NA 4.8000000 NA 4.00000000 NA 4.133333 0.02547619
Readwanul NA 3.50000000 3.7000000 NA 4.10000000 NA 3.766667 -0.34119048
Tabassumul NA NA NA NA NA 4.6000000 4.600000 0.49214286
movie_avg 3.8666667 4.15000000 4.3166667 3.1666667 4.20000000 4.2666667 NA NA
movie_avg_mean_movie -0.2411905 0.04214286 0.2088095 -0.9411905 0.09214286 0.1588095 NA NA
1-9 of 9 rows

Good

Ah! I see exactly what happened. Your `mutate(across(...))` code is **not correctly referencing the GBE formula**, and the way you're indexing `movie_avg_mean_movie_row[cur_column()]` doesn't work the way you expect in `dplyr`. Also, the formula inside `ifelse()` is incorrect: you have a subtraction instead of **adding global mean + user bias + item bias**. That's why NAs are not filled.

Let's fix this step by step with a **working vectorized approach using** `rowwise()`.

---

### Correct GBE formula:

$$\hat{r}_{ui} = \text{global mean} + \text{user bias} + \text{item bias}$$

- `global_mean` = mean of all ratings (excluding NAs, only real users)
- `user_bias = user_avg_mean_movie`
- `item_bias = movie_avg_mean_movie_row`

---

Good

### Fixed R code

```r
library(dplyr)

# Identify numeric rating columns
num_cols <- names(df)[!names(df) %in% c("Name", "user_avg", "user_avg_mean_movie"
```

```r
# Step 1: Compute global mean (only real users)
global_mean <- df %>%
  filter(!Name %in% c("movie_avg", "movie_avg_mean_movie")) %>%
  select(all_of(num_cols)) %>%
  as.matrix() %>%
  mean(na.rm = TRUE)

global_mean

# Step 2: Extract item bias as numeric vector
item_bias <- df %>%
  filter(Name == "movie_avg_mean_movie") %>%
  select(all_of(num_cols)) %>%
  unlist() %>%
  as.numeric()

names(item_bias) <- num_cols

# Step 3: Fill NAs for actual users using GBE formula
df_filled <- df %>%
  filter(!Name %in% c("movie_avg", "movie_avg_mean_movie")) %>%  # keep only real
  rowwise() %>%
  mutate(across(all_of(num_cols), ~ ifelse(
    is.na(.),
    global_mean + user_avg_mean_movie + item_bias[cur_column()],
    .
  ))) %>%
  ungroup()

# Optional: round for readability
```

Good

Copy code

```
df_filled[num_cols] <- round(df_filled[num_cols], 1)

df_filled
```

---

### ✅ Why this works

1. `global_mean` is added to every estimate.
2. `user_avg_mean_movie` gives the **user-specific bias**.
3. `item_bias[cur_column()]` gives the **movie-specific bias**.
4. `ifelse(is.na(.), ..., .)` only fills missing values, leaving existing ratings intact.

---

If you run this, **all NAs for actual users should now be replaced with estimated ratings**.

I can also rewrite this **fully vectorized without** `rowwise()`, which is faster and avoids the slow `for`-like operations if you plan to scale to larger datasets.

Good

Do you want me to do that?



that is user_avg and user_avg_mean_movie:

```
{r}
movie_avg_values <- df %>%
```

```
  select(-Name, -user_avg, -user_avg_mean_movie)
  colMeans(na.rm = TRUE)

movie_avg <- data.frame(
  Name = "movie_avg",
  t(movie_avg_values),
  `user_avg` = NA,
  `user_avg_mean_movie` = NA,
  check.names = FALSE
)

df <- bind_rows(df, movie_avg)


overall_user_avg <- mean(df$user_avg, na.rm = TR
overall_user_avg

movie_avg_mean_movie_values <- movie_avg_values
movie_avg_mean_movie <- data.frame(
  Name = "movie_avg_mean_movie",
  t(movie_avg_mean_movie_values),
  user_avg = NA,
  user_avg_mean_movie = NA,
  check.names = FALSE
)

df <- bind_rows(df, movie_avg_mean_movie)

df
```

Good

```r
num_cols <- names(df)[!names(df) %in% c("Name",

# Step 2: Extract item bias as numeric vector
item_bias <- df %>%
  filter(Name == "movie_avg_mean_movie") %>%
  select(all_of(num_cols)) %>%
  unlist() %>%
  as.numeric()

names(item_bias) <- num_cols

# Step 3: Fill NAs for actual users using GBE fo
df <- df %>%
  filter(!Name %in% c("movie_avg", "movie_avg_me
  rowwise() %>%
  mutate(across(all_of(num_cols), ~ ifelse(
    is.na(.),
    overall_user_avg + user_avg_mean_movie + ite
    .
  ))) %>%
  ungroup()

# Optional: round for readability
df[num_cols] <- round(df[num_cols], 1)

df
```

Good

Got it! You want to **fill the NAs using GBE** but also **drop the last two columns** ( `user_avg` and `user_avg_mean_movie` ) in the final `df` . We can do this by removing them after computing everything. Here's the modified code:

r

```r
library(dplyr)

# --- Step 1: Compute movie averages ---
movie_avg_values <- df %>%
  select(-Name, -user_avg, -user_avg_mean_movie) %>%
  colMeans(na.rm = TRUE)

movie_avg <- data.frame(
  Name = "movie_avg",
  t(movie_avg_values),
  check.names = FALSE
)

df <- bind_rows(df, movie_avg)

# --- Step 2: Compute overall user average ---
overall_user_avg <- mean(df$user_avg, na.rm = TRUE)

# --- Step 3: Compute movie bias (item bias) ---
movie_avg_mean_movie_values <- movie_avg_values - overall_user_avg
movie_avg_mean_movie <- data.frame(
  Name = "movie_avg_mean_movie",
  t(movie_avg_mean_movie_values),
```

Good

```
  check.names = FALSE
)

df <- bind_rows(df, movie_avg_mean_movie)

# --- Step 4: Identify numeric rating columns ---
num_cols <- names(df)[!names(df) %in% c("Name", "user_avg", "user_avg_mean_movie"

# --- Step 5: Extract item bias as numeric vector ---
item_bias <- df %>%
  filter(Name == "movie_avg_mean_movie") %>%
  select(all_of(num_cols)) %>%
  unlist() %>%
  as.numeric()
names(item_bias) <- num_cols

# --- Step 6: Fill NAs for actual users using GBE formula ---
df_filled <- df %>%
  filter(!Name %in% c("movie_avg", "movie_avg_mean_movie")) %>%  # keep only real
  rowwise() %>%
  mutate(across(all_of(num_cols), ~ ifelse(
    is.na(.),
    overall_user_avg + user_avg_mean_movie + item_bias[cur_column()],
    .
  ))) %>%
  ungroup()

# --- Step 7: Drop helper columns ---
df_filled <- df_filled %>%
  select(-user_avg, -user_avg_mean_movie)
```

Good

Copy code

```r
# --- Step 8: Round ratings for readability ---
df_filled[num_cols] <- round(df_filled[num_cols], 1)

df_filled
```

---

### ✅ What changed:

1. Dropped `user_avg` and `user_avg_mean_movie` in **Step 7** after filling NAs.
2. The GBE formula still correctly fills missing ratings.
3. Ratings are rounded for readability.

---

If you want, I can also **merge `df_filled` back with `movie_avg` and `movie_avg_mean_movie`** so your final dataframe includes **all rows** but without the helper columns — ready for further analysis or export.

Good

Do you want me to do that?

give me APA citation for this transcript

Here is the APA citation for this ChatGPT transcript in the format you've been using:

OpenAI. (2026). *ChatGPT (Version 5.2) [Large language model]*. https://chat.openai.com ↗. Accessed February 14, 2026.

If you need it formatted slightly differently (for example, for a reference list vs. in-text citation), let me know.

Good