Simulation Methods

```r
#devtools::load_all()
suppressPackageStartupMessages({
  library(GEM)
  library(ggplot2)
  library(ggridges)
  library(dplyr)
  library(tidyr)
  library(cowplot)
  library(magrittr)
})
```

Run simulation

Simulation Model

The simulation study was designed to evaluate the performance of our proposed method for analyzing aggregated binary outcomes. We generated data according to the following procedure:

Data Generation Process

We simulated a dataset with $n = 10,000$ candidate mutations with $p = 10$ features/covariates, assigned to $k = 100$ distinct samples/individuals. The data generation process follows these steps:

1. **Grouping Structure**: Mutations were randomly assigned to $k = 100$ individuals, with each mutation having equal probability of assignment to any sample. Let $\mathcal{S}_j$ represent the set of indices of mutations belonging to the j-th sample, for $j \in \{1, 2, \dots, k\}$.

2. **Covariate Generation**: We generated a covariate matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$ with each element independently drawn from a standard normal distribution:

$$X_{ij} \sim \mathcal{N}(0, 1) \quad \text{for } i = 1, \dots, n \text{ and } j = 1, \dots, p$$

3. **Mutation Status**: For each observation, a latent linear predictor η_i was computed as:

$$\eta_i = \mathbf{X}_i^T \beta + \beta_0$$

   where:

   - $\mathbf{X}_i$ is the vector of covariates for the i-th observation
   - $\beta \in \mathbb{R}^p$ is a vector of regression coefficients, with each element independently drawn from $\mathcal{N}(0, 2)$
   - $\beta_0 = -0.8$ is a fixed intercept term

   Binary mutation statuses Z_i were then generated according to a Bernoulli distribution with probability determined by the logistic function:

$$Z_i \sim \text{Bernoulli}(\text{logit}^{-1}(\eta_i)) \quad \text{for } i = 1, \dots, n$$

   where $\text{logit}^{-1}(x) = \frac{1}{1+e^{-x}}$ is the inverse logit (or logistic) function.

4. **Simulated age at blood draw**: For each sample j, we computed the true count of positive binary outcomes:

$$C_j = \sum_{i \in \mathcal{S}_j} Z_i \quad \text{for } j = 1, \dots, k$$

   The final response variable (simulated age at blood draw) Y_j for each aggregation unit was generated as:

$$Y_j = 60 + 10 \cdot \log_2(C_j) \cdot \beta_1 + \epsilon_j \quad \text{for } j = 1, \dots, k$$

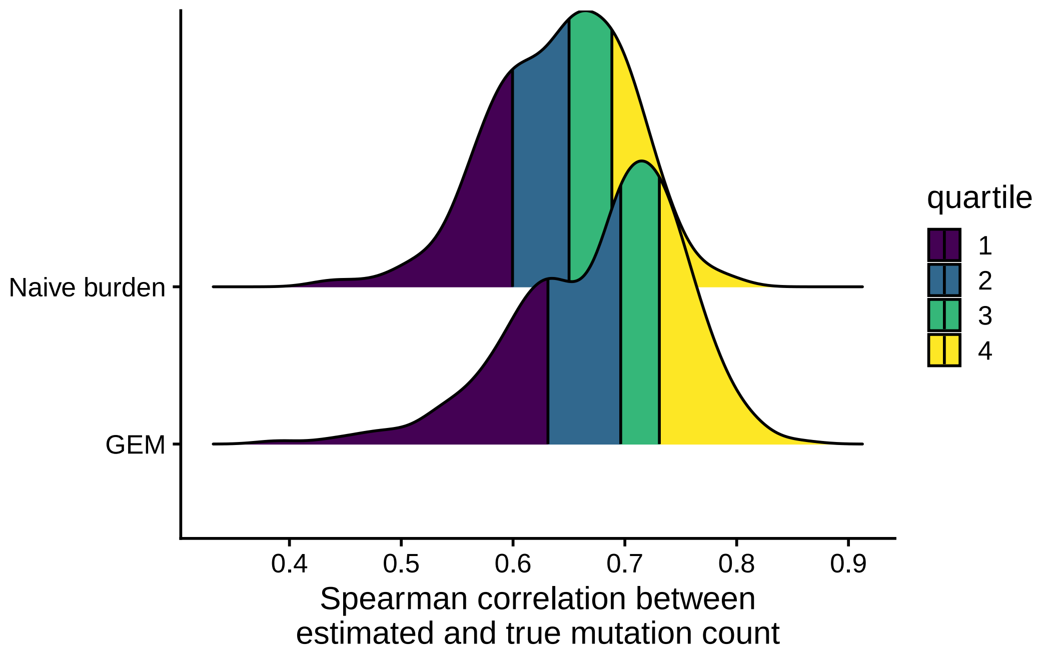
   where:

   - $\beta_1 = 1.0$ is a fixed coefficient
   - $\epsilon_j \sim \mathcal{N}(0, 5)$ represents random noise
   - The logarithmic transformation $\log_2(C_j)$ models a non-linear relationship between counts and the response

We then run this simulation 300 times and estimate the cordance with the "true" mutation burden using both GEM and the naive mutation count/burden as estimators.

```r
sims = GEM:::wrap_simulation(nsims = 300, .progress = FALSE)
```

```r
sims %>%
  rename(
    `GEM` = cor_gem,
    `Naive burden` = cor_burden
  ) %>%
  tidyr::pivot_longer(cols = everything(), names_to = "type", values_to = "cor") %>%
  ggplot(., aes(y = type, x = cor, fill = factor(stat(quantile)))) +
    stat_density_ridges(
      geom = "density_ridges_gradient",
      calc_ecdf = TRUE,
      quantiles = 4,
      quantile_lines = TRUE
    ) +
    scale_fill_viridis_d(name = "quartile") +
    theme_cowplot(font_size = 12) +
    theme(axis.title.y = element_blank()) +
    labs(x = "Spearman correlation between\nestimated and true mutation count")
```



```r
summary(sims)
#>     cor_gem          cor_burden
#>  Min.   :0.3898   Min.   :0.4326
#>  1st Qu.:0.6312   1st Qu.:0.5995
#>  Median :0.6963   Median :0.6501
#>  Mean   :0.6797   Mean   :0.6434
#>  3rd Qu.:0.7309   3rd Qu.:0.6884
#>  Max.   :0.8545   Max.   :0.7971
```