How is my family support? I would be measuring how often my family contacts me over a couple of months, be it mom, dad, or sister. How long does it take for family to check up on me and want to hear of my day.

Simple Linear Regression in R

**In A Society of Strangers, Kin Is Still Key: Identified Family Relations In Large-Scale Mobile Phone Data**

* [Tam'as D'avid-Barrett](https://www.semanticscholar.org/author/Tam'as-D'avid-Barrett/2221315179), [Sebastian Diaz](https://www.semanticscholar.org/author/Sebastian-Diaz/2221314724), +4 authors [Loreto Bravo](https://www.semanticscholar.org/author/Loreto-Bravo/2055369949)
* Published in [arXiv.org](https://www.semanticscholar.org/venue?name=arXiv.org) 7 July 2023
* Sociology

TLDR

Kinship matters even when demographic processes, such as low fertility, urbanisation and migration reduce the access to family members, and the results provide tools for distinguishing between different kinds of kin relationships from mobile call data, when information about names are unavailable.

Abstract

Mobile call networks have been widely used to investigate communication patterns and the network of interactions of humans at the societal scale. Yet, more detailed analysis is often hindered by having no information about the nature of the relationships, even if some metadata about the individuals are available. Using a unique, large mobile phone database with information about individual surnames in a population in which people inherit two surnames: one from their father, and one from their mother, we are able to differentiate among close kin relationship types. Here we focus on the difference between the most frequently called alters depending on whether they are family relationships or not. We find support in the data for two hypotheses: (1) phone calls between family members are more frequent and last longer than phone calls between non-kin, and (2) the phone call pattern between family members show a higher variation depending on the stage of life-course compared to non-family members. We give an interpretation of these findings within the framework of evolutionary anthropology: kinship matters even when demographic processes, such as low fertility, urbanisation and migration reduce the access to family members. Furthermore, our results provide tools for distinguishing between different kinds of kin relationships from mobile call data, when information about names are unavailable.

**Sharing Family Life Information Through Video Calls and Other Information and Communication Technologies and the Association With Family Well-Being: Population-Based Survey**

Authors of this article:

[Chen Shen1](https://mental.jmir.org/search?term=Chen%20Shen&type=author&precise=true) ;  [Man Ping Wang2](https://mental.jmir.org/search?term=Man%20Ping%20Wang&type=author&precise=true) ;  [Joanna TW Chu1](https://mental.jmir.org/search?term=Joanna%20TW%20Chu&type=author&precise=true) ;  [Alice Wan1](https://mental.jmir.org/search?term=Alice%20Wan&type=author&precise=true)

 ;  [Kasisomayajula Viswanath](https://mental.jmir.org/search?term=Kasisomayajula%20Viswanath&type=author&precise=true)[3, 4](https://mental.jmir.org/search?term=Kasisomayajula%20Viswanath&type=author&precise=true) ;  [Sophia Siu Chee Chan2](https://mental.jmir.org/search?term=Sophia%20Siu%20Chee%20Chan&type=author&precise=true) ;  [Tai Hing Lam1](https://mental.jmir.org/search?term=Tai%20Hing%20Lam&type=author&precise=true)

# Load necessary libraries

library(readxl) # For reading excel files

library(ggplot2) # For creating plots

library(broom) # For tidying model output

# Step 1: Read the data

# Read the excel file named "path"

# The file should have two columns: "frequency" and "duration"

path <- read\_excel("C:/Users/Beboj/OneDrive/Documents/Family support calls.xlsx")

# Print the first few rows of the data to verify it's loaded correctly

print(head(path))

# Step 2: Explore the data

# Calculate summary statistics for both variables

summary(path)

# Create a scatter plot to visualize the relationship between frequency and duration

ggplot(path, aes(x = frequency, y = duration)) +

geom\_point() +

labs(title = "Family Call Duration vs. Frequency",

x = "Frequency of Calls",

y = "Duration of Calls") +

theme\_minimal()

# Step 3: Perform simple linear regression

# Fit a linear model where duration is the dependent variable and frequency is the independent variable

model <- lm(duration ~ dad + mom + sister, data = path)

# Step 4: Examine the model results

# Print a summary of the model, including coefficients, R-squared, and p-values

summary(model)

# Use broom to get a tidy version of the model coefficients

tidy\_model <- tidy(model)

print(tidy\_model)

# Step 5: Visualize the regression line

ggplot(path, aes(x = frequency, y = duration)) +

geom\_point() +

geom\_smooth(method = "lm", se = FALSE, color = "red") +

labs(title = "Linear Regression: Call Duration vs. Frequency",

x = "Frequency",

y = "Call Duration") +

theme\_minimal()

# Step 6: Model diagnostics

# Create diagnostic plots to check assumptions

par(mfrow = c(2, 2)) # Set up a 2x2 plot layout

plot(model)

# Step 7: Predictions

# Create a data frame with new frequency values for prediction

new\_times <- data.frame(frequency = seq(min(path$frequency), max(path$frequency), length.out = 100))

# Make predictions using the model

predictions <- predict(model, newdata = new\_times, interval = "confidence")

# Combine predictions with new\_times

prediction\_data <- cbind(new\_times, predictions)

# Plot the original data, regression line, and confidence interval

ggplot() +

geom\_point(data = path, aes(x = frequency, y = duration)) +

geom\_line(data = prediction\_data, aes(x = frequency, y = fit), color = "blue") +

geom\_ribbon(data = prediction\_data, aes(x = frequency, ymin = lwr, ymax = upr), alpha = 0.2) +

labs(title = "Linear Regression with Confidence Interval",

x = "Frequency",

y = "Call Duration") +

theme\_minimal()

# Print the R-squared value

cat("R-squared:", summary(model)$r.squared, "\n")

# Print the equation of the regression line

cat("Regression equation: duration =",

round(coef(model)[1], 3), "+",

round(coef(model)[2], 3), "\* dad +",

round(coef(model)[2], 3), "\* mom +",

round(coef(model)[2], 3), "\* sister +",

Y = b0 + b1x1 + b2X2 + b3x3

Data File time

# Load necessary libraries

library(readr) # For reading CSV files

library(ggplot2) # For creating plots

library(broom) # For tidying model output

# Step 1: Read the data

# Read the CSV file named "walking\_data.csv"

# The file should have two columns: "time" and "speed"

walking\_data <- read\_csv("walking\_data.csv")

# Print the first few rows of the data to verify it's loaded correctly

print(head(walking\_data))

# Step 2: Explore the data

# Calculate summary statistics for both variables

summary(walking\_data)

# Create a scatter plot to visualize the relationship between time and speed

ggplot(walking\_data, aes(x = time, y = speed)) +

geom\_point() +

labs(title = "Scatter Plot of Walking Speed vs. Time",

x = "Time",

y = "Walking Speed") +

theme\_minimal()

# Step 3: Perform simple linear regression

# Fit a linear model where speed is the dependent variable and time is the independent variable

model <- lm(speed ~ time, data = walking\_data)

# Step 4: Examine the model results

# Print a summary of the model, including coefficients, R-squared, and p-values

summary(model)

# Use broom to get a tidy version of the model coefficients

tidy\_model <- tidy(model)

print(tidy\_model)

# Step 5: Visualize the regression line

ggplot(walking\_data, aes(x = time, y = speed)) +

geom\_point() +

geom\_smooth(method = "lm", se = FALSE, color = "red") +

labs(title = "Linear Regression: Walking Speed vs. Time",

x = "Time",

y = "Walking Speed") +

theme\_minimal()

# Step 6: Model diagnostics

# Create diagnostic plots to check assumptions

par(mfrow = c(2, 2)) # Set up a 2x2 plot layout

plot(model)

# Step 7: Predictions

# Create a data frame with new time values for prediction

new\_times <- data.frame(time = seq(min(walking\_data$time), max(walking\_data$time), length.out = 100))

# Make predictions using the model

predictions <- predict(model, newdata = new\_times, interval = "confidence")

# Combine predictions with new\_times

prediction\_data <- cbind(new\_times, predictions)

# Plot the original data, regression line, and confidence interval

ggplot() +

geom\_point(data = walking\_data, aes(x = time, y = speed)) +

geom\_line(data = prediction\_data, aes(x = time, y = fit), color = "blue") +

geom\_ribbon(data = prediction\_data, aes(x = time, ymin = lwr, ymax = upr), alpha = 0.2) +

labs(title = "Linear Regression with Confidence Interval",

x = "Time",

y = "Walking Speed") +

theme\_minimal()

# Print the R-squared value

cat("R-squared:", summary(model)$r.squared, "\n")

# Print the equation of the regression line

cat("Regression equation: speed =",

round(coef(model)[1], 3), "+",

round(coef(model)[2], 3), "\* time\n")

New one

# Load necessary libraries

library(readr) # For reading CSV files

library(ggplot2) # For creating plots

library(broom) # For tidying model output

library(gganimate) # For creating animations

# Step 1: Read the data

data <- read\_csv("Family support calls.csv")

# Remove rows with NA values in `walking\_data`

data <- na.omit(data)

# Step 3: Perform simple linear regressions

model\_dad <- lm(dad ~ duration, data = data)

model\_mom <- lm(mom ~ duration, data = data)

model\_sister <- lm(sister ~ duration, data = data)

# Step 7: Predictions for each model

new\_times <- data.frame(duration = seq(min(data$duration), max(data$duration), length.out = 100))

predictions\_dad <- predict(model\_dad, newdata = new\_times, interval = "confidence")

predictions\_mom <- predict(model\_mom, newdata = new\_times, interval = "confidence")

predictions\_sister <- predict(model\_sister, newdata = new\_times, interval = "confidence")

# Combine predictions with new\_times and include a column for variable name

prediction\_data\_dad <- cbind(new\_times, predictions\_dad, variable = "dad")

prediction\_data\_mom <- cbind(new\_times, predictions\_mom, variable = "mom")

prediction\_data\_sister <- cbind(new\_times, predictions\_sister, variable = "sister")

# Combine the three prediction datasets

prediction\_data <- rbind(prediction\_data\_dad, prediction\_data\_mom, prediction\_data\_sister)

# Step 8: Create a time variable for animation purposes

prediction\_data$time <- rep(seq(1, 100), 3)

# Create the plot

p <- ggplot(data, aes(x = duration)) +

# Scatter points for original data

geom\_point(aes(y = dad), color = "blue", alpha = 0.6) +

geom\_point(aes(y = mom), color = "purple", alpha = 0.6) +

geom\_point(aes(y = sister), color = "pink", alpha = 0.6) +

# Add the regression lines and ribbons for confidence intervals

geom\_line(data = prediction\_data, aes(y = fit, color = variable), size = 1) +

geom\_ribbon(data = prediction\_data, aes(ymin = lwr, ymax = upr, fill = variable), alpha = 0.2) +

labs(title = "Regression Lines for Call Duration with Dad, Mom, and Sister",

x = "Duration of Call",

y = "Call Duration",

color = "Variable",

fill = "Variable") +

theme\_minimal() +

# Set color and fill options for the three groups

scale\_color\_manual(values = c("dad" = "blue", "mom" = "purple", "sister" = "pink")) +

scale\_fill\_manual(values = c("dad" = "blue", "mom" = "purple", "sister" = "pink")) +

# Use gganimate to animate over time

transition\_time(time) +

ease\_aes('linear') # This line ensures smooth transition

# Render the animation

animate(p, nframes = 100, width = 600, height = 400)

like this one

# Load necessary libraries

library(readr) # For reading CSV files

library(ggplot2) # For creating plots

library(broom) # For tidying model output

library(gganimate) # For creating animations

# Step 1: Read the data

data <- read\_csv("Family support calls.csv")

# Remove rows with NA values in `walking\_data`

data <- na.omit(data)

# Step 3: Perform simple linear regressions

model\_dad <- lm(dad ~ duration, data = data)

model\_mom <- lm(mom ~ duration, data = data)

model\_sister <- lm(sister ~ duration, data = data)

# Step 7: Predictions for each model

new\_times <- data.frame(duration = seq(min(data$duration), max(data$duration), length.out = 100))

predictions\_dad <- predict(model\_dad, newdata = new\_times, interval = "confidence")

predictions\_mom <- predict(model\_mom, newdata = new\_times, interval = "confidence")

predictions\_sister <- predict(model\_sister, newdata = new\_times, interval = "confidence")

# Combine predictions with new\_times and include a column for variable name

prediction\_data\_dad <- cbind(new\_times, predictions\_dad, variable = "dad")

prediction\_data\_mom <- cbind(new\_times, predictions\_mom, variable = "mom")

prediction\_data\_sister <- cbind(new\_times, predictions\_sister, variable = "sister")

# Combine the three prediction datasets

prediction\_data <- rbind(prediction\_data\_dad, prediction\_data\_mom, prediction\_data\_sister)

# Step 8: Create a time variable for animation purposes

prediction\_data$time <- rep(seq(1, 100), 3)

# Create the plot

p <- ggplot(data, aes(x = duration)) +

# Scatter points for original data

geom\_point(aes(y = dad), color = "blue", alpha = 0.6) +

geom\_point(aes(y = mom), color = "purple", alpha = 0.6) +

geom\_point(aes(y = sister), color = "pink", alpha = 0.6) +

# Add the regression lines and ribbons for confidence intervals

geom\_line(data = prediction\_data, aes(y = fit, color = variable), size = 1) +

geom\_ribbon(data = prediction\_data, aes(ymin = lwr, ymax = upr, fill = variable), alpha = 0.2) +

labs(title = "Regression Lines for Call Duration with Dad, Mom, and Sister",

x = "Duration of Call",

y = "Call Duration",

color = "Variable",

fill = "Variable") +

theme\_minimal() +

# Set color and fill options for the three groups

scale\_color\_manual(values = c("dad" = "blue", "mom" = "purple", "sister" = "pink")) +

scale\_fill\_manual(values = c("dad" = "blue", "mom" = "purple", "sister" = "pink")) +

# Use gganimate to animate over time

transition\_time(time) +

ease\_aes('linear') # This line ensures smooth transition

# Render the animation

animate(p, nframes = 100, width = 600, height = 400)