Estimating Farm Size Production Distribution: MRP vs Pareto Approaches

Vincent Ricciardi

2025-02-28

Contents

1 Introduction

This document provides boiler-plate code to compare two methods to estimate crop production by farm size class: Pareto interpolation and Multilevel Regression with Poststratification (MRP). Once we have our final dataset, we can use this code to test out the two methods and compare results. This document might be useful as supplemental material to our paper so we can show readers how using different interpolation methods might change results and why we choose one over the other.

1.1 Pareto Interpolation Method

The Pareto interpolation method is a statistical approach typically used to estimate distribution characteristics, particularly for phenomena with a heavy-tailed distribution. For our purpose, it assumes that farm sizes follow a power-law distribution, characterized by a small number of large farms producing a disproportionate amount of agricultural output. The method is based on the Pareto distribution, which is defined by two key parameters: the minimum value (xmin) and the shape parameter (α). Researchers estimate these parameters using techniques like maximum likelihood estimation to model how farm sizes are distributed, with a particular focus on capturing the characteristics of larger farms that are often underrepresented in survey data.

1.2 Multilevel Regression with Poststratification (MRP)

MRP offers a more robust alternative to Pareto interpolation by avoiding strict distributional assumptions and leveraging information across different strata. Unlike the Pareto method, which requires data to follow a specific power-law distribution, MRP can adapt to complex, real-world patterns by "borrowing strength" from similar observations across countries, regions, and farm types. Critically, MRP provides more comprehensive error estimates, allowing researchers to quantify uncertainty at multiple levels - from individual farm sizes to entire regional agricultural systems - which is significantly more challenging with traditional Pareto approaches.

MRP will also enable us to include more countries with partial data into our analysis.

While MRP has become a standard methodological approach in political science, public opinion research, and certain ecological applications, it remains relatively underutilized in the agricultural economics literature. The traditional reliance on Pareto interpolation in agricultural studies presents an opportunity to introduce the more flexible MRP framework, which may better capture the complex hierarchical structures inherent in farm production data. By applying this method from adjacent disciplines, we potentially offer methodological improvements to the standard toolkit for analyzing agricultural production distributions by farm size

1.3 Comparison

This document generates synthetic data, then compares the Pareto and MRP methods. Once ready, we can use this to plug our actual data in to see if there are critical differences.

2 Data Requirements for Farm Size Production Analysis

2.1 Minimum Data Requirements

For our analysis, we need the following data structures:

2.1.1 Complete Data Requirement

For countries where complete farm-level production data is available, we need: - Country identifier - Region - Crop type (e.g., oilcrop, vegetable, nuts, etc.) - Farm size class - Production metric (e.g., kcal, tons, monetary value) - Additional predictor variables - Development index / GDP PPP - Other contextual variables - Dataset specific Boolean flags, such as if it's a agricultural census or a household survey

2.1.2 Minimum Viable Dataset

For countries where complete farm-level data is unavailable, we need: - Total production by crop type - Country-level characteristics - Estimated farm size distribution (even if approximate) - Additional predictor variables - Development index / GDP PPP - Other contextual variables - Dataset specific Boolean flags, such as if it's a agricultural census or a household survey

3 Synthetic Data Generation

```
library(dplyr)
library(tidyr)

# Function to generate synthetic farm production data
generate_farm_production_data <- function(
  n_countries = 50,
  n_crops = 8,
  farm_sizes = c("0-1", "1-2", "2-5", "5-10", "10-20", "20-50", "50+"),
  regions = c("Africa", "Asia", "Europe", "North America", "South America")
```

```
) {
  # Create country-level dataset
  countries <- data.frame(
    country_id = 1:n_countries,
    country_name = paste0("Country", 1:n_countries),
    region = sample(regions, n_countries, replace = TRUE),
    development_index = runif(n_countries, 0.3, 0.9)
  )

  # Crop types
  crops <- c("Cereals", "Fruit", "Oilcrops", "Other",
             "Pulses", "Roots and Tubers", "Treenuts", "Vegetables")

  # Create full grid of combinations
  full_grid <- expand.grid(
    country_id = 1:n_countries,
    crop = crops,
    farm_size = farm_sizes
  )

  # Add country information
  full_grid <- merge(full_grid, countries, by = "country_id")

  # Generate synthetic kcal production values
  generate_kcal <- function(data) {
    # Farm size effect
    size_effect <- match(data$farm_size, farm_sizes)

    # Development effect
    dev_effect <- ifelse(data$farm_size %in% c("0-1", "1-2", "2-5"),
                         1 - data$development_index,
                         data$development_index)

    # Crop effect
    crop_effect <- case_when(
      data$crop == "Cereals" ~ 1.2,
      data$crop == "Fruit" ~ 0.8,
      data$crop == "Oilcrops" ~ 1.1,
      data$crop == "Other" ~ 0.7,
      data$crop == "Pulses" ~ 0.9,
      data$crop == "Roots and Tubers" ~ 1.3,
      data$crop == "Treenuts" ~ 0.6,
      data$crop == "Vegetables" ~ 0.5
    )

    # Region effect
    region_effect <- case_when(
      data$region == "Africa" ~ 0.8,
      data$region == "Asia" ~ 1.1,
      data$region == "Europe" ~ 0.9,
      data$region == "North America" ~ 1.2,
      data$region == "South America" ~ 1.0
    )
```

```
    # Base kcal production
    base_kcal <- exp(size_effect/2) * 1000

    # Combine effects with randomness
    kcal <- base_kcal * dev_effect * crop_effect * region_effect *
      exp(rnorm(nrow(data), 0, 0.3))

    return(kcal)
  }

  # Generate kcal values
  full_grid$kcal_true <- generate_kcal(full_grid)

  return(full_grid)
}

# Generate synthetic data
set.seed(123)
synthetic_data <- generate_farm_production_data()
head(synthetic_data)
```

```
##   country_id       crop farm_size country_name region development_index
## 1          1    Cereals       0-1     Country1 Europe         0.7261094
## 2          1      Fruit       2-5     Country1 Europe         0.7261094
## 3          1      Other       50+     Country1 Europe         0.7261094
## 4          1   Oilcrops     10-20     Country1 Europe         0.7261094
## 5          1   Treenuts      5-10     Country1 Europe         0.7261094
## 6          1 Vegetables     20-50     Country1 Europe         0.7261094
##    kcal_true
## 1   419.4703
## 2   799.7205
## 3 11160.0160
## 4  6349.3838
## 5  3173.4456
## 6  7507.4864
```

4 Pareto Distribution Analysis

```
library(poweRlaw)
library(ggplot2)

calculate_pareto_ks <- function(data, alpha, xmin) {
  # Caclulate the Kolmogorov-Smirnov statistic
  # Generally, a lower KS statistic indicates a better fit. In practice:
  # KS < 0.05: Excellent fit
  # KS < 0.1: Good fit
  # KS > 0.1: Poor fit

  # Sort the data
  sorted_data <- sort(data[data >= xmin])
  n <- length(sorted_data)
```

```r
  # Calculate empirical CDF
  ecdf_vals <- (1:n) / n

  # Calculate theoretical Pareto CDF: F(x) = 1 - (xmin/x)^alpha
  pareto_cdf <- 1 - (xmin / sorted_data)^alpha

  # Calculate KS statistic: maximum absolute difference between empirical and theoretical CDF
  ks_stat <- max(abs(ecdf_vals - pareto_cdf))

  # Calculate p-value (optional, requires the kolmogorov-smirnov distribution)
  # p_value <- 1 - pkstwo(ks_stat * sqrt(n))

  return(list(
    ks_statistic = ks_stat,
    data_points = n,
    alpha = alpha,
    xmin = xmin
  ))
}

# Function to test Pareto distribution fit
test_pareto_fit <- function(data, min_size = 1) {
  # Prepare data for Pareto analysis
  pareto_data <- data %>%
    group_by(country_name, crop, farm_size) %>%
    summarise(total_kcal = sum(kcal_true), .groups = "drop")

  # Select data above minimum size
  pareto_subset <- pareto_data %>%
    filter(total_kcal >= min_size)

  # Fit Pareto distribution
  m_pl <- conpl$new(pareto_subset$total_kcal)
  est_pl <- estimate_pars(m_pl)
  m_pl$setPars(est_pl$pars)

  # Calculate KS statistic
  ks_results <- calculate_pareto_ks(
    data = pareto_subset$total_kcal,
    alpha = est_pl$pars,
    xmin = m_pl$getXmin()
  )

  # Prepare plotting data
  plot_data <- data.frame(
    x = pareto_subset$total_kcal,
    y = 1 - rank(pareto_subset$total_kcal, ties.method = "random")/(nrow(pareto_subset)+1)
  )

  # Create log-log plot
  p <- ggplot(plot_data, aes(x = x, y = y)) +
    geom_point(alpha = 0.5) +
    scale_x_log10() +
```

```
    scale_y_log10() +
    geom_line(data = data.frame(
      x = seq(min(plot_data$x), max(plot_data$x), length.out = 100)
    ),
    aes(x = x, y = (x/est_pl$pars)^(-est_pl$pars)),
    color = "red", size = 1) +
    labs(
      title = "Farm Size Distribution - Pareto Fit",
      x = "Total kcal (log scale)",
      y = "Complementary CDF (log scale)"
    ) +
    theme_minimal()

  return(list(
    plot = p,
    pareto_alpha = est_pl$pars,
    xmin = m_pl$getXmin(),
    ks_statistic = ks_results$ks_statistic,
    message = paste("Pareto alpha:", round(est_pl$pars, 2),
                    "| KS statistic:", round(ks_results$ks_statistic, 3))
  ))
}

# Perform Pareto analysis
pareto_results <- test_pareto_fit(synthetic_data)
print(pareto_results$message)
```

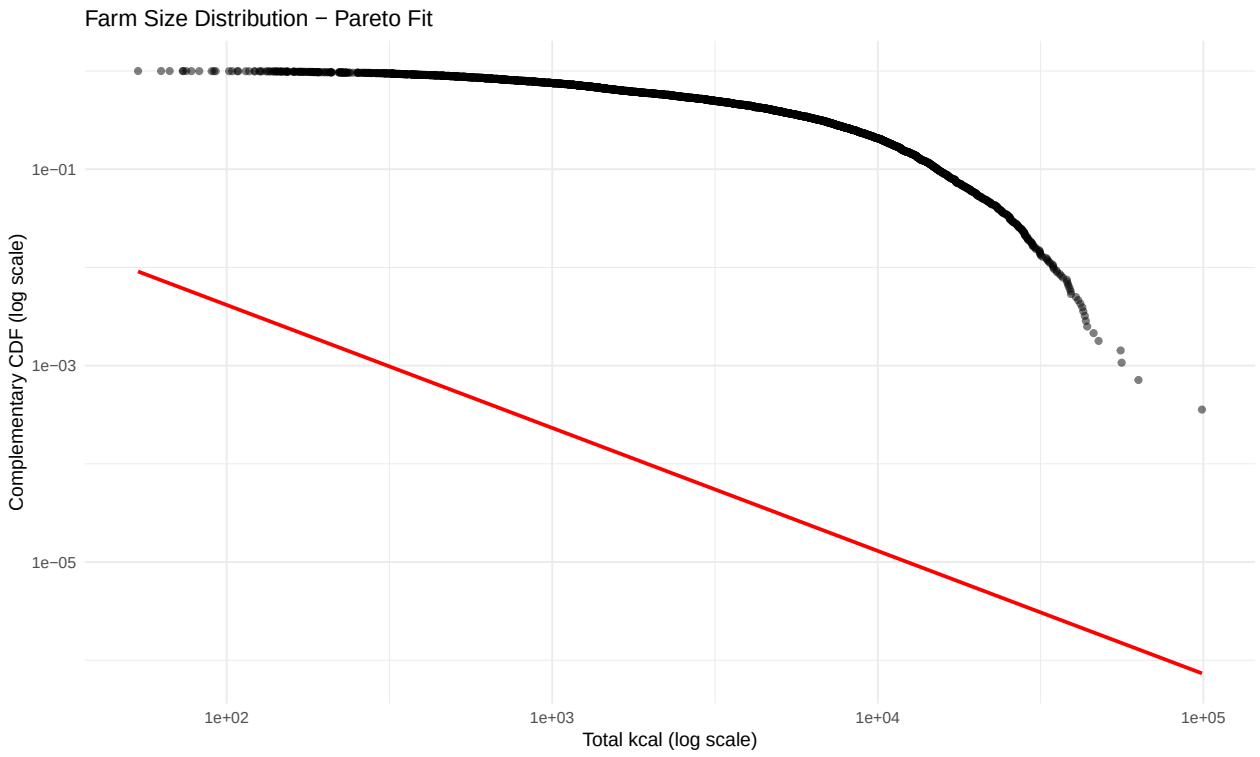
```
## [1] "Pareto alpha: 1.25 | KS statistic: 0.836"
```

If the Pareto alpha falls between 1 and 3, with a visual fit where the actual data (black) fits the Pareto distribution (red), and low Kolmogorov-Smirnov statistic (KS), the Pareto approach may be appropriate; but, I need to look into the literature to see if there is a common cutoff for agricultural economics literature.

Generally, a lower KS statistic indicates a better fit. In practice:

- KS < 0.05: Excellent fit
- KS < 0.1: Good fit
- KS > 0.1: Poor fit

```
pareto_results$plot
```



5 Multilevel Regression with Poststratification (MRP)

```
library(brms)
library(tidybayes)

# Prepare data for MRP
prepare_mrp_data <- function(data, observed_countries_pct = 0.4) {
  # Determine observed and missing countries
  n_countries <- length(unique(data$country_id))
  n_observed <- floor(n_countries * observed_countries_pct)
  observed_countries <- sample(unique(data$country_id), n_observed)

  # Complete data for observed countries
  observed_data <- data %>%
    filter(country_id %in% observed_countries) %>%
    mutate(log_kcal = log(kcal_true))

  # Country and crop totals for countries with missing farm size data
  country_crop_totals <- data %>%
    filter(!(country_id %in% observed_countries)) %>%
    group_by(country_id, country_name, region, development_index, crop) %>%
    summarise(kcal_total = sum(kcal_true), .groups = "drop") %>%
    mutate(farm_size = "TOTAL", log_kcal = log(kcal_total))

  # Poststratification frame
  poststrat_frame <- data %>%
    filter(!(country_id %in% observed_countries)) %>%
    select(country_id, country_name, region, development_index, crop, farm_size)
```

```
  return(list(
    observed_data = observed_data,
    country_crop_totals = country_crop_totals,
    poststrat_frame = poststrat_frame
  ))
}

# Cross-validation function
cross_validate_mrp <- function(data, k_folds = 5) {
  # Prepare data
  data_prep <- prepare_mrp_data(data)
  observed_data <- data_prep$observed_data

  # Calculate mean of the target variable
  target_mean <- mean(exp(observed_data$log_kcal))

  # Create folds
  set.seed(123)
  observed_data$fold <- sample(1:k_folds, nrow(observed_data), replace = TRUE)

  # Store cross-validation results
  cv_results <- list()

  # Perform k-fold cross-validation
  for(i in 1:k_folds) {
    # Split data
    train_data <- observed_data %>% filter(fold != i)
    test_data <- observed_data %>% filter(fold == i)

    # Fit model
    model_formula <- bf(log_kcal ~ farm_size + crop + region + development_index +
                        (1|country_id) + (1|farm_size:region) + (1|crop:farm_size))

    fit_mrp <- brm(
      formula = model_formula,
      data = train_data,
      family = gaussian(),
      chains = 2,
      iter = 1000,
      warmup = 500,
      seed = 123
    )

    # Predict on test data
    predictions <- posterior_epred(
      fit_mrp,
      newdata = test_data,
      re_formula = NULL
    )

    # Calculate metrics
    predicted_values <- exp(apply(predictions, 2, mean))
    actual_values <- exp(test_data$log_kcal)
```

```
    cv_results[[i]] <- list(
      rmse = sqrt(mean((predicted_values - actual_values)^2)),
      mae = mean(abs(predicted_values - actual_values))
    )
  }

  # Aggregate cross-validation results
  rmse_values <- sapply(cv_results, `[[`, "rmse")
  mae_values <- sapply(cv_results, `[[`, "mae")

  return(list(
    rmse = mean(rmse_values),
    mae = mean(mae_values),
    rmse_sd = sd(rmse_values),
    mae_sd = sd(mae_values),
    rmse_pct = mean(rmse_values) / target_mean * 100,
    mae_pct = mean(mae_values) / target_mean * 100,
    target_mean = target_mean
  ))
}

# Perform cross-validation
cv_results <- cross_validate_mrp(synthetic_data)
print("Cross-Validation Results:")
print(cv_results)
```

Generally, RMSE and MAE below 20% of the mean target variable are considered good, while values above 30% suggest the need for model refinement; lower standard deviation across folds indicates more consistent and reliable predictions. But, these are fuzzy thresholds. We should look at the specific context of the problem, the complexity of the data, the potential impact of prediction errors, and conduct further diagnostic checks such as residual analysis, feature importance, and comparison with our domain expertise (aka gut checks).

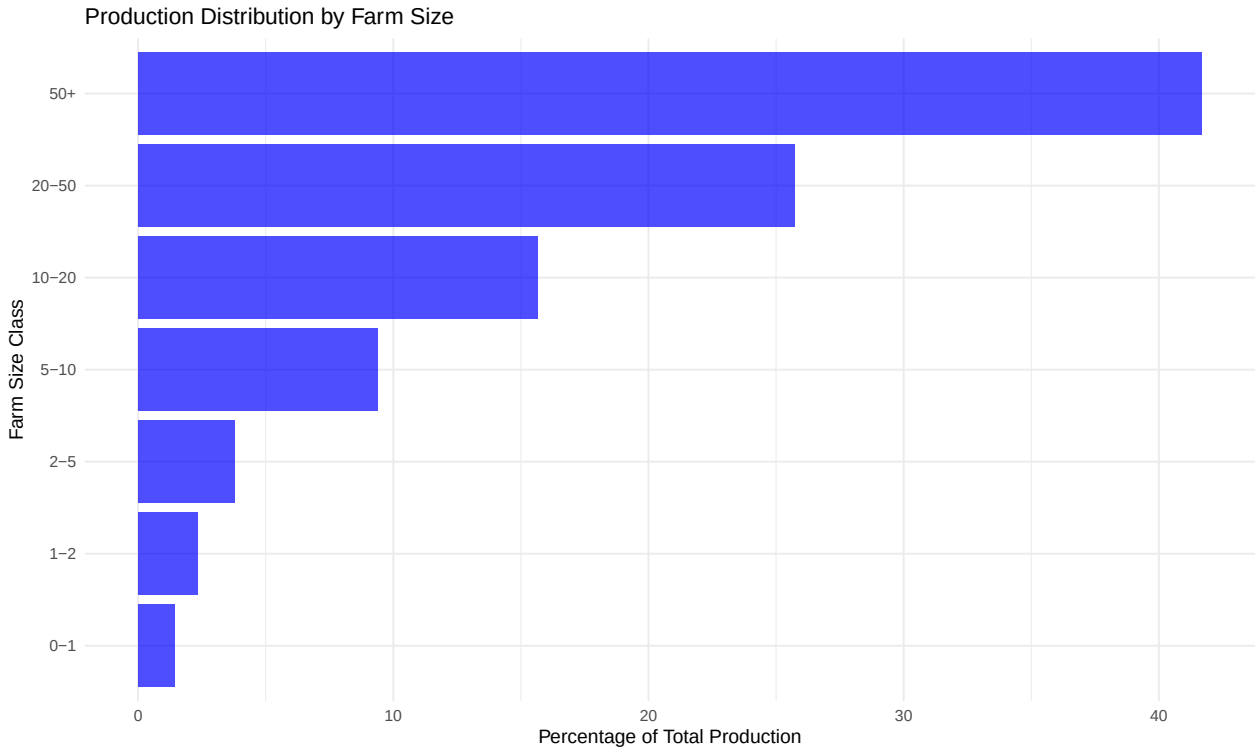
6 Comparison and Visualization

```
# Aggregate results by farm size
pareto_aggregation <- synthetic_data %>%
  group_by(farm_size) %>%
  summarise(
    total_kcal = sum(kcal_true),
    pct_production = total_kcal / sum(synthetic_data$kcal_true) * 100
  )

# Visualization
library(ggplot2)

# Production distribution by farm size
ggplot(pareto_aggregation, aes(x = farm_size, y = pct_production)) +
  geom_bar(stat = "identity", fill = "blue", alpha = 0.7) +
  labs(
    title = "Production Distribution by Farm Size",
    x = "Farm Size Class",
```

```
    y = "Percentage of Total Production"
  ) +
  theme_minimal() +
  coord_flip()
```




```
# Print aggregated results
print(pareto_aggregation)
```

```
## # A tibble: 7 x 3
##   farm_size total_kcal pct_production
##   <fct>          <dbl>          <dbl>
## 1 0-1          244734.           1.45
## 2 1-2          396471.           2.34
## 3 2-5          639077.           3.78
## 4 5-10        1588363.           9.39
## 5 10-20       2647530.          15.6
## 6 20-50       4352288.          25.7
## 7 50+         7055155.          41.7
```

6.1 Things to keep in mind

1. Test distributional assumptions (Pareto fit)
2. Use cross-validation to assess model performance
3. Be prepared to adjust model specifications based on data characteristics
4. We could report on MRP and have this document as a SI - it might help our paper push and share emerging methods while comparing against the commonly used Pareto method.

7 Next Steps

- Replace synthetic data with actual farm production data

- Validate assumptions
- Refine model specifications
- Interpret results in context of specific agricultural system