How to Easily Understand the Difference Between glm and lm in R

Authored by

stats writer

December 6, 2025

Understanding the fundamental differences between statistical modeling functions is crucial for effective data analysis in the R programming environment. Two of the most frequently used functions for fitting regression models are lm and glm. While both serve the purpose of modeling the relationship between predictors and a response variable, they operate under distinctly different sets of assumptions regarding the error distribution and the nature of the response variable itself.

The core distinction lies in the concept of the Generalized Linear Model (GLM). The `lm` function is specifically designed to fit standard Linear Models, which necessitate that the response variable errors follow a Normal Distribution and utilize an identity Link Function. Conversely, the `glm` function represents a powerful generalization of this framework, allowing analysts to fit models where the response variable may follow a non-normal distribution (such as Poisson or Binomial) and employ various non-identity link functions (such as logit or log). This flexibility makes `glm` an indispensable tool for modeling diverse data types, ranging from binary outcomes to count data.

The R programming language offers several foundational functions for regression analysis, but `lm` and `glm` are primary workhorses. Understanding their precise implementation details allows researchers to select the most appropriate method for their specific data structure and modeling objectives.

The Linear Model (lm) Function in R

The `lm()` function is the standard tool for fitting classical Linear Models, often referred to as Ordinary Least Squares (OLS) regression. This function is foundational in classical statistics and is used when the relationship between the predictors and the response is assumed to be linear, and the residuals are assumed to be independent and identically distributed following a Normal Distribution. These stringent assumptions ensure that the estimates derived are Best Linear Unbiased Estimators (BLUE).

When employing `lm`, we implicitly assume that the expected value of the response variable is directly equal to the linear combination of the predictors, as this function uses the identity Link Function by default. Furthermore, the variance of the errors is assumed to be constant (homoscedasticity). The simplicity and straightforward interpretation of the output make `lm` the first choice for continuous, normally distributed data.

The syntax for the `lm` function is concise and focused primarily on defining the relationship between variables:

lm(formula, data, ...)

where:

formula: Specifies the model structure, defining the dependent variable and the independent variables (e.g., y ~ x1 + x2).
data: Identifies the name of the R data frame that contains all the variables referenced in the formula.

The Generalized Linear Model (glm) Function in R

The `glm()` function is significantly more versatile, designed to fit models that extend beyond the constraints of the standard Linear Model. A GLM consists of three key components: a random component (specifying the probability distribution of the response variable), a systematic component (the linear predictor, similar to the `lm` structure), and a Link Function (connecting the expected value of the response to the linear predictor). This structure allows `glm` to handle response variables that are counts, proportions, or binary outcomes.

By allowing the user to specify the statistical family, `glm` enables fitting models where the variance is not constant but depends on the mean, which is characteristic of non-normal distributions like the Poisson or Binomial distributions. For instance, when modeling binary outcomes, the Binomial family is used, and the logit Link Function transforms the probability of success (which is bound between 0 and 1) into a linear scale suitable for modeling.

The syntax for the `glm` function expands upon the `lm` structure by requiring the crucial `family` argument:

glm(formula, family=gaussian, data, ...)

where:

formula: Defines the relationship between the response and predictors, identical in structure to the `lm` formula.
family: This is the defining argument. It specifies the statistical distribution and the default canonical link function to use to fit the model. The default setting is gaussian, but other widely used options include binomial (for logistic regression), Gamma, and poisson (for count data).
data: The name of the data frame containing the variables.

The Critical Role of the Family Argument

The sole explicit difference in the standard function call structure between `lm()` and `glm()` is the inclusion of the **family** argument in `glm()`. This argument dictates the distribution assumed for the response variable and the transformation used by the link function. For standard linear regression, which assumes Normal Distribution errors, the appropriate family setting is `family=gaussian`.

Crucially, if you use the `glm()` function and explicitly set the family argument to the default

gaussian (or omit it, as it is the default), you are effectively asking `glm()` to perform the exact same computation as `lm()`. Both functions will calculate the model parameters using Ordinary Least Squares estimation, thereby guaranteeing that **they will produce the exact same numerical results** for linear regression.

However, the true power of `glm()` emerges when we shift away from the Gaussian assumption. This allows us to fit complex models that are otherwise impossible using `lm()`, adapting the model structure to the inherent properties of the response data.

Examples of non-Gaussian models readily handled by `glm()` include:

Logistic Regression (family=binomial), suitable for binary response variables (0/1 outcomes).
Poisson Regression (family=poisson), ideal for count data, such as the number of events occurring over a fixed interval.

Practical Demonstration: lm() vs. glm(family=gaussian)

To illustrate the equivalence when modeling standard linear relationships, we will use the built-in `mtcars` dataset in R. We aim to fit a multiple linear regression model predicting miles per gallon (`mpg`) based on displacement (`disp`) and horsepower (`hp`). The following examples show how to use the `lm()` function and then replicate the results perfectly using the `glm()` function.

Example of Using the lm() Function

The `lm()` function provides a clean summary focused on the quality of the least squares fit, including the residuals, coefficient estimates, standard errors, and overall model fit statistics like R-squared.

#fit multiple linear regression model
model <- lm(mpg ~ disp + hp, data=mtcars)

#view model summary
summary(model)

Call:
lm(formula = mpg ~ disp + hp, data = mtcars)

Residuals:
Min 1Q Median 3Q Max
-4.7945 -2.3036 -0.8246 1.8582 6.9363

Coefficients:

Estimate Std. Error t value Pr(>|t|)
(Intercept) 30.735904 1.331566 23.083 < 2e-16 ***
disp -0.030346 0.007405 -4.098 0.000306 ***
hp -0.024840 0.013385 -1.856 0.073679 .

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.127 on 29 degrees of freedom
Multiple R-squared: 0.7482, Adjusted R-squared: 0.7309
F-statistic: 43.09 on 2 and 29 DF, p-value: 2.062e-09

Example of Using the glm() Function for Linear Regression

The following code demonstrates how to fit the exact same linear regression model using the `glm()` function by explicitly or implicitly setting the family to gaussian. Since Gaussian errors imply the identity link function, the mathematical procedure used to derive the parameter estimates is identical to that used by `lm()`.

#fit multiple linear regression model (family=gaussian is default/implicit)
model <- glm(mpg ~ disp + hp, data=mtcars)

#view model summary
summary(model)

Call:
glm(formula = mpg ~ disp + hp, data = mtcars)

Deviance Residuals:
Min 1Q Median 3Q Max
-4.7945 -2.3036 -0.8246 1.8582 6.9363

Coefficients:
Estimate Std. Error t value Pr(>|t|)
(Intercept) 30.735904 1.331566 23.083 < 2e-16 ***
disp -0.030346 0.007405 -4.098 0.000306 ***
hp -0.024840 0.013385 -1.856 0.073679 .

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 9.775636)

Null deviance: 1126.05 on 31 degrees of freedom
Residual deviance: 283.49 on 29 degrees of freedom
AIC: 168.62

Number of Fisher Scoring iterations: 2

Comparing the two outputs, one immediately notices that the coefficient estimates (Intercept, disp, hp) and their corresponding standard errors are exactly the same across both models. The primary difference in the output summary is that `glm()` reports statistics like Deviance Residuals, Null Deviance, Residual Deviance, and AIC, which are specific metrics used in the context of the Generalized Linear Model framework, even when the family is Gaussian. The output also explicitly reports the Dispersion parameter for the gaussian family, which is estimated from the residual sum of squares divided by the degrees of freedom.

Fitting Non-Gaussian Models: The True Utility of glm()

The true advantage of the `glm()` function becomes clear when the response variable violates the assumptions of normality, necessitating the use of different statistical distributions. We now demonstrate two crucial applications where `glm()` is mandatory: Logistic Regression and Poisson Regression.

Example: Logistic Regression using glm(family=binomial)

When the response variable is binary (e.g., 0 or 1, success or failure), the Binomial distribution is appropriate, and we use Logistic Regression. The canonical link function for the binomial family is the logit link, which transforms the probability of the outcome occurring into a range suitable for linear modeling. In this example, we model the probability of automatic transmission (`am`) based on displacement and horsepower.

#fit logistic regression model
model <- glm(am ~ disp + hp, data=mtcars, family=binomial)

#view model summary
summary(model)

Call:
glm(formula = am ~ disp + hp, family = binomial, data = mtcars)

Deviance Residuals:
Min 1Q Median 3Q Max
-1.9665 -0.3090 -0.0017 0.3934 1.3682

Coefficients:
Estimate Std. Error z value Pr(>|z|)
(Intercept) 1.40342 1.36757 1.026 0.3048
disp -0.09518 0.04800 -1.983 0.0474 *
hp 0.12170 0.06777 1.796 0.0725 .

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 43.230 on 31 degrees of freedom
Residual deviance: 16.713 on 29 degrees of freedom
AIC: 22.713

Number of Fisher Scoring iterations: 8

Note that for the Binomial family, the test statistics switch from t-values to z-values, reflecting the large-sample asymptotic properties of maximum likelihood estimation used in GLMs, unlike the small-sample assumptions often used in OLS regression. Additionally, the Dispersion parameter is fixed at 1 for the Binomial distribution. The coefficient estimates here represent the change in the log-odds of the outcome for a unit increase in the predictor, a clear departure from the interpretation of standard linear coefficients.

Example: Poisson Regression using glm(family=poisson)

When the response variable is count data (non-negative integers representing frequencies or occurrences), the Poisson distribution is appropriate. This is commonly used in epidemiology, quality control, or analyzing event counts. For the Poisson family, the canonical Link Function is the log link. Here, we model the relationship between the predictors and the log of the expected count.

#fit Poisson regression model
model <- glm(am ~ disp + hp, data=mtcars, family=poisson)

#view model summary
summary(model)

Call:
glm(formula = am ~ disp + hp, family = poisson, data = mtcars)

Deviance Residuals:

Min 1Q Median 3Q Max
-1.1266 -0.4629 -0.2453 0.1797 1.5428

Coefficients:
Estimate Std. Error z value Pr(>|z|)
(Intercept) 0.214255 0.593463 0.361 0.71808
disp -0.018915 0.007072 -2.674 0.00749 **
hp 0.016522 0.007163 2.307 0.02107 *

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

Null deviance: 23.420 on 31 degrees of freedom
Residual deviance: 10.526 on 29 degrees of freedom
AIC: 42.526

Number of Fisher Scoring iterations: 6

In Poisson Regression, the coefficient estimates are interpreted multiplicatively on the original scale; specifically, they represent the change in the log of the expected count. Like the Binomial family, the Poisson family assumes a Dispersion parameter fixed at 1 (equidispersion). If the data exhibit overdispersion (variance greater than the mean), alternative functions or families, such as the quasipoisson or negative binomial (using dedicated packages like MASS), would be required.

Summary of Functional Differences

In essence, `lm()` is a specialized, restricted case of `glm()`. The `lm()` function is optimized for speed and simplicity when dealing with continuous data that meet the normal distribution assumption. The `glm()` function, while slightly more computationally intensive due to its reliance on iterative fitting algorithms (like Fisher Scoring), offers unparalleled flexibility. It allows the analyst to appropriately model data arising from various natural phenomena, ensuring that the model assumptions align accurately with the data-generating process. Researchers should default to `lm()` for traditional linear regression when assumptions are met, but must use `glm()` whenever the response variable requires a non-Gaussian distribution or a non-identity link function.