The Journal of Open Source Software

gkwreg: An R Package for Generalized Kumaraswamy Regression Models for Bounded Data

José Evandeilton Lopes[1] **and Wagner Hugo Bonat**[1]

1 Paraná Federal University, Brazil

DOI:

Software

- Review
- Repository
- Archive

Submitted:
Published:

Summary

Modeling data restricted to the unit interval $(0, 1)$, such as proportions and rates, is a common challenge. The beta distribution, although predominant, has limitations regarding flexibility and analytical tractability (CDF), motivating alternatives, especially in regression contexts. The Kumaraswamy (Kw) distribution offers analytical simplicity (closed-form CDF/PDF), but it may have insufficient flexibility. In this context, the Generalized Kumaraswamy (GKw) distribution, a five-parameter flexible model nesting Beta, Kw, and other relevant distributions, emerges as a promising alternative. However, the complexity of its likelihood function makes maximum likelihood estimation (MLE) difficult, particularly in multiparametric regression. This article introduces the **R** package `gkwreg`, developed for efficient (MLE) estimation and inference in GKw regression models and its seven main subfamilies. `gkwreg` uses the robust Template Model Builder (`TMB`) framework, employing automatic differentiation for fast and accurate gradient/Hessian computation and effective optimization algorithms. This integrated approach via `TMB` enables reliable, robust, and efficient estimation of complex models, even when multiple parameters are linked to covariates by various link functions. The package also includes routines in `RcppArmadillo` and `C++` that implement distribution functions `p`, `d`, `r`, and `q`, as well as their respective log-likelihood functions, gradients, and analytic Hessians for the seven GKw subfamilies and a function for fitting distributions to data. We detail the statistical foundations of the GKw family, its hierarchy, the regression framework, inference, diagnostics, and the package's computational implementation. Real data examples demonstrate `gkwreg`'s ability to capture complex distributional patterns that simpler models may overlook. The package thus provides a unified, flexible, and computationally efficient tool for analyzing responses restricted to the unit interval.

Introduction

Statistical analysis of data restricted to the standard unit interval $(0, 1)$ is often required in numerous empirical studies. Such data commonly appear as proportions, percentages, rates, indices, normalized concentrations, or fractions (Ferrari & Cribari-Neto, 2004; Kieschnick & McCullough, 2003; Smithson & Verkuilen, 2006). Examples include the proportion of family income allocated to certain goods or services (Branscum, Johnson, & Thurmond, 2007), the fraction of an area's surface affected by a specific condition (e.g., disease in a leaf), indices measuring socioeconomic development or academic performance, or levels of chemical substances scaled to the $(0, 1)$ interval, default probabilities in credit risk, fraud risk analysis, and others.

Although traditional approaches frequently involve transforming the response variable $y \in (0, 1)$ (for example, using logit transformations, $\log(y/(1-y))$, or probit transformations, $\Phi^{-1}(y)$) to map the unit interval onto the real line $\mathbb{R}$, followed by standard linear models (e.g., linear regression), these methods can exhibit important drawbacks. Model parameter

interpretation becomes less direct, since parameters relate to the transformed scale of the variable rather than its original meaning in $(0, 1)$. Additionally, transformations can induce heteroskedasticity or fail to properly address the non-constant variance often observed in proportions. They may also fail to accurately capture the variable's behavior near 0 and 1 or more complex distributional shapes (e.g., bimodal) within the interval (Ferrari & Cribari-Neto, 2004; Smithson & Verkuilen, 2006).

A more direct and often statistically more suitable approach involves using probability distributions defined explicitly on the $(0, 1)$ interval. Historically, the beta distribution has been the most prominent choice for this purpose (Gupta & Nadarajah, 2004; Johnson, Kotz, & Balakrishnan, 1995). Its probability density function (PDF) is notably flexible, able to take diverse shapes (symmetric, left- or right-skewed, U-shaped, J-shaped, or approximately uniform) depending on the values of its two positive shape parameters. However, the beta distribution also presents limitations. Its cumulative distribution function (CDF) is defined through the regularized incomplete beta function, which does not have a closed-form expression. Historically, this posed computational challenges for tasks such as probability calculation, quantile determination, and, crucially, maximum likelihood estimation (MLE), especially before efficient numerical algorithms became available (Cribari-Neto & Zeileis, 2010; Dutka, 1981). Although modern numerical methods, such as those implemented in packages like `gkwreg` (Cribari-Neto & Zeileis, 2010), handle this efficiently, the absence of a closed-form CDF remains a theoretical (and sometimes practical) inconvenience. Moreover, despite its versatility, the range of shapes the beta distribution can assume may not suffice to adequately capture all empirical patterns observed in bounded data, such as certain types of bimodality or specific tail behaviors.

As an alternative, the Kumaraswamy (Kw) distribution (Kumaraswamy, 1980), originally proposed in hydrology to model variables like soil moisture, gained renewed attention in the statistical literature after Jones's seminal work (2009). The main advantage of the Kw distribution lies in its remarkable analytical tractability: both its CDF and quantile function (the inverse CDF) have simple, closed-form expressions. This greatly facilitates data simulation, parameter estimation using quantile- or moment-based methods, and theoretical property derivation (Fletcher & Ponnambalam, 1996; Jones, 2009; Sundar & Subbiah, 1989). Although it offers flexibility comparable to the beta distribution for certain configurations of its two shape parameters, the Kw distribution, because it has only two parameters, can also be restrictive for modeling more complex data structures.

Recognizing the inherent limitations of both the beta and Kw distributions, researchers have developed a range of methodologies to build more flexible distributions on $(0, 1)$ and other domains. Methods based on generating new distribution families by transforming existing ones, such as the popular "beta-generated" approach proposed by Eugene et al. (2002), have led to an extensive variety of extended models (for example, beta-normal, beta-exponential (Nadarajah & Kotz, 2006), beta-Gumbel (Nadarajah & Kotz, 2004), beta-Fréchet (Nadarajah & Gupta, 2004)). Inspired by the broader framework of generalized beta distributions introduced by McDonald (1984), Carrasco et al. (2010) proposed the Generalized Kumaraswamy (GKw) distribution. This five-parameter distribution exhibits notable flexibility, encompassing the beta, Kumaraswamy, and several others (like the exponentiated Kumaraswamy and the McDonald distribution) as special or limiting cases. Its capacity to model a wide spectrum of distribution shapes, including unimodal, bimodal, J-shaped, U-shaped, and "bathtub"-shaped forms, makes it a highly attractive candidate for modeling complex single or multi-modal patterns often found in bounded data.

Despite its theoretical advantages in terms of flexibility, practical application of the GKw distribution, particularly in a regression context where parameters may vary dynamically with covariates, is hampered by significant computational challenges. The GKw likelihood function, involving five parameters, is mathematically complex, making MLE computationally intensive and potentially unstable, especially when multiple parameters are

modeled simultaneously as functions of linear predictors. Efficient, stable, user-friendly software is thus essential for researchers to effectively use GKw regression models in applied analyses.

This article presents the `R` package (R Core Team, 2025) `gkwreg`, available from the Comprehensive `R` Archive Network (CRAN). `gkwreg` provides a robust, computationally efficient framework for fitting MLE-based regression models using the GKw distribution. The package implements not only the full, five-parameter GKw distribution, but also its six important nested submodels (detailed in Section), covering seven sophisticated families for bounded data. A cornerstone of `gkwreg`'s implementation is its reliance on the `TMB` (Template Model Builder) (Kristensen, Nielsen, Berg, Skaug, & Bell, 2016) package. `TMB` employs automatic differentiation (AD) to compute analytically exact gradients and Hessians of the objective function (log-likelihood), and makes internal use of high-performance `C++` routines, facilitating fast and stable optimization via standard numerical algorithms such as `nlminb` (Gay, 1990) or methods implemented through `optim()` known in `R` as `BFGS`, `L-BFGS-B`, `Nelder-Mead`, among others. This approach allows `gkwreg` to handle models in which each of the GKw distribution parameters $(\alpha, \beta, \gamma, \delta, \lambda)$ can be specified as a function of covariates, using `R`'s familiar formula syntax and a variety of appropriate link functions. The package thus aims to make these flexible, powerful regression models accessible for practical application across various research fields.

The remainder of this article is structured as follows. Section provides an overview of the beta and Kumaraswamy distributions, introduces generalization methodologies, and details the Generalized Kumaraswamy distribution, its properties, and its hierarchical structure. Still in Section , we describe the implemented regression framework, including link functions, maximum likelihood estimation, inference procedures, and model diagnostic tools. Section describes the `gkwreg` package, focusing on its computational engine (`TMB`), its main functions, the model specification interface, and how output is processed. Section demonstrates the package's capabilities with extensive examples using real datasets. Finally, Section offers concluding remarks, summarizes the work's contributions, and discusses potential future directions for developing the `gkwreg` package.

Theoretical and Methodological Context

Beta and Kumaraswamy

The beta distribution is defined by the probability density function (PDF):

$$f_{\text{Beta}}(y; a, b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} y^{a-1}(1-y)^{b-1} = \frac{1}{B(a,b)} y^{a-1}(1-y)^{b-1}, \quad 0 < y < 1,$$

where $a > 0$ and $b > 0$ are the shape parameters, $\Gamma(\cdot)$ denotes the gamma function, and $B(a,b) = \Gamma(a)\Gamma(b)/\Gamma(a+b)$ is the beta function (Johnson et al., 1995). Its cumulative distribution function (CDF) is the regularized incomplete beta function, $F_{\text{Beta}}(y; a, b) = I_y(a,b) = \frac{B_y(a,b)}{B(a,b)}$, where $B_y(a,b) = \int_0^y t^{a-1}(1-t)^{b-1}\,dt$ is the incomplete beta function. The absence of a closed-form expression for $I_y(a,b)$ requires numerical methods for its evaluation and inversion, underpinning MLE procedures in packages such as `gkwreg` (Cribari-Neto & Zeileis, 2010).

The Kumaraswamy (Kw) distribution (Kumaraswamy, 1980), on the other hand, is characterized by a remarkably simple closed-form CDF:

$$G_1(y; \alpha, \beta) = 1 - (1 - y^\alpha)^\beta, \quad 0 < y < 1,\ \alpha > 0,\ \beta > 0.$$

Its corresponding PDF is obtained by differentiating with respect to y:

$$g_1(y;\alpha,\beta) = \frac{d}{dy} G_1(y;\alpha,\beta) = \alpha\beta\, y^{\alpha-1}\,(1-y^{\alpha})^{\beta-1}.$$

Parameters $\alpha > 0$ and $\beta > 0$ are also shape parameters. A significant advantage is the existence of a closed-form quantile function (the inverse CDF):

$$Q(u;\alpha,\beta) = G_1^{-1}(u) = \left[1-(1-u)^{1/\beta}\right]^{1/\alpha}, \quad u \in (0,1).$$

This analytical tractability simplifies data generation, as well as parameter estimation based on quantiles or moments, and potentially theoretical derivations (Jones, 2009). The r-th raw moment is given by $E(Y^r) = \beta\, B\big(1+\frac{r}{\alpha},\beta\big)$ for $Y \sim \mathrm{Kw}(\alpha,\beta)$.

Generalization

Various systematic approaches have been proposed to construct more flexible distribution families, often by transforming a known base distribution. One major method uses the probability integral transform (PIT) (Ferreira & Steel, 2006; Wahed & Ali, 2006). Given a base CDF $G_1(y;\boldsymbol{\omega})$ with PDF $g_1(y;\boldsymbol{\omega})$, and a "generator" distribution defined on $(0,1)$ with PDF $g_2(t;\boldsymbol{\tau})$, a new (potentially more flexible) CDF $F(y;\boldsymbol{\omega},\boldsymbol{\tau})$ is formed as $F(y) = \int_0^{G_1(y;\boldsymbol{\omega})} g_2(t;\boldsymbol{\tau})\,dt$. The corresponding PDF is $f(y) = g_2\big(G_1(y;\boldsymbol{\omega});\boldsymbol{\tau}\big)\, g_1(y;\boldsymbol{\omega})$. Eugene et al. (2002) popularized using the PDF of the beta distribution as the generator g_2, creating the "beta-G" (beta-generated) family with CDF $F(y) = I_{G_1(y)}(a,b)$. This led to distributions such as beta-normal, beta-Gumbel (Nadarajah & Kotz, 2004), beta-Fréchet (Nadarajah & Gupta, 2004), and beta-exponential (Nadarajah & Kotz, 2006). Another influential generator derives from McDonald's generalized beta of the first kind (GB1) (McDonald, 1984), whose PDF is $h(z) = \frac{|c|}{B(a,b)} z^{ac-1}(1-z^c)^{b-1}$ for $0 < z < 1$. Cordeiro and de Castro (2011) extensively explored generalizations based on this framework.

Generalized Kumaraswamy (GKw)

The Generalized Kumaraswamy (GKw) distribution, introduced by Carrasco et al. (2010), arises from applying a generator related to McDonald's GB1 framework to the Kumaraswamy CDF $G_1(y;\alpha,\beta)$. Specifically, the GKw distribution is defined by its CDF:

$$F(y;\boldsymbol{\theta}) = I_{[1-(1-y^{\alpha})^{\beta}]^{\lambda}}(\gamma,\delta+1), \quad 0 < y < 1,$$

where $I_z(a,b)$ is the regularized incomplete beta function, and $\boldsymbol{\theta} = (\alpha,\beta,\gamma,\delta,\lambda)^{\top}$ is the vector of five parameters, all positive ($\alpha,\beta,\gamma,\delta,\lambda > 0$). This definition reveals the GKw as a beta-generated distribution applied to a power transformation of the Kumaraswamy CDF, $G_1(y;\alpha,\beta)^{\lambda}$. The corresponding PDF is obtained by differentiating the CDF with respect to y:

$$f(y;\boldsymbol{\theta}) = \frac{\lambda\,\alpha\,\beta\, y^{\alpha-1}}{B(\gamma,\delta+1)} \left(1-y^{\alpha}\right)^{\beta-1} \left[1-\left(1-y^{\alpha}\right)^{\beta}\right]^{\gamma\lambda-1} \left\{1-\left[1-\left(1-y^{\alpha}\right)^{\beta}\right]^{\lambda}\right\}^{\delta}.$$

The five parameters $\alpha,\beta,\gamma,\delta,\lambda$ collectively provide substantial flexibility in modeling the shape of the distribution on $(0,1)$: (i) α and β primarily govern the basic shape inherited from the underlying Kumaraswamy distribution, influencing the locations of the modes and general skewness. (ii) γ and δ act similarly to the shape parameters a and b of a beta distribution, modulating the density's shape, particularly affecting tail behavior and the concentration of mass around the modes. Smaller values of γ and δ tend to yield U- or J-shaped forms, while larger values tend to yield unimodal forms. (iii) λ introduces additional flexibility as a power parameter applied to the transformed Kumaraswamy CDF

$G_1(y; \alpha, \beta)$. It can influence skewness, as well as the location and sharpness (kurtosis) of the density peaks.

This rich parameterization allows the GKw distribution to capture a broad range of shapes, including symmetrical, skewed (positive or negative), unimodal, J-shaped (increasing or decreasing), U-shaped, "bathtub"-shaped (indicating hazard-like behavior), and potentially bimodal forms, making it particularly suitable for approximating complex empirical distributions of data bounded on $(0, 1)$.

Carrasco et al. (2010) derived various theoretical properties of GKw. They showed that both the PDF and CDF can be expressed as infinite mixture representations involving densities or powers of the Kumaraswamy CDF. For instance, the PDF can be written as an infinite mixture of Kw densities:

$$f(y; \boldsymbol{\theta}) = \sum_{k=0}^{\infty} w_k \, g_1\big(y; \alpha, (k+1)\beta\big),$$

where $g_1\big(y; \alpha, (k+1)\beta\big)$ is the PDF of a $\mathrm{Kw}\big(\alpha, (k+1)\beta\big)$ distribution, and the weights $w_k \geq 0$ ($\sum w_k = 1$) are functions of γ, δ, λ. This representation can be useful for deriving theoretical properties, such as moments: $E(Y^r) = \sum_{k=0}^{\infty} w_k \, E(Z_k^r)$, where $Z_k \sim \mathrm{Kw}(\alpha, (k+1)\beta)$. However, for likelihood-based inference, direct evaluation of the PDF and CDF using their defining expressions is typically more computationally efficient than using these infinite series expansions.

Special Cases

Table 1: The Generalized Kumaraswamy (GKw) distribution and its main nested subfamilies, implemented in the 'gkwreg' package. Parameter constraints define the hierarchy.

Family	Probability Density Function (PDF) $f(y; \boldsymbol{\theta})$	Parameters & Constraints	# Par.
GKw (Generalized Kumaraswamy)	$\frac{\lambda\alpha\beta\, y^{\alpha-1}}{B(\gamma,\, \delta+1)} (1-y^{\alpha})^{\beta-1} [1-(1-y^{\alpha})^{\beta}]^{\gamma\lambda-1} \times \{1-[1-(1-y^{\alpha})^{\beta}]^{\lambda}\}^{\delta}$	$\boldsymbol{\theta} = (\alpha, \beta, \gamma, \delta, \lambda)^{\top}$ $\alpha, \beta, \gamma, \delta, \lambda > 0$	5
BKw (Beta Kumaraswamy)	$\frac{\alpha\beta\, y^{\alpha-1}}{B(\gamma,\, \delta+1)} (1-y^{\alpha})^{\beta(\delta+1)-1} [1-(1-y^{\alpha})^{\beta}]^{\gamma-1}$	$\alpha, \beta, \gamma, \delta > 0$ $(\lambda = 1)$	4
KKw (Kumaraswamy Kumaraswamy)	$\lambda\alpha\beta(\delta+1)\, y^{\alpha-1}(1-y^{\alpha})^{\beta-1} [1-(1-y^{\alpha})^{\beta}]^{\lambda-1} \{1-[1-(1-y^{\alpha})^{\beta}]^{\lambda}\}^{\delta}$	$\alpha, \beta, \delta, \lambda > 0$ $(\gamma = 1)$	4
EKw (Exponentiated Kumaraswamy)	$\lambda\alpha\beta\, y^{\alpha-1}(1-y^{\alpha})^{\beta-1} [1-(1-y^{\alpha})^{\beta}]^{\lambda-1}$	$\alpha, \beta, \lambda > 0$ $(\gamma = 1, \delta \to 0)$	3
Mc (McDonald / Beta Power)	$\frac{\lambda\, y^{\lambda\gamma-1}(1-y^{\lambda})^{\delta}}{B(\gamma,\, \delta+1)}$	$\gamma, \delta, \lambda > 0$ $(\alpha = 1, \beta = 1)$	3
Kw (Kumaraswamy)	$\alpha\beta\, y^{\alpha-1}(1-y^{\alpha})^{\beta-1}$	$\alpha, \beta > 0$ $(\lambda = 1, \gamma = 1, \delta \to 0)$	2
Beta (Standard Beta)	$\frac{y^{\gamma-1}(1-y)^{\delta}}{B(\gamma,\, \delta+1)}$	$\gamma, \delta > 0$ $(\alpha = 1, \beta = 1, \lambda = 1)$	2

Note: The restriction $\delta \to 0$ for EKw and Kw implies that the term $\{1-[\ldots]^{\lambda}\}^{\delta}$ tends to 1.

A significant advantage of the GKw distribution is its rich hierarchical structure. By imposing specific constraints on its parameters (typically fixing one or more at 1, or considering limits such as $\delta \to 0$), GKw reduces to several well-known and useful distributions defined on $(0, 1)$. This nesting property is particularly valuable for principled model selection, for example, through likelihood ratio tests (LRTs) or information-criterion comparisons (AIC, BIC). Table 1 presents the GKw distribution and its subfamilies implemented in the `gkwreg` package, along with the parameter constraints that define them. Note that one special GKw case is a beta distribution whose second shape parameter is increased by one $\big(B(\gamma, \delta+1)\big)$. This hierarchical structure allows practitioners to begin modeling with the full GKw model, offering maximum flexibility, and then, if indicated by the data and parsimony considerations, simplify to a more restricted submodel using standard model-comparison techniques such as likelihood ratio tests (for nested models) or information criteria (AIC, BIC).

GKw Regression Framework

The `gkwreg` package was primarily developed to facilitate regression modeling, in which the distribution parameters of the chosen GKw family (or its subfamilies) are related to a set of covariates (predictors). Let y_i be the response variable for the i-th observation ($i = 1, \ldots, n$), with $y_i \in (0, 1)$. We assume y_i follows (conditional on the covariates) a distribution belonging to the GKw family (as in Table 1) with parameter vector $\boldsymbol{\theta}_i = (\alpha_i, \beta_i, \gamma_i, \delta_i, \lambda_i)^\top$. In the regression context, each parameter θ_{ip} (where p indexes parameters, $p \in \{\alpha, \beta, \gamma, \delta, \lambda\}$) can depend on a specific vector of covariates $\mathbf{x}_{ip}$ (a subset or all the available covariates for observation i) through a monotonic, differentiable link function $g_p(\cdot)$:

$$g_p(\theta_{ip}) = \eta_{ip} = \mathbf{x}_{ip}^\top \boldsymbol{\beta}_p,$$

where η_{ip} is the linear predictor for the p-th parameter of the i-th observation, and $\boldsymbol{\beta}_p$ is the vector of regression coefficients (including the intercept) associated with that specific parameter. Equivalently, the parameter on the original scale is obtained via the inverse link function: $\theta_{ip} = g_p^{-1}(\eta_{ip}) = g_p^{-1}(\mathbf{x}_{ip}^\top \boldsymbol{\beta}_p)$. This structure allows different aspects of the response distribution (e.g., shape, skewness, kurtosis) to be influenced by different sets of covariates.

Link Functions

Link functions (g_p) play a crucial role in mapping the linear predictor $\eta_{ip} \in \mathbb{R}$ to the valid domain of the distribution parameter θ_{ip}. For all five GKw family parameters $(\alpha, \beta, \gamma, \delta, \lambda)$, the natural domain is the set of positive real numbers $(0, \infty)$. The `gkwreg` package offers several options for the link function $g_p(\cdot)$ for each of these parameters, specified via the `link` argument (which can be a single string if the same link is used for all modeled parameters, or a named list of strings specifying the link for each parameter):

1. **Log link (`"log"`):** $g_p(\theta_{ip}) = \log(\theta_{ip})$, implying $\theta_{ip} = \exp(\eta_{ip})$. This is the **default link** for all parameters $(\alpha, \beta, \gamma, \delta, \lambda)$ in `gkwreg`. It ensures strict positivity ($\theta_{ip} > 0$) for any linear predictor $\eta_{ip} \in \mathbb{R}$ and is often considered the most natural link for strictly positive parameters.
2. **Identity link (`"identity"`):** $g_p(\theta_{ip}) = \theta_{ip}$, implying $\theta_{ip} = \eta_{ip}$. This link requires the linear predictor η_{ip} to be inherently positive. Although available, its use is generally **discouraged** for strictly positive parameters unless positivity is guaranteed by model structure or covariate constraints, since it can easily lead to numerical instability or invalid parameter values during optimization. The implementation includes an internal clamping mechanism to mitigate issues, but the `log` link is preferable.
3. **Square Root link (`"sqrt"`):** $g_p(\theta_{ip}) = \sqrt{\theta_{ip}}$, mapping $(0, \infty) \to (0, \infty)$. The inverse is $\theta_{ip} = \eta_{ip}^2$. It guarantees non-negativity. The implementation ensures

strict positivity via clamping.

4. **Inverse link (`"inverse"`):** $g_p(\theta_{ip}) = 1/\theta_{ip}$, implying $\theta_{ip} = 1/\eta_{ip}$. Requires $\eta_{ip} > 0$. Internal clamping applies for η_{ip} close to zero.
5. **Inverse Square Root link (`"inverse-sqrt"`):** Implemented as $g_p(\theta_{ip}) = 1/\sqrt{\theta_{ip}}$ (mapping $(0, \infty) \to (0, \infty)$), with inverse $\theta_{ip} = 1/\eta_{ip}^2$. Requires $\eta_{ip} \neq 0$ and ensures $\theta_{ip} > 0$.

Additionally, inspired by link functions commonly used in generalized linear models (GLMs) for binomial or proportion responses (though here applied to the *parameters* of GKw, not the response mean), `gkwreg` provides four link functions based on scaled CDFs. These map $\eta_{ip} \in \mathbb{R}$ to a bounded interval $(0, S)$, where $S > 0$ is a scaling factor (`scale_factor`) that **must** be specified by the user when choosing one of these links.

6. **Scaled Logit link (`"logit"`):** $\theta_{ip} = S \times \frac{\exp(\eta_{ip})}{1+\exp(\eta_{ip})}$, whose corresponding inverse (or unlink function) is: $\eta_{ip} = \ln\left(\frac{\theta_{ip}}{S-\theta_{ip}}\right)$.
7. **Scaled Probit link (`"probit"`):** $\theta_{ip} = S \times \Phi(\eta_{ip})$, where $\Phi(\cdot)$ is the standard normal CDF. The corresponding inverse is: $\eta_{ip} = \Phi^{-1}\left(\frac{\theta_{ip}}{S}\right)$.
8. **Scaled Cauchit link (`"cauchy"`):** $\theta_{ip} = S \times \left[0.5 + \frac{1}{\pi}\arctan(\eta_{ip})\right]$ and its corresponding inverse: $\eta_{ip} = \tan\left(\pi\left[\frac{\theta_{ip}}{S} - 0.5\right]\right)$.
9. **Scaled Complementary Log-Log link (`"cloglog"`):** $\theta_{ip} = S \times \left[1 - \exp\left(-\exp(\eta_{ip})\right)\right]$ with corresponding inverse: $\eta_{ip} = \ln\left(-\ln\left[1 - \frac{\theta_{ip}}{S}\right]\right)$.

NOTE: Critical Considerations and Justifications for Scaled Links:

Observe that in all scaled CDF links, $\theta_{ip} \in (0, S)$. If θ_{ip} exceeds this interval, the inverse function is no longer valid, reinforcing the need to properly specify S and check the parameter values θ_{ip} generated by the model or by data sampling. It is essential to recognize that the scaled CDF links (`"logit"`, `"probit"`, `"cauchy"`, `"cloglog"`) impose an **explicit upper limit** S on the parameter θ_{ip}. Conceptually, this is quite different from the first five links listed, which map to the unbounded interval $(0, \infty)$ (or its subsets, as in the identity or inverse cases). Although in the vast majority of practical applications the default `log` link is recommended and most natural for GKw parameters, there are specific circumstances where using scaled links may be warranted. These include: (i) contexts where physical, biological, or theoretical constraints impose a natural upper limit on parameters (e.g., biological growth rates limited by metabolic considerations); (ii) situations in which controlling the model's asymptotic behavior is desirable to avoid numerical instabilities when parameters take extremely large values; (iii) as an implicit regularization mechanism, penalizing extreme values and potentially improving estimation; (iv) to increase numerical stability during optimization, potentially improving Hessian conditioning; and (v) to facilitate interpretability and comparability of parameters in metanalysis or results communication. Even in such cases, using these scaled CDF links for parameters such as $\alpha, \beta, \gamma, \delta, \lambda$, whose natural domain is $(0, \infty)$, requires strong *a priori* theoretical or empirical justification for imposing such an upper bound S. If, for specific substantive reasons, a scaled CDF link is chosen, the `scale_factor` S must be supplied as an additional argument, and its value should be carefully justified in the application context, preferably by eliciting expert knowledge, analyzing historical data, or theoretical considerations in the domain.

The internal implementation in `gkwreg` uses numerically stable versions of these trans-

formations (e.g., employing functions like `log1p`, `expm1`, and clamping values near the boundaries or for extreme η_{ip}) to mitigate potential floating-point issues and improve the stability of the optimization process. However, these computational safeguards do not replace the need for substantive justification when choosing the link and its scale factor. It is recommended that any decision to use scaled links be accompanied by sensitivity analyses regarding the choice of S, as well as comparisons with models using the default log link, to verify the robustness of inferences under different specifications. Model inferential validity must always prevail over considerations of computational or interpretive convenience.

Maximum Likelihood Estimation (MLE)

Given an i.i.d. sample of n observations $(y_i, \mathbf{X}_i)$, where $\mathbf{X}_i$ denotes the matrix containing all relevant covariate values for observation i (considering predictors for all modeled parameters), the vector of unknown regression coefficients $\boldsymbol{\Theta}$ (which concatenates all unknown $\boldsymbol{\beta}_\alpha, \ldots, \boldsymbol{\beta}_\lambda$) is estimated by maximizing the joint log-likelihood function. For the full GKw model, the log-likelihood is:

$$
\begin{aligned}
\ell(\boldsymbol{\Theta}; \mathbf{y}, \mathbf{X}) &= \sum_{i=1}^{n} \log f\big(y_i; \boldsymbol{\theta}_i(\boldsymbol{\Theta})\big) \\
&= \sum_{i=1}^{n} \Bigg[\log(\lambda_i) + \log(\alpha_i) + \log(\beta_i) - \log B(\gamma_i,\, \delta_i + 1) \\
&\quad + (\alpha_i - 1)\log(y_i) + (\beta_i - 1)\log\big(1 - y_i^{\alpha_i}\big) \\
&\quad + (\gamma_i \lambda_i - 1)\log\Big(1 - \big(1 - y_i^{\alpha_i}\big)^{\beta_i}\Big) \\
&\quad + \delta_i \log\Big(1 - \Big[1 - \big(1 - y_i^{\alpha_i}\big)^{\beta_i}\Big]^{\lambda_i}\Big) \Bigg],
\end{aligned}
\tag{1}
$$

where each parameter $\theta_{ip} \in \{\alpha_i, \beta_i, \gamma_i, \delta_i, \lambda_i\}$ is an implicit function of the global coefficient vector $\boldsymbol{\Theta}$ through the link functions and linear predictors: $\theta_{ip} = g_p^{-1}(\mathbf{x}_{ip}^\top \boldsymbol{\beta}_p)$. For GKw subfamilies (Table 1), the log-likelihood expression simplifies according to the constraints imposed on the parameters (e.g., $\lambda_i = 1$ for BKw, $\gamma_i = 1$ for KKw, etc.), and $\boldsymbol{\Theta}$ contains only the regression coefficients for the effectively modeled parameters. Maximizing $\ell(\boldsymbol{\Theta})$ is computationally demanding due to the function's complexity and the potentially high dimensionality of $\boldsymbol{\Theta}$, especially when multiple parameters $(\alpha_i, \ldots, \lambda_i)$ depend on several sets of covariates. This underscores the need for efficient and stable optimization techniques, such as those implemented in `gkwreg` via `TMB` and `C++`.

Inference and Diagnostics

Inference

Under standard regularity conditions (see, e.g., (Lehmann, 1999)), the maximum likelihood estimator (MLE) $\hat{\boldsymbol{\Theta}}$ is consistent and asymptotically normally distributed. Its variance-covariance matrix can be consistently estimated by the inverse of the observed Fisher information matrix, evaluated at the MLE (Pawitan, 2013):

$$
\widehat{\mathrm{Var}}(\hat{\boldsymbol{\Theta}}) \approx \mathcal{I}(\hat{\boldsymbol{\Theta}})^{-1} = \left[-\frac{\partial^2 \ell(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}\, \partial \boldsymbol{\Theta}^\top} \right]^{-1}_{\boldsymbol{\Theta} = \hat{\boldsymbol{\Theta}}}.
$$

In `gkwreg`, the Hessian matrix (second partial derivatives of the log-likelihood), $\mathcal{H}(\boldsymbol{\Theta}) = \frac{\partial^2 \ell}{\partial \boldsymbol{\Theta}\, \partial \boldsymbol{\Theta}^\top}$, is calculated with high precision using `TMB`'s automatic differentiation capabilities.

The estimated variance-covariance matrix is then $\widehat{\text{Var}}(\hat{\boldsymbol{\Theta}}) = [-\mathcal{H}(\hat{\boldsymbol{\Theta}})]^{-1}$. Standard errors (SE) for individual coefficient estimates $\hat{\beta}_j$ (components of $\hat{\boldsymbol{\Theta}}$) are the square roots of the diagonal elements of this inverse matrix.

These standard errors allow construction of approximate $(1-\alpha_{IC})100\%$ Wald confidence intervals (CI): $\hat{\beta}_j \pm z_{1-\alpha_{IC}/2} \times \text{SE}(\hat{\beta}_j)$, where $z_{1-\alpha_{IC}/2}$ is the $1-\alpha_{IC}/2$ quantile of the standard normal distribution (e.g., $z_{0.975} \approx 1.96$ for 95% CI). Wald hypothesis tests for individual coefficient significance (typically $H_0 : \beta_j = 0$ vs. $H_1 : \beta_j \neq 0$) can be done using the statistic $z = \hat{\beta}_j/\text{SE}(\hat{\beta}_j)$ (compared with $N(0,1)$) or $W_j = z^2 = (\hat{\beta}_j/\text{SE}(\hat{\beta}_j))^2$ (compared with χ^2_1). This information (SE, z/t-statistic, p-value) is standard output provided by the `summary()` method for a `gkwreg` object. For potentially more accurate inferences, especially with smaller sample sizes or when estimated parameters are near boundary values, profile likelihood-based confidence intervals can be computed (if requested via `profile = TRUE` when fitting the model), although that is more computationally intensive (Pawitan, 2013; Venzon & Moolgavkar, 1988).

Model Selection

The hierarchical structure of the GKw family (Table 1) greatly facilitates model selection using likelihood-based methods. To compare two nested models (e.g., Kw vs. EKw, or Beta vs. BKw, or models differing only by adding/dropping covariates for a specific parameter), the Likelihood Ratio Test (LRT) is an appropriate tool. The LRT statistic is $LR = 2\big(\ell(\hat{\boldsymbol{\Theta}}_{\text{full}}) - \ell(\hat{\boldsymbol{\Theta}}_{\text{reduced}})\big)$, where $\ell(\cdot)$ denotes the maximized log-likelihood for each model. Under the null hypothesis that the reduced (simpler) model is adequate, LR is asymptotically chi-squared (χ^2_q) with q degrees of freedom, where q is the difference in the number of free parameters estimated between the full and reduced models.

For non-nested model comparison or as a general measure that penalizes model complexity, information criteria are commonly employed. The Akaike Information Criterion (AIC) (Akaike, 1974) and the Bayesian Information Criterion (BIC) (Schwarz, 1978), also known as the Schwarz Information Criterion (SIC), are defined as:

$$\text{AIC} = -2\ell(\hat{\boldsymbol{\Theta}}) + 2k,$$

$$\text{BIC} = -2\ell(\hat{\boldsymbol{\Theta}}) + k\log(n),$$

where $\ell(\hat{\boldsymbol{\Theta}})$ is the maximized log-likelihood, k is the total number of estimated parameters in $\boldsymbol{\Theta}$ (i.e., the dimension of $\boldsymbol{\Theta}$), and n is the sample size. Models with lower AIC or BIC values are generally preferred, indicating a better balance between data fit and parsimony. BIC tends to penalize more complex models more strictly than AIC, especially for larger samples. Both criteria are readily computed and returned by the `AIC()`, `BIC()`, and `summary()` methods in `gkwreg`. For a more in-depth discussion on model selection using information criteria, see (Burnham & Anderson, 2002).

Model Diagnostics and Goodness-of-Fit

Evaluating the quality of fit of the chosen model and verifying its underlying assumptions is a critical step in any statistical analysis. Residuals defined in a simple manner, such as raw residuals $(y_i - \hat{\mu}_i)$, where $\hat{\mu}_i = E(Y_i|\mathbf{X}_i; \hat{\boldsymbol{\Theta}})$ is the estimated conditional mean, often do not behave well (e.g., they do not have constant variance or follow a normal distribution) for non-normally distributed responses, particularly for those bounded on $(0,1)$. Following best practices established for generalized linear models (GLMs) and related frameworks like beta regression (Cribari-Neto & Zeileis, 2010; Ferrari & Cribari-Neto, 2004; McCullagh & Nelder, 1989), using **randomized quantile residuals** (Dunn & Smyth, 1996) is strongly recommended for diagnostic purposes with continuous $(0,1)$ responses.

For a continuous response Y with CDF $F(y;\boldsymbol{\theta})$, the quantile residual for the i-th observation is:

$$r_i^Q = \Phi^{-1}\big(F(y_i;\hat{\boldsymbol{\theta}}_i)\big),$$

where $F(y_i;\hat{\boldsymbol{\theta}}_i)$ is the fitted GKw family CDF (e.g., GKw, BKw, Kw, Beta) evaluated at the observed response y_i using the estimated parameters $\hat{\boldsymbol{\theta}}_i = (\hat{\alpha}_i, \ldots, \hat{\lambda}_i)$ for that observation, and $\Phi^{-1}(\cdot)$ is the quantile function (inverse CDF) of the standard normal distribution, $N(0,1)$. The "randomization" proposed by (Dunn & Smyth, 1996) is primarily relevant for discrete or mixed distributions to break ties; for strictly continuous distributions like GKw and its subfamilies on $(0,1)$, the definition above is generally sufficient and is equivalent to the Cox-Snell residual transformed by Φ^{-1}. The fundamental property of these residuals is that if the model is correctly specified (i.e., the assumed distributional form and the regression structure for the parameters are correct), the quantile residuals r_i^Q should be approximately i.i.d. samples from a standard normal distribution, $N(0,1)$.

Hence, diagnostic plots based on these residuals are essential: (i) A **quantile-quantile (QQ) plot** of the quantile residuals r_i^Q against the theoretical quantiles of a $N(0,1)$ should approximately lie on a straight line through the origin with slope 1. Systematic departures from this line indicate inadequacy of the assumed distributional form. (ii) **Plots of the quantile residuals r_i^Q vs. the fitted mean** $\hat{\mu}_i$ (if calculated), or vs. the fitted linear predictors $\hat{\eta}_{ip}$ for each parameter, or vs. individual covariates $\mathbf{x}_{ij}$ can help detect problems such as misspecification of functional forms in the linear predictors, omission of relevant covariates, or patterns of heteroskedasticity not captured by the model. Ideally, these plots should show no systematic patterns, with points scattered randomly around zero.

The `gkwreg` package provides these residuals through `residuals(model_object, type = "quantile")` and includes standard diagnostic plotting functions via the `plot()` method on the fitted model object, typically encompassing the QQ plot and residuals-vs.-fitted plots. Significant departures from normality or obvious patterns in the residual plots indicate potential model misspecification, suggesting the need to reconsider the distributional choice, the linear predictor structure, or the inclusion of additional terms (Cribari-Neto & Zeileis, 2010; Dunn & Smyth, 1996). Furthermore, formal goodness-of-fit tests (e.g., Kolmogorov-Smirnov, Anderson-Darling) can be applied to the quantile residuals to test the hypothesis of normality $N(0,1)$, although graphical inspection is often more informative in practice.

The gkwreg Package

The `R` package `gkwreg` is the primary contribution of this work, providing the computational and statistical tools necessary for fitting regression models based on the GKw family described in Section .

Computational Engine: Template Model Builder (TMB)

At the core of `gkwreg` lies the `TMB` (Kristensen et al., 2016) package. `TMB` is a highly efficient statistical and computational framework for fitting complex models in `R`. It uses `C++` templates and automatic differentiation (AD), implemented through the `CppAD`(Bell, 2007) library, to facilitate fast and stable maximum likelihood estimation (or restricted maximum likelihood - REML), making it particularly powerful for models involving random effects or latent variables.

Although `TMB` is renowned for its efficient handling of models with high-dimensional latent variables (e.g., spatial models, generalized mixed models) by combining Laplace approximation (LA) with AD (Osgood-Zimmerman & Wakefield, 2021), its AD capabilities are equally beneficial for models that involve only fixed effects but have complex likelihood

functions, such as the GKw regression models considered here. In the `gkwreg` context, where all regression coefficients $\boldsymbol{\Theta}$ are treated as fixed parameters to be estimated, the objective function to be maximized (or minimized, in the case of negative log-likelihood) is simply the log-likelihood $\ell(\boldsymbol{\Theta})$ given in Equation (1). The Laplace approximation component of `TMB` is not used in this scenario.

However, `TMB`'s automatic differentiation engine remains crucial. The log-likelihood function (Equation (1)), together with transformations induced by the link functions (Section), is implemented in `C++` using `TMB`'s specialized data types (based on `CppAD::AD<double>`). With this implementation, `TMB` automatically generates efficient `C++` code to compute the analytical gradient vector $\nabla_{\boldsymbol{\Theta}}\ell(\boldsymbol{\Theta})$ and the Hessian matrix $\mathcal{H}(\boldsymbol{\Theta})$ with respect to all parameters in $\boldsymbol{\Theta}$ (Griewank & Walther, 2008; Skaug & Fournier, 2016). These exact derivatives are then available to robust, well-tested numerical optimization routines provided by `R`, such as `nlminb` (Gay, 1990) (the `gkwreg` default) or methods available in `optim()` (e.g., `BFGS`, `L-BFGS-B`, `Nelder-Mead`), which use them to efficiently find the MLE $\hat{\boldsymbol{\Theta}}$.

Computational Advantages

Using `TMB` as the computational engine for `gkwreg` offers a synergistic combination of major advantages for fitting GKw regression models, resulting in higher **speed**, **accuracy**, **stability**, and **scalability**:

- **Speed:** Automatic differentiation (AD) computes exact derivatives far more efficiently than numeric differentiation (e.g., finite differences), especially for complex functions with many parameters. Benchmarks often show substantial speedups (e.g., orders of magnitude) compared to pure `R` implementations that rely on numeric derivatives or even manually coded analytical derivatives in R. This speed advantage is further enhanced by evaluating the likelihood and its derivatives in compiled `C++` code.
- **Accuracy:** AD provides machine-precision derivatives, eliminating truncation and cancellation errors inherent to numeric differentiation approaches. This translates directly into more accurate parameter estimates $\hat{\boldsymbol{\Theta}}$ and, crucially, more reliable standard errors, since these are derived from the (inverted) Hessian matrix.
- **Stability:** Having analytically exact gradients and Hessians markedly improves the stability and convergence rate of numerical optimization algorithms. This enables successful fitting of complex models—those with multiple covariates influencing multiple parameters simultaneously—that might fail to converge or produce spurious solutions if optimized using less precise numeric derivatives or algorithms that do not exploit the Hessian.
- **Scalability:** `TMB`'s efficient derivative handling makes it exceptionally well-suited for models with potentially large numbers of regression coefficients, as can occur when all five GKw parameters $(\alpha, \ldots, \lambda)$ depend on different (or the same) sets of covariates. Although the Hessian matrix for GKw regression may not exhibit the sparsity structure that makes `TMB` particularly advantageous for spatial or GMRF models (Osgood-Zimmerman & Wakefield, 2021), its computational efficiency for dense derivatives remains highly beneficial.

In summary, integration with `TMB` enables `gkwreg` to provide a robust and efficient tool, making estimation of complex GKw regression models feasible and reliable for applied researchers within reasonable time frames.

Model Specification and Main Functions

The `gkwreg` package is designed with a user interface intended to be intuitive and consistent with familiar statistical modeling conventions in `R`, offering high-level functions for model estimation and standard methods for result extraction, inference, and diagnostics.

Core Implementations and Base Functions

At the fundamental level, the package includes efficient `C++` implementations (via `Rcpp` (Dirk Eddelbuettel & Balamuta, 2018) and `RcppArmadillo` (D. Eddelbuettel & Sanderson, 2014)) of the essential functions for each of the seven distributions in the GKw family (inclusive) listed in Table 1. These encompass the standard density (`d*`), cumulative distribution (`p*`), quantile (`q*`), and random generation (`r*`) functions, following the naming conventions of `R` (e.g., `dgkw`, `pgkw`, `qkw`, `rbeta_`). Note that for the Beta distribution, the underscore suffix `_` is used (`dbeta_`, `pbeta_`, etc.) to avoid conflicts with `R`'s native functions. Additionally, the package internally implements functions to compute the log-likelihood (`ll*`), the analytical gradient (`gr*`), and the analytical Hessian (`hs*`) for all distribution families for learning and extensibility. Although these analytical derivatives could be used directly for optimization (and a Newton-Raphson implementation, `nrgkw`, is available for testing or debugging), the main estimation functions (`gkwreg` and `gkwfit`) primarily use `TMB`'s automatic differentiation capabilities, as discussed in Section , due to their computational advantages like precision, robustness, and speed.

Regression Modeling with gkwreg

The main function for regression analysis in the package is `gkwreg()` (acronym for **G**eneralized **K**umaras**w**amy **Reg**ression). It enables users to fit regression models where the parameters of a distribution from the GKw family (or its subfamilies) are modeled as functions of covariates, implementing the framework described in Section . This function relies **exclusively** on the `TMB` backend for maximum likelihood estimation, ensuring robustness and computational efficiency.

A key feature of `gkwreg()` is its flexible formula interface, which takes advantage of the `Formula` (Zeileis & Croissant, 2010) package. This allows potentially different regression structures to be specified for each parameter of the selected distribution. The `formula` argument accepts a multi-part structure, where the right-hand side parts are separated by the pipe symbol (`|`), corresponding sequentially to the linear predictors for parameters $\alpha, \beta, \gamma, \delta, \lambda$:

```
# General formula structure for gkwreg (gkw family)
response ~ alpha_predictors | beta_predictors | gamma_predictors |
           delta_predictors | lambda_predictors
```

Within each part (`*_predictors`), standard `R` formula syntax is used (e.g., `x1 + x2`, `x1*x2`, `factor(group)`, `log(time)`, `poly(age, 2)`, etc.). If a specific parameter's part in the formula is omitted or explicitly set to `~ 1`, `gkwreg()` will fit an **intercept-only** model for that parameter. This means the corresponding parameter is treated as constant for all observations, and its value (on the link scale) is estimated from the data without covariates. The number of formula parts actually interpreted depends on the `family` argument specified. For example, for `family = "kw"`, only the first two parts (for α and β) are relevant; subsequent parts, even if provided, would be ignored.

This flexible syntax enables users to tailor the regression structure precisely to their data and hypotheses. For instance:

```
# Example 1: Model alpha with x1, beta with x2 and x3,
#            and the rest (gamma, delta, lambda) as constants.
#            Assuming family = "gkw".
fit1 <- gkwreg(y ~ x1 | x2 + x3 | 1 | 1 | 1, data = mydata, family = "gkw")

# Example 2: The same model as fit1, omitting the final parts
#            (which default to '~ 1').
#            Identical to fit1.
fit2 <- gkwreg(y ~ x1 | x2 + x3, data = mydata, family = "gkw")
```

```r
# Example 3: Fit a Kw model (2 parameters) where alpha depends on x1
#            and beta is constant.
fit3 <- gkwreg(y ~ x1 | 1, data = mydata, family = "kw")

# Example 4: Fit a Beta model (gamma and delta parameters) where gamma
#            depends on x1 and delta depends on x2.
#            Note that the formula's first two parts (for alpha and beta) must
#            still be included, even if trivial ('~0' or '~1'), so that the
#            parts for gamma and delta are correctly parsed.
#            A clearer way is to specify family='beta' and use only
#            the two formula parts. Interpret x1 for gamma, x2 for delta.
fit4 <- gkwreg(y ~ x1 | x2, data = mydata, family = "beta")
```

This structure provides considerable control over model specification, allowing different covariates to influence different aspects of the shape and location of the response distribution as captured by the GKw parameters.

The `gkwreg()` function signature reveals its key arguments and options:

```r
gkwreg(formula, data, family = "gkw", link = NULL, start = NULL, fixed = NULL,
       method = "nlminb", conf.level = 0.95, optimizer.control = list(),
       subset = NULL, weights = NULL, offset = NULL, x = FALSE, y = TRUE,
       na.action = getOption("na.action"), model = TRUE, silent = TRUE, ...)
```

Key arguments include:

- `formula`: The multi-part formula (a `Formula` object).
- `data`: The data frame containing the variables.
- `family`: A string specifying which GKw subfamily to use (e.g., `"gkw"`, `"kw"`, `"beta"`, `"ekw"`, etc., as in Table 1). Default is `"gkw"`.
- `link`: `NULL` (uses `"log"` for all modeled parameters) or a named list specifying the link function for each parameter (e.g., `list(alpha = "log", beta = "log", gamma = "identity")`).
- `start`: `NULL` (the package attempts to find initial values automatically) or a named list with initial coefficient vectors for each modeled parameter. Providing good starting values can be crucial for convergence in complex models.
- `fixed`: `NULL` or a named list specifying which coefficients should be fixed at specific values during estimation (e.g., `list(alpha = c("(Intercept)" = 0, x1 = 1))` would fix the alpha intercept at exp(0) = 1 and the x1 coefficient at exp(1) if the link is log).
- `method`: The optimization algorithm to pass to TMB (via `TMB::MakeADFun` and `nlminb` or `optim` from `stats`). Default is `"nlminb"`.
- `hessian`: Logical indicating whether the Hessian matrix should be calculated at the optimum (needed for standard errors). Default `TRUE`.
- `conf.level`: Confidence level for Hessian-based intervals. Default 0.95.
- `optimizer.control`: List of control parameters for the optimizer (e.g., `iter.max`, `eval.max`).
- `x`, `y`, `model`: Logical flags indicating whether to return the model matrix, the response variable, and the model frame in the fitted object.
- `silent`: Logical to suppress messages during fitting. Default `TRUE`.

Distribution Fitting with gkwfit

For the simpler task of fitting a distribution from the GKw family to a univariate data vector $y_1, \ldots, y_n$ (i.e., without covariates, estimating only the distribution's parameters), the `gkwfit()` function is available. It also supports all seven families listed in Table

1. Parameter estimation is performed using the same `TMB` backend as `gkwreg`, ensuring consistency and efficiency. The available optimization methods are the same, selected via the `method` argument.

Its signature is:

```
gkwfit(data, family = "gkw", start = NULL, fixed = NULL, method = "nlminb",
       use_moments = FALSE, hessian = TRUE, profile = FALSE, npoints = 20,
       plot = TRUE, conf.level = 0.95, optimizer.control = list(),
       submodels = FALSE, silent = TRUE, ...)
```

Notable arguments specific to `gkwfit()` include:

- `data`: A numeric vector with values strictly between 0 and 1. Values at the boundaries $(0, 1)$ may cause issues; consider slight adjustments if necessary.
- `family`: A character string specifying the distribution family. One of: `gkw` (default), `bkw`, `kkw`, `ekw`, `mc`, `kw`, or `beta`.
- `start`: Optional list with initial parameter values (using natural parameter names like alpha, beta, etc.). If `NULL`, reasonable starting values will be determined, potentially using the method of moments if `use_moments = TRUE`.
- `use_moments`: Logical; if `TRUE`, attempts to use the method of moments for initial values (may not be implemented for all families).
- `profile`: Logical; if `TRUE`, calculates likelihood-based profile confidence intervals (computationally intensive).
- `npoints`: Number of points to use in the likelihood profile.
- `plot`: Logical; if `TRUE`, generates a basic plot of the fit (e.g., histogram with fitted density).
- `method`: Optimization method to use. One of: `nlminb` (default), `Nelder-Mead`, `BFGS`, `CG`, `L-BFGS-B` or `SANN`. If `nlminb` is selected, R's nlminb function is used; otherwise, R's optim function is used with the specified method.
- `submodels`: Logical; if `TRUE` and `family = "gkw"`, attempts to also fit the nested subfamilies for comparison.
- `profile`: computes likelihood profiles for parameters using `TMB`'s profiling capabilities. Default: `FALSE`.

The `gkwfit()` function returns a `gkwfit` object containing parameter estimates, the maximized log-likelihood, the Hessian matrix (if `hessian=TRUE`), and other relevant fitting information.

Output, Methods, and Post-Estimation Tools

Both `gkwreg()` and `gkwfit()` return objects of specific classes (`gkwreg` and `gkwfit`, respectively) that encapsulate the estimation results. Following best practices of object-oriented programming in `R`, `gkwreg` implements standard generic methods for these objects. This creates a consistent, familiar interface for users, mirroring the behavior of widely used modeling functions in `R`, such as `lm()` and `glm()`.

Table 2 summarizes the main generic methods available for `gkwreg` objects (and, to a large extent, `gkwfit` objects). These are familiar to `R` users for interacting with results from the package, performing statistical inference, assessing fit quality, and systematically comparing alternative models. For `gkwfit` (univariate fits), a subset of these methods is available (e.g., `coef`, `logLik`, `AIC`, `BIC`, `plot`). Additionally, the `gkwgof()` function can be used to carry out (univariate) Goodness-of-Fit tests and produce more specialized diagnostics, and `gkwfitall` allows fitting all seven families simultaneously.

Table 2: Primary S3 generic methods available for objects of class `gkwreg`.

Method	Description
`print()`	Displays concise information about the fitted model, including the function call, the family used, and the estimated regression coefficients for each parameter.
`summary()`	Provides more detailed output, typically including: coefficient estimates ($\hat{\boldsymbol{\beta}}_p$), their standard errors (SE), z (or t) statistics, and associated p-values (based on the asymptotic normal/Wald test). It also reports the link functions used, the maximized log-likelihood $\ell(\hat{\boldsymbol{\Theta}})$, AIC, BIC, and possibly the number of iterations in the optimizer and the convergence code.
`coef()`	Extracts the vector (or list, if multiple parameters are modeled) of all regression coefficients $\hat{\boldsymbol{\Theta}} = (\hat{\boldsymbol{\beta}}_\alpha^\top, \ldots, \hat{\boldsymbol{\beta}}_\lambda^\top)^\top$.
`vcov()`	Extracts the estimated variance-covariance matrix for all regression coefficients, $\widehat{\mathrm{Var}}(\hat{\boldsymbol{\Theta}})$, computed from the inverse of the negative Hessian at the MLE.
`logLik()`	Extracts the maximized log-likelihood value, $\ell(\hat{\boldsymbol{\Theta}})$, with attributes indicating the number of estimated parameters.
`AIC()` `BIC()`	Compute the Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC), respectively, for the fitted model, useful for model comparison.
`predict()`	Computes predicted values based on the fitted model. The 'type' argument controls the prediction type: '"link"' returns the linear predictors ($\hat{\eta}_{ip}$); '"response"' returns the estimated parameters on the original scale ($\hat{\theta}_{ip} = g_p^{-1}(\hat{\eta}_{ip})$); '"mean"' (if implemented for the specific family) attempts to return the estimated conditional mean $E(Y_i|\mathbf{X}_i; \hat{\boldsymbol{\Theta}})$. The 'newdata' argument allows predictions for new observations.
`residuals()`	Computes model residuals. The residual type is controlled by the 'type' argument. 'type = "quantile"' (the default and recommended) returns the randomized quantile residuals (see Section), ideal for model evaluation. Other types, such as '"response"' (raw residuals $y_i - \hat{\mu}_i$), may be available but have limited diagnostic value.
`plot()`	Produces a set of standard diagnostic plots, typically based on quantile residuals. Commonly includes a normal QQ plot and residuals-vs.-fitted or linear predictor plots, useful for checking model assumptions (see Section).
`formula()`	Extracts the model formula used in fitting.
`terms()`	Extracts the model terms.
`model.frame()`	Extracts the model frame used in fitting (if 'model = TRUE' in the original call).
`model.matrix()`	Extracts the model matrix/matrices used in fitting (if 'x = TRUE' in the original call).

Real Data Examples

To demonstrate the practical application and flexibility of the `gkwreg` package, we now analyze four well-known datasets in the literature on bounded data modeling, from different subject areas. These examples illustrate how the various distributions in the GKw family can be used to capture complex patterns in $(0, 1)$-bounded data and how the package's hierarchical structure facilitates model selection. We additionally include a fifth example focusing on univariate distribution fitting using `gkwfit()`.

Food Expenditure Data

Our first example uses the `FoodExpenditure` dataset, available in the `gkwreg` package (Cribari-Neto & Zeileis, 2010). This dataset contains information on the ratio of family income (`income`) spent on food (`food`) for a sample of families, along with the number of people in the family (`persons`). The response variable of interest is $y = \text{food/income}$, a proportion naturally restricted to $(0, 1)$. We will model how this proportion varies as a function of the number of people in the family.

```
# Load packages (if not already)
library(gkwreg)

# Wrapper function to compare different gkw models for bounded data
fit_and_compare_models_2p <- function(
                formula, data, method = "BFGS",
                families = c("gkw", "bkw", "kkw", "ekw", "mc", "kw", "beta"),
                use_betareg = TRUE) {

  # Map family codes to descriptive names
  family_names <- c(
    gkw = "Generalized Kumaraswamy", bkw = "Beta-Kumaraswamy",
    kkw = "Kumaraswamy-Kumaraswamy", ekw = "Exponential Kumaraswamy",
    mc = "McDonald/Beta Power", kw = "Kumaraswamy", beta = "Beta (gkwreg)"
  )

  # Number of parameters for each family
  family_params <- list(
    gkw = 5, bkw = 4, kkw = 4, ekw = 3, mc = 3, kw = 2, beta = 2
  )

  results <- list()

  # Fit gkwreg models
  for (fam in families) {
    num_links <- family_params[[fam]]

    # Try to fit model with basic error handling
    fit <- try(gkwreg(formula, data = data, family = fam,
                      link = rep("log", num_links),
                      method = method, silent = TRUE), silent = TRUE)

    # Extract model statistics if fit successful
    if (!inherits(fit, "try-error")) {
      results[[fam]] <- data.frame(
        Model = family_names[[fam]],
        Code = fam,
```

```
        Parameters = length(unlist(coef(fit))),
        logLik = as.numeric(logLik(fit)),
        AIC = AIC(fit),
        BIC = BIC(fit),
        Conv = max(as.numeric(fit$convergence)),
        row.names = fam
      )
    }
  }

  # Fit betareg model if requested
  if (use_betareg) {

    # Fit standard beta regression
    br_fit <- try(
      betareg::betareg(formula, data = data,
                       link = "log", link.phi = "log"),
      silent = TRUE)

    # Add betareg results if successful
    if (!inherits(br_fit, "try-error")) {
      results[["betareg"]] <- data.frame(
        Model = "Beta (betareg)",
        Code = "betareg",
        Parameters = length(coef(br_fit)),
        logLik = as.numeric(logLik(br_fit)),
        AIC = AIC(br_fit),
        BIC = BIC(br_fit),
        Conv = max(as.numeric(br_fit$converged)),
        row.names = "betareg"
      )
    }
  }

  # Combine all results and sort by AIC
  results_df <- do.call(rbind, results)
  results_df <- results_df[order(results_df$AIC), ]
  rownames(results_df) <- NULL

  return(results_df)
}

# Get FoodExpenditure data and create 'y' as the response
food_data <- get_bounded_datasets("FoodExpenditure")
food_data <- within(food_data, {y = food / income})

# Define the formula: y depends on 'persons'
# We'll model only gamma and delta for Beta, keeping other parameters constant
formu_fe <- y ~ persons | income

# Fit all families
results_food_data <- fit_and_compare_models_2p(formu_fe, food_data,
                                               method = "nlminb")
```

Table 3: Comparison of regression models for the FoodExpenditure data. Models ordered by AIC.

Model Family	Code	N. Par.	LogLik	AIC	BIC	Conv.
Kumaraswamy	kw	4	46.34	-84.69	-78.14	1
Exponential Kumaraswamy	ekw	5	46.99	-83.98	-75.79	1
Beta-Kumaraswamy	bkw	6	46.99	-81.98	-72.15	1
Kumaraswamy-Kumaraswamy	kkw	6	46.98	-81.97	-72.14	0
Beta (gkwreg)	beta	4	44.64	-81.28	-74.73	1
Generalized Kumaraswamy	gkw	7	46.99	-79.98	-68.52	0
McDonald/Beta Power	mc	5	44.74	-79.49	-71.30	1
Beta (betareg)	betareg	4	39.03	-70.05	-63.50	1

```
# Best model
kw_model <- gkwreg(formu_fe, food_data, family = "kw", method = "nlminb")
summary(kw_model)
plot(kw_model, use_ggplot = TRUE, arrange_plots = TRUE, sub.caption = "")
```

Based on the AIC values in Table 3, the Kumaraswamy (Kw) model offers the best fit ($AIC = -84.69$), closely followed by the Exponential Kumaraswamy (EKw) model ($AIC = -83.98$), while the standard Beta model (betareg) has the least favorable AIC. This indicates that a more flexible shape improves model fit compared to the Beta baseline, and the Kw model in particular strikes a good balance between fit and parsimony when modeling food expenditure proportions using "persons" as a predictor. Moreover, likelihood ratio tests (LRTs) were conducted to compare the more complex models—EKw (5 parameters) and Beta-Kumaraswamy (BKw, 6 parameters)—against the simpler Kw model (4 parameters). The LRT statistic was computed as $LR = 2\left(\ell_{\text{complex}} - \ell_{\text{kw}}\right)$, where ℓ_{complex} is the maximized log-likelihood of the complex model and ℓ_{kw} is that of the Kw model. For example, the EKw model had a log-likelihood difference of $46.99 - 46.34 = 0.65$, yielding $LR = 2 \times 0.65 = 1.29$; given 1 degree of freedom, this corresponds to a p-value of approximately 0.26. Similarly, the BKw model also yielded $LR = 1.29$ with 2 degrees of freedom ($p \approx 0.48$). These non-significant p-values suggest that the additional parameters in the EKw and BKw models do not provide a significant improvement in fit over the simpler Kw model.

Gasoline Yield Data

Our second example uses the `GasolineYield` dataset, also in `gkwreg` and originally analyzed by (Prater, 1956) and later by (Atkinson, 1985). This dataset documents the proportion (`yield`) of crude oil converted into gasoline after distillation and fractionation processes, with 32 observations. Predictors include crude oil gravity (`gravity`, API), vapor pressure (`pressure`, lbf/in^2), a 10% critical temperature of evaporation (`temp10`), the final boiling point temperature (`temp`), and a categorical factor representing the crude oil batch (`batch`) with 10 levels. Following previous analyses (Ferrari & Cribari-Neto, 2004), we model the proportion `yield` as a function of batch (`batch`) and final temperature (`temp`), allowing `temp` to also influence the secondary dispersion/shape parameter. Because the response $y \in (0, 1)$ and Atkinson observed possible skewed errors, this dataset is particularly relevant for testing regression models for bounded data.

```
# Load GasolineYield data
gasoline_data <- get_bounded_datasets("GasolineYield")

# Formula: yield ~ batch + temp | temp
# First part (for alpha/gamma) includes batch and temp
```

Table 4: Comparison of regression models for the Gasoline Yield data. Models ordered by AIC.

Model Family	Code	N. Par.	LogLik	AIC	BIC	Conv.
Kumaraswamy	kw	13	96.48	-166.97	-147.91	1
Exponential Kumaraswamy	ekw	14	97.17	-166.33	-145.81	1
Kumaraswamy-Kumaraswamy	kkw	15	97.26	-164.52	-142.53	1
McDonald/Beta Power	mc	14	96.16	-164.32	-143.80	1
Beta-Kumaraswamy	bkw	15	96.07	-162.14	-140.16	1
Generalized Kumaraswamy	gkw	16	95.59	-159.18	-135.73	1
Beta (gkwreg)	beta	13	87.68	-149.37	-130.31	1
Beta (betareg)	betareg	13	81.01	-136.03	-116.97	1

```
# Second part (for beta/delta/phi) includes only temp
formu_gy <- yield ~ batch + temp | temp

# Load the GasolineYield data
gasoline_data <- get_bounded_datasets("GasolineYield")

# Define the formula: yield ~ batch + temp | temp
# The first part (for alpha/gamma) includes batch and temp
# The second part (for beta/delta/phi) includes only temp
formu_gy <- yield ~ batch + temp | temp

# Execute the comparative analysis
results_gasoline <- fit_and_compare_models_2p(formu_gy, gasoline_data,
                                              method = "BFGS")

# Best model
kw_model_gas <- gkwreg(formu_gy, gasoline_data, family = "kw",
                       method = "nlminb")
summary(kw_model_gas)
plot(kw_model_gas, use_ggplot = TRUE, arrange_plots = TRUE, sub.caption = "")
```

Based on the AIC values for the `GasolineYield` data in Table 4, the Kumaraswamy (kw) model provides the best fit with an AIC of -166.97, closely followed by the Exponential Kumaraswamy (ekw) model with an AIC of -166.33, whereas both the Beta (gkwreg) and Beta (betareg) models yield considerably higher AICs (-149.37 and -136.03, respectively), indicating poorer fit. To further assess whether the additional parameters in the more complex models enhance the fit, likelihood ratio tests (LRTs) were conducted by calculating $LR = 2\big(\ell_{\text{complex}} - \ell_{\text{kw}}\big)$, where ℓ_{complex} is the maximized log-likelihood of the complex model and ℓ_{kw} is that of the kw model. For instance, comparing the ekw model (14 parameters) to the kw model (13 parameters) yields a log-likelihood difference of $(97.17 - 96.48) = 0.69$, resulting in $LR = 2 \times 0.69 = 1.38$ with 1 degree of freedom and a corresponding p-value of approximately 0.24. Similarly, comparing the Beta-Kumaraswamy (bkw) model (15 parameters, logLik $= 96.07$) to the kw model results in a difference of $(96.48 - 96.07) = 0.41$, giving $LR = 2 \times 0.41 = 0.82$ with 2 degrees of freedom and a p-value of about 0.67. These non-significant p-values indicate that the extra parameters in the ekw and bkw models do not significantly improve the fit compared to the simpler kw model, thereby favoring the kw model for its superior balance of model fit and parsimony in analyzing the `GasolineYield` data.

Table 5: Comparison of regression models for the residual cancer detection data (sdac). Models ordered by AIC.

Model Family	Code	N. Par.	LogLik	AIC	BIC	Conv.
Exponential Kumaraswamy	ekw	5	199.04	-388.09	-370.70	1
Kumaraswamy-Kumaraswamy	kkw	6	199.04	-386.09	-365.23	1
Beta-Kumaraswamy	bkw	6	199.04	-386.09	-365.23	1
Beta (gkwreg)	beta	4	196.99	-385.98	-372.07	1
Generalized Kumaraswamy	gkw	7	199.04	-384.09	-359.75	0
McDonald/Beta Power	mc	5	197.04	-384.08	-366.70	0
Beta (betareg)	betareg	4	195.40	-382.81	-368.90	1
Kumaraswamy	kw	4	195.40	-382.79	-368.89	1

Residual Cancer Detection Data

For our third example, we examine a dataset related to residual cancer detection in patients undergoing stem cell transplantation. The `sdac` dataset (Stem Cell Transplant Data), available from the `simplexreg` package (Zhang, Qiu, & Shi, 2016), documents results from 239 patients who underwent autologous peripheral blood stem cell (PBSC) transplantation after myeloablative chemotherapy at the Edmonton Hematopoietic Stem Cell Laboratory between 2003 and 2008. This clinically relevant dataset includes variables such as patient age (`age`), adjusted age (`ageadj`, categorized <40 as baseline), gender (`gender`), the type of chemotherapy protocol (`chemo`, one-day vs. three-day regimen), and crucially, the viable CD34+ cell recovery rate (`rcd`), a $(0, 1)$-bounded proportion directly correlated with hematopoietic engraftment success (Allan et al., 2002; Yang et al., 2005). The bounded nature of `rcd` makes the dataset particularly suitable for testing specialized regression models for proportional data. We investigate how `ageadj` and `chemo` affect the recovery rate, allowing `age` also to influence the second parameter of the distribution.

```
# Load the sdac data
sdac_data <- get_bounded_datasets("sdac")

# Formula: rcd ~ ageadj + chemo | age
formu_sd <- rcd ~ ageadj + chemo | age

# Compare different families in gkwreg
results_sdac <- fit_and_compare_models_2p(formu_sd, sdac_data, method="nlminb",
                                          use_betareg = TRUE)

# Best model
ekw_model_gas <- gkwreg(formu_sd, sdac_data, family="ekw", method="BFGS")
summary(ekw_model_gas)
plot(ekw_model_gas, use_ggplot = TRUE, arrange_plots=TRUE, sub.caption="")
```

Based on the sdac dataset results in Table 5 — which examine the viable `CD34+` cell recovery rate (`rcd`) as a function of adjusted age (`ageadj`) and chemotherapy protocol (`chemo`), with age also influencing the second parameter—the Exponential Kumaraswamy (EKw) model yields the lowest $AIC(386.17)$ with 6 parameters and a log-likelihood of 199.08, compared to the simpler Kumaraswamy (Kw) model with 5 parameters ($logLik = 195.74, AIC = 381.49$). A likelihood ratio test between EKw and Kw computes a log-likelihood difference of 3.34, resulting in an LR value of $LR = 2 \times 3.34 = 6.68$ (with 1 degree of freedom), which corresponds to a p-value of approximately 0.01, indicating a significant improvement. Similarly, comparing the Beta-Kumaraswamy (BKw) model (7 parameters, $logLik = 199.48, AIC = 384.96$) to the Kw model gives a log-likelihood difference of 3.74, yielding $LR = 2 \times 3.74 = 7.48$ with 2 degrees of freedom ($p \approx 0.02$).

Although these tests show that the extra parameters in the EKw and BKw models contribute a statistically significant improvement over the Kw model, the EKw model's lower AIC demonstrates that it provides the best balance between model fit and parsimony for these clinical data.

Retinal Detachment Data

Our next example analyzes longitudinal ophthalmologic data from (Meyers, Ambler, & Tan, 1992), which tracked intraocular gas (C_3F_8) decay rates used in surgeries for complex retinal detachment. The `retinal` dataset, available in the `gkwreg` package, comprises 181 observations of 31 patients monitored over three months. Each patient was followed 3 to 8 times (mean 5.8 visits). The response of interest is the proportion (`Gas`) of the initial gas volume that remains in the eye at each visit—a classic proportion restricted to $(0, 1)$. Predictors include the time since gas injection (`Time`, in days), often transformed (e.g., `LogT = log(Time)`), and the initial gas concentration level (`Level`, a factor with levels $15\%, 20\%, 25\%$). The key research question is whether the initial gas concentration affects the decay rate over time. This dataset is ideal for illustrating multiparametric modeling, where different distribution parameters may depend on different predictors. Due to potential complexity, we focus on 3+ parameter models in the GKw family here.

```
# Function to fit and compare models with 3+ parameters
fit_and_compare_models_3p <- function(formula, data, method = "nlminb",
                                      families = c("gkw", "bkw", "kkw", "ekw", "mc")) {

  # Map family codes to descriptive names
  family_names <- c(
    gkw = "Generalized Kumaraswamy", bkw = "Beta-Kumaraswamy",
    kkw = "Kumaraswamy-Kumaraswamy", ekw = "Exponential Kumaraswamy",
    mc = "McDonald/Beta Power"
  )

  # Define number of parameters for each family
  family_params <- list(gkw = 5, bkw = 4, kkw = 4, ekw = 3, mc = 3)

  results <- list()

  # Fit each model family
  for (fam in families) {
    # Try to fit the model
    fit <- try(gkwreg(formula,
                      data = data,
                      family = fam,
                      link = rep("log", family_params[[fam]]),
                      method = method,
                      silent = TRUE),
               silent = TRUE)

    # Extract statistics if fit successful
    if (!inherits(fit, "try-error")) {
      results[[fam]] <- data.frame(
        Model = family_names[[fam]],
        Family = fam,
        Parameters = length(unlist(coef(fit))),
        logLik = as.numeric(logLik(fit)),
        AIC = AIC(fit),
```

Table 6: Comparison of models from the GKw family (with 3+ parameters) for the retinal detachment data. Models sorted by AIC.

Model Family	Code	N. Par.	LogLik	AIC	BIC	Conv.
Exponential Kumaraswamy	ekw	9	132.71	-247.42	-218.63	1
Beta-Kumaraswamy	bkw	10	132.71	-245.42	-213.43	1
Generalized Kumaraswamy	gkw	11	132.71	-243.42	-208.23	1
Kumaraswamy-Kumaraswamy	kkw	10	131.02	-242.05	-210.06	1
McDonald/Beta Power	mc	9	108.85	-199.71	-170.92	1

```
        BIC = BIC(fit),
        Conv = max(as.numeric(fit$convergence)),
        row.names = NULL
      )
    }
  }

  # Combine results and sort by AIC
  results_df <- do.call(rbind, results)
  results_df <- results_df[order(results_df$AIC), ]
  rownames(results_df) <- NULL

  return(results_df)
}

# Load the retinal data
retinal_data <- get_bounded_datasets("retinal")

# Formula modeling alpha, beta, gamma (for EKw/Mc) or more (KKw/BKw/GKw)
# alpha ~ LogT + LogT2 + Level
# beta  ~ LogT + Level
# gamma ~ Time (or constant if Kw/Beta)
formu_rt <- Gas ~ LogT + LogT2 + Level | LogT + Level | Time

# Compare different families in gkwreg
results_retinal <- fit_and_compare_models_3p(
                          formu_rt, retinal_data, method = "nlminb",
                          use_betareg = TRUE)

# Best model
ekw_model_ret <- gkwreg(formu_rt, retinal_data, family="ekw", method="BFGS")
summary(ekw_model_ret)
plot(ekw_model_ret, use_ggplot = TRUE, arrange_plots=TRUE, sub.caption="")
```

Among the five compared models, results in Table 6 show that the Exponentiated Kumaraswamy (ekw) model provides the optimal fit with the highest log-likelihood (132.71) using only 9 parameters, resulting in the best AIC (−247.42) and BIC (−218.63) values. The Beta-Kumaraswamy (bkw) and Generalized Kumaraswamy (gkw) models achieve identical log-likelihood but require more parameters (10 and 11 respectively), resulting in higher information criteria. The Kumaraswamy-Kumaraswamy (kkw) model shows slightly lower log-likelihood (131.02), while the McDonald/Beta Power (mc) model performs substantially worse despite having 9 parameters. With all models converging successfully, the ekw model clearly offers the best balance between parsimony and goodness-

of-fit for the retinal data.

Fitting a Distribution to WeatherTask Data

As a final example focusing on univariate distribution fitting, we use the `WeatherTask` dataset from `gkwreg`, containing $n = 345$ observations of the `agreement` variable. These are subjective probabilities assigned by undergraduate psychology students (with no prior probability theory training) to the statement that "Sunday would be the warmest day of the following week." We aggregate all conditions from the original 2×2 factorial design (`priming` vs. `eliciting` types) to illustrate purely distributional fitting.

```
# Geta the "WeatherTask" data
df_weather <- get_bounded_datasets("WeatherTask")

# Fit all seven distribution families to the 'agreement' data
# We'll use gkwfitall, which is a convenient wrapper for this purpose
fitall_weather <- gkwfitall(df_weather$agreement, method = "BFGS")

# Analyze the comparative results
# summary(fitall_weather) # Displays the comparison table

# Obtain the name of the best family (based on the lowest AIC, for example)
# The table is already sorted by AIC
best_family_code <- fitall_weather$comparison$Family[1]

# Refit the best model to obtain additional details and plots
fit_best_weather <- gkwfit(
  df_weather$agreement, family = best_family_code,
  method = "BFGS", profile = TRUE, plot = TRUE, silent = TRUE)
  # set plot = FALSE here if you want to generate later

# Generate the Goodness-of-Fit (GoF) report for the best model
gof_report <- gkwgof(
  fit_best_weather, theme = ggplot2::theme_classic(),
  plot = TRUE, print_summary = FALSE, verbose = FALSE)
# summary(gof_report) # Displays GoF statistics

# Extract GoF and fitting statistics for all the families
# fitted by gkwfitall
results_weather <- do.call(rbind,
  lapply(fitall_weather$fits, function(f){
      extract_gof_stats(gkwgof(f, plot = FALSE,
                               print_summary = FALSE, verbose = FALSE))
}))
results_weather <- results_weather[order(results_weather$AIC), ]
row.names(results_weather) <- NULL

# Generate the fit plot for the best model and save it as a PDF
# (The plot generated by gkwgof includes a histogram, empirical and
# fitted density curves, P-P, and Q-Q plots)
plot(gkwgof(fit_best_weather),
     title = paste("Fit of the",
     toupper(best_family_code), "Distribution to the WeatherTask Data"))
```

The diagnostic plots in Figure 1 and results in Table 7 for the Exponentiated Kumaraswamy (EKW) model, indicate a generally good fit to the WeatherTask data. The

Table 7: Results of fitting the GKw family distributions to the WeatherTask data (univariate). Models sorted by AIC.

Code	N. Par.	LogLik	AIC	BIC	KS	AD	RMSE	pseudo-R^2
ekw	3	204.45	-402.90	-391.37	0.18	11.19	0.08	0.69
bkw	4	203.82	-399.64	-384.27	0.18	11.31	0.08	0.69
kkw	4	203.80	-399.61	-384.23	0.18	11.32	0.08	0.69
gkw	5	204.35	-398.71	-379.49	0.18	10.96	0.08	0.69
mc	3	195.97	-385.94	-374.41	0.19	13.44	0.09	0.68
beta	2	191.98	-379.96	-372.27	0.20	14.48	0.09	0.67
kw	2	187.03	-370.07	-362.38	0.19	14.71	0.09	0.66

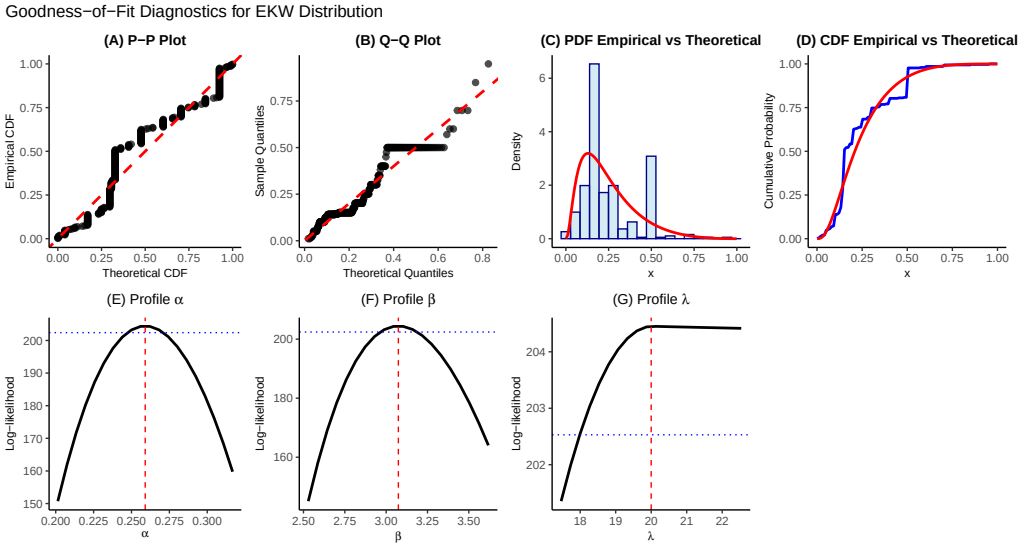

Figure 1: Diagnostic plots for the fit of the Exponentiated Kumaraswamy (EKw) distribution to the WeatherTask data. It includes a histogram with density curves, a P-P plot, and a Q-Q plot.

histogram is closely tracked by the fitted EKW density curve, and both the `P-P` and `Q-Q` plots show points aligning well with the diagonal, reflected in high correlation coefficients (0.96 and 0.96, respectively). Although some systematic deviations appear in the intermediate probabilities (`P-P` plot) and at the distribution's tails (`Q-Q` plot), these are relatively minor given the large sample size ($n = 345$). The Kolmogorov-Smirnov (KS) and Anderson-Darling (AD) tests formally reject the null hypothesis of perfect adherence—commonly observed in large samples, where even small discrepancies can lead to rejection. Nonetheless, the EKW model achieves an AIC of -402.90—the lowest among all candidates—and a pseudo-R^2 of 0.69, reflecting a strong improvement over a null (uniform) model.

From a moment-based perspective, the theoretical mean (0.2435) is almost identical to the sample mean (0.2434), while the variance, skewness, and kurtosis differ only modestly, suggesting that the EKW model captures the main features of the empirical distribution. Prediction metrics (MAE, RMSE, CRPS) also confirm the model's robust performance, with RMSE $\approx$ 0.077. Compared to simpler two-parameter distributions like Beta or Kumaraswamy, the extra exponentiation parameter λ clearly helps accommodate the shape of the subjective probability judgments in this dataset. Although small residual mismatches remain — particularly for moderate-probability values and the upper tail — these results underscore the EKW model as the best overall choice for fitting the `WeatherTask` data among the evaluated GKw-family alternatives.

Conclusion

This article introduced and detailed the `R` package `gkwreg`, a computational and statistical tool developed to facilitate regression modeling of data restricted to the $(0, 1)$ interval using the flexible Generalized Kumaraswamy (GKw) distribution family and its main nested subfamilies. Analysis of proportional, rate, or index data is ubiquitous in numerous scientific fields, and while beta regression is the current standard, its limitations regarding flexibility for capturing complex distributional shapes motivate the search for more general alternatives. The GKw distribution, with its five parameters, offers such generality, encompassing models like Beta, Kumaraswamy, McDonald, and Exponentiated Kumaraswamy as particular cases. However, the inherent complexity of its likelihood function poses a significant computational challenge for estimation, especially in scenarios involving multiparametric regression.

The main contribution of `gkwreg` lies in overcoming these computational challenges through strategic integration with the Template Model Builder (`TMB`) framework. By leveraging automatic differentiation (AD) to compute exact and efficient gradients and Hessians of the log-likelihood function, combined with robust optimization algorithms, the package enables stable and rapid maximum likelihood estimation (MLE), even for models in which multiple GKw parameters are simultaneously related to different sets of covariates via various link functions.

The package provides a user interface that is consistent and familiar in the `R` environment, employing multi-part formulas (via the `Formula` package) for flexible specification of the regression structure for each parameter. In addition to estimation, `gkwreg` offers a complete set of post-estimation tools through standard S3 methods, including `summary()`, `coef()`, `vcov()`, `logLik()`, `AIC()`, `BIC()`, `predict()`, and `residuals()`. Crucially, the implementation of randomized quantile residuals as the default diagnostic method (`residuals(..., type = "quantile")`) and the associated diagnostic plots (`plot()`) provide statistically sound tools for evaluating the suitability of the fitted model.

Empirical analyses presented with four different datasets (food expenditure, gasoline yield, stem cell recovery, and retinal gas decay) consistently showed the practical utility of `gkwreg`. The results repeatedly indicated that subfamilies of GKw, such as Kumaraswamy (Kw), McDonald (Mc), or Exponentiated Kumaraswamy (EKw), often deliver superior fits compared to standard Beta regression, as assessed by criteria like AIC, BIC, and log-likelihood. The GKw family's hierarchical structure, along with the package's model-comparison tools, allows researchers to select models that strike an appropriate balance between flexibility, data fit, and parsimony—avoiding both underfitting and overparameterization. The additional univariate example with the `WeatherTask` data illustrated how the EKw family can effectively capture skewed distributions of subjective probability judgments that simpler two-parameter models struggle to replicate.

In summary, the `gkwreg` package fills an important gap in the `R` statistical software ecosystem by providing a robust, efficient, and user-friendly implementation of regression models based on the Generalized Kumaraswamy family. By combining substantial distributional flexibility with computational rigor and a clear interface, `gkwreg` empowers researchers and practitioners in diverse fields to perform more sophisticated and accurate analyses of responses bounded on the unit interval, leading to deeper insights into the phenomena under investigation.

References

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, *19*(6), 716–723. doi:10.1109/TAC.1974.1100705

Allan, D., Keeney, M., Howson-Jan, K., Popma, J., Weir, K., Bhatia, M., Sutherland, D., et al. (2002). Number of viable CD34+ cells reinfused predicts engraftment in autologous hematopoietic stem cell transplantation. *Bone Marrow Transplantation*, *20*, 967–972.

Atkinson, A. C. (1985). *Plots, transformations, and regression: An introduction to graphical methods of diagnostic regression analysis*. Oxford: Oxford University Press.

Bell, B. M. (2007). CppAD: A package for c++ algorithmic differentiation. *Computational Infrastructure for Operations Research*. Retrieved from http://www.coin-or.org/CppAD

Branscum, A. J., Johnson, W. O., & Thurmond, M. C. (2007). Bayesian beta regression: Applications to household expenditure data and genetic distance between foot-and-mouth disease viruses. *Australian & New Zealand Journal of Statistics*, *49*(3), 287–301. doi:10.1111/j.1467-842X.2007.00481.x

Burnham, K. P., & Anderson, D. R. (2002). *Model selection and multimodel inference: A practical information-theoretic approach* (2nd ed.). Springer.

Carrasco, J. M. F., Ferrari, S. L. P., & Cordeiro, G. M. (2010). A new generalized Kumaraswamy distribution. *arXiv preprint arXiv:1004.0911*.

Cordeiro, G. M., & Castro, M. de. (2011). A new family of generalized distributions. *Journal of Statistical Computation and Simulation*, *81*(7), 883–898. doi:10.1080/00949650903530745

Cribari-Neto, F., & Zeileis, A. (2010). Beta regression in R. *Journal of Statistical Software*, *34*(2), 1–24. doi:10.18637/jss.v034.i02

Dunn, P. K., & Smyth, G. K. (1996). Randomized quantile residuals. *Journal of Computational and Graphical Statistics*, *5*(3), 236–244. doi:10.1080/10618600.1996.10474708

Dutka, J. (1981). The incomplete beta function—a historical profile. *Archive for History of Exact Sciences*, *24*(1), 11–29. doi:10.1007/BF00327713

Eddelbuettel, Dirk, & Balamuta, J. J. (2018). Extending R with C++: A Brief Introduction to Rcpp. *The American Statistician*, *72*(1), 28–36. doi:10.1080/00031305.2017.1375990

Eddelbuettel, D., & Sanderson, C. (2014). RcppArmadillo: Accelerating R with high-performance C++ linear algebra. *Computational Statistics and Data Analysis*, *71*, 1054–1063. doi:10.1016/j.csda.2013.02.005

Eugene, N., Lee, C., & Famoye, F. (2002). Beta-normal distribution and its applications. *Communications in Statistics - Theory and Methods*, *31*(4), 497–512. doi:10.1081/STA-120003130

Ferrari, S. L. P., & Cribari-Neto, F. (2004). Beta regression for modelling rates and proportions. *Journal of Applied Statistics*, *31*(7), 799–815. doi:10.1080/0266476042000214501

Ferreira, J. T. A. S., & Steel, M. F. J. (2006). A constructive representation of univariate skewed distributions. *Journal of the American Statistical Association*, *101*(474), 823–829. doi:10.1198/016214505000001050

Fletcher, S. G., & Ponnambalam, K. (1996). Estimation of reservoir yield and storage distribution using moments analysis. *Journal of Hydrology*, *182*(1-4), 259–275. doi:10.1016/0022-1694(95)02946-X

Gay, D. M. (1990). *Usage summary for selected optimization routines*. Computing Science Technical Report No. 153, AT&T Bell Laboratories.

Griewank, A., & Walther, A. (2008). *Evaluating derivatives: Principles and techniques of algorithmic differentiation* (2nd ed.). SIAM. doi:10.1137/1.9780898717761

Gupta, A. K., & Nadarajah, S. (2004). *Handbook of beta distribution and its applications*. CRC Press.

Johnson, N. L., Kotz, S., & Balakrishnan, N. (1995). Continuous univariate distributions, *2*.

Jones, M. C. (2009). Kumaraswamy's distribution: A beta-type distribution with some tractability advantages. *Statistical Methodology*, *6*(1), 70–81. doi:10.1016/j.stamet.2008.04.001

Kieschnick, R., & McCullough, B. D. (2003). Regression analysis of variates observed on (0, 1): Percentages, proportions and fractions. *Statistical Modelling*, *3*(3), 193–213. doi:10.1191/1471082X03st053oa

Kristensen, K., Nielsen, A., Berg, C. W., Skaug, H., & Bell, B. M. (2016). TMB: Automatic differentiation and Laplace approximation. *Journal of Statistical Software*, *70*(5), 1–21. doi:10.18637/jss.v070.i05

Kumaraswamy, P. (1980). A generalized probability density function for double-bounded random processes. *Journal of Hydrology*, *46*(1-2), 79–88. doi:10.1016/0022-1694(80)90036-0

Lehmann, E. L. (1999). *Elements of large-sample theory*. Springer.

McCullagh, P., & Nelder, J. A. (1989). *Generalized linear models* (2nd ed.). Chapman; Hall.

McDonald, J. B. (1984). Some generalized functions for the size distribution of income. *Econometrica*, *52*(3), 647–663. doi:10.2307/1913469

Meyers, S. M., Ambler, J. S., & Tan, M. (1992). Variation of perfluorpropane disappearance after vitrectomy. *Retina*, *12*(4), 359–363. doi:10.1097/00006982-199212040-00009

Nadarajah, S., & Gupta, A. K. (2004). The beta Fréchet distribution. *Far East Journal of Theoretical Statistics*, *14*(1), 15–24.

Nadarajah, S., & Kotz, S. (2004). The beta Gumbel distribution. *Mathematical Problems in Engineering*, *2004*(4), 323–332. doi:10.1155/S1024123X04403068

Nadarajah, S., & Kotz, S. (2006). The beta exponential distribution. *Reliability Engineering & System Safety*, *91*(6), 689–697. doi:10.1016/j.ress.2005.05.008

Osgood-Zimmerman, A., & Wakefield, J. (2021). Gaussian process modelling for geostatistical count data using INLA and TMB. *Spatial Statistics*, *45*, 100544. doi:10.1016/j.spasta.2021.100544

Pawitan, Y. (2013). *In all likelihood: Statistical modelling and inference using likelihood* (2nd ed.). Oxford University Press.

Prater, N. H. (1956). Estimate gasoline yields from crudes (predict yields from heavy high sulfur crudes). *Petroleum Refiner*, *35*(5), 236–238.

R Core Team. (2025). *R: A language and environment for statistical computing*. Vienna, Austria: R Foundation for Statistical Computing. Retrieved from https://www.R-project.org/

Schwarz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics*, *6*(2), 461–464. doi:10.1214/aos/1176344136

Skaug, H. J., & Fournier, D. A. (2016). Automatic approximation of the marginal likelihood in non-Gaussian hierarchical models. *Computational Statistics & Data Analysis*, *51*(2), 699–709. doi:10.1016/j.csda.2006.03.005

Smithson, M., & Verkuilen, J. (2006). A better lemon squeezer? Maximum-likelihood regression with beta-distributed dependent variables. *Psychological Methods*, *11*(1), 54–71. doi:10.1037/1082-989X.11.1.54

Sundar, V., & Subbiah, K. (1989). Application of double bounded probability density function for analysis of ocean waves. *Ocean Engineering*, *16*(2), 193–200. doi:10.1016/0029-8018(89)90029-8

Venzon, D. J., & Moolgavkar, S. H. (1988). A method for computing profile-likelihood-based confidence intervals. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, *37*(1), 87–94. doi:10.2307/2347496

Wahed, A. S., & Ali, M. M. (2006). The skewed-logistic distribution. *Journal of Statistical Research*, *40*(2), 71–80.

Yang, H., Acker, J., Cabuhat, M., Letcher, B., Larratt, L., & McGann, L. (2005). Association of post-thaw viable CD34+ cells and CFU-GM with time to hematopoietic engraftment. *Bone Marrow Transplantation*, *35*, 881–887.

Zeileis, A., & Croissant, Y. (2010). Extended model formulas in R: Multiple parts and multiple responses. *Journal of Statistical Software*, *34*(1), 1–13. doi:10.18637/jss.v034.i01

Zhang, P., Qiu, Z., & Shi, C. (2016). simplexreg: An R package for regression analysis of proportional data using the simplex distribution. *Journal of Statistical Software*, *71*(11), 1–21. doi:10.18637/jss.v071.i11