Asymmetric Laplace and GAL Distributions

Antonio Aguirre

2025-03-06

Introduction

This vignette introduces the **Asymmetric Laplace (AL) distribution** and its generalization, the **GAL (GAL) distribution**. These distributions are useful for modeling **skewed data** and play an important role in **quantile regression, risk analysis, and Bayesian statistics**.

Asymmetric Laplace (AL) Distribution

The **Asymmetric Laplace (AL) distribution** is defined by a parameter p such that $p \in (0, 1)$.

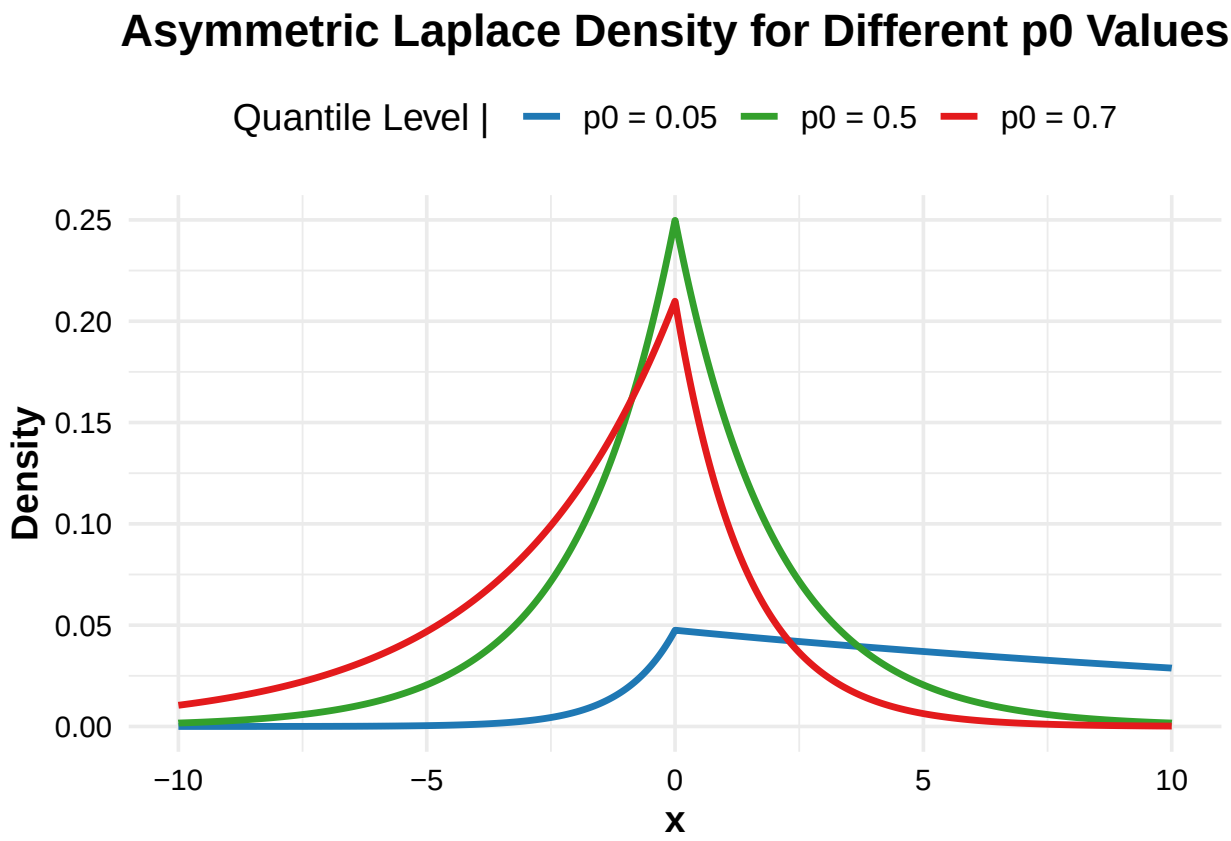
Probability Density Function (PDF)

The AL density function for a standard case ($\mu = 0$, $\sigma = 1$) is:

$$f_{\text{AL}}(y \mid p, 0, 1) = p(1-p) \times \begin{cases} \exp(-py), & y \geq 0 \\ \exp((1-p)y), & y < 0 \end{cases}$$

```
# Generate x values
x <- seq(-10, 10, length = 5000)

# Compute AL densities for different p0 values
df <- data.frame(
  x = rep(x, 3),
  y = c(sapply(x, function(x) dexal(x, p0 = 0.5)),
        sapply(x, function(x) dexal(x, p0 = 0.05)),
        sapply(x, function(x) dexal(x, p0 = 0.7))),
  p0 = factor(rep(c("p0 = 0.5", "p0 = 0.05", "p0 = 0.7"), each = length(x)))
)

# Create the plot with a professional style
ggplot(df, aes(x = x, y = y, color = p0)) +
  geom_line(size = 1.2) +
  scale_color_manual(values = c("#1f78b4", "#33a02c", "#e31a1c")) +  # Professional color palette
  labs(title = "Asymmetric Laplace Density for Different p0 Values",
       x = "x",
       y = "Density",
       color = "Quantile Level | ") +
  theme_minimal(base_size = 14) +
```

```r
  theme(
    legend.position = "top",
    plot.title = element_text(hjust = 0.5, face = "bold"),
    axis.title = element_text(face = "bold"),
    axis.text = element_text(color = "black"),
    legend.text = element_text(size = 12)
  )
```



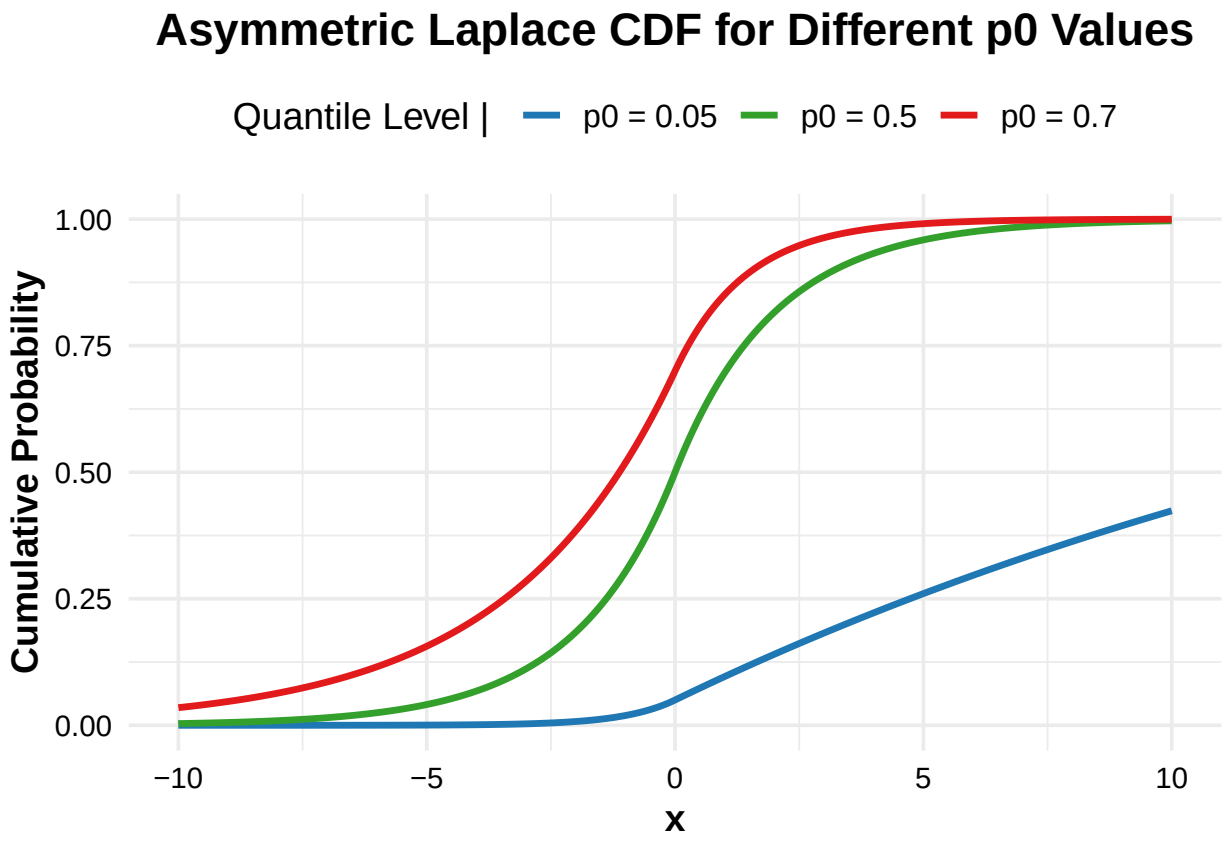
Cumulative Distribution Function (CDF)

The CDF is given by:

$$F_{\text{AL}}(y \mid p, 0, 1) = \begin{cases} p \exp((1-p)y), & y < 0 \\ 1 - (1-p)\exp(-py), & y \geq 0 \end{cases}$$

```r
# Generate x values
x <- seq(-10, 10, length = 5000)

# Compute AL CDFs for different p_0 values
df <- data.frame(
  x = rep(x, 3),
  y = c(sapply(x, function(x) pexal(x, p0 = 0.5)),
        sapply(x, function(x) pexal(x, p0 = 0.05)),
        sapply(x, function(x) pexal(x, p0 = 0.7))),
  p0 = factor(rep(c("p0 = 0.5", "p0 = 0.05", "p0 = 0.7"), each = length(x)))
```

```
)

# Create the plot with a professional style
ggplot(df, aes(x = x, y = y, color = p0)) +
  geom_line(size = 1.2) +
  scale_color_manual(values = c("#1f78b4", "#33a02c", "#e31a1c")) +  # Professional color palette
  labs(title = "Asymmetric Laplace CDF for Different p0 Values",
       x = "x",
       y = "Cumulative Probability",
       color = "Quantile Level | ") +
  theme_minimal(base_size = 14) +
  theme(
    legend.position = "top",
    plot.title = element_text(hjust = 0.5, face = "bold"),
    axis.title = element_text(face = "bold"),
    axis.text = element_text(color = "black"),
    legend.text = element_text(size = 12)
  )
```



GAL (GAL) Distribution

The **GAL (GAL) distribution** extends the **Asymmetric Laplace (AL) distribution** by incorporating an additional **half-normal distributed component**. This generalization introduces an extra **skewness parameter,** denoted as γ, which governs the asymmetry and spread of the distribution.

The GAL distribution possesses **infinite mixture representations**, which offer different perspectives on its structure and highlight its connections to **normal, exponential, and asymmetric Laplace distributions**.

Mixture Representations of the GAL Distribution

The GAL distribution can be understood as a **hierarchical model**, obtained through the following **three equivalent mixture representations**:

1) AL as a Normal-Exponential Mixture

The **Asymmetric Laplace (AL) distribution** itself can be written as a **normal-exponential mixture**:

$$f_{\text{AL}}(y \mid p, 0, 1) = \int_0^\infty N(y \mid \lambda\tau, \tau) Exp(\tau|1) d\tau$$

where the latent variable τ follows an **exponential(1) distribution**, and:

$$\lambda = \frac{1-2p}{p(1-p)}.$$

2) GAL as an AL-Half-Normal Mixture

The **GAL distribution** generalizes the AL by introducing an additional half-normal component $s \sim N^+(0,1)$:

$$f_{\text{GAL}}(y \mid p, \alpha, 0, 1) = \int_0^\infty f_{\text{AL}}(y - \alpha s \mid p, 0, 1) N^+(s|0,1) ds.$$

This representation shows that the GAL distribution **introduces variability through a half-normal component,** controlling the spread asymmetrically via α.

3) GAL as a Normal-Exponential-Half-Normal Mixture

By combining the **Normal-Exponential Mixture** (from AL) with the **Half-Normal Mixture**, the GAL distribution can also be written as a **double mixture**:

$$f_{\text{GAL}}(y \mid p, \alpha, 0, 1) = \int_0^\infty \int_0^\infty N(y \mid \alpha s + \lambda\tau, \tau) e^{-\tau} Exp(\tau|1) N^+(s|0,1) d\tau ds.$$

This shows that the GAL can be thought of as a **three-component hierarchical model**, with: - An **exponential** latent variable τ, - A **half-normal** latent variable s, - A **normal** component $N(y \mid \alpha s + \lambda\tau, \tau)$.

Reparameterization: Defining the GAL in Terms of gamma and p0

Instead of working directly with (p, α), the GAL distribution can also be expressed in terms of: - $p_0 \in (0, 1)$**:** The quantile level that controls the asymmetry of the distribution. - γ**:** A skewness parameter that determines the spread and tilt of the distribution.

The transformation from (p_0, γ) **to** (p, α) is given by:

$$p = \mathbf{1}(\gamma < 0) + \frac{p_0 - \mathbf{1}(\gamma < 0)}{g(\gamma)}$$

$$\alpha = \frac{|\gamma|}{\mathbf{1}(\gamma > 0) - p}$$

where:

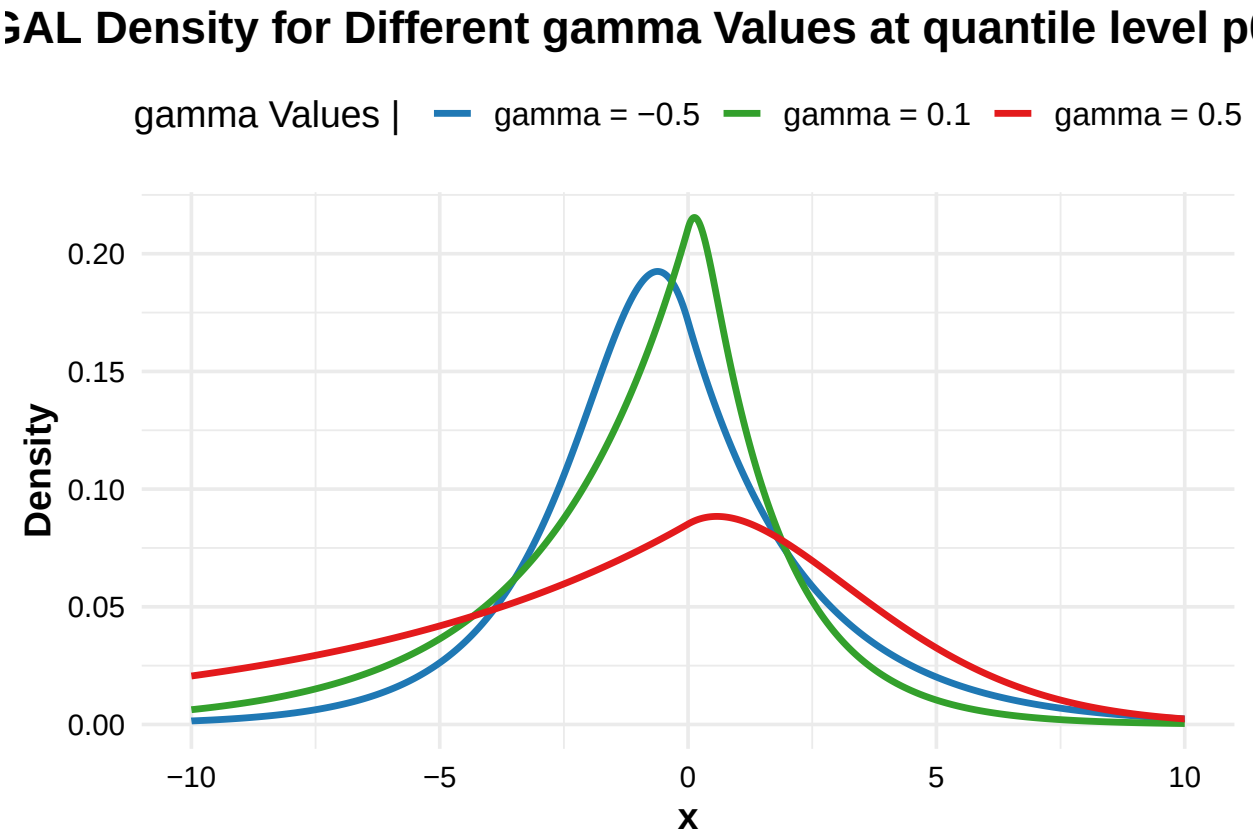
$$g(\gamma) = 2\Phi(-|\gamma|)e^{\gamma^2/2}.$$

The parameter γ is **restricted to the interval** (L, U), where L and U are the respective negative and positive solutions to:

$$g(x) - (1 - p_0) = 0, \quad g(x) - p_0 = 0.$$

ensuring that $L < 0 < U$.

```r
# Generate x values
x <- seq(-10, 10, length = 5000)

# Compute GAL densities for different p_0 and gamma values
df <- data.frame(
  x = rep(x, 3),
  y = c(sapply(x, function(x) dexal(x, p0 = 0.6, gamma = 0.5)),
        sapply(x, function(x) dexal(x, p0 = 0.6, gamma = -0.5)),
        sapply(x, function(x) dexal(x, p0 = 0.6, gamma = 0.1))),
  gamma = factor(rep(c("gamma = 0.5", "gamma = -0.5", "gamma = 0.1"), each = length(x)))
)

# Create the plot with a professional style
ggplot(df, aes(x = x, y = y, color = gamma)) +
  geom_line(size = 1.2) +
  scale_color_manual(values = c("#1f78b4", "#33a02c", "#e31a1c")) +  # Professional color palette
  labs(title = "GAL Density for Different gamma Values at quantile level p0 = 0.6",
       x = "x",
       y = "Density",
       color = "gamma Values | ") +
  theme_minimal(base_size = 14) +
  theme(
    legend.position = "top",
    plot.title = element_text(hjust = 0.5, face = "bold"),
    axis.title = element_text(face = "bold"),
    axis.text = element_text(color = "black"),
    legend.text = element_text(size = 12)
  )
```



Closed-Form Probability Density Function (PDF) of the GAL Distribution

Using the **relationship between** (p, α) **and** (p_0, γ), the GAL **density function** can be expressed in terms of **gamma and p0**.

For $\alpha > 0$:

$$f_{\text{GAL}}(y \mid p, \alpha, 0, 1) = 2p(1-p)\left[\Phi\left(\frac{y}{\alpha} - p\alpha\right) - \Phi(-p\alpha)\right] e^{-py + \frac{1}{2}(p\alpha)^2} + \Phi\left((p-1)\alpha - \frac{y}{\alpha}\right) e^{-(p-1)y + \frac{1}{2}((p-1)\alpha)^2}$$

Notice that α and p are functions of γ and p_0.

For $\alpha < 0$, the density can be derived using the symmetry property:

$$f_{\text{GAL}}(y \mid p, \alpha, 0, 1) = f_{\text{GAL}}(-y \mid 1-p, |\alpha|, 0, 1).$$

Closed-Form cumulative distribution function (CDF) of the GAL

For $\alpha > 0$:

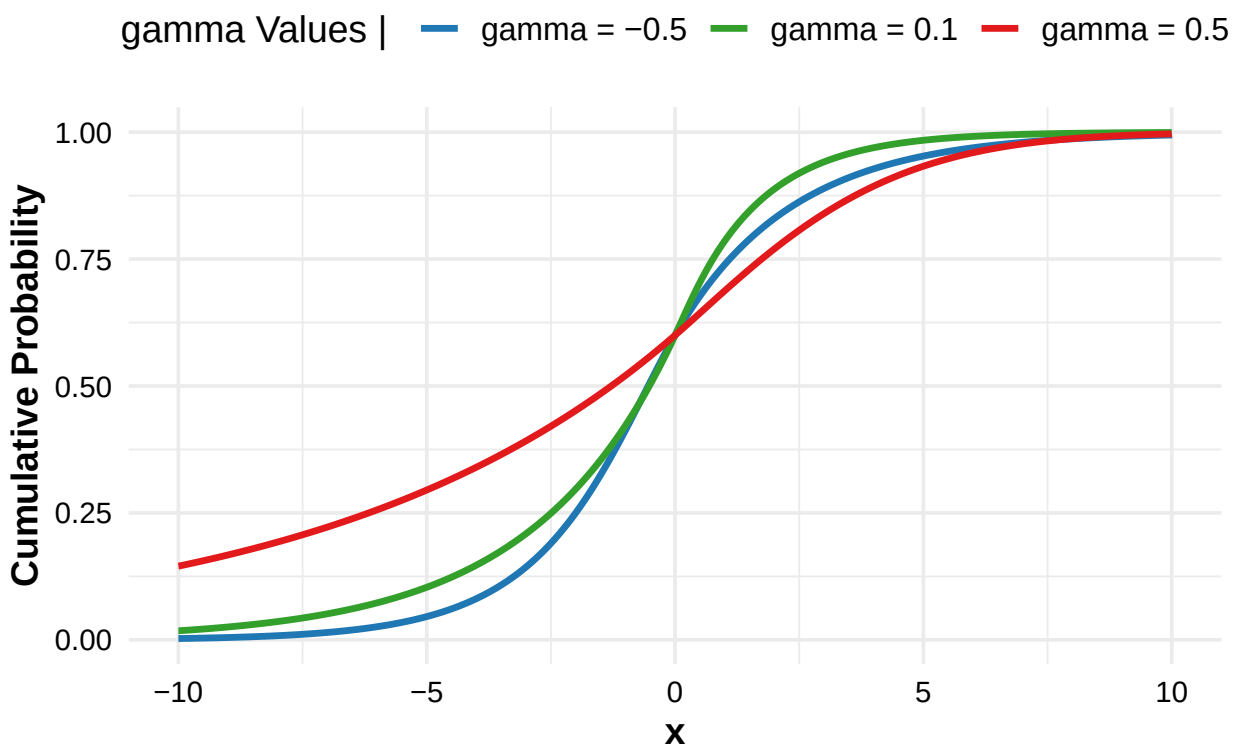
$$F_{\text{GAL}}(y \mid p, \alpha, 0, 1) = 2pe^{\frac{1}{2}((p-1)\alpha)^2 + (1-p)y}\Phi\left(-(1-p)\alpha - \frac{y}{\alpha}\mathbf{1}(\frac{y}{\alpha} > 0)\right)$$

$$+\left[2(1-p)e^{\frac{1}{2}(p\alpha)^2-py}\left(\Phi(-p\alpha)-\Phi\left(\frac{y}{\alpha}-p\alpha\right)\right)+2\Phi\left(\frac{y}{\alpha}\right)-1\right]\mathbf{1}(\frac{y}{\alpha}>0).$$

For $\alpha < 0$, the symmetry property follows:

$$F_{\mathrm{GAL}}(y \mid p, \alpha) = 1 - F_{\mathrm{GAL}}(-y \mid 1-p, |\alpha|).$$

```r
# Generate x values
x <- seq(-10, 10, length = 5000)

# Compute GAL CDFs for different p_0 and gamma values
df <- data.frame(
  x = rep(x, 3),
  y = c(sapply(x, function(x) pexal(x, p0 = 0.6, gamma = 0.5)),
        sapply(x, function(x) pexal(x, p0 = 0.6, gamma = -0.5)),
        sapply(x, function(x) pexal(x, p0 = 0.6, gamma = 0.1))),
  gamma = factor(rep(c("gamma = 0.5", "gamma = -0.5", "gamma = 0.1"), each = length(x)))
)

# Create the plot with a professional style
ggplot(df, aes(x = x, y = y, color = gamma)) +
  geom_line(size = 1.2) +
  scale_color_manual(values = c("#1f78b4", "#33a02c", "#e31a1c")) +  # Professional color palette
  labs(title = "GAL CDF for Different gamma Values at Quantile Level p0 = 0.6",
       x = "x",
       y = "Cumulative Probability",
       color = "gamma Values | ") +
  theme_minimal(base_size = 14) +
  theme(
    legend.position = "top",
    plot.title = element_text(hjust = 0.5, face = "bold"),
    axis.title = element_text(face = "bold"),
    axis.text = element_text(color = "black"),
    legend.text = element_text(size = 12)
  )
```



Introducing Location and Scale Parameters in the GAL Distribution

Like the **Asymmetric Laplace (AL) distribution**, the **GAL distribution belongs to the location-scale family**, meaning that **location (μ) and scale (σ) parameters can be introduced via affine transformations.**

1. Location-Scale Transformation

Let $Y \sim \mathbf{GAL}(p, \alpha, 0, 1)$, i.e., a standard GAL variable with location $\mu = 0$ and scale $\sigma = 1$. We introduce **a new variable** X by applying the transformation:

$$X = \mu + \sigma Y.$$

Then, X follows the general **location-scale GAL distribution**:

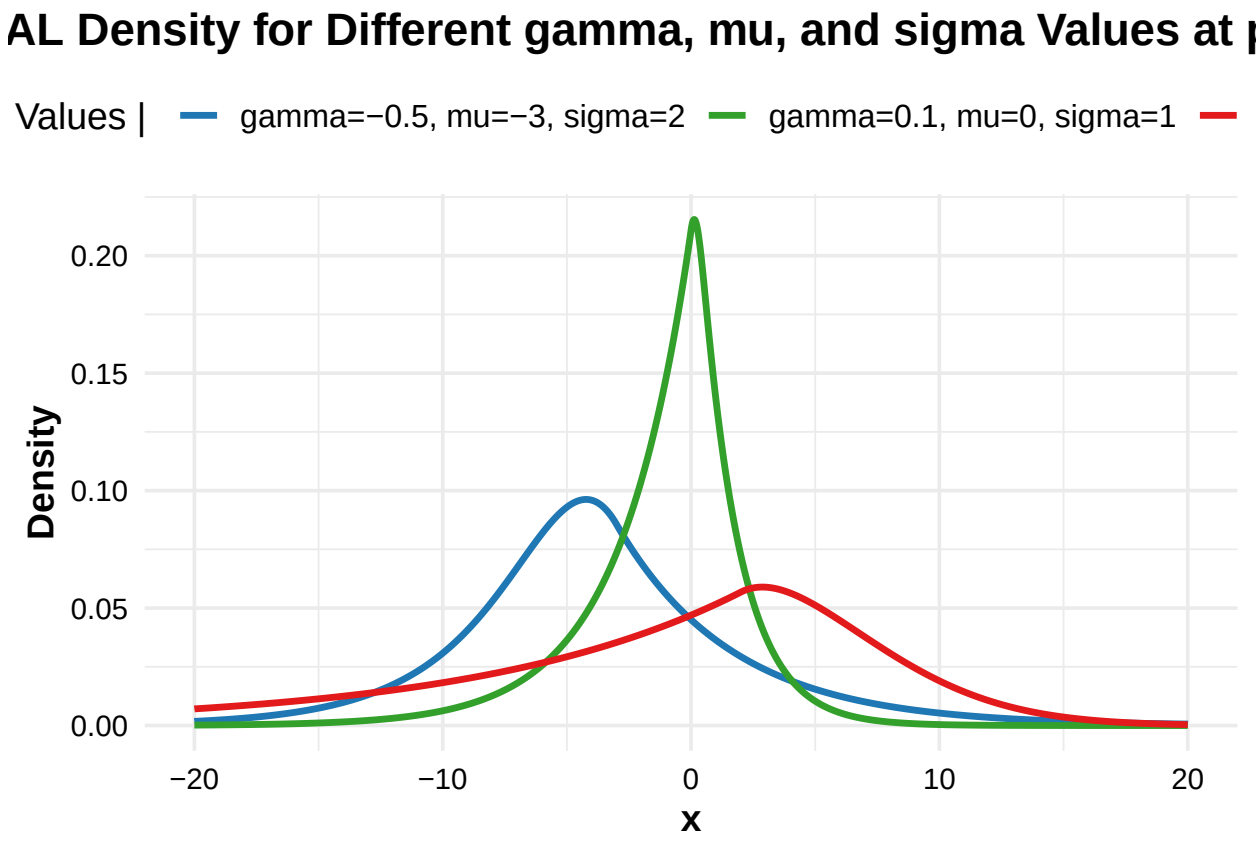
$$X \sim \mathrm{GAL}(p, \alpha, \mu, \sigma).$$

2. Probability Density Function (PDF) of the General GAL Distribution

By applying the location-scale transformation to the standard GAL density:

$$f_{\mathrm{GAL}}(x \mid p, \alpha, \mu, \sigma) = \frac{1}{\sigma} f_{\mathrm{GAL}}\left(\frac{x-\mu}{\sigma} \mid p, \alpha, 0, 1\right).$$

```r
# Generate x values
x <- seq(-20, 20, length = 5000)

# Compute GAL densities for different p_0, gamma, mu, and sigma values
df <- data.frame(
  x = rep(x, 3),
  y = c(sapply(x, function(x) dexal(x, p0 = 0.6, gamma = 0.5, mu = 2, sigma = 1.5)),
        sapply(x, function(x) dexal(x, p0 = 0.6, gamma = -0.5, mu = -3, sigma = 2)),
        sapply(x, function(x) dexal(x, p0 = 0.6, gamma = 0.1, mu = 0, sigma = 1))),
  gamma = factor(rep(c("gamma=0.5, mu=2, sigma=1.5",
                       "gamma=-0.5, mu=-3, sigma=2",
                       "gamma=0.1, mu=0, sigma=1"),
                     each = length(x)))
)

# Create the plot with a professional style
ggplot(df, aes(x = x, y = y, color = gamma)) +
  geom_line(size = 1.2) +
  scale_color_manual(values = c("#1f78b4", "#33a02c", "#e31a1c")) +  # Professional color palette
  labs(title = "GAL Density for Different gamma, mu, and sigma Values at p0 = 0.6",
       x = "x",
       y = "Density",
       color = "gamma, mu, sigma Values | ") +
  theme_minimal(base_size = 14) +
  theme(
    legend.position = "top",
    plot.title = element_text(hjust = 0.5, face = "bold"),
    axis.title = element_text(face = "bold"),
    axis.text = element_text(color = "black"),
    legend.text = element_text(size = 12)
  )
```



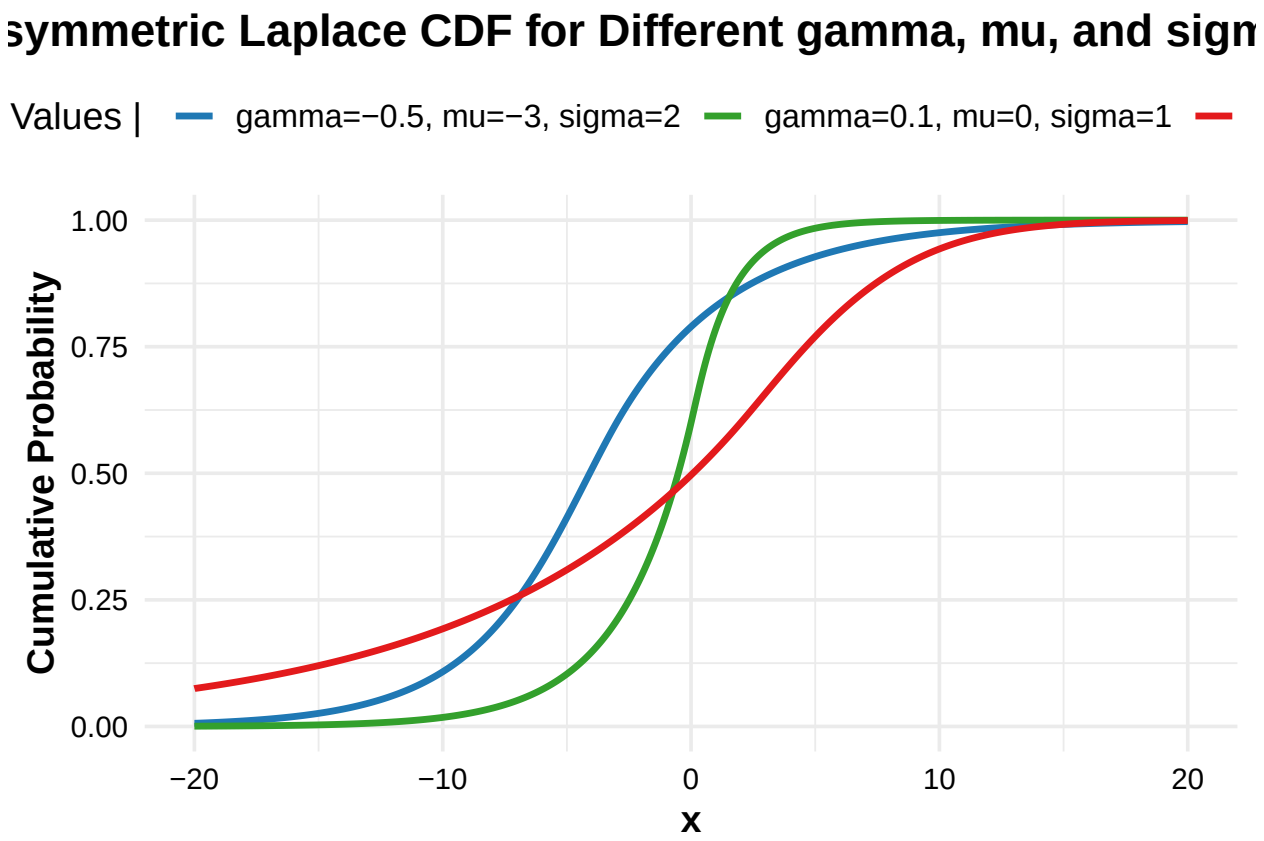
3. Cumulative Distribution Function (CDF) of the General GAL Distribution

Similarly, the CDF transforms as:

$$F_{\text{GAL}}(x \mid p, \alpha, \mu, \sigma) = F_{\text{GAL}}\left(\frac{x-\mu}{\sigma} \mid p, \alpha, 0, 1\right).$$

```r
# Generate x values
x <- seq(-20, 20, length = 5000)

# Compute GAL CDFs for different p_0, gamma, mu, and sigma values
df <- data.frame(
  x = rep(x, 3),
  y = c(sapply(x, function(x) pexal(x, p0 = 0.6, gamma = 0.5, mu = 2, sigma = 1.5)),
        sapply(x, function(x) pexal(x, p0 = 0.6, gamma = -0.5, mu = -3, sigma = 2)),
        sapply(x, function(x) pexal(x, p0 = 0.6, gamma = 0.1, mu = 0, sigma = 1))),
  gamma = factor(rep(c("gamma=0.5, mu= 2, sigma=1.5",
                       "gamma=-0.5, mu=-3, sigma=2",
                       "gamma=0.1, mu=0, sigma=1"),
                     each = length(x)))
)

# Create the plot with a professional style
ggplot(df, aes(x = x, y = y, color = gamma)) +
  geom_line(size = 1.2) +
```

```r
scale_color_manual(values = c("#1f78b4", "#33a02c", "#e31a1c")) +  # Professional color palette
labs(title = "Generalized Asymmetric Laplace CDF for Different gamma, mu, and sigma Values at p0 = 0.
     x = "x",
     y = "Cumulative Probability",
     color = "gamma, mu, sigma Values | ") +
theme_minimal(base_size = 14) +
theme(
  legend.position = "top",
  plot.title = element_text(hjust = 0.5, face = "bold"),
  axis.title = element_text(face = "bold"),
  axis.text = element_text(color = "black"),
  legend.text = element_text(size = 12)
)
```



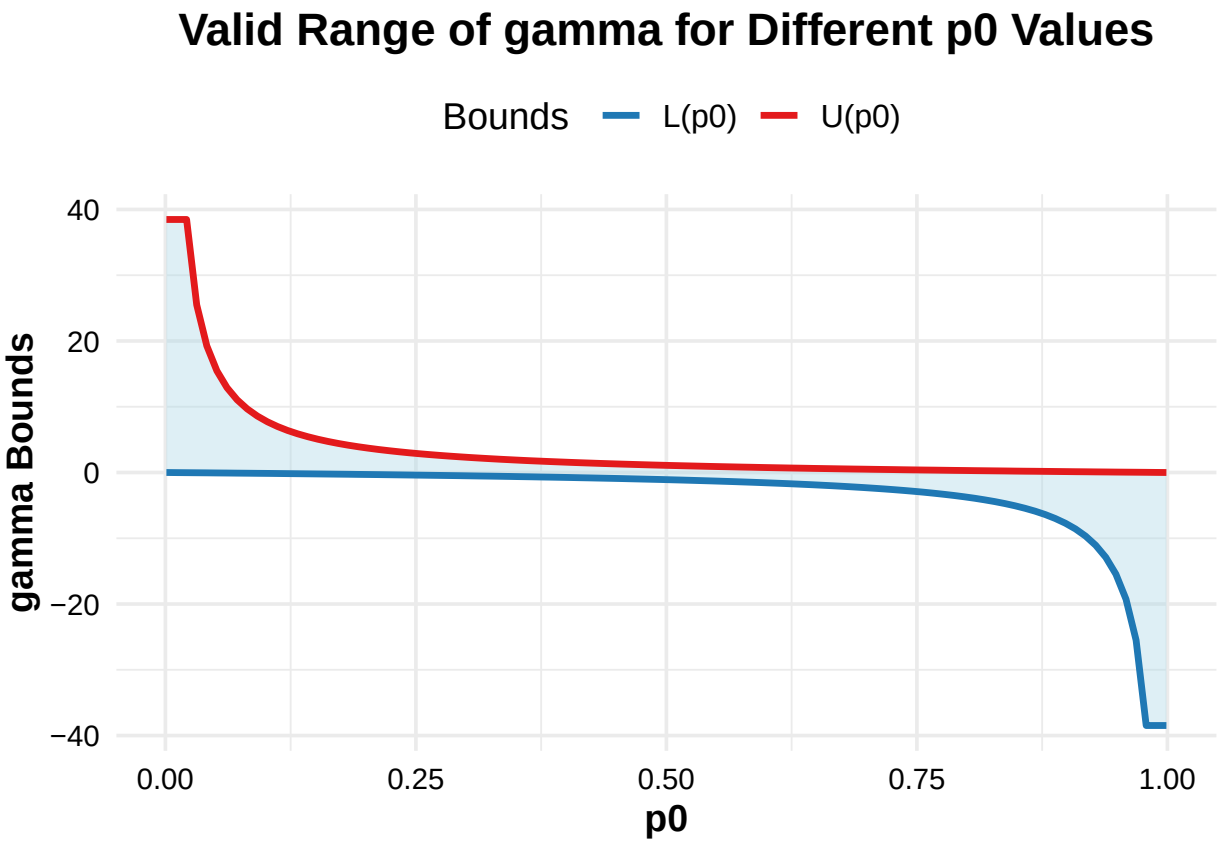
Gamma Bounds Computation

The GAL parameter γ is constrained by **two bounds**, L and U, which depend on p_0. These bounds are the solutions to:

$$g(L) = 1 - p_0, \quad g(U) = p_0$$

These bounds define the valid region for γ at a given p_0. To visualize this, we compute the bounds for several values of p_0 ranging from **0.001 to 0.999** and plot $L(p_0)$ and $U(p_0)$ along with the **shaded region** representing the valid range of γ.

```
## **Gamma Bounds Computation**

# The GAL parameter gamma is constrained by two bounds, L and U, which depend on p_0.
# These bounds define the valid region for gamma at a given p_0.

# Define a sequence of p0 values from 0.001 to 0.999
p0_values <- seq(0.001, 0.999, length.out = 100)

# Compute gamma bounds (L, U) for each p0
gamma_bounds <- as.data.frame(t(sapply(p0_values, get_gamma_bounds)))
colnames(gamma_bounds) <- c("L", "U")
gamma_bounds$p0 <- p0_values  # Add p0 column for mapping

# Create the plot with shading
ggplot(gamma_bounds, aes(x = p0)) +
  geom_ribbon(aes(ymin = L, ymax = U), fill = "lightblue", alpha = 0.4) +  # Shaded region for valid ga
  geom_line(aes(y = L, color = "L(p0)"), size = 1.2) +
  geom_line(aes(y = U, color = "U(p0)"), size = 1.2) +
  scale_color_manual(values = c("#1f78b4", "#e31a1c")) +  # Professional color palette
  labs(title = "Valid Range of gamma for Different p0 Values",
       x = "p0",
       y = "gamma Bounds",
       color = " Bounds ") +
  theme_minimal(base_size = 14) +
  theme(
    legend.position = "top",
    plot.title = element_text(hjust = 0.5, face = "bold"),
    axis.title = element_text(face = "bold"),
    axis.text = element_text(color = "black"),
    legend.text = element_text(size = 12)
  )
```

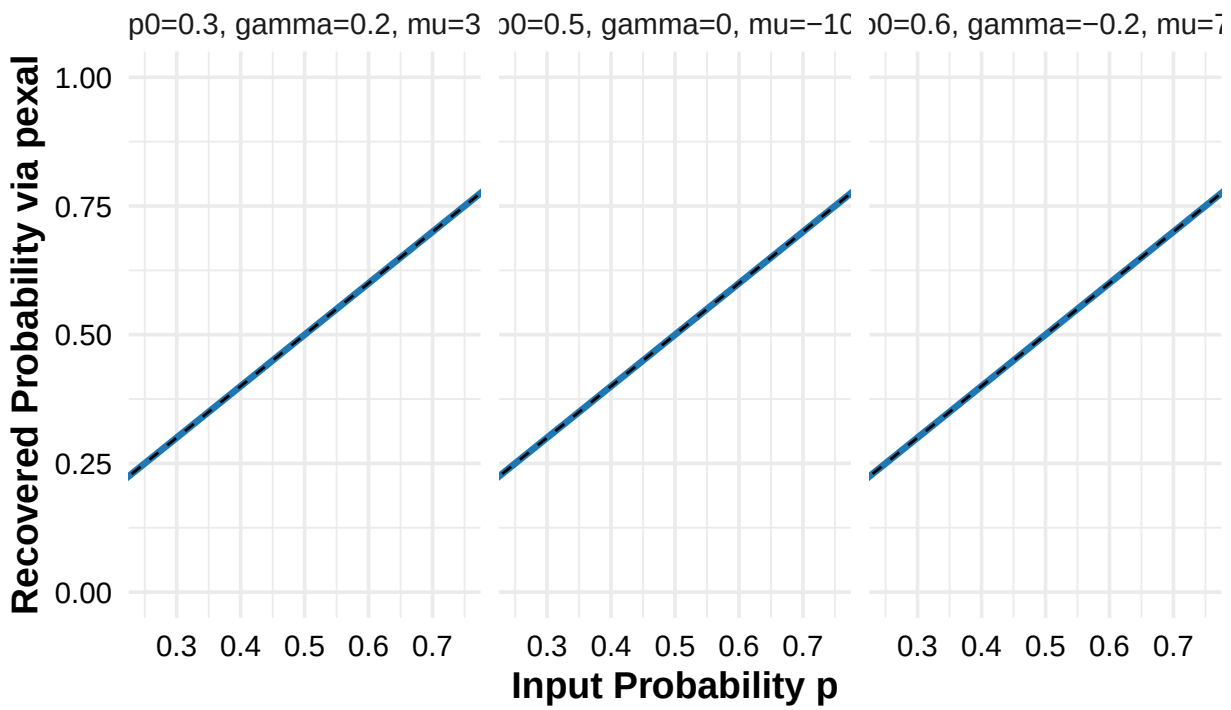


Quantile Function

The quantile function is computed **numerically** using a root-finding method.

```
# Define test cases
test_cases <- list(
  list(p0 = 0.3, gamma = 0.2, mu = 3),
  list(p0 = 0.5, gamma = 0, mu = -10),
  list(p0 = 0.6, gamma = -0.2, mu = 7)
)

# Define probability values to test

# Create an empty list to store results
results <- list()

# Loop over test cases
for (case in test_cases) {
  p0 <- case$p0
  gamma <- case$gamma
  mu <- case$mu

  p_vals <- seq(0.2, 0.8, length.out = 100)
```

```r
  # Define the label for this case
  case_label <- paste0("p0=", p0, ", gamma=", gamma, ", mu=", mu)

  # Compute quantiles using qexal
  q_vals <- sapply(p_vals, function(p) qexal(p, p0 = p0, gamma = gamma, mu = mu))

  # Apply pexal to check if we retrieve original probabilities
  p_check <- sapply(q_vals, function(q) pexal(q, p0 = p0, gamma = gamma, mu = mu))

  # Compute absolute error
  error <- abs(p_vals - p_check)

  # Store results in a data frame
  df <- data.frame(
    p = p_vals,
    q = q_vals,
    p_recovered = p_check,
    error = error,
    case_label = case_label
  )

  results[[case_label]] <- df
}

# Combine all results into one data frame
df_plot <- do.call(rbind, results)

# Plot each validation separately with fixed x and y limits (0,1)
ggplot(df_plot, aes(x = p, y = p_recovered)) +
  geom_line(size = 1.2, color = "#1f78b4") +
  geom_abline(slope = 1, intercept = 0, linetype = "dashed", color = "black") +
  labs(title = "Validation of qexal as the Inverse of pexal",
       subtitle = "Each panel corresponds to a different (p0, gamma, mu) configuration",
       x = "Input Probability p",
       y = "Recovered Probability via pexal") +
  theme_minimal(base_size = 14) +
  facet_wrap(~ case_label, scales = "fixed") +  # Keep fixed scales for comparison
  coord_cartesian(xlim = c(0.25, 0.75), ylim = c(0, 1)) +  # Set fixed axis limits
  theme(
    plot.title = element_text(hjust = 0.5, face = "bold"),
    plot.subtitle = element_text(hjust = 0.5, face = "italic"),
    axis.title = element_text(face = "bold"),
    axis.text = element_text(color = "black"),
    legend.position = "none"
  )
```




```r
# Define test cases
test_cases <- list(
  list(q = 0.1, p0 = 0.1, gamma = 0.1, mu = -123),
  list(q = 0.1, p0 = 0.1, gamma = 0.0, mu = -123),
  list(q = 0.1, p0 = 0.1, gamma = -0.1, mu = -123),

  list(q = 0.5, p0 = 0.5, gamma = 0.1, mu = -123),
  list(q = 0.5, p0 = 0.5, gamma = 0.0, mu = -123),
  list(q = 0.5, p0 = 0.5, gamma = -0.1, mu = -123),

  list(q = 0.8, p0 = 0.8, gamma = 0.1, mu = -123),
  list(q = 0.8, p0 = 0.8, gamma = 0.0, mu = -123),
  list(q = 0.8, p0 = 0.8, gamma = -0.1, mu = -123)
)

# Compute quantiles
results <- lapply(test_cases, function(tc) {
  quantile_value <- qexal(tc$q, p0 = tc$p0, gamma = tc$gamma, mu = tc$mu)
  return(data.frame(
    q = tc$q,
    p0 = tc$p0,
    gamma = tc$gamma,
    mu = tc$mu,
    quantile_value = quantile_value
  ))
})
```

```r
# Convert results to a single data frame
results_df <- do.call(rbind, results)

# Print results with comments
print(results_df)
```

```
##     q  p0 gamma   mu quantile_value
## 1 0.1 0.1   0.1 -123           -123
## 2 0.1 0.1   0.0 -123           -123
## 3 0.1 0.1  -0.1 -123           -123
## 4 0.5 0.5   0.1 -123           -123
## 5 0.5 0.5   0.0 -123           -123
## 6 0.5 0.5  -0.1 -123           -123
## 7 0.8 0.8   0.1 -123           -123
## 8 0.8 0.8   0.0 -123           -123
## 9 0.8 0.8  -0.1 -123           -123
```

```r
# Check if mu is equal to qexal(p0)
all(results_df$mu == results_df$quantile_value)  # Should return TRUE
```

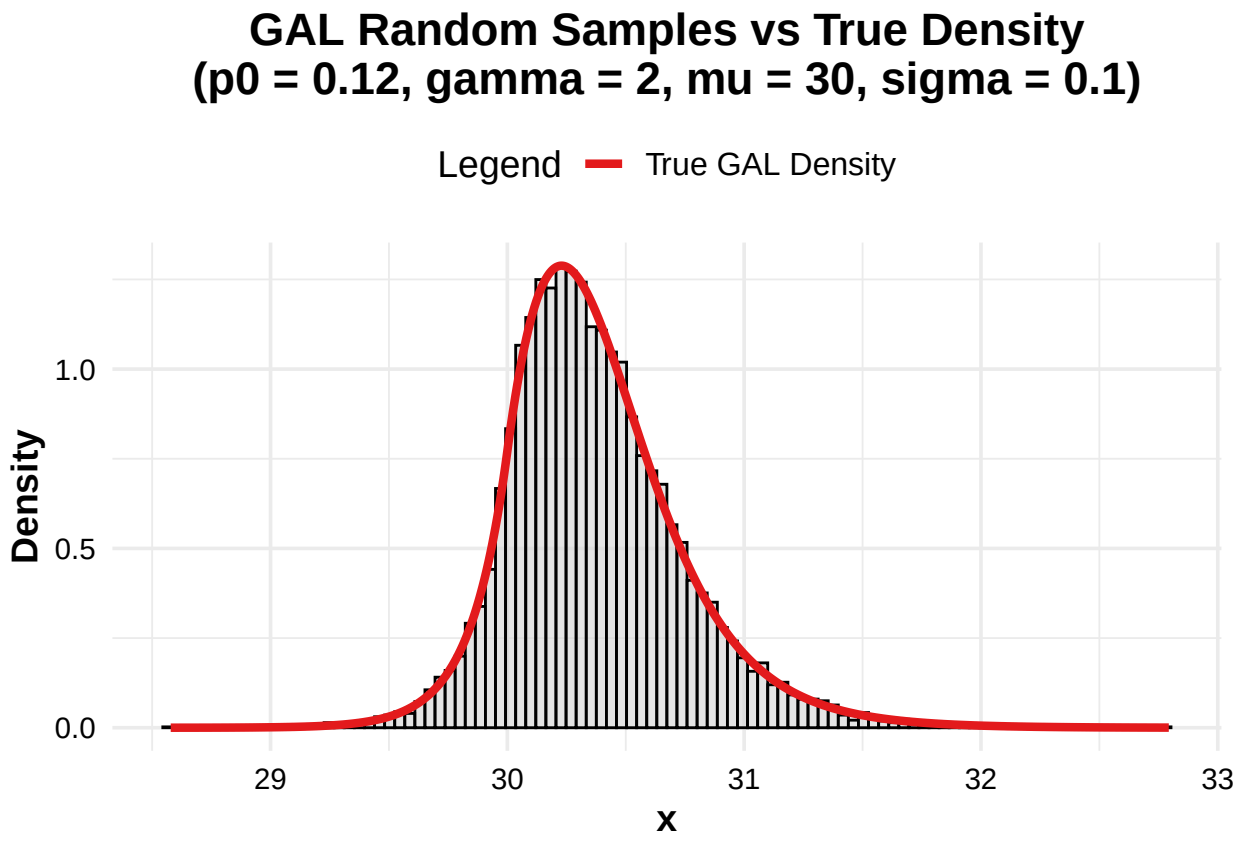
```
## [1] TRUE
```

Random Sampling

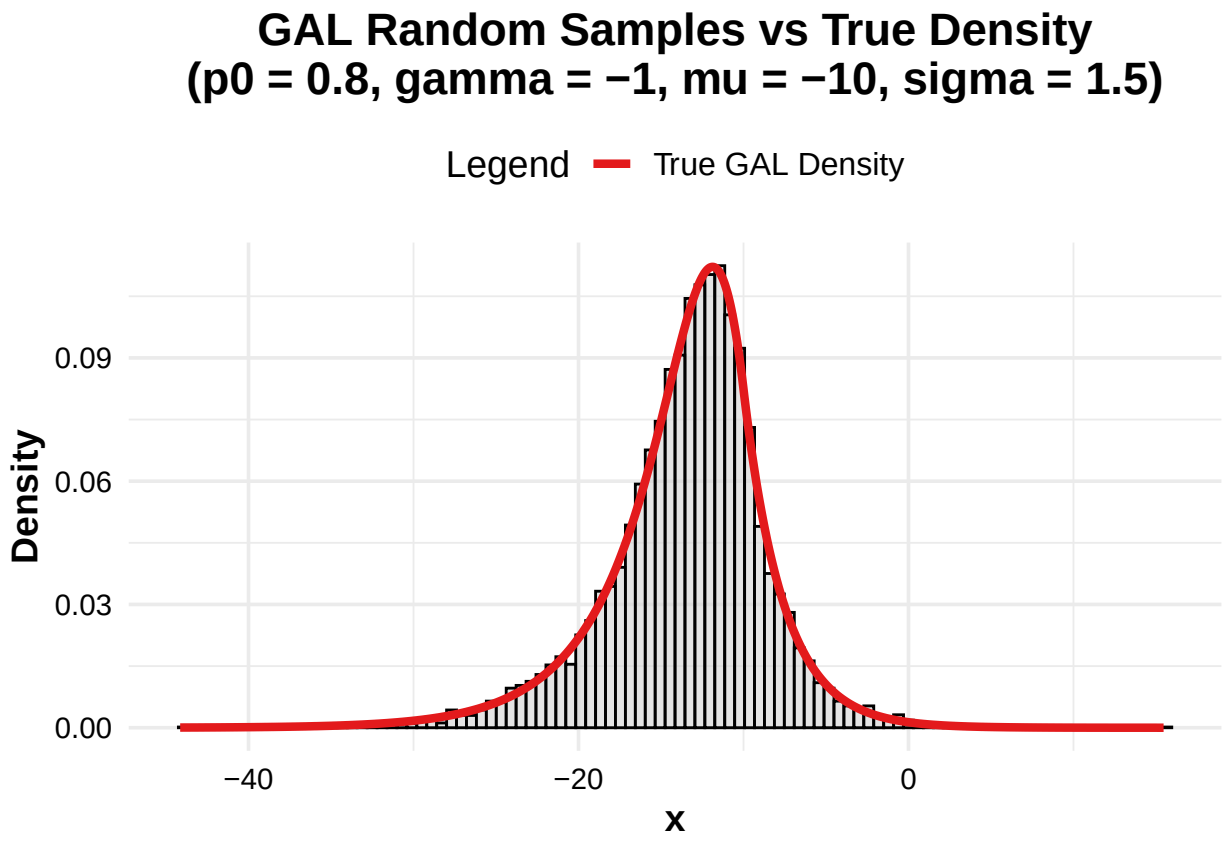
The GAL distribution can be used to **generate random samples**, which can then be visualized.

```r
set.seed(123)
# Define different parameter sets for GAL distribution
params <- list(
  list(p0 = 0.12, gamma = 2, mu = 30, sigma = 0.1),
  list(p0 = 0.8, gamma = -1, mu = -10, sigma = 1.5),
  list(p0 = 0.35, gamma = 0, mu = 100, sigma = 2)
)

# Create an empty list to store plots
plot_list <- list()

# Loop through each parameter set and generate plots
for (i in seq_along(params)) {

  # Extract parameters
  p0 <- params[[i]]$p0
  gamma <- params[[i]]$gamma
  mu <- params[[i]]$mu
  sigma <- params[[i]]$sigma

  # Generate random samples
  samples <- rexal(10000, p0 = p0, gamma = gamma, mu = mu, sigma = sigma)

  # Define x range for density computation
```

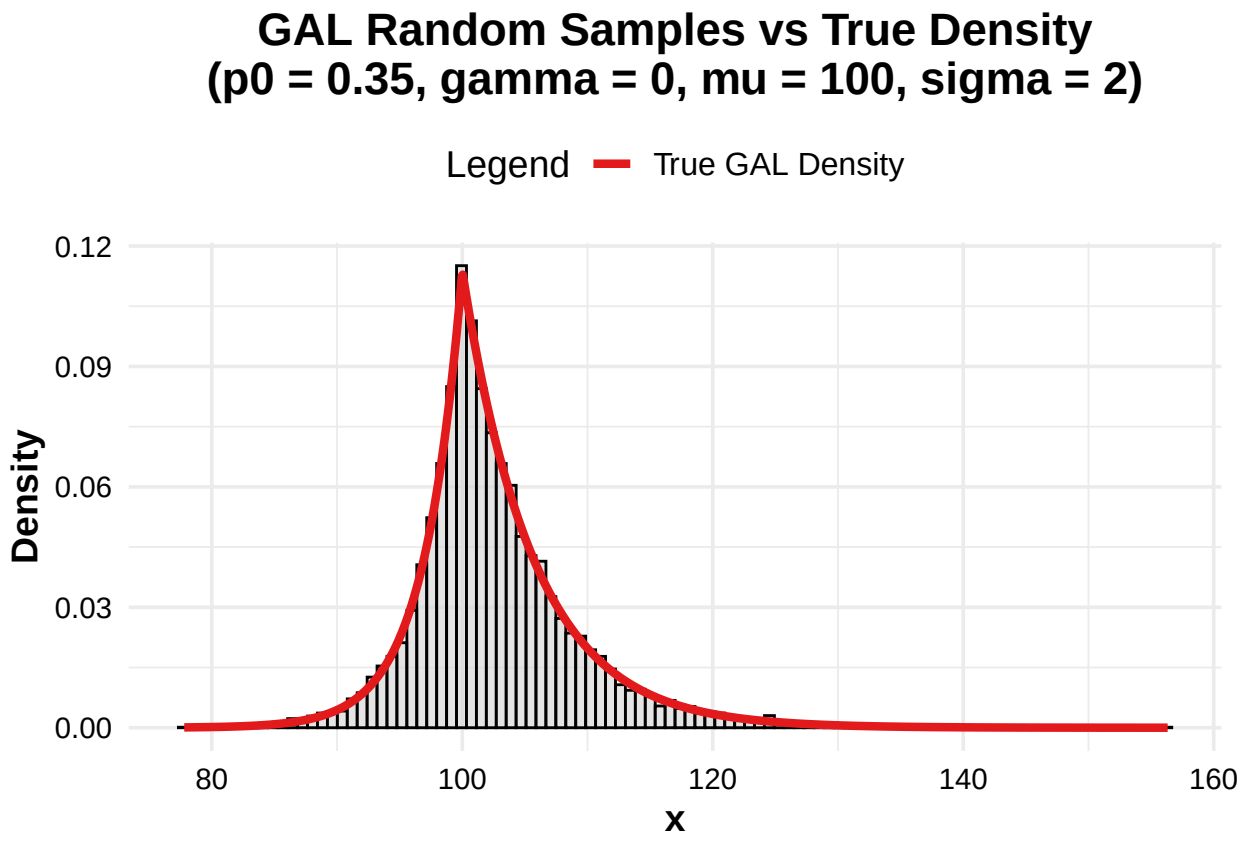
```r
  x_values <- seq(min(samples), max(samples), length.out = 1000)

  # Compute GAL density using the same parameters
  density_values <- sapply(x_values, function(x) dexal(x, p0 = p0, gamma = gamma, mu = mu, sigma = sigma

  # Create data frames
  df_density <- data.frame(x = x_values, density = density_values)
  df_samples <- data.frame(samples = samples)

  # Generate histogram with density overlay
  plot_list[[i]] <- ggplot(df_samples, aes(x = samples)) +
    geom_histogram(aes(y = after_stat(density)), bins = 100, fill = "lightgray", color = "black", alpha
    geom_line(data = df_density, aes(x = x, y = density, color = "True GAL Density"), size = 1.5) +
    scale_color_manual(values = c("True GAL Density" = "#e31a1c")) +  # Professional color
    labs(
      title = paste0("GAL Random Samples vs True Density\n(p0 = ", p0, ", gamma = ", gamma, ", mu = ", m
      x = "x",
      y = "Density",
      color = "Legend"
    ) +
    theme_minimal(base_size = 14) +
    theme(
      legend.position = "top",
      plot.title = element_text(hjust = 0.5, face = "bold"),
      axis.title = element_text(face = "bold"),
      axis.text = element_text(color = "black"),
      legend.text = element_text(size = 12)
    )
}

# Display plots
plot_list[[1]]
```


GAL Random Samples vs True Density
(p0 = 0.12, gamma = 2, mu = 30, sigma = 0.1)
Legend
True GAL Density
Density
1.0
0.5
0.0
29
30
31
32
33
x

```
plot_list[[2]]
```


GAL Random Samples vs True Density
(p0 = 0.8, gamma = −1, mu = −10, sigma = 1.5)
Legend
True GAL Density
Density
0.09
0.06
0.03
0.00
−40
−20
0
x

```
plot_list[[3]]
```


GAL Random Samples vs True Density
(p0 = 0.35, gamma = 0, mu = 100, sigma = 2)
Legend
True GAL Density
Density
0.12
0.09
0.06
0.03
0.00
80
100
120
140
160
x