Universidad de Costa Rica
Facultad de Ingeniería
Escuela de Ingeniería Eléctrica

IE0499 – Proyecto Eléctrico

Generación de funciones de densidad de probabilidad ajustadas a mediciones de variables eléctricas: caso Rayleigh y Gamma

por

Rodrigo López Soto

Ciudad Universitaria Rodrigo Facio

Julio de 2021

Generación de funciones de densidad de probabilidad ajustadas a mediciones de variables eléctricas: caso Rayleigh y Gamma

por

Rodrigo López Soto

B23776

IE0499 – Proyecto Eléctrico

Aprobado por

Ing. Jorge Arturo Romero Chacón, Ph. D.
Profesor guía

Ing. Gustavo Valverde Mora, Ph.D.
Profesor lector

Ing. Marvin Coto Jiménez, Ph.D.
Profesor lector

Julio de 2021

Resumen

Generación de funciones de densidad de probabilidad ajustadas a mediciones de variables eléctricas: caso Rayleigh y Gamma

por

Rodrigo López Soto

Universidad de Costa Rica
Escuela de Ingeniería Eléctrica
Profesor guía: Ing. Jorge Arturo Romero Chacón, Ph. D.
Julio de 2021

Las variables aleatorias gaussianas y las mezclas de estas son ampliamente utilizadas para modelar fenómenos naturales. Sin embargo, existen ocasiones en las que el ajuste obtenido con otras funciones de distribución de probabilidad es mejor (principalmente por efecto de asimetrías en el conjunto de datos) .

En este proyecto se desarrolló un algoritmo capaz de realizar un ajuste de parámetros óptimos de mezclas de cualquier número finito de funciones de distribución de probabilidad de Rayleigh o Gamma. Para ello se utilizó el algoritmo de esperanza-maximización (EM), y se determinaron datos ocultos que, de conocerse, permitirían modelar adecuadamente el conjunto de datos reales.

El algoritmo se desarrolló en Matlab, para lo cual se creó una clase correspondiente a mezclas de funciones de distribución de probabilidad de Rayleigh y otra clase para funciones de distribución de probabilidad Gamma. Cada clase es capaz de realizar el ajuste óptimo de la mezcla de funciones de distribución de probabilidad y generar los datos aleatorios según la función de distribución de probabilidad escogida. Así mismo, se utilizó Python para visualización de resultados y para preprocesamiento de los datos. El algoritmo diseñado se probó utilizando un conjunto de datos de mediciones reales de acometidas eléctricas correspondiente al aporte porcentual en tensión rms de las armónicas de la 2 a la 25 de la frecuencia fundamental. Se determinó que el ajuste por medio de funciones de distribución de probabilidad gamma fue mejor que el ajuste por medio de funciones de distribución de probabilidad de Rayleigh, debido a que modeló mejor conjuntos de datos positivos que no inician exactamente en cero.

***Palabras claves**: Esperanza-maximización, mezcla de funciones de distribución de probabilidad, función de distribución de probabilidad Gamma, función de distribución de probabilidad de Rayleigh, armónicos.*

Acerca de IE0499 – Proyecto Eléctrico

El Proyecto Eléctrico es un curso semestral bajo la modalidad de trabajo individual supervisado, con el propósito de aplicar estrategias de diseño y análisis a un problema de temática abierta de la ingeniería eléctrica. Es un requisito de graduación para el grado de Bachiller en Ingeniería Eléctrica de la Universidad de Costa Rica.

Abstract

Generación de funciones de densidad de probabilidad ajustadas a mediciones de variables eléctricas: caso Rayleigh y Gamma

Original in Spanish. Translated as: "Generation of probability density functions adjusted to measurements of electrical variables: Rayleigh and Gamma cases"

by

Rodrigo López Soto

University of Costa Rica
Department of Electrical Engineering
Tutor: Ing. Jorge Arturo Romero Chacón, Ph. D.
July of 2021

In this project, an algorithm capable of performing an optimal parameter adjustment of mixtures of any finite number of Rayleigh or Gamma probability distribution functions was developed. For this, the hope-maximization (EM) algorithm was used.

The algorithm was developed in Matlab, for which a class corresponding to mixtures of Rayleigh probability distribution functions and another class for Gamma probability distribution functions was created. Each class is capable of optimizing the mixture of probability distribution functions and generating the random data according to the chosen probability distribution function. The designed algorithm was tested using a data set of real measurements of electrical connections corresponding to the percentage contribution in rms voltage of harmonics from 2 to 25 of the fundamental frequency. It was determined that the fit by means of gamma probability distribution functions was better than the fit by means of Rayleigh probability distribution functions, because it better modeled positive data sets that do not start exactly at zero.

***Keywords**: Hope-maximization, mixture of probability distribution functions, Gamma probability distribution function, Rayleigh probability distribution function, harmonics.*

About IE0499 – Proyecto Eléctrico ("Electrical Project")

The "Electrical Project" (or "capstone project") is a course of supervised individual work of one semester, with the purpose of applying design and analysis strategies to a problem in an open topic in electrical engineering. It is a requisite of graduation for the Bachelor of Science in Electrical Engineering, granted by the University of Costa Rica.

Dedicado a mis padres.

Agradecimientos

Quisiera realizar el agradecimiento al Ing. Jorge Arturo Romero Chacón, Ph.D., por haber propuesto la idea que dio origen a este proyecto y por el apoyo dado durante el desarrollo del mismo, así como por la oportunidad de poder trabajar y aprender en este proceso de tratamiento y análisis de datos. Así mismo, al Ing. Gustavo Valverde Mora, Ph.D., por haber suministrado los datos del laboratorio UVECASE de la Universidad de Costa Rica con los cuales se evaluó el algoritmo diseñado y por haber colaborado como lector del proyecto. Además, agradezco al Ing. Marvin Coto Jiménez, Ph.D., por haber amablemente colaborado como lector de este proyecto. Agradezco además a todas y todos los profesores que compartieron su conocimiento en los cursos de la carrera de Ingeniería Eléctrica. Finalmente, agradezco en especial a mis padres, con cuyo apoyo constante e incondicional fue posible alcanzar este resultado.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

En el presente documento se reportarán los resultados correspondientes al Proyecto Eléctrico, el cual es un curso de integración de conceptos y finalización de la carrera de Ingeniería Eléctrica en la Universidad de Costa Rica, en el cual se realizará un tratamiento estadístico y se generarán funciones de densidad de Rayleigh y Gamma para mediciones de variables eléctricas.

1.1. Justificación

El problema que se pretende resolver es determinar si las distribuciones de Rayleigh y Gamma constituyen buenas aproximaciones al comportamiento de las mediciones de variables eléctricas.

Los modelos probabilísticos funcionan como herramienta de estudio para diversa cantidad de fenómenos. En las mediciones de variables eléctricas existe diversa cantidad de factores que influyen en las mediciones, por lo que el uso de modelos estadísticos permite realizar mejores estimaciones a la hora de realizar las mediciones. En todo proceso de análisis de datos se parte de una fuente de datos, que en este caso se trata de variables eléctricas, en la mayoría de casos se requiere una limpieza de datos y una selección de variables relevantes, en ocasiones podría ser realizar análisis exploratorios de datos para finalmente llegar a un modelo basado en la evidencia científica que se tenga y el criterio de profesionales en materia de modelación matemática.

En este caso se propone la generación de funciones de densidad para los casos de Rayleigh y Gamma, puesto que en ocasiones una distribución normal no es una buena aproximación para el comportamiento real del sistema.

Un mejor entendimiento de los modelos matemáticos asociados a las mediciones de variables eléctricas es de vital importancia en ingeniería eléctrica, y es una aplicación que integra diversa cantidad de conocimientos de la carrera, principalmente en los temas de modelos probabilísticos de señales y sistemas, programación, estructuras abstractas de datos, circuitos eléctricos, modelado matemático, tratamiento de datos, entre otros.

Así mismo, implica un proceso de verificación de que el modelo propuesto se ajusta al comportamiento real y permite comprender si el proceso computacional genera una herramienta útil en ingeniería.

1.2. Objetivos

1.2.1. Objetivo general:

Diseñar un algoritmo para la generación de funciones de densidad de probabilidad no gaussianas ajustadas a mediciones de variables eléctricas, con base en las funciones de densidad Rayleigh y Gamma.

1.2.2. Objetivos específicos:

1. Determinar las características principales en las mediciones de las variables eléctricas para las cuales se implementará el algoritmo.

2. Obtener el algoritmo utilizando la combinación de funciones de densidad de probabilidad no gaussianas, específicamente las funciones de densidad Rayleigh y Gamma.

3. Validar el algoritmo mediante pruebas basadas en los datos reales obtenidos de la Unidad de Verificación de la Calidad del Suministro Eléctrico de la Universidad de Costa Rica y comparar los resultados obtenidos de la validación del algoritmo con los resultados de otras familias de funciones de densidad probabilística utilizadas en estudios previos.

1.3. Alcances

El tratamiento de datos y la modelación matemática son procesos que requieren tomar en cuanta diversa cantidad de factores e integra el conocimiento de profesionales de diferentes áreas. En este caso por ejemplo la escogencia de funciones de distribución de Rayleigh y Gamma se debe a que existen ocasiones en las que la muestra de datos no cumple con el teorema del límite central, y por lo tanto una distribución normal no es una aproximación adecuada. En el presente proyecto no se pretende encontrar el mejor modelo matemático para las mediciones en cuestión, se pretende únicamente generar las funciones de distribución de Rayleigh y Gamma que mejor se ajusten a las mediciones y determinar desde un punto de vista científico si los modelos generados son adecuados. Con esto se puede llegar a tener un mejor entendimiento de la estadística asociada a fenómenos aleatorios que afectan las mediciones eléctricas. Así mismo, la herramienta de un lenguaje de alto nivel como Python permite realizar cálculos estadísticos de manera rápida haciendo uso de sus herramientas de estadística, visualización de datos, multiprocesamiento, entre otros.

1.4. Metodología

1. Revisión bibliográfica sobre el algoritmo de esperanza-maximización y acerca de las funciones de densidad de probabilidad no gaussianas Rayleigh y Gamma.

2. Estudio del entorno de Matlab que se usará para la implementación del algoritmo.

3. Diseño del algoritmo para utilizar las funciones de densidad de probabilidad no gaussianas.

4. Implementación del algoritmo diseñado utilizando Matlab.
5. Validación del algoritmo diseñado con datos reales obtenidos de la UVECASE.
6. Comparación de los resultados obtenidos entre las funciones estudiadas y otra familia de funciones presentadas en trabajos previos.

1.5. Cronograma

Tabla 1.1: Cronograma del Proyecto Eléctrico

Semana	Actividades y entregas	Fecha de referencia
2	Entrega de avance preliminar: (descripción general, objetivo general y específicos)	16 de abril
4	Revisión de la propuesta del proyecto	25 de abril
4	Entrega de anteproyecto 1: Introducción, justificación, objetivos, alcances, metodología y cronograma	30 de abril
7	Revisión bibliográfica para el marco teórico sobre el algoritmo de esperanza-maximización y funciones de densidad probabilística no gaussianas (Rayleigh y Gamma)	Mayo
7	Estudio de las librerías de estadística y graficación disponibles en lenguaje Python	Mayo
9	Entrega de avance 1 (capítulo II y secciones adicionales): revisión de los objetivos y alcances, metodología y cronograma, marco teórico.	7 de junio
10	Diseño del algoritmo en Matlab	10 de junio
11	Generación y visuaización de resultados de implementación de algoritmo	17 de junio
12	Entrega de avance 2: revisión de los objetivos, alcances, metodología y cronograma, revisión de la investigación bibliográfica y el diseño, resultados de la implementación y su evaluación, conclusiones y recomendaciones preliminares.	30 de junio
15	Entrega de borrador final completo	17 de julio
18	Presentación del proyecto en línea por medio de la plataforma Zoom	4, 5 y 6 de agosto
20	Entrega de informe final corregido	20 de agosto

Capítulo 2

Marco Teórico

Este capítulo definen las bases teóricas que hay detrás del algoritmo matemático que se utilizará: el de esperanza-maximización. Se definen las propiedades y los teoremas necesarios para entender el algoritmo y su convergencia.

2.1. Definiciones y fundamentos

2.1.1. Verosimilitud

Dado un conjunto de mediciones independientes $x = (x_1, x_2, ..., x_n)^T$ la probabilidad de que se reciba una medición x_i en específico viene dada por [1]:

$$f(x_i|\Theta) \tag{2.1}$$

Donde el parámetro Θ es asignado por la función de densidad probabilísitica (por ejemplo, para el caso de una distribución normal esta queda definida por los parámetros μ y σ, que corresponden a la media y la desviación estándar). Por otra parte, tomando en cuenta que el vector x es de un conjunto de mediciones independientes, la probabilidad de x viene dada por [2]:

$$f(x|\Theta) = \prod_{i=1}^{n} f(x_i|\Theta) \tag{2.2}$$

Esta es la probabilidad de obtener un conjunto de mediciones si se conocen los parámetros de la función de densidad probabilística. En ocasiones el problema de modelado consiste en calcular la probabilidad de que una función de densidad sea la que produjo el conjunto de mediciones independientes x, lo cual viene relacionado con el concepto de verosimilitud [1]:

Definición 2.1. *Sea $x = (x_1, x_2, ..., x_n)^T$ un conjunto de mediciones independientes, y f una función de densidad probabilística con parámetros dados por Θ, se define verosimilitud como:*

$$L(\Theta|x) = f(x|\Theta) = \prod_{i=1}^{n} f(x_i|\Theta) \tag{2.3}$$

Esta función de verosimilitud se puede ver como la probabilidad de que una función de densidad probabilística con parámetros dados por Θ genere el conjunto de mediciones x. Así mismo, en ocasiones se prefiere trabajar en escala logarítmica por facilidad en los cálculos, dado que se transforma la productoria de probabilidades en una sumatoria de probabilidades, lo cual se conoce como **log-verisimilitud**, de la siguiente manera [3]:

$$\log L(\Theta|x) = \log f(x|\Theta) = \sum_{i=1}^{n} \log f(x_i|\Theta) \tag{2.4}$$

2.1.2. Convexidad

La convexidad de una función continua viene relacionada con el signo de su segunda derivada, de la siguiente manera:

Definición 2.2. *Sea f una función real con valores reales definidos en el intervalo I =[a,b], se dice que f es convexa si* $\forall x_1, x_2 \in I, \lambda \in [0,1]$:

$$f(\lambda x_1 + (1-\lambda)x_2)) \leq \lambda f(x_1) + (1-\lambda) f(x_2)$$

Se dice que f es **estrictamente convexa** si la el la inecuación es con signo menor estricto [1]. Intuitivamente se tiene que la gráfica de la función siempre está por debajo (estrictamente convexa) o nunca sobrepasa (convexa) a la recta secante definida entre los puntos $(x_1, f(x_1))$ y $(x_2, f(x_2))$. Observe por ejemplo la Figura 2.1, en la cual se observa una función estrictamente convexa y una recta secante a la misma, la cual se observa siempre por encima de la función en el intervalo [4].

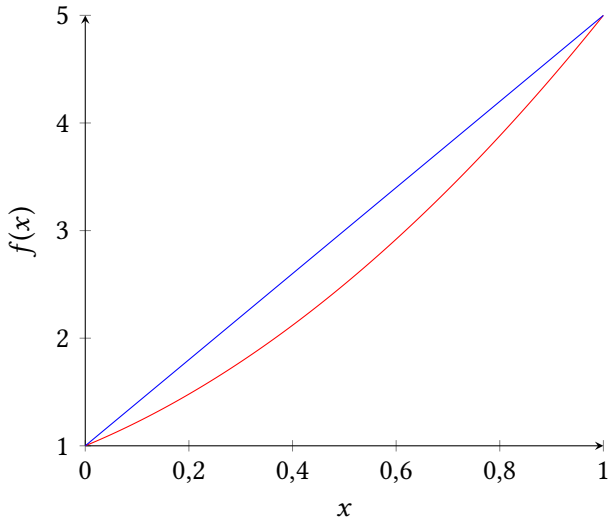

Figura 2.1: Representación visual de una función convexa y una recta secante a la misma. Autoría propia

Definición 2.3. *Una función f es cóncava (estrictamente cóncava) si -f es convexa (estrictamente convexa) [1].*

Teorema 1. *Si f es una función doblemente diferenciable en el intervalo [a,b] y* $f'' > 0\ \forall x \in [a,b]$ *entonces f es convexa en el intervalo [a,b] [1]*

Demostración. Para x ≤ y ∈ [a,b] y $\lambda \in [0,1]$, sea z = $\lambda y + (1-\lambda)x$. <por definición f es convexa si $f(\lambda y + (1-\lambda)x)) \leq \lambda f(y) + (1-\lambda)f(x)$. Tomando en cuenta que $z = \lambda y + (1-\lambda)x$ y que $f(z) = \lambda f(z) + (1-\lambda)f(z)$, se tiene por lo tanto que $f(z) = \lambda f(z) + (1-\lambda)f(z) \leq \lambda f(y) + (1-\lambda)f(x)$. Reordenando términos se tiene que si f es convexa:

$$\lambda[f(y) - f(x)] \geq (1-\lambda)[f(z) - f(x)] \tag{2.5}$$

Por teorema del valor medio se tiene que $\exists s, x \leq s \leq z$ tal que:

$$f(z) - f(x) = f'(s)(z-x) \tag{2.6}$$

De manera análoga para $f(y) - f(z)$ por teorema del valor medio $\exists t, z \leq t \leq y$ tal que:

$$f(y) - f(z) = f'(t)(y-z) \tag{2.7}$$

Por lo tanto tomando en cuenta que $x \leq s \leq z \leq t \leq y$. Asumiento que $f''(x) \geq 0$ en el intervalo $[a,b]$ se tiene que:

$$f'(s) \leq f'(t); s \leq t \tag{2.8}$$

Además, se puede reescribir z de la forma:

$$(1-\lambda)(z-x) = \lambda(y-z) \tag{2.9}$$

De manera que combinando las ecuaciones 2.6, 2.7, 2.8 y 2.9 se tiene que:

$$(1-\lambda)[f(z) - f(x)] = (1-\lambda)f'(s)(z-x) \leq f'(t)(z-x) = \lambda f'(t)(y-z) = \lambda[f(y) - f(x)] \tag{2.10}$$

Que es justamente la ecuación 2.5, por lo que queda demostrado. □

Teorema 2. *Una función* $\phi : \mathbb{R} \longrightarrow \mathbb{R}$ *es convexa en un intervalo I si y solo si para todo xx* $\in I$ *la función [4]:*

$$f_x(y) = \frac{\phi(y) - \phi(x)}{(y-x)} \tag{2.11}$$

es creciente.

Demostración. Sea $x, y, z \in I$ tales que $x < y < z$. Luego, existe $t \in [0,1]$ tal que $z = (1-t)x + ty$. De la definición de convexidad se tiene que [4]:

$$\phi(z) = \phi((1-t)x + ty) \leq (1-t)\phi(x) + t\phi(y) \tag{2.12}$$

Reacomodando se obtiene que:

$$f_x(z) = \frac{\phi(z) - \phi(x)}{z - x} \leq \frac{(1-t)\phi(x) + t\phi(y) - \phi(x)}{(1-t)x + ty - x} = \frac{phi(y) - \phi(x)}{y - x} = f_x(y) \tag{2.13}$$

De lo cual se observa que f_x es creciente en $w/w \in I, w > x$. Por esto, para ver que f es creciente en todo el intervalo queda pendiente verificarlo para $w/w \in I, w < x$

De manera análoga si se tienen $z, y, x \in I$ tales que $z < y < x$, existen $t, s \in [0,1]$ tales que $(1-t)z + ty = x = sz + (1-s)y$. Realizando el mismo procedimiento se obtiene que:

$$\phi(x) =\leq (1-t)\phi(z) + t\phi(y) \tag{2.14}$$

$$\phi(x) \geq s\phi(z) + (1-s)\phi(y) \tag{2.15}$$

$$\frac{\phi(x) - \phi(z)}{x - z} \leq \frac{t(\phi(y) - \phi(z))}{t(y - z)} \tag{2.16}$$

$$\frac{\phi(x) - \phi(y)}{x - z} \geq \frac{t(\phi(z) - \phi(y))}{t(z - y)} \tag{2.17}$$

Con lo cual es claro que:

$$f_x(z) \leq f_x(y) \tag{2.18}$$

De lo cual se observa que f_x es creciente en $w/w \in I, w < x$, que era lo que se buscaba. □

Teorema 3. *Una función $\phi : \mathbb{R} \longrightarrow \mathbb{R}$ es convexa en un intervalo I si y solo si para todo $x \in I$ existen $a, b \in \mathbb{R}$ tales que [4]:*

$$\begin{aligned} ay + b &\leq \phi(y) \\ ax + b &= \phi(x) \end{aligned} \tag{2.19}$$

es creciente.

Demostración. Sea $x \in I$ tales que $x < y < z$. Luego, existe $t \in [0,1]$ tal que $z = (1-t)x + ty$. De la ecuación 2.18

$$f_x(z) = \frac{\phi(z) - \phi(x)}{z - x} \leq \frac{\phi(y) - \phi(x)}{y - x} = f_x(y) \tag{2.20}$$

Por esta razón los conjuntos $A = f(z) : z \in I, z < x$ y $B = f(y) : y \in I, y > x$ tienen cotas superior e inferior, respectivamente. Por esta razón existe algún a tal que:

$$\begin{aligned} &\sup_{z \in A} f_x(z) \leq a \leq \text{ínf } y \in B f_x(y) \\ \Longleftrightarrow\ & frac\phi(z) - \phi(x) z - x \leq a \leq \frac{\phi(y) - \phi(x)}{y - x} \forall z, y \in A \times B \\ \Longleftrightarrow\ & a(w - x) \leq \phi(w) - \phi(x), \forall w \in I \\ \Longleftrightarrow\ & aw + \phi(x) - ax \leq \phi(w), \forall w \in I \end{aligned} \tag{2.21}$$

Ahora, tome $b = \phi(x) - ax$. Es claro que con esta escogencia se cumple con lo requerido, por lo que queda demostrado la existencia de estos valores. □

2.1.3. Desigualdad de Jensen

El concepto de convexidad puede extenderse a n puntos, lo cual da como resultado el concepto conocido como desigualdad de Jensen [4].

Teorema 4 (Desigulad de Jensen)**.** *Sea f una función convexa definida en el intervalo I. Si* $x_1, x_2, ..., x_n$ *y* $\lambda_1, \lambda_2, ..., \lambda_n \geq 0$ *tales que:*

$$\sum_{i=1}^{n} \lambda_i = 1 \tag{2.22}$$

Con esto, se tiene que:

$$f(\sum_{i=1}^{n} \lambda_i x_i) \leq \sum_{i=1}^{n} \lambda_i f(x_i) \tag{2.23}$$

Demostración. Para n = 1 el resultado es trivial. El caso de n = 2 corresponde a la definición de convexidad. De esta manera para generalizar a n puntos se procede a demostrar por inducción el concepto [1]. Asumiendo que para n se cumple el resultado, para $n+1$ se tiene que:

$$\begin{aligned}
f(\sum_{i=1}^{n+1} \lambda_i x_i) &= f(\lambda_{n+1} x_{n+1} + \sum_{i=1}^{n} \lambda_i x_i) \\
& f(\lambda_{n+1} x_{n+1} + (1-\lambda_{n+1}) \frac{1}{1-\lambda_{n+1}} \sum_{i=1}^{n} \lambda_i x_i) \\
&\leq \lambda_{n+1} f(x_{n+1}) + (1-\lambda_{n+1}) f(\frac{1}{1-\lambda_{n+1}} \sum_{i=1}^{n} \lambda_i x_i)) \\
&= \lambda_{n+1} f(x_{n+1}) + (1-\lambda_{n+1}) f(\sum_{i=1}^{n} \frac{\lambda_i}{1-\lambda_{n+1}} x_i)) \\
&\leq \lambda_{n+1} f(x_{n+1}) + (1-\lambda_{n+1}) \sum_{i=1}^{n} \frac{\lambda_i}{1-\lambda_{n+1}} f(x_i) \\
&= \lambda_{n+1} f(x_{n+1}) + \sum_{i=1}^{n} \lambda_i f(x_i) \\
&= \sum_{i=1}^{n+1} \lambda_i f(x_i)
\end{aligned} \tag{2.24}$$

□

Por ejemplo, la función $\ln(x)$ es cóncava, de manera que se puede aplicar la desigualdad de Jensen para obtener una cota inferior para el logaritmo de una suma (lo cual se utilizará para la derivación del algoritmo de esperanza-maximización [4], de la siguiente manera:

$$\ln(\sum_{i=1}^{n} \lambda_i x_i) \geq \sum_{i=1}^{n} \lambda_i \ln(x_i) \tag{2.25}$$

Este resultado que se demostró para variable discreta también se cumple en variable continua, se la siguiente manera:

Teorema 5 (Desigulad de Jensen). *Sea $\phi : \mathbb{R} \longrightarrow \mathbb{R}$ una función convexa y $f, g : \mathbb{R} \longrightarrow \mathbb{R}$ dos funciones de densidad tales que se cumple que g es una función de densidad de probabilidad de una variable aleatoria X . Por lo tanto, se cumple que:*

$$\phi(E[f(X)]) \leq E[\phi(f(X))] \tag{2.26}$$

O lo que es equivalente:

$$\phi(\int f(x)g(x)dx \leq \int \phi(f(x)g(x))dx \tag{2.27}$$

Donde la igualdad se da si y solo si ϕ f es lineal o f es constante.

Demostración. Sea $w = E[f(X)]$. Utilizando el teorema 2.19 se sabe que existen $a, b \in \mathbb{R}$ tales que $ay + b \leq \phi(y) \forall y \in \mathbb{R}$, tales que $aw + b = \phi(w)$. Por ello, se tiene que:

$$\phi(E[f(X)] = aE[f(X)] + b = E[af(x) + b] \leq E[\phi(f(X))] \tag{2.28}$$

Que era lo que se quería. Para verifivcar el caso de la igualdad observe que se cumple que:

$$E[af(x) + b] = E[\phi(f(X))] \tag{2.29}$$

Con esto se tiene que:

$$ay + b = \phi(y) \tag{2.30}$$

Con lo cual se obtiene que esto tiene valides si y solo si ϕ es una función lineal o f es constante. □

2.1.4. Desigualdad de Gibbs

Para algunos estudios es importante poder comparar dos distribuciones de probabilidad entre sí. Uno de los métodos utilizados para este fin es el de la entropía relativa, el cual se describe a continuación [5]:

Definición 2.4. *Entropía relativa: Dadas dos funciones de distribución de probabilidad $P = p_1, p_2, ..., p_n$ y $Q = q_1, q_2, ..., q_n$, se define entropía relativa, o divergencia de Kullback-Leibler de P y Q como [4]:*

$$D_{KL}(P||Q) = \sum_{i=1}^{n} p_i \log(\frac{p_i}{q_i}) \tag{2.31}$$

Un resultado de esta métrica es la desigualdad de Gibbs [6], la cual afirma que la entropía relativa entre dos distribuciones de probabilidad es mayor o igual a cero, siendo igual solamente cuando las funciones de distribución de probabilidad son iguales. Esto es:

$$D_{KL}(P||Q) \geq 0 \tag{2.32}$$

Lo cual es equivalente a:

$$-\sum_{i=1}^{n} p_i \log(p_i) \leq -\sum_{i=1}^{n} p_i \log(q_i) \tag{2.33}$$

La definición formal y demostración de este resultado se muestra a continuación:

Teorema 6 (Desigualdad de Gibbs)**.** *Si $p(x)$ y $q(x)$ son dos funciones de densidad de probabilidad, entonces se cumple la desigualdad:*

$$-\int p(x)\log(p(x))dx \leq -\int p(x)\log(q(x))dx \tag{2.34}$$

Con igualdad si y solo si $p(x) = q(x)$

Demostración. Utilizando el hecho de que el logaritmo es una función cóncava, la desigualdad de Jensen implica que:

$$\int p(x)log(\frac{q(x)}{p(x)})dx \leq \log(\int p(x)\frac{q(x)}{p(x)}) \tag{2.35}$$

Con igualdad si y solo si $\exists\lambda \in \mathbb{R}/\frac{p(x)}{q(x)} = \lambda$ Eliminando $p(x)$ del lado derecho de la ecuación se obtiene que:

$$\int p(x)log(\frac{q(x)}{p(x)})dx \leq \log(\int q(x)) = \log(1) = 0 \tag{2.36}$$

En el último paso se utilizó el hecho de que la función $q(x)$ es una función de densidad de probabilidad que está por lo tanto normalizada a uno. De este modo, se obtiene que:

$$\int p(x)log(\frac{q(x)}{p(x)})dx \leq 0 \tag{2.37}$$

Lo cual es equivalente a 2.33, que era lo que se buscaba. Finalmente, la igualdad se propuso que se cumple si y solo si $\exists\ \lambda \in \mathbb{R}/\frac{p(x)}{q(x)} = \lambda$, tomando en cuenta esto se obtiene que si se cumple esta condición se tiene que:

$$\int p(x)log(\frac{q(x)}{p(x)})dx = \int p(x)log(\lambda) = 0 \tag{2.38}$$

Lo cual se cumple solamente si $\lambda = 1$. Con esto se verifica el resultado de que la igualdad en la ecuación 2.33 se cumple solamente si las distribuciones son iguales. □

2.2. Algoritmo de esperanza-maximización (EM)

El algoritmo de esperanza-variación es una herramienta estadística útil que permite ajustar mezclas de funciones de densidad de probabilidad a una serie de mediciones reales (obtenidas experimentalmente de un sistema real mediante procedimientos adecuados de medición), lo cual se realiza utilizando un procedimiento científico que se describe por primera vez en 1977 [6]. El algoritmo de EM permite

maximizar log-verosimilitud de **una muestra** de datos incompleta. Para esto, se emplea una serie de pasos repetidamente donde se evalúa el valor esperado y se maximiza este valor, tal como se muestra en la Figura 2.2 [5].

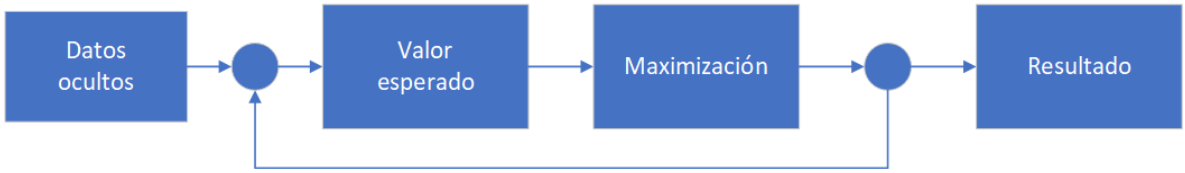

Figura 2.2: Algoritmo esperanza-maximización. Autoría propia.

El algoritmo EM consiste en un procedimiento científico que consiste en maximizar la verosimilitud de los parámetros que definen la función de densidad de probabilidad. Con esto, al maximizar la verosimilitud, esto implica que estos son los parámetros que mejor modelan la muestra de datos disponibles. Por otra parte, observe de la Figura 2.2 que la maximización no ocurre directamente sobre la muestra de datos, si no sobre su esperanza. El hecho de que una maximización del valor esperado implique una maximización de verosimilitud no es trivial, en esta sección se demostrará esto y se estudiará los criterios de convergencia del algoritmo [2].

El algoritmo de esperanza-maximización, como se muestra en la Figura 2.2, toma como base un conjunto de datos ocultos. Esto es: para una variable aleatoria Y se determina un conjunto de datos ocultos y los cuales se utilizan para maximizar la función de verosimilitud. Estos valores ocultos son en muchas ocasiones una construcción matemática que se puede ver como el conjunto de toda la información que no fue observada o que es imposible hacerlo, pero que si se conociera con detalle se podría determinar el valor de los parámetros de mejor ajuste [3].

Ahora, se define el **vector de datos completos** como la unión de los datos observados x y los datos ocultos y de la siguiente manera:

$$z^T = (x^T, y^T) \tag{2.39}$$

De manera tal que se estudiará log-verosimilitud de los **datos completos**, y no de los datos observados, de la siguiente manera [4]:

$$\log L_C(\Theta|z) = \log f(z|\Theta) \tag{2.40}$$

Tomando el **vector de datos completos** se procede a encontrar los parámetros que maximicen log-verosimilitud. Para ello, el algoritmo esperanza-maximización procede a aplicar repetidamente los pasos de calcular esperanza y maximización que se muestran en la Figura 2.2. Para ello, primeramente se debe obtener la esperanza de la función log-verosimilitud utilizando los parámetros actuales Θ^t y rl vector de datos observados x [4]:

$$Q(\Theta|\Theta^t) = E_y[\log L_C(\Theta|z)|x, \Theta^t] \tag{2.41}$$

Posterior a este paso viene el de maximización en el cual se busca encontrar nuevos parámetros de ajuste Θ^{t+1} que maximicen $Q(\Theta|\Theta^t)$. Es decir, se trata de un pronlema de optimización que busca resolver la ecuación [4]:

$$\Theta^{t+1} = argmax(Q(\Theta|\Theta^t)) \tag{2.42}$$

Cabe destacar que el algoritmo esperanza-maximización (EM) no maximiza directamente log-verosimilitud, si no maximiza su esperanza. Sin embargo, en el siguiente teorema se demuestra que un aumento en el valor de la esperanza implica un incremento de log-verosimilitud [1].

Teorema 7. *Sea x una muestra de datos observados, y el conjunto de datos ocultos y $z^T = (x^T, y^T)$ el conjunto de datos completos. Sea f una función de densidad de probabilidad con parámetros Θ y la función $Q(\Theta|\Theta^t) = E_y[\log L_C(\Theta|z)|x, \Theta^t]$, entonces, para cualqueier conjunto de parámetros Φ se cumple la desigualdad [4]:*

$$Q(\Phi,\Theta) - Q(\Theta,\Theta) \leq \log f(x|\Phi) - \log f(x|\Theta) \tag{2.43}$$

Demostración. De la definición de probabilidad condicional se tiene que:

$$\begin{aligned} f(y|x,\Phi) &= \frac{f(x,y|\Phi)}{f(x|\Phi)} \\ \Longrightarrow \log f(x|\Phi) &= \log f(x,y|\Phi) - \log f(y|x,\Phi) \end{aligned} \tag{2.44}$$

Multiplicando en ambos lados por $f(x,y|\Theta)$ e integrando sobre la variable aleatoria Y se obtiene:

$$\begin{aligned} \int_Y \log f(x|\Phi)\log f(y|x,\Theta)dy &= \int_Y \log f(x,y|\Phi)\log f(y|x,\Theta)dy - \int_Y \log f(y|x,\Phi)\log f(x|y,\Theta)dy \\ \Longleftrightarrow E_Y[\log f(x|\Phi)|x,\Theta] &= E_Y[\log(f(x,y|\Phi))|x,\Theta] - \int_Y \log f(y|x,\Phi)\log f(y,x|\Theta)dy \\ \Longleftrightarrow \log f(x|\Phi) &= E_Y[\log f(z|\Phi)|x,\Theta] - \int_Y \log f(y|x,\Phi)\log f(y|x,\Theta)dy \\ \Longleftrightarrow \log f(x|\Phi) &= E_Y[\log L_C(\Phi|z)|x,\Theta] - \int_Y \log f(y|x,\Phi)\log f(y|x,\Theta)dy \end{aligned} \tag{2.45}$$

El primer término del lado derecho de esta ecuación corresponde al valor esperado de log-verosimilitud del conjunto de datos completos, es decir $Q(\Phi,\Theta)$. Además, si el segundo término se escribe de la forma $R(\Phi,\Theta)$ la ecuación tiene la forma [4]:

$$\log f(x|\Phi) = Q(\Phi,\Theta) - R(\Phi,\Theta) \tag{2.46}$$

La cuál es válida para cualquier valor de Φ. Si se sustituye partucularmente por $\Phi = \Theta$ se obtiene que:

$$\log f(x|\Theta) = Q(\Theta,\Theta) - R(\Theta,\Theta) \tag{2.47}$$

Tomando en cuenta la desigualdad de Gibbs se tiene que:

$$\begin{aligned} -\int_Y \log f(y|x,\Theta)\log f(y|x,\Theta)dy &\leq -\int_Y \log f(y|x,\Phi)\log f(y|x,\Theta)dy \\ \Longleftrightarrow -R(\Theta,\Theta) &\leq -R(\Phi,\Theta) \end{aligned} \tag{2.48}$$

Por lo tanto, uniendo las ecuaciones 2.46 y 2.47 se obtiene que:

$$\begin{aligned} & -R(\Theta,\Theta) \leq -R(\Phi,\Theta) \\ \iff & \log f(x|\Theta) - Q(\Theta,\Theta) \leq \log f(x|\Phi) - Q(\Phi,\Theta) \\ \iff & Q(\Phi,\Theta) - Q(\Theta,\Theta) \leq \log f(x|\Theta) - \log f(x|\Theta) \end{aligned} \tag{2.49}$$

Que era lo que se quería demostrar. □

Con este teorema se corrobora que si se mejora el valor de Q al seleccionar nuevos parámetros, también se estará mejorando la log-verosimilitud al menos en la misma cantidad (o más). Es decir, el proceso repetido de cálculo de esperanza y maximización que se observa en la Figura 2.2 efectivamente conlleva a una mejora en la verosimilitud de los datos observados, lo cual se ha demostrado matemáticamente y con ello implica la validez en la aplicación del algoritmo [4].

Por otra parte, observe de la Figura 2.2 que el algoritmo de EM no especifica cómo seleccionar los datos ocultos, ni tampoco la forma de elegir los nuevos parámetros Θ^{t+1}. La manera de escoger los datos ocultos dependerá de la persona o máquina que emplee el algoritmo, y la selección de los nuevos parámetros Θ^{t+1} se realiza con cualquier técnica de optimización. En este caso particular, la herramienta que se utilizará para optimización es la de multiplicadores de Lagrange [4].

Con respecto a la convergencia del algoritmo es hacia el **máximo local**, que como muchos otros algoritmos similares no garantiza que la convergencia sea hacia el **máximo global**, que sería el resultado ideal, pero no hay ningún criterio en el algoritmo que haga que se garantice esto. Incluso, el algoritmo converge hacia un punto crítico, de manera que en ciertas situaciones podría converger a un mínimo local o a un punto de ensilladura, de manera que hay que tener precaución al aplicarlo. Para ello, el criterio de convergencia es que con cada iteración el valor de log-verosimilitud no decrezca [1].

2.3. Multiplicadores de Lagrange

Los multiplicadores de Lagrange son una herramienta de optimización que consiste en hallar los puntos críticos de una función $f(x)$ sujeta a ciertas restricciones que se deben satisfacer, las cuales se denotan mediante $g(x) = 0$. Esta es una herramienta poderosa para encontrar las soluciones al problema de optimización cuando se tienen restricciones [4].

Particularmente, dadas dos funciones $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ y $f : \mathbb{R}^n \longrightarrow \mathbb{R}^m$, tales que $m \leq n$ el problema de optimización consiste en encontrar el máximo (o mínimo) de la función $f(x)$ sujeta a satisfacer la restricción denotada por $g(x) = 0$ [4].

Para ello, se define la **función lagrangiana** de la siguiente manera:

$$L(x,\lambda) = f(x) - \lambda^T g(x) \tag{2.50}$$

En lo cual el vector λ se conoce como multiplicador de Lagrange. Este método de multiplicadores de Lagrange busca resolver este problema, lo cual se reduce a resolver la ecuación [4]:

$$\nabla_{x,\lambda} L(x,\lambda) = 0 \tag{2.51}$$

Lo cual es equivalente a resolver el sistema:

$$\begin{aligned} \nabla_x L(x,\lambda) &= \nabla f(x) = \lambda^T g(x) \\ \nabla_\lambda L(x,\lambda) &= g(x) = 0 \end{aligned} \tag{2.52}$$

Definición 2.5. *Sean $S \subseteq \mathbb{R}^n$ y $x \in S$. El **plano tangente** a S que pasa por x se considera como el conjunto de las derivadas en x de todas las curvas diferenciables sobre S. Es decir, si Γ es el conjunto de todas las curvas diferenciables sobre S que pasan por x en t = 0 [4]:*

$$\Gamma_{S,x} = \{\gamma : \mathbb{R}^n \longrightarrow S, \gamma(0) = x, \gamma \in C^1\} \tag{2.53}$$

Entonces el plano tangente al conjunto S en el punto x está definido por:

$$T = \{\dot{\gamma}(0) : \gamma \in \Gamma_{S,x}\} \subseteq \mathbb{R}^n \tag{2.54}$$

Definición 2.6. *Sea y un vector tal que $g(y) = 0$ y $g \in C^1$. Sean $g_1, g_2, .., g_n$ los componentes del vector g.Se dice que y es un **punto regular** de g si los gradientes $\nabla g_1(y), \nabla g_2(y), ..., \nabla g_n(y)$ son linealmente independientes entre sí [4]*

Considerando que para garantizar la existencia de la inversa de una matriz cuadrada el determinante de la misma debe ser distinto de cero, es fácil ver que si los vectores $\nabla g_i(y)$ son linealmente independientes entre sí entonces matriz $\nabla g(y) \nabla g(y)^T$ tiene matriz inversa (es invertible). Con ello, se tiene la siguiente definición, la cual no se demostrará porque se sale del enfoque de este proyecto [4]:

Teorema 8. *Sea y un punto regular de la función g sobre la superficie S= {x:g(x) = 0}, entonces el plano tangente a S que pasa por el punto y es igual a:*

$$M = \{z \in \mathbb{R}^n : \nabla g(y) z = 0\} \tag{2.55}$$

Teorema 9. *Sea y un punto regular de la función g tal que g(y) = 0, que además es un máximo (o mínimo) local de f sujeto a la restricción impuesta por g. En estas condiciones se cumple que [4]:*

$$\nabla g(y) z = 0 \Longrightarrow \nabla f(y) z = 0 \tag{2.56}$$

Para todo $z \in \mathbb{R}^n$

Demostración. Sea $z \in \mathbb{R}^n$ / $\nabla g(y) z = 0$. Por el Teorema 8 se tiene que z pertenece al plano tangente al conjunto $S = \{x : g(x) = 0\}$ por el punto y. Por lo tanto, existe $\varphi \in \Gamma_{S,y}$ que satisface $\dot{\varphi}(0) = 0$

Por otra parte, note que y es un máximo (o mínimo) local de $f(x)$ sujeto a la condición $g(x) = 0$ si y solo si:

$$0 = \frac{d}{dt} f(\gamma(t))\Big|_{t=0} = \frac{df}{dx}\frac{d}{dt}\gamma(t)\Big|_{t=0} = \nabla f(y)\dot{\gamma}(0) \tag{2.57}$$

para todas las curvas $\gamma \in \Gamma_{S,y}$. Particularmente para $\gamma = \varphi$ se tiene que:

$$0 = \nabla f(y)\dot{\gamma}(y) = \nabla f(y) z \tag{2.58}$$

Tal como se buscaba. □

Por otra parte, la existencia del multiplicador de Lagrange y su utilidad para resolver el problema se garantiza en los teoremas 10, 11 t 12, cuya demostración se sale del enfoque de este proyecto [4].

Teorema 10. *Sea y un extremo local de una función f(x) sujeta a la restricción g(x) = 0, que además es un punto regular de f. Entonces, existe un* $\lambda \in \mathbb{R}^m$ *tal que:*

$$\nabla f(y) + \lambda^T \nabla g(y) = 0 \tag{2.59}$$

Teorema 11. *Sea y un mínimo local del problema de obtener valores extremos de una función* $f(x)$ *sujeta a la restricción* $g(x) = 0$*, y suponga que* $f \in C^2$*,* $g \in C^2$*. Entonces, existe un* λ *que satisface las ecuaciones:*

$$\begin{aligned} \nabla_x L(y, \lambda) &= 0 \\ z^T \nabla^2_{x,x} L(y, \lambda) z &\geq 0 \end{aligned} \tag{2.60}$$

Para todo $z \in \mathbb{R}^n$ *tal que* $g(y)z = 0$*.*

Teorema 12. *Suponga que* $f \in C^2$*,* $g \in C^2$*, sea y un vector que satisface las ecuaciones:*

$$\begin{aligned} \nabla_x L(y, \lambda) &= 0 \\ z^T \nabla^2_{x,x} L(y, \lambda) z &\geq 0 \end{aligned} \tag{2.61}$$

Para todo $z \in \mathbb{R}^n$*,* $z \neq 0$*, tal que* $g(y)z = 0$*, entonces y es un mínimo local del problema de encontrar los valores extremos de la función* $f(x)$ *sujeto a la restricción* $g(x) = 0$

2.4. Distribuciones de probabilidad

2.4.1. Función de densidad de probabilidad de Rayleigh

La distribución Rayleigh es encontrada en diversa cantidad de fenómenos. Es una distribución cuya PDF normalizada viene dada por la ecuación 2.62.

$$f_x(x, \sigma) = \frac{x}{\sigma^2} e^{\frac{-x^2}{2\sigma^2}}; x > 0 \tag{2.62}$$

Donde σ es un parámetro que que se ajusta según la distribución [7].

2.4.2. Función de densidad de probabilidad de Gamma

La PDF de la distribución Gamma viene dada por la ecuación 2.63.

$$f_x(x, k) = \frac{\beta^k}{\Gamma(k)} x^{k-1} e^{-\beta x}; x > 0 \tag{2.63}$$

Donde k y β son parámetros que que se ajusta según la distribución [7].

Capítulo 3

Diseño del algoritmo

En la presente sección se detallan los procedimientos realizados para la implementación del algoritmo.

3.1. Datos ocultos

Tomando en cuenta la sección 2 las observaciones constan de una serie de mediciones x generadas por la variable aleatoria X, la cual sigue una función de densidad de probabilidad $f(x|\Theta)$, donde como ya se discutió Θ son los parámetros de dicha función.

En este caso particular se trabajará con funciones de densidad de probabilidad de Rayleigh y Gamma. Por lo tanto, para este caso se supondrá que el conjunto de datos es una mezcla de N funciones de densidad de probabilidad (solamente de Rayleigh o solamente de Gamma), donde para una observación se tiene que $\Theta^T = (\alpha_1, ..., \alpha_n, ..\theta_1, ..., \theta_n)$ tal que:

$$f(x_i|\Theta) = \sum_{n=1}^{N} \alpha_n f_n(x_i|\theta_n) \tag{3.1}$$

Donde:

$$\sum_{n=1}^{N} \alpha_n = 1 \tag{3.2}$$

Tomando en cuenta que se tiene un conjunto de mediciones independientes $x^T = (x_1, ..., x_m)$ se tiene que la expresión para log-verosimilitud es la siguiente:

$$\begin{aligned} \log L(\Theta, x) &= \log f(x|\Theta) \\ &= \log \prod_{m=1}^{M} f(x_m|\Theta) \\ &= \log \prod_{m=1}^{M} \sum_{n=1}^{N} \alpha_n f_n(x_m|\theta_n) \\ &= \sum_{m=1}^{M} \log \sum_{n=1}^{N} \alpha_n f_n(x_m|\theta_n) \end{aligned} \tag{3.3}$$

Para lo anterior no hay manera razonable de trabajar con la suma en el argumento de un logaritmo, por lo que es fundamental elegir apropiadamente los datos ocultos. Para la presente aplicación del algoritmo, se supondrá que cada observación x_i es generada por un único componente de f, f_j de manera tal que los datos ocultos se tratarán como una familia de indicadores y_{ij} que es 1 si el componente i fue el que generó la observación x_j, y 0 de lo contrario (como una delta de Kronecker). De esta manera, estos datos serían generados por una variable aleatoria Y con una función de distribución de probabilidad $g(y)$.

De esta manera, se tiene que:

$$\begin{aligned} \log L_C(\Theta, x) &= \sum_{m=1}^{M} \log \sum_{n=1}^{N} \alpha_n f_n(x_m|\theta_n) \\ &= \sum_{m=1}^{M} \sum_{n=1}^{N} y_{nm} \log \alpha_n f_n(x_m|\theta_n) \end{aligned} \tag{3.4}$$

Lo cual se debe a que solamente uno de los elementos de $y_i = (y_{1i}, ..., y_{Ni})$ es 1 y todos los demás son iguales a 0.

3.2. Esperanza

Según lo discutido en la sección 2 el algoritmo de EM realiza aproximaciones reiteradas sobre la esperanza del parámetro Θ, tal como se muestra en el algoritmo de la Figura 2.2. Estas aproximaciones se denotan Θ^t, y se definen de la misma manera que el vector Θ, es decir, vienen dadas por $\Theta^t =$.

Para implementar el algoritmo EM lo el primer paso, según lo mostrado en la Figura 2.2, consta de calcular el valor esperado de log-verosimilitud, lo cual viene dado por:

$$\begin{aligned} Q(\Theta|\Theta^t) &= E_y \left[\log L_C(\Theta|z)|x, \Theta^{(t)} \right] \\ &= E_y \left[\sum_{m=1}^{M} \sum_{n=1}^{N} y_{nm} \log a_n f_n(x_m|\theta_n)|x, \Theta^{(t)} \right] \\ &= \sum_{m=1}^{M} \sum_{n=1}^{N} E_y \left[y_{nm} \log a_n f_n(x_m|\theta_n)|x, \Theta^{(t)} \right] \\ &= \sum_{m=1}^{M} \sum_{n=1}^{N} E_y \left[y_{nm}|x, \Theta^{(t)} \right] \log a_n f_n(x_m|\theta_n) \end{aligned} \tag{3.5}$$

En lo cual se utilizó el hecho de que el valor esperado es un operador lineal y que opera respecto a la variable aleatoria Y.

Con esto, es necesario obtener el valor esperado de los datos ocultos con el cálculo correspondiente

de su primer momento:

$$\begin{aligned} E_y\left[y_{nm}|x,\Theta^{(t)}\right] &= \sum_{y_{nm}\in Y} y_{nm} g(y_{nm}|x,\Theta^{(t)}) \\ &= 0\cdot g(y_{nm}=0|x,\Theta^{(t)}) + 1\cdot g(y_{nm}=1|x,\Theta^{(t)}) \\ &= g(y_{nm}=1|x,\Theta^{(t)}) \end{aligned} \tag{3.6}$$

Nótese que la variable $y_{nm} = 1$ si y solo si la observación x_m fue originada por f_n. Es claro que tratándose X de una variable aleatoria, todas las observaciones x son independientes entre sí. Por esta razón no existe relación entre y_{nm} y las observaciones x_i, $i \neq n$. Por estas razones, de la ecuación 3.6 se puede obtener, utilizando el Teorema de Bayes, que:

$$g(y_{nm}=1|x,\Theta^{(t)}) = g(y_{nm}=1|x_n,\Theta^{(t)}) = \frac{f(x_m|y_{nm}=1,\Theta^{(t)})g(y_{nm}=1|\Theta^{(t)})}{f(x_m|\Theta^{(t)})} \tag{3.7}$$

Ahora bien, tomando en cuenta la ecuación 3.7 se puede ver que el término $f(x_m|y_{nm}=1,\Theta^{(t)})$ se puede interpretar como la probabilidad de obtener una medición x_m considerando que el componente f_n fue el que la generó (lo cual es equivalente a evaluar la componente f_n en el punto x_m). Por otra parte, $g(y_{nm}=1|\Theta^{(t)})$ se puede ver como la probabilidad de escoger el componente f_n en particular, lo cual se puede representar como el peso estadístico $\alpha_n^{(t)}$. Por otra parte, el término $f(x_m|\Theta^{(t)})$ se puede expandir en términos de la suma de la ecuación 3.1, de tal manera que la ecuación 3.7 se puede reescribir de la siguiente manera:

$$E_y\left[y_{nm}|x,\Theta^{(t)}\right] = \frac{\alpha_n^{(t)} f_n(x_m|\theta_n^{(t)})}{\sum_{k=1}^{N}\alpha_k^{(t)} f_k(x_i|\theta_k^{(t)})} \tag{3.8}$$

De manera tal que se puede sustituir la expresión obtenida de la ecuación 3.8 en la ecuación 3.5 para obtener la siguiente expresión:

$$Q(\Theta|\Theta^t) = \sum_{m=1}^{M}\sum_{n=1}^{N}\frac{\alpha_n^{(t)} f_n(x_m|\theta_n^{(t)})}{\sum_{k=1}^{N}\alpha_k^{(t)} f_n(x_i|\theta_k^{(t)})}\log\alpha_n f_n(x_m|\theta_n) \tag{3.9}$$

3.3. Maximización

Tomando en cuenta lo obtenido de la ecuación 3.9, y que del diagrama de flujos considerado en la Figura 2.2 se observa que en el algoritmo de EM el paso posterior a evaluar la esperanza es obtener los parámetros de maximización, se procederá a obtener el valor de Θ que maximice la ecuación 3.9 utilizando la herramienta de optimización ya mencionada en la sección 2: los multiplicadores de Lagrange. Con esto, se debe realizar la optimización respecto a los parámetros θ_k y los pesos estadísticos α_k, tomando en cuenta que la restricción de que la suma de pesos estadísticos α_k debe ser 1. Tomando en cuenta que los multiplicadores de Lagrange son un método de maximización de una función $f(x)$ sujeta a una restricción de la forma $g(x) = 0$, el problema de optimización en este caso corresponde a:

$$\begin{cases} Q(\Theta,\Theta^t), & \text{Maximización} \\ \sum_{n=1}^{N}\alpha_n - 1 = 0, & \text{Restricción} \end{cases} \tag{3.10}$$

De manera tal que si se define el vector de pesos estadísticos dado por $\alpha^t = (\alpha_1, \alpha_2, ..., \alpha_N)$ y la función $g(\alpha) = \sum_{n=1}^{N} \alpha_n - 1$. Con esto,el problema de optimización es equivalente a resolver el siguiente problema:

$$\begin{cases} \nabla_\alpha Q(\Theta, \Theta^t) = \lambda \nabla_\alpha g(\alpha) \\ g(\alpha) = 0 \end{cases} \tag{3.11}$$

Con lo cual se satisfacen las condiciones consideradas en el Teorema 12, considerando que $Q(\Theta, \Theta^t) \in C^2$, $h \in C^2$.

Con esto, considerando la diferenciación con respecto a la variable α, se puede resolver el problema de optimización de la siguiente manera:

$$\begin{aligned} \frac{\partial}{\partial \alpha_i}\left(\sum_{m=1}^{M}\sum_{n=1}^{N} \frac{\alpha_n^{(t)} f_n(x_m|\theta_n^{(t)})}{\sum_{k=1}^{N}\alpha_k^{(t)} f_n(x_i|\theta_k^{(t)})} \log \alpha_n f_n(x_m|\theta_n)\right) &= \lambda \frac{\partial}{\partial \alpha_i}\left(\sum_{n=1}^{N}\alpha_n - 1\right) \\ \Longrightarrow \frac{\partial}{\partial \alpha_i}\left(\sum_{m=1}^{M} \frac{\alpha_i^{(t)} f_i(x_m|\theta_i^{(t)})}{\sum_{k=1}^{N}\alpha_k^{(t)} f_n(x_i|\theta_k^{(t)})} \log \alpha_i f_i(x_m|\theta_i)\right) &= \lambda \\ \Longrightarrow \sum_{m=1}^{M}\left[\frac{\alpha_i^{(t)} f_i(x_m|\theta_i^{(t)})}{\sum_{k=1}^{N}\alpha_k^{(t)} f_n(x_i|\theta_k^{(t)})} \frac{\partial}{\partial \alpha_i}\left(\log \alpha_i f_i(x_m|\theta_i)\right)\right] &= \lambda \\ \Longrightarrow \sum_{m=1}^{M}\left(\frac{\alpha_i^{(t)} f_i(x_m|\theta_i^{(t)})}{\sum_{k=1}^{N}\alpha_k^{(t)} f_n(x_i|\theta_k^{(t)})} \frac{1}{\alpha_i}\right) &= \lambda \\ \Longrightarrow \alpha_i \lambda = \sum_{m=1}^{M}\left(\frac{\alpha_i^{(t)} f_i(x_m|\theta_i^{(t)})}{\sum_{k=1}^{N}\alpha_k^{(t)} f_n(x_i|\theta_k^{(t)})}\right) & \end{aligned} \tag{3.12}$$

Con esto, considerando que la suma de todos los pesos estadísticos es 1 por la restricción a la cual está sujeto el problema de optimización (ver ecuación 3.10) se puede obtener que, sumando sobre todos los subíndices i se obtiene que:

$$\begin{aligned} \lambda = \sum_{n=1}^{N}\alpha_i \lambda = \sum_{n=1}^{N}\sum_{m=1}^{M}\left(\frac{\alpha_i^{(t)} f_i(x_m|\theta_i^{(t)})}{\sum_{k=1}^{N}\alpha_k^{(t)} f_n(x_i|\theta_k^{(t)})}\right) \\ \Longrightarrow \sum_{m=1}^{M}\sum_{n=1}^{N}\left(\frac{\alpha_i^{(t)} f_i(x_m|\theta_i^{(t)})}{\sum_{k=1}^{N}\alpha_k^{(t)} f_n(x_i|\theta_k^{(t)})}\right) \\ \Longrightarrow \sum_{m=1}^{M}\left(\frac{\sum_{n=1}^{N}\alpha_i^{(t)} f_i(x_m|\theta_i^{(t)})}{\sum_{k=1}^{N}\alpha_k^{(t)} f_n(x_i|\theta_k^{(t)})}\right) \\ = \sum_{m=1}^{M} 1 = M \end{aligned} \tag{3.13}$$

De manera tal que, considerando lo anterior, se puede evaluar este valor de λ en la ecuación 3.12 para obtener los valores de α_i requeridos:

$$\alpha_i = \frac{1}{M}\sum_{m=1}^{M}\left(\frac{\alpha_i^{(t)} f_i(x_m|\theta_i^{(t)})}{\sum_{k=1}^{N}\alpha_k^{(t)} f_n(x_i|\theta_k^{(t)})}\right) = \frac{1}{M}\sum_{m=1}^{M} E_y\left[y_{im}|x,\Theta^{(t)}\right] \tag{3.14}$$

De esta manera se han obtenido los valores de θ_i que son soluciones a la ecuación $\nabla_\alpha Q(\Theta,\Theta^t) = \lambda\nabla_\alpha g(\alpha)$, es decir, que son puntos críticos. Sin embargo, estos puntos podrían tratarse de máximos, mínimos, o puntos de ensilladura. En este caso el algoritmo lo que busca es maximización, de manera tal que se debe garantizar que, de acuerdo al Teorema 12 para que la solución sea un máximo estricto se debe cumplir que:

$$z^T\nabla^2_{\alpha,\alpha}(Q(\Theta,\Theta^t) - \lambda g(\alpha))z < 0 \tag{3.15}$$

Para todo $z \in \mathbb{R}^N$ / $\nabla_\alpha g(\alpha)z$. De esta manera, se encontró los valores de θ_i que corresponden a puntos críticos, y para que sea máximo estricto se debe garantizar, utilizando las ecuaciones 3.9, 3.10, 3.14 y 3.15, que la siguiente expresión debe ser menor estricto que 0:

$$\begin{aligned}
&z^T\nabla^2_{\alpha,\alpha}(Q(\Theta,\Theta^t) - \lambda g(\alpha))z \\
&= z^T\nabla^2_{\alpha,\alpha}(Q(\Theta,\Theta^t))z \\
&= z^T\nabla_\alpha\left(\left[\sum_{m=1}^{M}\frac{\alpha_i^{(t)} f_i(x_m|\theta_i^{(t)})}{\sum_{k=1}^{N}\alpha_k^{(t)} f_k(x_m|\theta_k^{(t)})}\cdot\frac{1}{\alpha_i}\right]\Bigg|_{1\le i\le N}\right)z \\
&= z^T Diag\left(\left[\sum_{m=1}^{M}\frac{\alpha_i^{(t)} f_i(x_m|\theta_i^{(t)})}{\sum_{k=1}^{N}\alpha_k^{(t)} f_k(x_m|\theta_k^{(t)})}\cdot\frac{-1}{\alpha_i^2}\right]\right)z \\
&= z^T Diag\left(\frac{-M^2}{\sum_{m=1}^{M} E_y\left[y_{im}|x,\Theta^{(t)}\right]}\right)z \\
&= \sum_{i=1}^{N} z_i^2\left(\frac{-M^2}{\sum_{m=1}^{M} E_y\left[y_{im}|x,\Theta^{(t)}\right]}\right) < 0
\end{aligned} \tag{3.16}$$

Lo cual es válido para todo $z \in \mathbb{R}^N$, puesto que tanto z_i^2 como M^2 son números reales positivos y los valores esperados $E_y\left[y_{im}|x,\Theta^{(t)}\right]$ también corresponden a valores positivos puesto que se le atribuyen valores esperados de funciones de densidad de probabilidad. Considerando lo anterior, el Teorema 12 garantiza que la solución encontrada en la ecuación 3.14 corresponde a un máximo estricto, por lo que la maximización respecto a las variables α_i queda demostrada y descrita en la ecuación 3.14 para la posterior aplicación en el algoritmo.

Por otra parte, se debe realizar el procedimiento para maximizar $Q(\Theta|\Theta^t)$ respecto a las variables $\theta_i^{(t)}$, que en este caso no tiene restricción, por lo que la optimización se realiza mediante la derivación y posterior resolución de la derivada igualada a cero, de la siguiente manera:

$$\begin{aligned} &\frac{\partial}{\partial\theta_i}Q(\Theta,\Theta^t)=0\\ \iff &\frac{\partial}{\partial\theta_i}\sum_{m=1}^{M}\sum_{n=1}^{N}\frac{\alpha_n^{(t)}f_n(x_m|\theta_n^{(t)})}{\sum_{k=1}^{N}\alpha_k^{(t)}f_n(x_m|\theta_k^{(t)})}\log\alpha_n f_n(x_m|\theta_n)=0\\ \iff &\sum_{m=1}^{M}\frac{\alpha_i^{(t)}f_i(x_m|\theta_i^{(t)})}{\sum_{k=1}^{N}\alpha_k^{(t)}f_n(x_m|\theta_k^{(t)})}\frac{\partial}{\partial\theta_i}\left(\log\alpha_i f_i(x_m|\theta_i)\right)=0 \end{aligned} \tag{3.17}$$

De esta manera, la solución particular dependerá de las funciones de densidad de probabilidad en particular que se tengan en el modelo matemático que se desea realizar. En particular, para este caso se considerarán los casos de funciones de distribución de probabilidad de **Rayleigh y Gamma**, de manera tal que se analizarán por aparte los parámetros Θ_i que las maximizan.

3.3.1. Maximización de parámetros para la distribución de Rayleigh

Con respecto al caso de la distribución de Rayleigh la PDF es la mostrada en la ecuación 2.62. Esta comienza en cero y es altamente creciente, y es modificada únicamente por el parámetro σ, de manera que ese será el parámetro a diseñar con el algoritmo EM.

De esta manera, con respecto al parámetro σ. se tiene que:

$$\begin{aligned} \frac{\partial}{\partial\sigma_i}\left(\log\alpha_i f_i(x_m|\theta_i)\right) &= \frac{\partial}{\partial\sigma_i}\left(\log\left(\alpha_i\frac{x_m}{\sigma_i^2}e^{\frac{-x_m^2}{2\sigma_i^2}}\right)\right)\\ \frac{\partial}{\partial\sigma_i}\left(\log(\alpha_i)+\log\left(\frac{x_m}{\sigma_i^2}\right)+\log\left(e^{\frac{-x_m^2}{2\sigma_i^2}}\right)\right)&\\ =\frac{\sigma_i^2}{x_m}\cdot\frac{-2x_m}{\sigma_i^3}-\frac{-2x_m^2}{2\sigma_i^3}&\\ =\frac{-2}{\sigma_i}+\frac{x_m^2}{\sigma_i^3}& \end{aligned} \tag{3.18}$$

De manera tal que esta expresión se sustituye en la ecuación 3.17 y se obtiene lo siguiente:

$$
\begin{aligned}
&\frac{\partial}{\partial \theta_i} Q(\Theta, \Theta^t) = 0 \\
\iff &\sum_{m=1}^{M} \frac{\alpha_i^{(t)} f_i(x_m|\theta_i^{(t)})}{\sum_{k=1}^{N} \alpha_k^{(t)} f_n(x_m|\theta_k^{(t)})} \left(\frac{-2}{\sigma_i} + \frac{x_m^2}{\sigma_i^3} \right) = 0 \\
\iff 2\sigma_i^2 &\sum_{m=1}^{M} \frac{\alpha_i^{(t)} f_i(x_m|\theta_i^{(t)})}{\sum_{k=1}^{N} \alpha_k^{(t)} f_n(x_m|\theta_k^{(t)})} = \sum_{m=1}^{M} \frac{\alpha_i^{(t)} f_i(x_m|\theta_i^{(t)})}{\sum_{k=1}^{N} \alpha_k^{(t)} f_n(x_m|\theta_k^{(t)})} x_m^2 \\
\iff \sigma_i^2 &= \frac{\sum_{m=1}^{M} \frac{\alpha_i^{(t)} f_i(x_m|\theta_i^{(t)}) x_m^2}{\sum_{k=1}^{N} \alpha_k^{(t)} f_n(x_m|\theta_k^{(t)})}}{2 \sum_{m=1}^{M} \frac{\alpha_i^{(t)} f_i(x_m|\theta_i^{(t)})}{\sum_{k=1}^{N} \alpha_k^{(t)} f_n(x_m|\theta_k^{(t)})}} \\
\sigma_i^2 &= \frac{\sum_{m=1}^{M} E_y \left[y_{im} | x, \Theta^{(t)} \right] x_m^2}{2 \sum_{m=1}^{M} E_y \left[y_{im} | x, \Theta^{(t)} \right]}
\end{aligned}
\tag{3.19}
$$

Con estos nuevos parámetros se garantiza un aumento en log-verosimilitud . De esta manera, el algoritmo debe realizar las siguientes tareas:

1. Determinar parámetros iniciales (tomando en cuenta que deberían estar lo más cercano que sea posible a los finales)

2. Calcular el valor esperado $Q(\Theta, \Theta^t)$.

3. Determinar los parámetros α_i^{t+1} y σ_i^{t+1} maximicen $Q(\Theta, \Theta^t)$.

4. Repetir el punto anterior hasta obtener convergencia

5. Reportar parámetros finales

Con esto, cabe destacar que el algoritmo se muestra en la Figura 3.1. Nótese que para el caso de la función de distribución de probabilidad de Rayleigh el parámetro de ajuste es solamente σ, a diferencia de otras distribuciones en las cuales se tienen dos parámetros de ajuste y por lo tanto se puede realizar mejores estimaciones, lo cual en este caso se ve limitado.

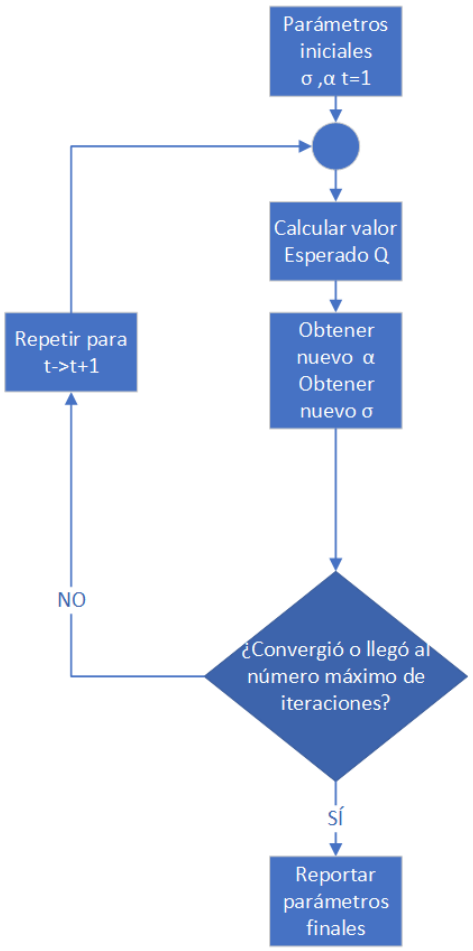

Figura 3.1: Diagrama de flujos del algoritmo EM para la función de distribución de probabilidad de Rayleigh. Autoría propia.

3.3.2. Maximización de parámetros para la función de distribución de probabilidad Gamma

Para el caso de la PDF Gamma se siguió un procedimiento muy similar, realizando los ajustes de los parámetros para maximizar log-verosimilitud. Considerando la expresión para la PDF Gamma que se presenta en la ecuación , se tiene que:

$$\log(\alpha_i f_x(x, k_i, \beta_i)) = \log(\alpha_i) + \log(f_x(x, k_i, \beta_i)) = \log(\alpha_i) + k_i \log(\beta_i) + (k_i - 1)\log(x) - \beta_i x - \log(\Gamma(k_i)) \quad (3.20)$$

Observe que el último término es una función gamma, de manera que a la hora de realizar el proceso de derivación parcial paraobtener los parámetros de ajuste óptimo este término obligaría a obtener los

resultados numéricamente. Sin embargo, para este caso no se realizará eso, si no que se utilizará la aproximación de la ecuación 3.21 para el logaritmo de la función gamma, que presenta una diferencia insignificante para valores positivos en las en el dominio, tal como se muestra en la Figura 3.2, en la cual se observa que la aproximación y la expresión original siguen la misma tendencia (lo cual produce que solo se vea el logaritmo de la función gamma en naranja, porque la aproximación está debajo en azul, y la diferencia de ambas se mantiene aproximadamente en cero). Esta aproximación en realizar es válida siempre y cuando el valor de α_i sea positivo y no se vuelva excesivamente pequeño (porque requeriría añadir otros términos en función de la función Zeta de Hurwitz, lo cual evitaría la simplificación en las expresiones que se desea). Sin embargo, se observa de la Figura 3.2 que para valores en un intervalo razonable de α_i la aproximación es válida [8].

$$\log(\Gamma(k_i)) \approx (k_i - \frac{1}{2})\log(k_i) - k_i + \frac{1}{2}\log(2\pi) \tag{3.21}$$

Considerando lo anterior, la expresión con la que se realizó el diseño fue:

$$\log(\alpha_i f_x(x, k_i, \beta_i)) = \log(\alpha_i) + k_i \log(\beta_i) + (k_i - 1)\log(x) - \beta_i x - (k_i - \frac{1}{2})\log(k_i) - k_i + \frac{1}{2}\log(2\pi) \tag{3.22}$$

Con esto, se procede a realizar la diferenciación parcial con respecto al parámetro β_i de la siguiente manera:

$$\begin{aligned} &\frac{\partial}{\partial \beta_i}(\log(\alpha_i f_x(x, k_i, \beta_i))) \\ &= \frac{\partial}{\partial \beta_i}\left(\log(\alpha_i) + k_i \log(\beta_i) + (k_i - 1)\log(x) - \beta_i x - (k_i - \frac{1}{2})\log(k_i) - k_i + \frac{1}{2}\log(2\pi)\right) \\ &= \frac{k_i}{\beta_i} - x_m \end{aligned} \tag{3.23}$$

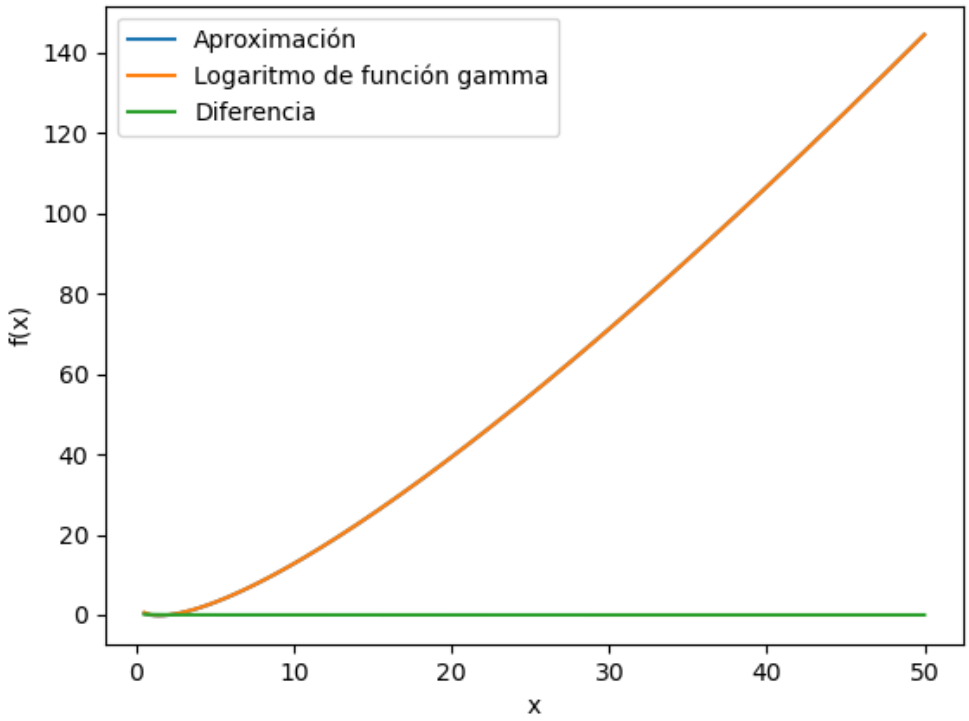

Figura 3.2: Comparación de la aproximación utilizada para el logaritmo de la función Gamma comparada a la expresión. Autoría propia.

Con esto, se sustituye la expresión de la ecuación3.23 en la ecuación 3.17 y se obtiene lo siguiente:

$$
\begin{aligned}
&\frac{\partial}{\partial \beta_i} Q(\Theta, \Theta^t) = 0 \\
\iff &\sum_{m=1}^{M} \frac{\alpha_i^{(t)} f_i(x_m|\theta_i^{(t)})}{\sum_{k=1}^{N} \alpha_k^{(t)} f_n(x_m|\theta_k^{(t)})} \left(\frac{k_i}{\beta_i} - x_m \right) = 0 \\
\iff &\frac{k_i}{\beta_i} \sum_{m=1}^{M} \frac{\alpha_i^{(t)} f_i(x_m|\theta_i^{(t)})}{\sum_{k=1}^{N} \alpha_k^{(t)} f_n(x_m|\theta_k^{(t)})} = \sum_{m=1}^{M} \frac{\alpha_i^{(t)} f_i(x_m|\theta_i^{(t)})}{\sum_{k=1}^{N} \alpha_k^{(t)} f_n(x_m|\theta_k^{(t)})} x_m \\
\iff &\frac{k_i}{\beta_i} = \frac{\sum_{m=1}^{M} \frac{\alpha_i^{(t)} f_i(x_m|\theta_i^{(t)}) x_m}{\sum_{k=1}^{N} \alpha_k^{(t)} f_n(x_m|\theta_k^{(t)})}}{\sum_{m=1}^{M} \frac{\alpha_i^{(t)} f_i(x_m|\theta_i^{(t)})}{\sum_{k=1}^{N} \alpha_k^{(t)} f_n(x_m|\theta_k^{(t)})}} \\
\iff &\frac{k_i}{\beta_i} = \frac{\sum_{m=1}^{M} E_y \left[y_{im}|x, \Theta^{(t)} \right] x_m}{\sum_{m=1}^{M} E_y \left[y_{im}|x, \Theta^{(t)} \right]}
\end{aligned}
\tag{3.24}
$$

Por otra parte, con respecto al parámetro k_i se tiene que :

$$\begin{aligned}
&\frac{\partial}{\partial i}\left(\log(\alpha_i f_x(x,k_i,\beta_i))\right)\\
&= \frac{\partial}{\partial i}\left(\log(\alpha_i) + k_i\log(\beta_i) + (k_i-1)\log(x) - \beta_i x - (k_i-\frac{1}{2})\log(k_i) - k_i + \frac{1}{2}\log(2\pi)\right)\\
&= \log(\beta_i) + \log(x_m) - \log(k_i) - \frac{k_i - \frac{1}{2}}{k_i} + 1\\
&= -\log\left(\frac{k_i}{\beta_i}\right) + \log(x_m) + \frac{1}{2k_i}
\end{aligned} \tag{3.25}$$

De esta manera se tiene que:

$$\begin{aligned}
&\frac{\partial}{\partial i}Q(\Theta,\Theta^t) = 0\\
&\iff \sum_{m=1}^{M}\frac{\alpha_i^{(t)} f_i(x_m|\theta_i^{(t)})}{\sum_{k=1}^{N}\alpha_k^{(t)} f_n(x_m|\theta_k^{(t)})}\left(-\log\left(\frac{k_i}{\beta_i}\right) + \log(x_m) + \frac{1}{2k_i}\right) = 0\\
&\iff \frac{1}{2k_i}\sum_{m=1}^{M}\frac{\alpha_i^{(t)} f_i(x_m|\theta_i^{(t)})}{\sum_{k=1}^{N}\alpha_k^{(t)} f_n(x_m|\theta_k^{(t)})} = \sum_{m=1}^{M}\frac{\alpha_i^{(t)} f_i(x_m|\theta_i^{(t)})}{\sum_{k=1}^{N}\alpha_k^{(t)} f_n(x_m|\theta_k^{(t)})}(\log\left(\frac{k_i}{\beta_i}\right) - \log(x_m))\\
&\iff k_i = \frac{\sum_{m=1}^{M}\frac{\alpha_i^{(t)} f_i(x_m|\theta_i^{(t)})}{\sum_{k=1}^{N}\alpha_k^{(t)} f_n(x_m|\theta_k^{(t)})}}{\sum_{m=1}^{M}\frac{\alpha_i^{(t)} f_i(x_m|\theta_i^{(t)})(\log\left(\frac{k_i}{\beta_i}\right)-\log(x_m))}{\sum_{k=1}^{N}\alpha_k^{(t)} f_n(x_m|\theta_k^{(t)})}}\\
&\iff k_i = \frac{\sum_{m=1}^{M} E_y\left[y_{im}|x,\Theta^{(t)}\right]}{\sum_{m=1}^{M} E_y\left[y_{im}|x,\Theta^{(t)}\right](\log\left(\frac{k_i}{\beta_i}\right) - \log(x_m))}
\end{aligned} \tag{3.26}$$

Donde la expresión $\frac{k_i}{\beta_i}$ se obtiene a partir de la ecuación 3.27. De este modo queda obtenido el parámetro de ajuste k_i y según la ecuación 3.27 el parámetro β_i viene dado por:

$$\beta_i = \frac{k_i}{\frac{\sum_{m=1}^{M} E_y\left[y_{im}|x,\Theta^{(t)}\right]x_m}{\sum_{m=1}^{M} E_y\left[y_{im}|x,\Theta^{(t)}\right]}} \tag{3.27}$$

De esta manera queda determinado el método de cálculo para los parámetros de la función de distribución de probabilidad Gamma. El algoritmo se muestra en la Figura 3.3 y debe realizar lo siguiente:

1. Determinar parámetros iniciales (tomando en cuenta que deberían estar lo más cercano que sea posible a los finales)

2. Calcular el valor esperado $Q(\Theta,\Theta^t)$.

3. Determinar los parámetros α_i^{t+1} y k_i^{t+1}, y con este último obtener β_i^{t+1} que maximicen $Q(\Theta,\Theta^t)$.

4. Repetir el punto anterior hasta obtener convergencia
5. Reportar parámetros finales

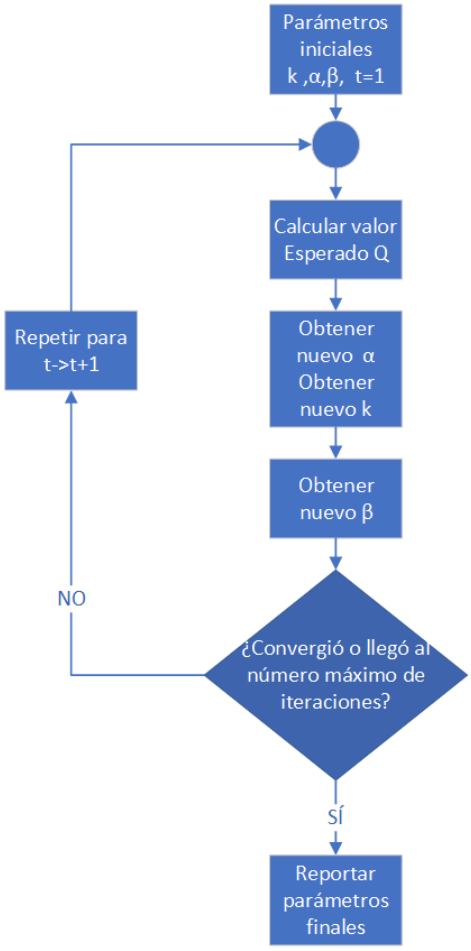

Figura 3.3: Diagrama de flujos del algoritmo EM para la función de distribución de probabilidad Gamma. Autoría propia.

3.4. Parámetros iniciales

Considerando que el algoritmo EM lo que ubica son máximos locales (y no garantiza máximo global), el algoritmo es sensible a la selección inicial de parámetros. Además, existen condiciones que se deben cumplir con respecto a estos parámetros, como por ejemplo:

- La suma de pesos debe ser igual a 1.

- No todos los parámetros pueden ser iguales entre sí.
- Los parámetros no pueden ser muy diferentes a los valores a los que deben converger

Esta última condición tiene que ver tanto con el tiempo de convergencia como con la convergencia del algoritmo al máximo que se desea ubicar y no a algún otro máximo.

Dado que los pesos estadísticos para las distribuciones no se conocen, se asumirán iguales, y tomando en consideración que la suma de los mismos es igual a 1 se toman de la siguiente manera:

$$\alpha_1^{(0)} = ... = \alpha_1^{(N)} = \frac{1}{N} \tag{3.28}$$

Con respecto a los parámetros Θ de las distribuciones estos no tienen restricciones especiales, pero si poseen rangos razonables a los que deben converger. Esta consideración inicial de los parámetros se realizará de acuerdo a parámetros estadísticos del conjunto de datos como lo son la media y la varianza.

3.4.1. Parámetros iniciales para el caso de Rayleigh

La distribución de Rayleigh se ajusta solamente con un parámetro, de manera que se estimará con la media, la cual viene dada por:

$$\mu = \sigma\sqrt{\frac{\pi}{2}} \tag{3.29}$$

Con lo cual se obtiene que:

$$\sigma = \frac{\mu}{\sqrt{\frac{\pi}{2}}} \tag{3.30}$$

Este parámetro inicial de ajuste toma en cuenta solo la una medida de tendencia central y no la variabilidad (porque la distribución de Rayleigh se ajusta con solamente un parámetro), de manera que podría no ser el ajuste más adecuado para el conjunto de datos.

3.4.2. Parámetros iniciales para el caso de Gamma

Para el caso de la función de distribución de probabilidad Gamma se tiene que la media y la varianza vienen dadas, respectivamente, por:

$$\mu = \frac{k}{\beta} \tag{3.31}$$

$$\sigma^2 = \frac{k}{\beta^2} \tag{3.32}$$

Con lo cual se puede obtener que:

$$\mu = \frac{k}{\beta} \tag{3.33}$$

$$\beta = \frac{\mu}{\sigma^2} \tag{3.34}$$

$$k = \mu\beta = \frac{\mu^2}{\sigma^2} \tag{3.35}$$

Con esto se obtiene una estimación razonable de los parámetros iniciales. Nuevamente, los parámetros no pueden ser exactamente iguales, de manera que se le añadirá una componente aleatoria que varíe ligeramente estos parámetros y permita la aplicación exitosa del algoritmo.

Con esto se tienen parámetros cercanos a los parámetros a los que se supone el algoritmo debe converger, y con ello se busca evitar que el algoritmo converja a una solución que no es la correcta. Esto se debe, cabe recordar, a que el algoritmo lo que encuentra son máximos locales (no garantiza que la solución sea un máximo global), y por ello, al tener parámetros iniciales cercanos a la solución se busca que el máximo local más cercano al que converja el algoritmo sea el de la solución buscada.

3.5. Consideraciones adicionales

Para el caso de la función de distribución de probabilidad Gamma y tomando en cuenta la aproximación que se utilizó en la ecuación 3.21 se debe garantizar que los parámetros de k_i no se vuelvan excesivamente pequeños, porque esto genera efectos no deseados en el comportamiento de la función gamma. De esta manera, se tiene que esto se realizará utilizando el siguiente razonamiento: el algoritmo se aplica varias veces, si en alguno de los casos el valor obtenido de k es excesivamente pequeño, se sustituirá por el valor mayor más cercano,porque si el valor se vuelve excesivamente pequeño el algoritmo no tendrá resultados estadísticamente válidos puesto que la aproximación utilizada no sería válida.

Un razonamiento similar se da para el caso de la función de distribución de probabilidad de Rayleigh, la cual tiene problemas de convergencia cuando el parámetro σ se vuelve sumamente pequeño. Por esta razón, en caso de que esto ocurra, se sustituirá por el valor mayor más cercano.

Esta tarea la realiza el algoritmo de manera automática por defecto, sin embargo, si el usuario lo desea, se puede desactivar y permitir que el algoritmo genere este tipo de valores.

Capítulo 4

Aplicación del algoritmo

Existen diferentes factores que se deben tomar en cuenta a la hora de decidir qué función de densidad de probabilidad elegir para que a la hora de diseñar un algoritmo que modele la naturaleza de los datos se tenga éxito. En este caso, el conjunto de datos que se utilizó para el modelado se obtuvo de la Unidad de Verificación de la Calidad del Suministro Eléctrico (UVECASE) de la Universidad de Costa Rica, del cual se obtuvo mediciones correspondientes al intervalo del 27 de maro al 3 de abril de 2017 en el sector de Turrialba. Particularmente en este análisis se estudiarán las componentes armónicas de frecuencia múltiplo de la frecuencia fundamental.

Los datos obtenidos del UVECASE reporten mediciones cada 10 minutos desde la componente de la segunda armónica hasta la número 25, lo cual se proporcionó en formato csv. De esta manera, en *Python* se realizó un tratamiento previo de la base de datos, en la cual se seleccionaron únicamente las columnas de interés (las de las armónicas), las cuales se representan como porcentajes positivos de la componente de frecuencia fundamental, y se eliminó el signo de porcentaje para que no hubiera confusión a la hora de realizar la lectura de los datos, y se consolidó en un solo archivo que contiene únicamente los datos de interés en formato numérico, de manera que no haya ningún tipo de confusión por el tipo de datos a la hora de realizar la lectura.

Es importante mencionar que al tratarse de mediciones en RMS **no se tienen observaciones negativas**. Por esta razón se seleccionaron PDFs que tuvieran únicamente componente positiva, tales como Rayleigh y Gamma. Así mismo, cabe destacar que al tratarse de mediciones reales de variables eléctricas existen limitantes desde un punto de vista metrológico que podrían provocar que las mediciones del conjunto de datos no inicien exactamente en cero (porque eventualmente los dispositivos de medición podrían tener problemas para leer tensiones RMS muy pequeñas), y por ello una distribución como la de Rayleigh presenta inconvenientes a la hora de modelar este fenómeno. Para el caso de Gamma se tienen dos factores de ajuste, por lo que el ajuste obtenido fue mejor con respecto a esta característica del conjunto de datos.

4.1. Análisis exploratorio de datos

Antes de iniciar con la implementación del algoritmo es importante realizar un análisis previo de los datos para tener una noción del comportamiento de los mismos. Para cada una de las componentes

se construyó el histograma haciendo uso de la librería *Matplotlib* de *Python*, teniendo el cuidado de definir los parámetros adecuados para que el histograma sea considerado una función de densidad y por tanto realice el ajuste correspondiente para que el área de la curva sea 1 y pueda ser por lo tanto considerada una función de densidad. Esto se logra seleccionando el parámetro $density = True$ de la siguiente manera:

```
plt.hist(arreglardatos(porcentajes)[variable], bins = 'auto', density = True)
```

Así mismo, el parámetro $bins = auto$ se selecciona de esta manera para que la visualización sea más estética. Con esto, se obtuvo un histograma para cada una de las armónicas para cada una de las fases, es decir, un total de 72 histogramas dado que se tienen datos de 24 armónicas y 3 fases. En las figuras 4.1 y 4.2 se muestran dos de estos a modo de ejemplo representativo, uno para componentes pares y otros para componentes impares. Note que para el caso de los componentes pares las mediciones están muy separadas entre sí, y por ello el histograma no se visualiza continuo, lo cual no ocurre en el caso de la componente impar.

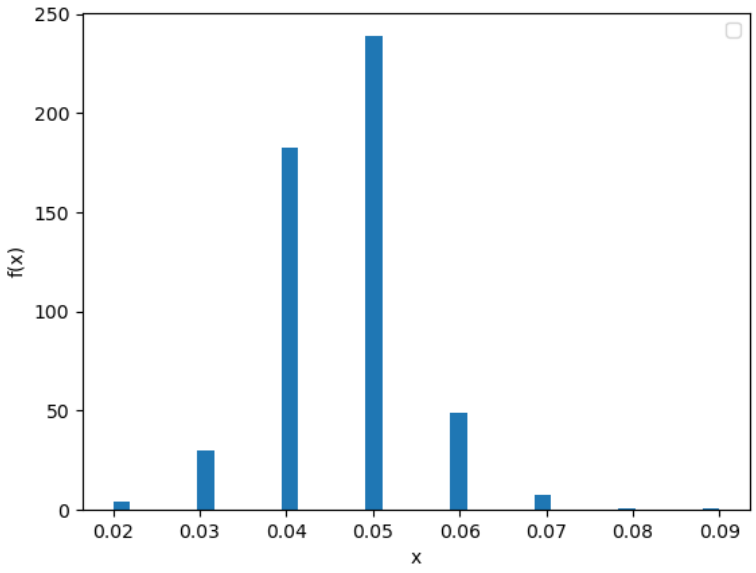

Figura 4.1: Histograma para la cuarta armónica de la fase BC. Autoría propia.

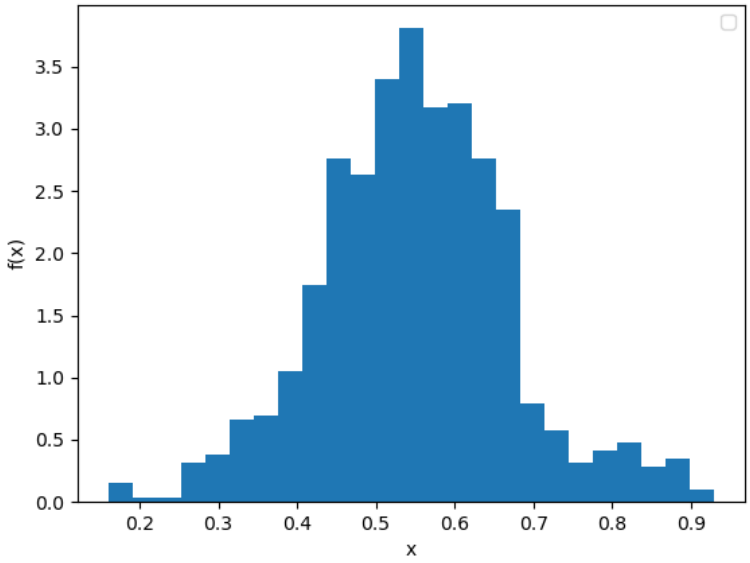

Figura 4.2: Histograma para la séptima armónica de la fase BC. Autoría propia.

En las redes de distribución de potencia se ha observado que el efecto de las componentes pares de armónicos tiende a cancelarse, de modo que se estudió el comportamiento de las medias, lo cual se muestra en la Tabla 4.1. Nótese que el aporte de las componentes pares está un orden de magnitud por debajo del del de componentes impares, lo cual corrobora el hecho comentado anteriormente. Por estas razones, y sumado al hecho de que las mediciones podrían estar muy cargadas de ruido al bajar el orden de magnitud el algoritmo se aplicará únicamente a las componentes impares, puesto que será en las cuales se podrá observar mejor la eficiencia del mismo a la hora de determinar un modelo estadístico adecuado.

Tabla 4.1: Promedios obtenidos para el conjunto de datos

Fase	**Promedio impares/(%)**	**Promedio pares/(%)**
Conjunto completo	0.2496	0.0275
C-A	0.2595	0.0270
B-C	0.2443	0.0290
A-B	0.2450	0.0265

4.2. Resultados de la aplicación del algoritmo

Como se comentó anteriormente las componentes pares fueron excluidas del análisis para la aplicación de este algoritmo. Para las otras los resultados se muestran en las Tablas 4.2, 4.3 y 4.4.

Tabla 4.2: Logverosimilitud obtenida para los casos de distribución Gamma y Rayleigh

Variable	Logverosimilitud Gamma	Logverosimilitud Rayleigh	Diferencia/(%)
PhaseC_AV_Harmonic3rd	481,85	73,58	84,73
PhaseC_AV_Harmonic5th	579,96	-84,09	114,50
PhaseC_AV_Harmonic7th	685,03	143,41	79,07
PhaseC_AV_Harmonic9th	851,49	626,69	26,40
PhaseC_AV_Harmonic11th	1140,05	893,80	21,60
PhaseC_AV_Harmonic13th	1485,07	1358,77	8,50
PhaseC_AV_Harmonic15th	1744,02	1599,32	8,30
PhaseC_AV_Harmonic17th	2067,17	1808,67	12,50
PhaseC_AV_Harmonic19th	2666,51	2345,29	12,05
PhaseC_AV_Harmonic21st	2414,72	2218,39	8,13
PhaseC_AV_Harmonic23rd	2343,64	2232,41	4,75
PhaseC_AV_Harmonic25th	2603,90	2473,30	5,02
PhaseB_CV_Harmonic3rd	496,36	66,25	86,65
PhaseB_CV_Harmonic5th	568,62	-36,30	106,38
PhaseB_CV_Harmonic7th	691,24	217,79	68,49
PhaseB_CV_Harmonic9th	1037,48	687,71	33,71
PhaseB_CV_Harmonic11th	Inf	797,69	-
PhaseB_CV_Harmonic13th	1739,51	1702,45	2,13
PhaseB_CV_Harmonic15th	1741,61	1692,12	2,84
PhaseB_CV_Harmonic17th	Inf	1631,91	-
PhaseB_CV_Harmonic19th	2210,97	2088,35	5,55
PhaseB_CV_Harmonic21st	2568,34	2487,39	3,15
PhaseB_CV_Harmonic23rd	2335,46	2292,31	1,85
PhaseB_CV_Harmonic25th	2436,04	2381,47	2,24
PhaseA_BV_Harmonic3rd	Inf	88,57	-
PhaseA_BV_Harmonic5th	628,16	-51,33	108,17
PhaseA_BV_Harmonic7th	751,02	192,72	74,34
PhaseA_BV_Harmonic9th	1041,70	589,41	43,42
PhaseA_BV_Harmonic11th	1125,54	867,51	22,92
PhaseA_BV_Harmonic13th	1608,74	1503,74	6,53
PhaseA_BV_Harmonic15th	1796,91	1680,25	6,49
PhaseA_BV_Harmonic17th	Inf	1788,71	-
PhaseA_BV_Harmonic19th	Inf	2475,24	-
PhaseA_BV_Harmonic21st	2698,84	2606,35	3,43
PhaseA_BV_Harmonic23rd	2540,28	2424,83	4,54
PhaseA_BV_Harmonic25th	2725,31	2624,10	3,71

Tabla 4.3: Parámetros obtenidos para la distribución gamma

Fase	Armónica	alfa1	alfa2	alfa3	k1	k2	k3	beta1	beta2	beta3
CA	3	0,33	0,33	0,33	15,47	17,48	16,44	23,63	29,43	26,69
CA	5	0,34	0,33	0,33	30,87	30,28	31,22	41,80	38,02	41,94
CA	7	0,34	0,33	0,33	21,83	23,35	24,06	37,86	38,78	38,78
CA	9	0,30	0,21	0,48	3,45	13,98	40,58	45,00	52,73	111,72
CA	11	0,08	0,68	0,24	5,90	27,87	26,75	23,95	92,06	171,16
CA	13	0,00	0,52	0,48	3,48	7,08	16,91	33,71	122,75	88,36
CA	15	0,16	0,61	0,23	68,83	9,43	8,24	342,51	85,78	76,53
CA	17	0,33	0,32	0,35	10,89	10,89	10,89	99,02	99,02	99,03
CA	19	0,29	0,27	0,44	12,22	12,20	15,23	190,96	190,91	217,81
CA	21	0,46	0,31	0,23	15,76	7,75	7,62	195,45	119,91	116,70
CA	23	0,28	0,37	0,35	6,51	6,51	18,48	111,76	111,76	217,34
CA	25	0,18	0,18	0,64	6,24	6,24	7,95	120,65	120,65	144,11
BC	3	0,52	0,15	0,32	50,10	102,55	26,00	108,82	105,06	42,70
BC	5	0,33	0,33	0,34	26,76	27,03	25,75	36,47	36,95	36,99
BC	7	0,37	0,31	0,33	40,10	10,07	24,63	69,98	19,54	44,10
BC	9	0,11	0,66	0,23	3,39	24,44	7,95	22,29	80,08	149,51
BC	11	0,00	0,00	0,17	78,42	9,21	121,92	513,74	32,05	339,44
BC	13	0,20	0,25	0,55	3,84	21,96	5,43	46,67	145,34	60,27
BC	15	0,38	0,56	0,06	8,65	6,19	3,48	61,02	122,13	38,37
BC	17	0,18	0,00	0,00	111,29	12,33	20,60	609,45	107,42	160,83
BC	19	0,20	0,18	0,62	5,27	5,27	11,20	79,84	79,85	129,67
BC	21	0,63	0,10	0,27	4,29	3,26	19,00	87,75	54,00	763,41
BC	23	0,30	0,42	0,28	4,77	4,77	4,77	81,14	81,14	81,14
BC	25	0,16	0,16	0,68	3,87	3,87	8,13	89,78	89,78	132,04
AB	3	0,00	0,00	0,15	32,96	36,22	143,53	47,67	70,27	154,38
AB	5	0,33	0,33	0,34	32,00	32,14	31,53	41,64	43,79	44,44
AB	7	0,34	0,34	0,32	34,35	36,00	13,45	58,96	61,29	24,67
AB	9	0,65	0,17	0,19	31,03	11,70	2,82	96,29	321,42	23,69
AB	11	0,49	0,19	0,32	12,04	54,64	39,40	43,64	359,03	120,82
AB	13	0,33	0,42	0,25	4,56	23,55	4,56	75,93	141,98	75,86
AB	15	0,16	0,15	0,68	81,39	39,69	4,79	450,09	679,89	54,42
AB	17	0,19	0,00	0,00	111,10	41,54	18,56	690,13	375,12	182,45
AB	19	0,00	0,00	0,01	18,38	33,01	116,88	367,85	480,27	5706,34
AB	21	0,33	0,33	0,34	6,15	6,15	6,15	133,91	133,91	133,91
AB	23	0,25	0,57	0,18	5,78	10,88	5,78	121,87	172,23	121,88
AB	25	0,20	0,22	0,58	5,34	5,34	13,38	150,47	150,47	251,55

Tabla 4.4: Parámetros obtenidos para la distribución Rayleigh

Fase	Armónica	alfa1	alfa2	alfa3	sigma1	sigma2	sigma3
CA	3	0,390	0,472	0,138	0,459	0,440	0,485
CA	5	0,470	0,530	0,000	0,547	0,546	0,251
CA	7	0,220	0,334	0,446	0,440	0,435	0,428
CA	9	0,258	0,427	0,315	0,056	0,232	0,239
CA	11	0,359	0,465	0,176	0,195	0,193	0,198
CA	13	0,212	0,304	0,484	0,119	0,040	0,119
CA	15	0,520	0,262	0,218	0,094	0,094	0,094
CA	17	0,149	0,247	0,603	0,081	0,081	0,081
CA	19	0,350	0,650	0,000	0,049	0,049	0,049
CA	21	0,466	0,176	0,358	0,054	0,054	0,054
CA	23	0,034	0,296	0,669	0,051	0,051	0,051
CA	25	0,296	0,416	0,288	0,041	0,041	0,041
BC	3	0,365	0,256	0,379	0,438	0,444	0,437
BC	5	0,304	0,692	0,003	0,521	0,518	0,469
BC	7	0,280	0,338	0,382	0,407	0,401	0,391
BC	9	0,343	0,327	0,330	0,177	0,188	0,186
BC	11	0,232	0,010	0,758	0,211	0,211	0,211
BC	13	0,644	0,221	0,135	0,080	0,080	0,080
BC	15	0,325	0,426	0,249	0,090	0,040	0,090
BC	17	0,212	0,228	0,560	0,097	0,097	0,097
BC	19	0,349	0,472	0,179	0,059	0,059	0,059
BC	21	0,061	0,263	0,675	0,054	0,045	0,028
BC	23	0,215	0,480	0,305	0,046	0,046	0,046
BC	25	0,367	0,282	0,352	0,043	0,043	0,043
AB	3	0,000	0,429	0,571	0,003	0,441	0,441
AB	5	0,000	0,555	0,445	0,292	0,526	0,534
AB	7	0,317	0,400	0,283	0,414	0,412	0,414
AB	9	0,235	0,510	0,255	0,193	0,189	0,193
AB	11	0,299	0,394	0,307	0,201	0,199	0,200
AB	13	0,421	0,423	0,156	0,104	0,042	0,114
AB	15	0,380	0,620	0,000	0,078	0,078	0,048
AB	17	0,262	0,260	0,477	0,085	0,085	0,085
AB	19	0,648	0,244	0,108	0,043	0,043	0,043
AB	21	0,266	0,276	0,458	0,035	0,035	0,035
AB	23	0,263	0,269	0,467	0,042	0,042	0,042
AB	25	0,343	0,239	0,417	0,035	0,035	0,035

Observe de la Tabla 4.2 que el algoritmo falla en 5 de los 72 armónicos analizados (el resultado de

log-verosimilitud diverge). Así mismo, en estos casos, el requisito de la suma de pesos igual a 1 se pierde, tal como se puede corroborar en la Figura 4.3.

Así mismo, a modo de ejemplo visualmente representativo se muestran dos resultados en las figuras 4.3 y 4.4. Observe que en el caso de la Figura 4.3 los datos de las mediciones no empiezan en cero, si no que empiezan en alrededor de 0,3 %, mientras que en el caso de la Figura 4.4 los datos inician más cerca de 0. Por esta razón, se vuelve evidente la limitación que tiene la distribución de Rayleigh cuando los datos no inician en cero, dado que no puede tomar este factor en consideración con éxito, tal como si se observa que lo hace la distribución gamma en la Figura 4.3 .

Por otra parte, este comportamiento también permite que la distribución gamma pueda detectar picos ubicados alrededor de medidas de tendencia central diferentes, como en el caso de la Figura 4.3. Obsérvese que el pico que está más a la izquierda el método aplicado con la distribución de Rayleigh sí lo detecta bien, sin embargo, esto no ocurre con el segundo, porque está más a la derecha.

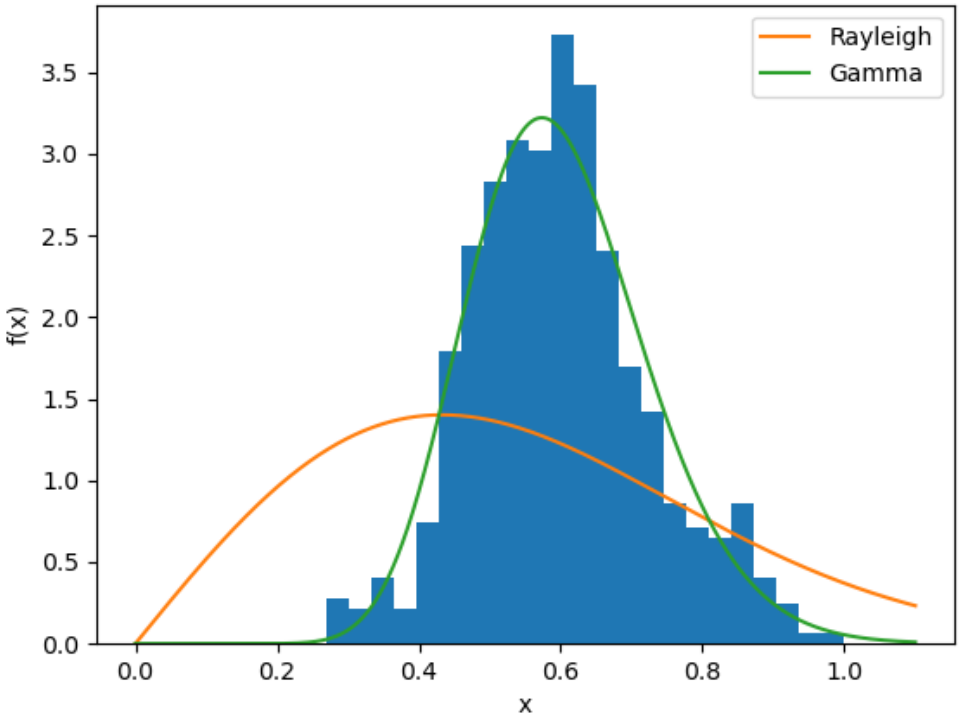

Figura 4.3: Histograma para la séptima armónica de la fase CA con ajuste por el algoritmo para PDF de Rayleigh y Gamma. Autoría propia.

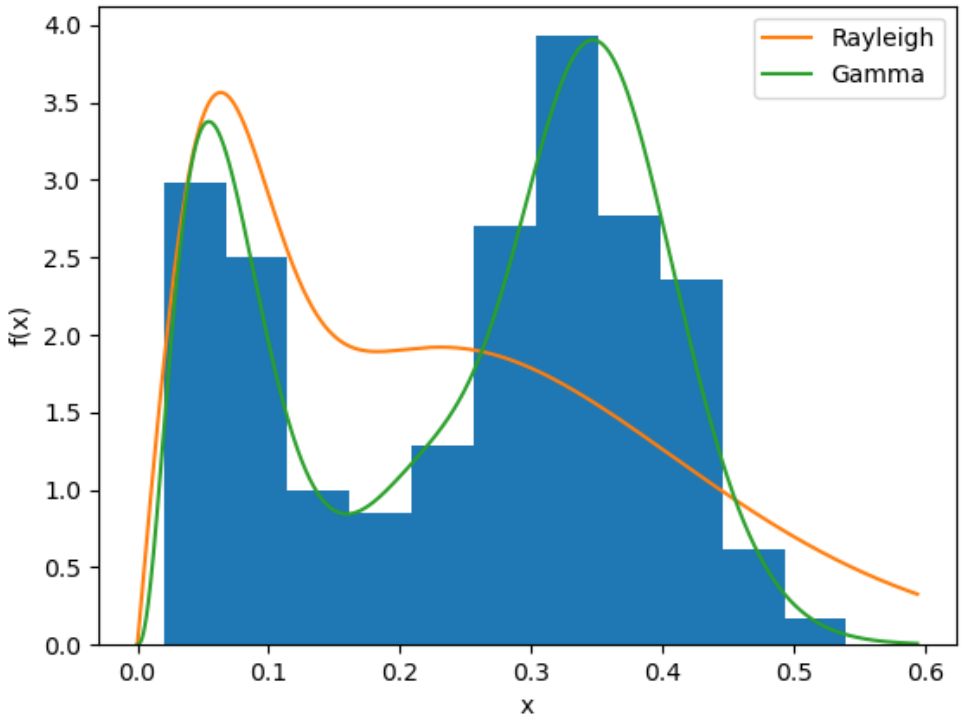

Figura 4.4: Histograma para la novena armónica de la fase CA con ajuste por el algoritmo para PDF de Rayleigh y Gamma. Autoría propia.

4.3. Análisis de resultados

Como ya se comentó previamente la distribución de Rayleigh se seleccionó porque es una distribución que abarca únicamente valores positivos, y para el caso de los datos utilizados, al tratarse de mediciones de valores de tensión en corriente alterna reportados mediante un valor de tensión RMS son únicamente valores positivos. Sin embargo, tiene la limitante de que inicia con una tendencia fuertemente creciente en 0.

Para los datos que se utilizaron, ya sea por limitaciones a la hora de realizar las mediciones, puesto que desde un punto de vista metrológico las tensiones RMS muy pequeñas no se pudieran medir, o porque la naturaleza de los datos sea así, en algunas de las variables analizadas todos los datos son positivos pero no inician exactamente en cero. Por esta razón, se determinó que el ajuste por distribución de Rayleigh no fue el más óptimo, porque no toma en cuenta esta característica de los datos.

Es importante mencionar que el objetivo de realizar este análisis es obtener un modelo estadístico que tome en cuenta todas las características que sea posible de los datos, y en el caso de la distribución

de Rayleigh esto es una limitante muy fuerte.

Por otra parte, se determinó que el algoritmo aplicado con la distribución Gamma falló en 5 de los 72 casos en los que se aplicó, dado que el parámetro de log-verosimilitud divergió. De esto, cabe mencionar que el ajuste de la función de distribución de probabilidad Gamma es sobre dos parámetros: k y β, particularmente el parámetro k será el que contiene la información relacionada la función Gamma, la cual es altamente no lineal y cualquier cambio pequeño en el valor de k puede producir grandes cambios en $\Gamma(k)$. Para este caso el ajuste se realizó por aproximación, y no específicamente sobre $\Gamma(k)$ si no sobre $\log\Gamma(k)$ utilizando la aproximación de la ecuación 3.21, la cual se graficó en la Figura 3.2 y se observó que es una buena aproximación. Sin embargo, cuando se acerca mucho a cero o cuando el valor de k crece mucho más que el intervalo mostrado ya no es adecuada.

Observe de la Tabla 4.2 que la armónica número 11 de la fase BC, la número 17 de la fase BC, y las 3, 17 y 19 de la fase AB son las variables en las cuales falla el algoritmo con la PDF Gamma, y se observa de la tabla 4.3 en todas estas existe un valor de k mayor a 100. Desde un punto de vista computacional, esto hace que evaluar la función gamma genere números demasiado grandes que desde un punto de vista computacional son difíciles de manejar y se vuelve más propenso a errores el algoritmo, y desde un punto de vista de la aproximación utilizada esta deja de tener validez, por lo que los resultados como un todo dejan de tener validez.

Así mismo, se comprueba que la suma de los pesos en estos casos deja de ser 1, puesto que la aproximación utilizada deja de ser válida y el algoritmo debe trabajar con números muy grandes que pueden generar errores en los cálculos.

Así mismo, en la Tabla 4.2 se muestra en la última columna una comparación porcentual entre el resultado obtenido la para PDF Gamma y la de Rayleigh. De esta comparación se observa que en todos los casos en los cuales el algoritmo aplicado con la PDF gamma convergió el resultado fue más adecuado que el de la PDF de Rayleigh desde un punto de vista cuantitativo. Para esto, los porcentajes de diferencia se calcularon de la siguiente manera:

$$\frac{\log L_{(gamma)} - \log L_{(Rayleigh)}}{\log L_{(gamma)}} \cdot 100 \tag{4.1}$$

De lo anterior se observan diferencias muy significativas entre la PDF Gamma y la PDF de Rayleigh, en ocasiones incluso mayores que el 100 %. Esta diferencia tan notoria se debe principalmente a que la PDF de Rayleigh no logra realizar un ajuste adecuado cuando el conjunto de datos correspondiente a las mediciones no inicia en 100.

Por otra parte, para todos los casos se aplicó el algoritmo utilizando un número de componentes igual a 3, de manera que se generan 3 pesos y 3 de cada uno de los componentes de los parámetros de las distribuciones de probabilidad (es decir, 3 σ para Rayleigh, y 3 k y 3 β para Gamma).

Ahora bien, observe por ejemplo el caso de la quinceava armónica que de la fase BC que se muestra en la Figura 4.5. Nótese que para este caso se observa solamente un pico principal y muy cercano a cero, de manera que el método con la distribución de Rayleigh sí logra atraparlo en el modelo, y luego de esto no se observan picos de importancia considerable. Por esta razón, para este caso particular, se observa una diferencia de solamente un 2,84 % en log-verosimilitud si se compara el ajuste realizado con distribución de Rayleigh con el de Gamma, y en ambos se obtiene un buen ajuste con un valor de log-verosimilitud cercano a 1700 en ambos casos, tal como se observa en la Tabla 4.2.

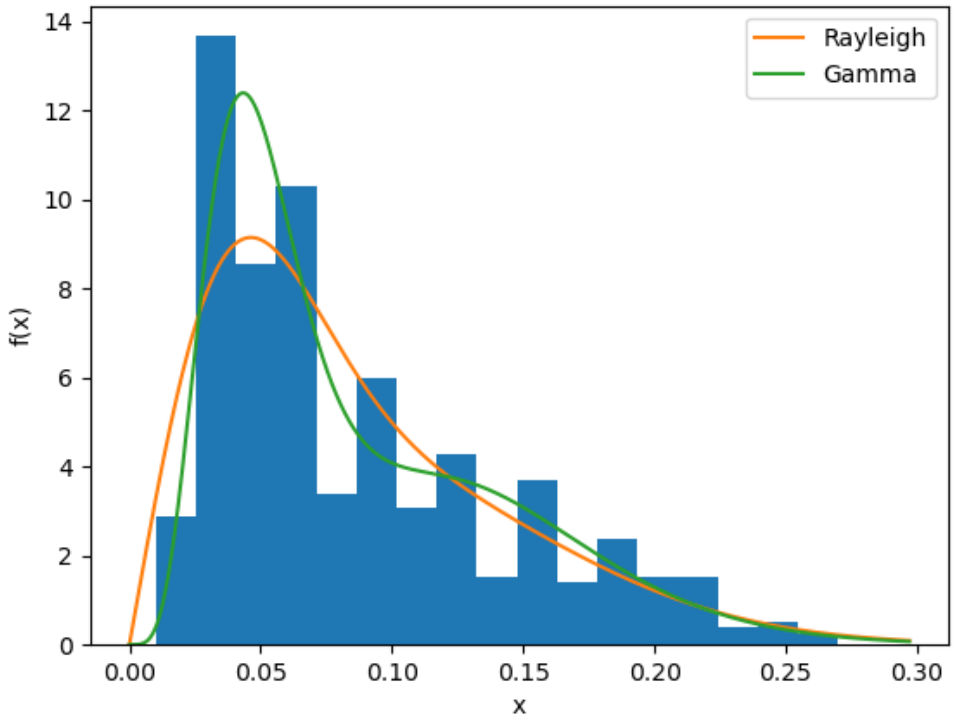

Figura 4.5: Histograma para la quinceava armónica de la fase BC con ajuste por el algoritmo para PDF de Rayleigh y Gamma. Autoría propia.

Por otra parte, véase el caso contrario, donde hubo mucha diferencia, por ejemplo en el caso de la quinta armónica de la fase CA que se observó que para la distribución de Rayleigh el valor de log-verosimilitud obtenido fue de -84,09 según se reporta en la Tabla 4.2 (lo cual indica un valor de verosimilitud muy cercano a cero). En la Figura 4.6 se vuelve evidente que en este caso las mediciones no empiezan en cero, y se encuentran en solo un pico muy concentrado alrededor de 0,7 % aproximadamente, con valores mínimos cercanos a 0,4 %. Con esto, se observa que el caso de la mezcla de distribuciones de Gamma realiza un ajuste razonable, sin embargo, el caso de mezcla de distribuciones de Rayleigh falla completamente, por este limitante correspondiente a que la distribución de Rayleigh empieza muy creciente en 0, lo cual no ocurre con el conjunto de datos.

Por esta razón, se vuelve evidente la limitante que se tiene para el caso de la mezcla de distribuciones de Rayleigh cuando se trabaja con conjuntos de datos como el que se utilizó.

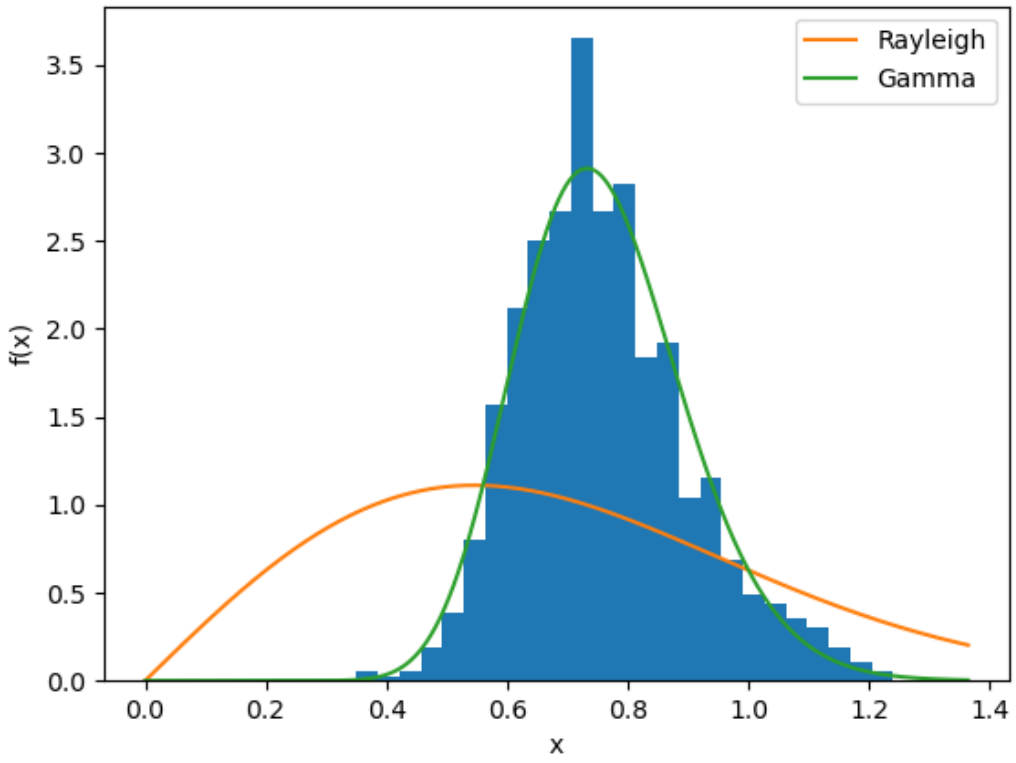

Figura 4.6: Histograma para la quinta armónica de la fase CA con ajuste por el algoritmo para PDF de Rayleigh y Gamma. Autoría propia.

Obsérvese por ejemplo la tercera armónica de la fase BC, la cual se muestra en la Figura 4.7. Nótese que se observa más de un pico, y que ninguno se observa que inicie en cero, de manera que nuevamente el método de la mezcla de distribuciones de Rayleigh no logra detectar con éxito estos picos en la distribución de los datos, mientras que en el caso de gamma esto sí se logra, dado que al tener dos parámetros de ajuste se logra manipular con mayor éxito la forma de la distribución.

Así mismo, obsérvese que a pesar de que el primer pico es muy prominente el ajuste por medio de la mezcla de distribuciones gamma logra atrapar de manera adecuada la información de los datos, y también para el segundo.

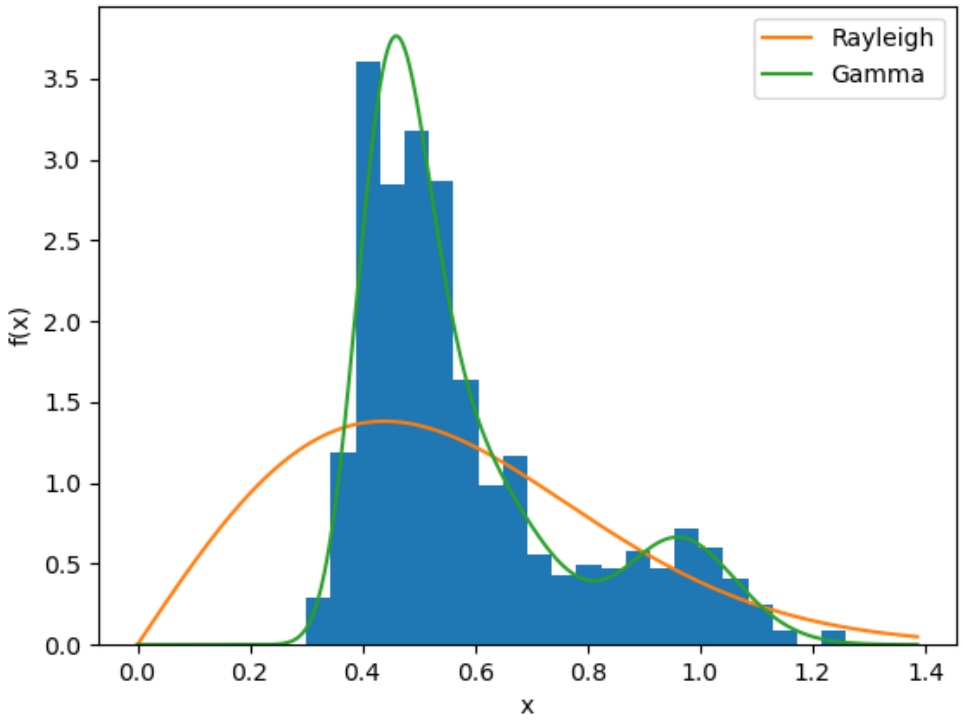

Figura 4.7: Histograma para la tercera armónica de la fase BC con ajuste por el algoritmo para PDF de Rayleigh y Gamma. Autoría propia.

Capítulo 5

Conclusiones y recomendaciones

A partir del diseño del algoritmo realizado y la posterior aplicación al conjunto de datos de mediciones eléctricas se puede concluir lo siguiente:

5.1. Conclusiones

- Se logró diseñar un algoritmo para la generación de funciones de densidad de probabilidad no gaussianas ajustadas a mediciones de tensión RMS, con base en las funciones de densidad Rayleigh y Gamma.

- Se determinó las principales características de las mediciones tensión RMS correspondientes a los armónicos del 2 al 25 para una muestra de datos de la última semana de marzo de 2017, obtenidos de la Unidad de Verificación de la Calidad del Suministro Eléctrico (UVECASE) de la Universidad de Costa Rica.

- Se obtuvo un algoritmo utilizando una combinación de funciones de distribución de probabilidad no gaussianas, específicamente las de Rayleigh y Gamma

- Se validó el algoritmo diseñado mediante pruebas en datos reales obtenidos de la Unidad de Verificación de la Calidad del Suministro Eléctrico (UVECASE) de la Universidad de Costa Rica.

- Se determinó que en 67 de las 72 variables analizadas el algoritmo utilizando distribución Gamma convergió, y dio mejores resultados que utilizando distribuciones de Rayleigh en estos casos.

- Se observó que todos los datos utilizados fueron positivos, sin embargo, en algunas de las variables las mediciones no inician exactamente en cero, de manera que para el caso de la distribución de Rayleigh los resultados obtenidos no fueron los más adecuados.

5.2. Recomendaciones

- Dado que el caso de la distribución de Rayleigh presenta problemas cuando el conjunto de datos no empieza en cero se recomienda sustituirla por otra que pueda modelar este comportamiento de manera más adecuada, como el caso de la distribución de Weibull.
- Se recomienda comparar los resultados obtenidos con otros algoritmos que se han implementado, particularmente, podría ser de mucha relevancia comparar los resultados con los obtenidos utilizando la clase *GaussianMixtures* de la librería *SkLearn* de *Python*. Esta es una librería sumamente utilizada y probada en análisis de datos, y contiene particularmente esta clase que contiene lo mismo que se realizó en este diseño pero para distribuciones gaussianas.
- Se sugiere seleccionar el número de componentes utilizando algún criterio científico como el criterio de información de Akaike (AIC), lo cual permitiría obtener mejores modelos y facilitar los cálculos desde un punto de vista computacional, dado que en este caso se seleccionó un número de componentes constante de 3 por simplicidad, luego de realizar un análisis visual de los histogramas de los datos, sin embargo sería recomendable automatizar esta decisión y determinar el número utilizando un criterio científico.

Apéndice A

Código para el caso de distribución de Rayleigh

A.1. Funciones auxiliares

Para el caso de esta distribución se definen una serie de funciones auxiliares que definen:

- La función de densidad de probabilidad de Rayleigh
- La log-verosimilitud de una mezcla de densidades de Rayleigh
- El valor esperado de las variables ocultas
- El valor esperado de log-verosimilitud

A.1.1. Función de densidad de Rayleigh

```
function y = Rayleigh(x, sigma ) %evaluar la pdf de Rayleigh
    if x<=0
        y = 0; %este valor se obtiene de continuidad en 0
    else
        y = (x/(sigma^2))*exp (-(x^2/(2* sigma^2)));
    end
end
```

A.1.2. Log-verosimilitud

```
function y = LogLikelihoodRayleigh(X,Alpha ,Sigma )
%log -verosimilitud Rayleigh
    M = length (X);
    N = length (Alpha);
    sum1 = 0;
```

```
    for m = 1:M
        sum2 = 0;
        for n = 1:N
            sum2 = sum2 + Alpha (n) * Rayleigh(X(m), Sigma (n));
        end
        sum1 = sum1 + log(sum2);
    end
    y = sum1 ;
end
```

A.1.3. Valor esperado de datos ocultos

```
function y = ExpectedValueYRayleigh (n,m,X,Alpha, Sigma )
%valor esperado de los datos ocultos
    N = length ( Alpha );
    num = Alpha (n)* Rayleigh(X(m),Sigma (n));
    den = 0;
    for k = 1:N
        den = den + Alpha (k)* Rayleigh(X(m),Sigma(k));
    end
    y = num/den;
end
```

A.1.4. Valor esperado de logverosimilitud

```
function y = ExpectedValueQRayleigh(X,Alpha,Sigma , AlphaNew,SigmaNew )
%valor esperado de la log - verosimilitud
    N = length ( Alpha );
    M = length (X);
    sum = 0;
    for m = 1:M
        for n = 1:N %el valor esperado Q depende del valor esperado de los datos ocultos
            sum = sum + ExpectedValueYRayleigh(n,m,X,Alpha, Sigma ) *
            log ( AlphaNew (n)* Rayleigh(X(m),SigmaNew (n)));
        end
    end
    y = sum;
end
```

A.2. Clase RayleighMix

Esta es la clase que desarrolla el algoritmo diseñado en el proyecto. Los objetos de esta clase poseen 4 propiedades: el número de componentes de la mezcla, la logverosimilitud del ajuste, el vector de pesos de los componentes $\{\alpha_i\}$ y los parámetros $\{\sigma_i\}$. Se utilizaron los siguientes métodos:

- **RayleighMix(N,Likelihood, alpha, sigma)**: genera el objeto de la clase y lo inicializa con los parámetros de entrada.
- **pdf(obj,X)**: evalúa a la mezcla de densidades de Rayleigh y genera en la salida un vector con los datos obtenidos.
- **random(obj,X)**: genera un vector de n valores distribuidos de acuerdo a la mezcla de densidades generada.

Así mismo, el método estático **RayleighMix.fit** es el que se utilizó para implementar el algoritmo y contiene todo lo comentado en el diseño.

```
classdef RayleighMix
    properties
        NumComponents
        LogLikelihood
        ComponentProportions
        sigma
    end
    methods
        function obj = RayleighMix(N, Likelihood ,alpha, sigma)
            %se define un nuevo objeto de la clase
            obj.NumComponents = N;
            obj.LogLikelihood = Likelihood ;
            obj.ComponentProportions = alpha ;
            obj.sigma = sigma ;
        end
        function f = pdf(obj ,X) %se calcula la mezcla de densidades
            %en un vector de datos
            f = 1:length (X);
            for k = 1:length(X)
                f(k) = 0;
                for n = 1:obj.NumComponents
                    %se calcula en cadacomponente y se multiplica por el peso
                    val = Rayleigh(X(k),obj.sigma(n));
                    f(k) = f(k) + obj.ComponentProportions(n)*val;
                end
```

```
        end
    end
    function r = random(obj ,n)
        %se genera un vector "n" de valores aleatorios
        %distribuidos de acuerdo con el ajuste
        priori = zeros (1, obj.NumComponents + 1);
        for k = 1:obj. NumComponents
            for j = 1:k
           %se separa el intervalo [0 ,1] en segmentos que
           %corresponden a la probabilidad de escoger un componente
                priori(k) = priori(k)+obj.ComponentProportions(j);
            end
            r = zeros(1,n);
            for k =1:n
                dr = rand;
                %este valor aleatorio uniforme indica el componente de
                %donde proviene el dato
                actComp = 1;
                while priori(actComp)<dr
                    actComp = actComp + 1;
                end
                r(k) = random('Rayleigh',obj.sigma( actComp ));
            end
        end
    end
end
    methods(Static)
        function obj = fit(X,N, EliminateImpulses )
            %se genera el ajuste a los datos "X" utilizando "N" componentes
            if nargin ==2
                EliminateImpulses = true ; %por defecto , se eliminan los impulsos
            end
            M = length (X);
            Mean = mean (X);
            Variance = var(X);
            Alpha = 1:N;
            %Mu = 1:N;
            Sigma = 1:N;
            for rep = 1:5 %cantidad de veces que se aplica el algoritmo
                for k = 1:N
                    %los valores iniciales se escogen aleatoriamente en intervalos
                    %razonables
```

```
        %Valores iniciales
        Alpha (k) =1/N;
        Sigma (k) = sqrt (Mean*sqrt(2/pi))* (0.0 + 2*rand );
    end
    for t = 1:50 %cantidad de iteraciones
        %cada vez que se aplica algoritmo
        AlphaNew = Alpha ;
        SigmaNew = Sigma ;
        ExpectedValueMatrix = zeros (N,M);
        %matriz de valores esperados de los datos invisibles
    for i = 1:N
        for m = 1:M
            ExpectedValueMatrix(i,m) =
            ExpectedValueYRayleigh(i,m,X,Alpha, Sigma );
        end
    end
    for i = 1:N %se calculan los nuevos valores de " Alpha "
        sum1 = 0;
        for m = 1:M
            sum1 = sum1 + ExpectedValueMatrix (i,m);
        end
        AlphaNew (i) = sum1 /M;
        if isnan ( AlphaNew (i))
            %en caso deerror , se mantiene el valoranterior
            AlphaNew (i) = Alpha (i);
        end
    end
    for i = 1:N %se calculan los nuevos valores de " Sigma "
        sum = 0;
        for m = 1:M
            sum = sum + ExpectedValueMatrix (i,m)*X(m)^2;
        end
        SigmaNew (i) = sqrt ( sum/ ( AlphaNew (i)* M*2) );
        if SigmaNew (i)==0 || isnan ( SigmaNew (i)) %en caso de error ,
            %se mantiene elvalor anterior
            SigmaNew (i) = Sigma (i);
        end
    end
    if EliminateImpulses
        %se verifica si esnecesario eliminar impulsos
        [maxSigmaNew , maxSigmaNewIndex] = max(SigmaNew );
        %se encuentra el mayor "Sigma " y su lugar en el vector
```

```
                    for i =1:N
                        if maxSigmaNew > 1000* SigmaNew (i)
                  %si el " Sigma " es mucho menor ,se cambia por uno razonable
                            SigmaNew (i) = SigmaNew (maxSigmaNewIndex );
                        end
                    end
                end
                if t == 1
                    Qprev =
                    ExpectedValueQRayleigh(X, Alpha ,Sigma , AlphaNew, SigmaNew );
                else
                    Qnext =
                    ExpectedValueQRayleigh(X, Alpha,Sigma , AlphaNew , SigmaNew );
                    if Qnext *1.01 < Qprev
                        %si no hubo mejoras significativas en el valor
                        break
                    end
                    Qprev = Qnext ;
                end
                Alpha = AlphaNew ;
                Sigma = SigmaNew ;
            end
            Likelihood = LogLikelihoodRayleigh(X,Alpha, Sigma );
            if rep ==1
                LikelihoodFinal = Likelihood ;
                AlphaFinal = Alpha ;
                SigmaFinal = Sigma ;
            elseif Likelihood > LikelihoodFinal
                %si los nuevos datos son mejores ,
                %estos se guardan como valores finales
                if not( isnan ( Likelihood ) || isnan (LikelihoodFinal ))
                    LikelihoodFinal = Likelihood ;
                    AlphaFinal = Alpha ;
                    %MuFinal = Mu;
                    SigmaFinal = Sigma ;
                end
            end
        end
        obj = RayleighMix (N, LikelihoodFinal , AlphaFinal , SigmaFinal );
    end
end
end
```

Apéndice B

Código para el caso de distribución Gamma

B.1. Funciones auxiliares

Para el caso de esta distribución se definen una serie de funciones auxiliares que definen:

- La función de densidad de probabilidad gamma
- La log-verosimilitud de una mezcla de densidades de gamma
- El valor esperado de las variables ocultas
- El valor esperado de log-verosimilitud

B.1.1. Función de densidad de Gamma

```
function y = Gamma(x,k,beta) %evaluar la pdf gamma
    if x<=0
        y = 0; %este valor se obtiene de continuidad en 0
    else
      y = ((beta^(k))*(x^(k-1))/gamma(k))*exp(-(beta*x));
    end
end
```

B.1.2. Log-verosimilitud

```
function y = LogLikelihoodGamma(X,Alpha ,k, beta )
%log -verosimilitud
    M = length (X);
    N = length ( Alpha );
    sum1 = 0;
```

```
    for m = 1:M
        sum2 = 0;
        for n = 1:N
            sum2 = sum2 + Alpha (n) * Gamma(X(m),k(n),beta(n));
        end
        sum1 = sum1 + log ( sum2 );
    end
    y = sum1 ;
end
```

B.1.3. Valor esperado de datos ocultos

```
function y = ExpectedValueYGamma(n,m,X,Alpha,k,beta)
%valor esperado de los datos invisibles
    N = length ( Alpha );
    num = Alpha (n)*Gamma(X(m),k(n),beta(n));
    den = 0;
    for i = 1:N
        den = den + Alpha(i)* Gamma(X(m),k(i),beta(i));
    end
    y = num/den;
end
```

B.1.4. Valor esperado de logverosimilitud

```
function y = ExpectedValueQGamma(X,Alpha,k,beta,AlphaNew,kNew , betaNew )
%valor esperado de la log - verosimilitud
    N = length ( Alpha );
    M = length (X);
    sum = 0;
    for m = 1:M
        for n = 1:N
            %el valor esperado Q depende del valor esperado de los datos ocultos
            sum = sum + ExpectedValueYGamma(n,m,X,Alpha ,k,beta )
            * log ( AlphaNew (n)* Gamma(X(m),kNew (n),betaNew (n)));
        end
    end
    y = sum;
end
```

B.2. Clase GammaMix

Esta es la clase que desarrolla el algoritmo diseñado en el proyecto. Los objetos de esta clase poseen 5 propiedades: el número de componentes de la mezcla, la logverosimilitud del ajuste, el vector de pesos de los componentes $\{\alpha_i\}$ y los parámetros $\{\sigma_i\}$. Se utilizaron los siguientes métodos:

- **GammaMix(N,Likelihood, alpha, sigma)**: genera el objeto de la clase y lo inicializa con los parámetros de entrada.
- **pdf(obj,X)**: evalúa a la mezcla de densidades de Gamma y genera en la salida un vector con los datos obtenidos.
- **random(obj,X)**: genera un vector de n valores distribuidos de acuerdo a la mezcla de densidades generada.

Así mismo, el método estático **gammaMix.fit** es el que se utilizó para implementar el algoritmo y contiene todo lo comentado en el diseño.

```
classdef gammaMix
    properties
        NumComponents
        LogLikelihood
        ComponentProportions
        k
        beta
    end
    methods
        function obj = gammaMix(N, Likelihood ,alpha ,k, beta)
            %se define un nuevo objeto de la clase
            obj.NumComponents = N;
            obj.LogLikelihood = Likelihood ;
            obj.ComponentProportions = alpha ;
            obj.k = k;
            obj.beta = beta;
        end
        function f = pdf(obj ,X)
            %se calcula la mezcla de densidades en un vector de datos
            f = 1:length (X);
            for i = 1:length(X)
                f(i) = 0;
                for n = 1:obj.NumComponents
                    %se calcula en cadacomponente y se multiplica por el peso
                    val = Gamma(X(i),obj.k(n),obj.beta(n));
```

```
                    f(i) = f(i) + obj.ComponentProportions (n)*val;
                end
            end
        end
        function r = random (obj ,n)
            %se genera un vector "n" de valores aleatorios distribuidos
            %de acuerdo con el ajuste
            priori = zeros (1, obj.NumComponents + 1);
            for k = 1:obj. NumComponents
                for j = 1:i
 %se separa el intervalo [0 ,1] en segmentos que corresponden a
 %la probabilidad de escoger un componente
                    priori(i) = priori(i)+obj.ComponentProportions(j);
                end
                r = zeros (1,n);
                for i =1:n
                    dr = rand;
%este valor aleatorio uniforme indica el componente de donde proviene el dato
                    actComp = 1;
                    while priori(actComp)<dr
                        actComp = actComp + 1;
                    end
                    r(i) = random('Gamma',obj.k( actComp ),1/obj.beta(actComp));
                end
            end
        end
    end
        methods(Static)
            function obj = fit(X,N, EliminateImpulses )
                %se genera el ajuste a los datos "X" utilizando "N" componentes
                if nargin ==2
                    EliminateImpulses = true ; %por defecto , se eliminan los impulsos
                end
                M = length (X);
                Mean = mean (X);
                Variance = var(X);
                Alpha = 1:N;
                K = 1:N;
                Beta = 1:N;
                for rep = 1:5 %cantidad de veces que se aplica el algoritmo
                    for i = 1:N
            %los valores iniciales se escogen aleatoriamente en intervalos razonables
```

```
    Alpha (i) =1/N;
    K(i) =((Mean^2)/Variance)+(0.0 + 2.0* rand );
    Beta (i) = (Mean/Variance)+(0.0 + 2.0*rand );
end
for t = 1:500
   %cantidad de iteraciones cada vez que se aplica algoritmo
    AlphaNew = Alpha;
    KNew = K;
    BetaNew = Beta;
    ExpectedValueMatrix = zeros (N,M); %matriz de
    %valores esperados de los datos invisibles
for i = 1:N
    for m = 1:M
        ExpectedValueMatrix (i,m) =
        ExpectedValueYGamma(i,m,X,Alpha,K,Beta);
    end
end
for i = 1:N %se calculan los nuevos valores de " Alpha " y "Mu"
    sum1 = 0;
    sum2 = 0;
    sum3 = 0;
    for m = 1:M
        sum1 = sum1 + ExpectedValueMatrix (i,m);
        sum2 = sum2 + ExpectedValueMatrix (i,m) * X(m);
    end

    for m = 1:M
        sum3 = sum3 + ExpectedValueMatrix(i,m)
        *(log(sum2/sum1)-log(X(m)));
    end
    AlphaNew (i) = sum1 /M;
    KNew (i) = sum1 /(2*sum3);
    if isnan( AlphaNew(i))
        %en caso deerror , se mantiene el valoranterior
        AlphaNew (i) = Alpha(i);
    end
    if isnan ( KNew (i))
        %en caso de error ,se mantiene el valor anterior
        KNew (i) = K(i);
    end
end
for i = 1:N %se calculan los nuevos valores de "beta"
```

```
            sum1 = 0;
            sum2 = 0;
            for m = 1:M
                sum1 = sum1 + ExpectedValueMatrix (i,m)*X(m);
                sum2 = sum2 + ExpectedValueMatrix (i,m);
            end
            BetaNew (i) = (sum2*KNew(i))/sum1;
            if BetaNew (i)==0 || isnan ( BetaNew (i))
                %en caso de error , se mantiene elvalor anterior
                BetaNew (i) = Beta (i);
            end
        end
        if EliminateImpulses %se verifica si es
            %necesario eliminar impulsos
            [maxKNew , maxKNewIndex] = max(KNew );
            %se encuentra el mayor "Sigma " y su lugar en el vector
            for i =1:N
                if maxKNew > 1000* KNew (i)
%si el " Sigma " es mucho menor ,se cambia por uno razonable
                    KNew (i) = KNew (maxKNewIndex );
                    BetaNew (i) = BetaNew(maxKNewIndex);
                end
            end
        end
        if t == 1
            Qprev =
            ExpectedValueQGamma(X, Alpha ,K,Beta, AlphaNew ,KNew , BetaNew );
        else
            Qnext =
            ExpectedValueQGamma(X, Alpha ,K ,Beta, AlphaNew ,KNew , BetaNew );
            if Qnext *1.01 < Qprev
                %si no hubo mejoras significativas en el valor
                break
            end
            Qprev = Qnext ;
        end
        Alpha = AlphaNew ;
        K = KNew ;
        Beta = BetaNew ;
    end
    Likelihood = LogLikelihoodGamma(X,Alpha ,K, Beta);
    if rep ==1
```

```
                    LikelihoodFinal = Likelihood ;
                    AlphaFinal = Alpha ;
                    KFinal = K;
                    BetaFinal = Beta;
                elseif Likelihood > LikelihoodFinal
        %si los nuevos datos son mejores , estos se guardan como valores finales
                    if not( isnan ( Likelihood ) || isnan (LikelihoodFinal ))
                        LikelihoodFinal = Likelihood ;
                        AlphaFinal = Alpha;
                        KFinal = K;
                        BetaFinal = Beta;
                    end
                end
            end
            obj = gammaMix(N, LikelihoodFinal , AlphaFinal, KFinal,BetaFinal);
        end
    end
end
```

Apéndice C

Consideraciones adicionales en el código

Adicionalmente se utilizó un *script* de Matlab para construir las tablas correspondientes a los resultados de logverosimilitud y a los parámetros, el cual se muestra a continuación:

```
g = table; %tabla que almacena resultados de mezcla de Rayleigh
r = table; %tabla que almacena resultados de mezcla de Gamma
h = table; %tabla que almacena resultados de mezcla de Log-normal
%Lectura de datos
datos = readtable('../data/tidy_data/datos.csv');
for i = 1:36
    display(2*i);
    u = datos{:,2*i};
    v = RayleighMix.fit(u,3);
    column1 = datos.Properties.VariableNames(2*i);
    column2 = v.LogLikelihood;
    column3 = v.ComponentProportions(1);
    column4 = v.ComponentProportions(2);
    column5 = v.ComponentProportions(3);
    column6 = v.sigma(1);
    column7 = v.sigma(2);
    column8 = v.sigma(3);

    tempt = table(column1, column2,column3,column4,column5,
    column6,column7,column8, 'VariableNames',
    {'Variable','Logverosimilitud','alfa1', 'alfa2',
        'alfa3','sigma1', 'sigma2','sigma3'});
    r = [r;tempt];
end
```

```
writetable(r,'../data/tidy_data/rayleigh.csv')

for i = 1:36
    display(2*i);
    u = datos{:,2*i};
    v = gammaMix.fit(u,3);
    column1 = datos.Properties.VariableNames(2*i);
    column2 = v.LogLikelihood;
    column3 = v.ComponentProportions(1);
    column4 = v.ComponentProportions(2);
    column5 = v.ComponentProportions(3);
    column6 = v.k(1);
    column7 = v.k(2);
    column8 = v.k(3);
    column9 = v.beta(1);
    column10 = v.beta(2);
    column11 = v.beta(3);
    tempt = table(column1, column2,column3,column4,column5,column6,
    column7,column8,column9,column10,column11, 'VariableNames',
    {'Variable','Logverosimilitud','alfa1', 'alfa2','alfa3','k1',
        'k2','k3','beta1', 'beta2', 'beta3'});
    g = [g;tempt];
end
writetable(g,'../data/tidy_data/gamma.csv')
for i = 1:36
    display(2*i);
    u = datos{:,2*i};
    v = lognormalMix.fit(u,3);
    column1 = datos.Properties.VariableNames(2*i);
    column2 = v.LogLikelihood;
    column3 = v.ComponentProportions(1);
    column4 = v.ComponentProportions(2);
    column5 = v.ComponentProportions(3);
    column6 = v.mu(1);
    column7 = v.mu(2);
    column8 = v.mu(3);
    column9 = v.sigma(1);
    column10 = v.sigma(2);
    column11 = v.sigma(3);
    tempt = table(column1, column2,column3,column4,column5,column6,
    column7,column8,column9,column10,column11, 'VariableNames',
    {'Variable','Logverosimilitud','alfa1', 'alfa2','alfa3','mu1',
```

```
        'mu2','mu3','sigma1', 'sigma2', 'sigma3'});
    h = [h;tempt];
end
writetable(h,'../data/tidy_data/lognormal.csv')
```

Así mismo, el la exploración, preprocesamiento de los datos y la graficación se realizó en en *script* de Python, el cual se adjunta a continuación:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import statistics as stats
import math
def arreglardatos(datos):
     indices = datos.columns
     datoscorregidos = datos.copy()
     for column in indices:
         datoscorregidos[column] = datoscorregidos[column]
         .str.replace("%","").astype(float)
     return datoscorregidos

def Rayleighdist(x,sigma):
    N= len(x)
    y = np.zeros(N)
    for k in range(N):
        if x[k]<0:
            y[k]= 0
        else:
            y[k]=((x[k])/sigma**2)*math.exp(-(x[k])**2/(2*sigma**2))
    return y

def gammadist(x,k,beta):
    N= len(x)
    y = np.zeros(N)
    for i in range(N):
        if x[i]<0:
            y[i]= 0
        else:
            y[i]=((beta**(k))*(x[i]**(k-1))/(math.gamma(k)))*math.exp(-beta*x[i])
    return y

#Lectura de datos
```

```
data = pd.read_csv('data/raw_data/M-0319-2017.csv',  header=0,sep = ',',
encoding='ISO-8859-1', low_memory=False)
porcentajes = data[['Phase C-A V-Harmonic 3rd', 'Phase C-A V-Harmonic 5th']]
ca = [x for x in data.columns if 'Phase C-A V-Harmonic' in x]
bc = [x for x in data.columns if 'Phase B-C V-Harmonic' in x]
ab = [x for x in data.columns if 'Phase A-B V-Harmonic' in x]
porcentajes= data[ca + bc + ab]

#Generación de archivo para análisis con EM
datoscorregidos = arreglardatos(porcentajes)
datoscorregidos.to_csv('data/tidy_data/datos.csv', index = False)

#Generación de histogramas

for variable in datoscorregidos.columns:
    plt.figure()
    plt.hist(arreglardatos(porcentajes)[variable], bins = 'auto', density = True)
    plt.xlabel('x')
    plt.ylabel('f(x)')
    plt.savefig('figures/exploratory_figures/hist' + variable.replace(" ","")
    .replace("-","_")+'.png')
    plt.close('all')
#Exploración de parámetros estadísticos
pares = arreglardatos(porcentajes).mean()[1::2]
impares = arreglardatos(porcentajes).mean()[::2]
print('Promedio pares', pares.mean())
print('Promedio impares', impares.mean())
paresca = datoscorregidos[ca].mean()[1::2]
imparesca = datoscorregidos[ca].mean()[::2]
print('Promedio pares ca', paresca.mean())
print('Promedio impares ca', imparesca.mean())
paresbc = datoscorregidos[bc].mean()[1::2]
imparesbc = datoscorregidos[bc].mean()[::2]
print('Promedio pares bc', paresbc.mean())
print('Promedio impares bc', imparesbc.mean())
paresab = datoscorregidos[ab].mean()[1::2]
imparesab = datoscorregidos[ab].mean()[::2]
print('Promedio pares ab', paresab.mean())
print('Promedio impares ab', imparesab.mean())
estadisticos = pd.DataFrame(ca + bc + ab, columns = ['Variable'])
estadisticos['Promedio'] = arreglardatos(porcentajes).mean().reset_index(drop=True)
estadisticos['Desviación estándar'] = arreglardatos(porcentajes).std().reset_index(drop=True)
```

```
#Lectura de resultados del análsis
gamma = pd.read_csv('data/tidy_data/gamma.csv',  header=0,sep = ',',
encoding='ISO-8859-1', low_memory=False)
ray = pd.read_csv('data/tidy_data/rayleigh.csv',  header=0,sep = ',',
encoding='ISO-8859-1', low_memory=False)
#Generación de histogramas con gráficos de resultados
for i, variable in enumerate(datoscorregidos.columns[1::2]):
    try:
        plt.figure()
        plt.hist(datoscorregidos[variable], bins = 'auto', density = True)
        x0 = datoscorregidos[variable].min()
        x1 = datoscorregidos[variable].max()
        x = np.arange(0,x1*1.1,0.00001)
        y1 = ray['alfa1'][i]*Rayleighdist(x, ray['sigma1'][i])+ray['alfa2'][i]
        *Rayleighdist(x, ray['sigma2'][i])+ray['alfa3'][i]
        *Rayleighdist(x, ray['sigma3'][i])
        y2 = gamma['alfa1'][i]*gammadist(x,gamma['k1'][i],gamma['beta1'][i])
        + gamma['alfa2'][i]*gammadist(x,gamma['k2'][i],
        gamma['beta2'][i])+gamma['alfa3'][i]
        *gammadist(x,gamma['k3'][i],gamma['beta3'][i])
        plt.plot(x,y1,label = 'Rayleigh')
        plt.plot(x,y2, label = 'Gamma')
        plt.xlabel('x')
        plt.ylabel('f(x)')
        plt.legend()
        plt.savefig('figures/explanatory_figures/hist' + variable.replace(" ","")
        .replace("-","_")+'.png')
        plt.close('all')
    except:
        print('No se pudo analizar' + variable)
```

Bibliografía

[1] S. Borman, "The expectation maximization algorithm-a short tutorial," *Submitted for publication*, vol. 41, 2004.

[2] R. Maitra, "On the expectation-maximization algorithm for rice-rayleigh mixtures with application to noise parameter estimation in magnitude mr datasets," *Sankhya B*, vol. 75, no. 2, pp. 293–318, 2013.

[3] C. Byrne and P. P. Eggermont, "Em algorithms." *Handbook of Mathematical Methods in Imaging*, vol. 1, p. 2, 2015.

[4] O. Arguedas, "Generación de funciones de densidad de probabilidad ajustadas a mediciones de diversas variables físicas," *Proyecto Eléctrico*, vol. 1, 2018.

[5] T. K. Moon, "The expectation-maximization algorithm," *IEEE Signal processing magazine*, vol. 13, no. 6, pp. 47–60, 1996.

[6] C. B. Do and S. Batzoglou, "What is the expectation maximization algorithm?" *Nature biotechnology*, vol. 26, no. 8, pp. 897–899, 2008.

[7] P. Z. Peebles, *Principios de probabilidad, variables aleatorias y señales aleatorias.* McGraw-Hill, 2006, vol. 4.

[8] E. T. Whittaker and G. N. Watson, *A course of modern analysis.* Courier Dover Publications, 2020.