Double Gumbel Function Optimization through PSO and Maximum Likelihood for Data Analysis Using AI

Dr. **Maritza ARGANIS**[1,*], M.Eng. **Margarita PRECIADO**[2,*]

[1] Universidad Nacional Autónoma de México. Instituto de Ingeniería. Facultad de Ingeniería, México

[2] Instituto Mexicano de Tecnología del Agua, Jiutepec, Mor.

* MArganisJ@iingen.unam.mx; preciado@tlaloc.imta.mx

Abstract: *Maximum annual hydrological frequency analysis data series requires techniques use to estimate distribution functions parameters; in case of univariate distribution functions that have more than two parameters, traditional methods complexity, such as the moment's method, increases. In this research, the five double Gumbel distribution parameters of a function were estimated to approximate maximum annual daily measured rainfall data from a climatological station located at state of Jalisco, Mexico. To achieve this objective, Bing AI tool was used to develop a program in Python language that uses PSO particle cluster optimization algorithm from evolutionary computing using as an objective function the likelihood maximization function. Bing AI was oriented by giving a search interval taken from result reported by a program that uses traditional least squares technique and with this, it was possible to eliminate errors in the AI program that finally gave adjustments with a standard error of 13.858 acceptable for data estimation with return periods between 5 and 30 years standard fitting error.*

Keywords: *Bing IA, evolutionary computation, frequency analysis, objective function, annual maximum dairy precipitation*

1. Introduction

Functions optimization is a fundamental aspect in data analysis, as it allows finding the optimal values that maximize or minimize a certain metric. In particular, the double Gumbel function is widely used in various fields, such as statistics and engineering, to model extreme distributions, specifically in hydrological analysis this distribution function is typical of precipitation data or runoff in coastal sites subject to the presence of hurricanes at certain year times and also in sites where winter rains known as "equipatas" occur [1, 2]. In this paper, we will explore how the combination of PSO (Particle Swarm Optimization) optimization technique and maximum likelihood approach can improve efficiency and accuracy in optimizing double Gumbel function for data analysis. Furthermore, we will leverage machine learning power of (AI) to further enhance this process.

2. Methodology

2.1 Double Gumbel function

Double Gumbel function [1,3] is a probability distribution that is used to model extreme events in a data set. The distribution function and density function take the form of equations 1 and 2. It is observed that it has 5 parameters p, α_1, β_1, α_1 and β_2, so the use of traditional parameter estimation techniques, for example moments are elaborate since it must be considered up to the fifth population and sample moment.

$$F(x) = p\left(e^{-e^{-y_1}}\right) + (1-p)\, e^{-e^{-y_2}} \tag{1}$$

$$f(x) = p\,\alpha_1\, e^{-e^{-y_1}} + (1-p)\,\alpha_2\, e^{-e^{-y_2}} \tag{2}$$

2.2 Optimization by Maximum Likelihood and PSO

Optimizing a function involves finding parameters that best fit the observed data. In frequency analysis, a classic optimization technique is maximum likelihood method, which consists of finding density function parameters f(x) of a variable x that maximize likelihood function L given by density function product valued at each known value xi; this means that objective function in this paper takes equation 3 form.

$$OF = máx(\prod_{i=1}^{n} f(xi, p, \alpha_1, \beta_1, \alpha_2, \beta_2) \quad (3)$$

Some tools for t frequency analysis maximum annual data series use least squares regression technique to solve problem, usually aided with logarithmic transformations use.

In this paper, it is proposed to use Particle Swarm Optimization tool, from evolutionary computing.

By combining PSO with maximum likelihood approach to double Gumbel function optimize, we can obtain more accurate and reliable results and, therefore, obtain a better double Gumbel function estimation

2.3 Particle Swarm Optimization (PSO)

It is a random algorithm with great potential and a simple form that makes an analogy of social behavior that species such as fish and birds naturally follow; it was proposed in 1995 [4]. Algorithm randomly generates a population of candidate solutions, called a particle swarm, in each iteration each particle has a position in solution search problem space, each particle has a velocity vector with which it moves in solution search space; particles interact and learn from each other; each particle has a memory of its best personal position and all particle swarm have a best global position. In each iteration, position and each particle velocity are obtained; for this, a vector is defined that goes from position in one iteration x_i (t) to best personal position p_i (t) and another vector that connects position in one iteration with best global position g(t); each particle moves parallel to its velocity vector, to vector that associates it with its best personal position and parallel to best global position obtaining each particle position in next iteration; vector that connects position in previous iteration with position in next iteration is particle velocity at next iteration. Mathematically, this is expressed as indicated in equations 4 and 5.

$$v_i(t+1) = wv_i(t) + c_1\big(p_i(t) - x_i(t)\big) + c_2(g(t) - x_i(t)) \quad (4)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (5)$$

The equation to update the velocity in terms of its components at the next instant $v_{ij}(t+1)$ is (equation 6):

$$v_{ij}(t+1) = wv_{ij}(t) + r_1c_1\left(p_{ij}(t) - x_{ij}(t)\right) + r_2c_2(g_i(t) - x_{ij}(t)) \quad (6)$$

And to update the position in terms of its components, we have (equation 7)

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1) \quad (7)$$

In equation 6, first term is called inertia term and coefficient w is inertia coefficient. Coefficients c1 and c2 are called acceleration coefficients, second term is known as cognitive component and the last term is known as social component. r_1 and r_2 are random numbers with a uniform distribution [0,1].

2.4 Bing AI

With advances in machine learning, field of we can now leverage AI power to enhance optimization process by grouping evolutionary computation use to perform likelihood function maximization.

By using AI in this process, we can take machine learning capabilities advantage to further improve optimization. For example, we can use supervised machine learning algorithms to train a model that can predict the optimal double Gumbel function parameters from previously observed data. This would allow us to speed up optimization process and obtain more accurate results in real time.

In this case, different instructions were given to Bing AI chat [5] to build a program in Python language that could be run on the Anaconda© Jupiter notebook platform [6] Input data to the problem, methods desired to solve it, and desired output were specified in detail. First program generated by Bing AI chat marked various errors associated with the dimensioning of the vector containing the search variables. Additionally, Bing AI chat had to be told in more detail that density function f(x) would be used for optimization and the distribution function F(x) and a transformed variable z for drawing the behavior of the information. After about five trials, program was successfully executed for randomly generated x data; subsequently, specific data from an analyzed site were added to perform the parameter retrieval procedure. Initial program code that worked is shown in Figure 1.

```
import numpy as np
import matplotlib.pyplot as plt
import pyswarms as ps
# Definition of the x vector with the data
nd = 32
x =
np.array([56.8,40,81.8,43,100,36.2,60,52.5,65,108.5,42.6,42.1,118.5,40.2,30.5,30.5,35,52.6,70,40,30,60,40,45,60,65,65,65,50,45,80,65
])
# Definition of the objective function
def f(params):
    p, alpha1, alpha2, beta1, beta2 = params
    y1 = alpha1 * (x - beta1)
    y2 = alpha2 * (x - beta2)
    return p * np.exp(-np.exp(-y1)) + (1 - p) * np.exp(-np.exp(-y2))
# Definition of the cumulative distribution function F(x)
def F(params):
    return np.cumsum(f(params))
# Definition of the likelihood function
def likelihood(swarm):
    n_particles = swarm.shape[0]
    return np.array([-np.sum(np.log(f(swarm[i]))) for i in range(n_particles)])
# Configuration of the optimizer
options = {'c1': 0.5, 'c2': 0.3, 'w':0.9}
optimizer = ps.single.GlobalBestPSO(n_particles=10,dimensions=5,options=options)
# Execution of the optimizer and storage of the likelihood function at each iteration
iters = 20
likelihood_values = []
for i in range(iters):
    cost,pos = optimizer.optimize(likelihood,iters=1)
    likelihood_values.append(cost)
# Graphing the results
plt.plot(range(iters), likelihood_values,'-')  # Calculated data with lines
plt.xlabel('Iteration number')  # Label for the horizontal axis
plt.ylabel('Likelihood')  # Label for the vertical axis
plt.show()
# Print the optimal parameters
print("The optimal parameters found with maximum likelihood are:")
print(f"p: {pos[0]}, alpha1: {pos[1]}, alpha2: {pos[2]}, beta1: {pos[3]}, beta2: {pos[4]}")
# Calculate and print the value of the function f(x) with optimized values
fx_opt = f(pos)
print(f"The value of the function f(x) calculated with these optimized values is: {fx_opt}")
# Calculate and print the value of the likelihood function with optimized values
likelihood_opt = likelihood(np.array([pos]))
print("The value of the likelihood function calculated with these optimized values is: {likelihood_opt}")
```

Fig. 1. Maximum likelihood code optimization program using PSO generated by Bing AI Chat

2.5 Input data

Thirty two historical record maximum daily precipitation data from climatological station 14064 from Clicom database, located in the state of Jalisco in Mexico [7], were considered (Figure 2).

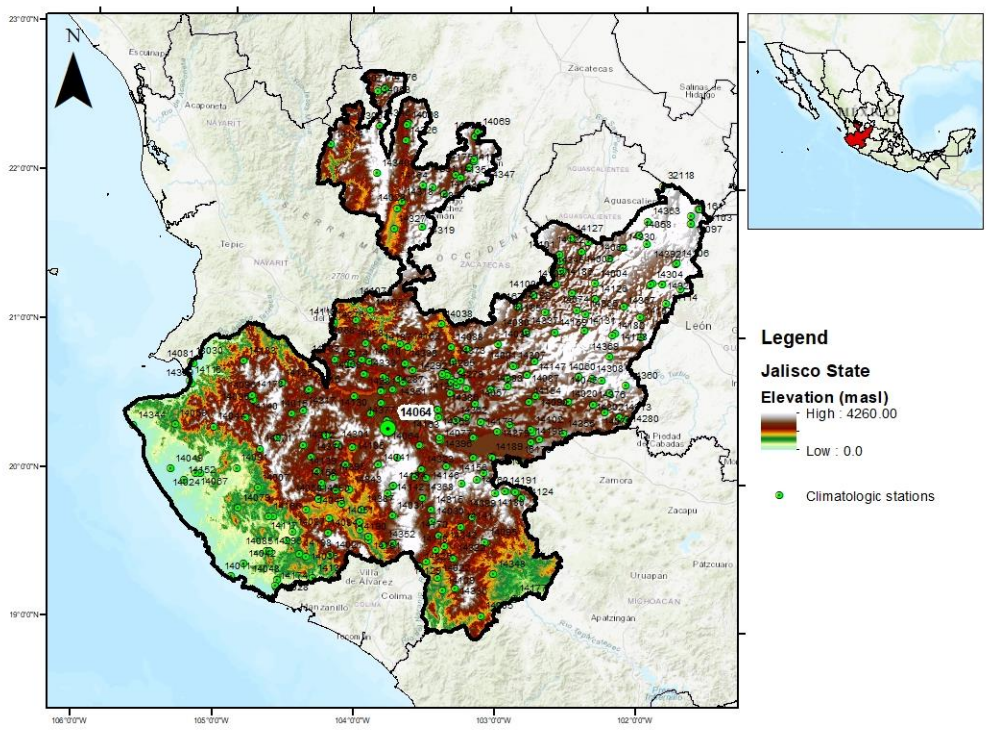

Fig. 2. Climatological station location used in our analysis, Jal. Mexico. Source: Own elaboration

3. Results

Final program made by Bing AI chat, feeding it with annual maximum daily rainfall data from station 14064, program resulted in Figure 3, which reports objective function value for each iteration. It is worth noting that program uses a method that originally minimizes, and for this reason, result of likelihood function value appears with opposite sign to indicate that a maximum value was obtained.

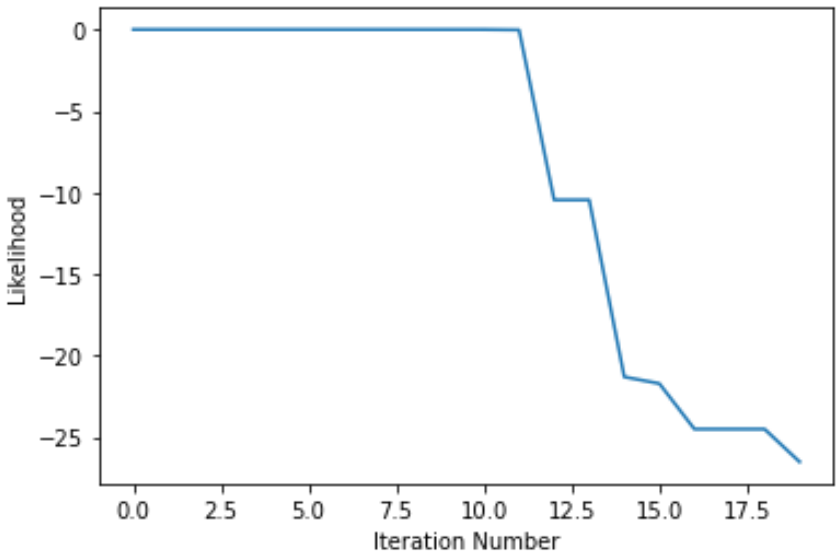

Fig. 3. Iteration number Graph vs optimized likelihood function value. Source: PSO program and maximum likelihood in Python from Bing AI chat. In Jupiter notebook. Anaconda

Program also reports information in Figure 4.

Optimal parameters found with maximum likelihood are: p: 2.290741286980267, alpha1: 0.5105690232088207, alpha2: -0.1679911851544585, beta1: -0.2712261197011611, beta2: 2.4006206416271803 The value of the function f(x) calculated with these optimized values is: [2.29074129 2.29074128 2.29074129 2.29074129 2.29074129 2.29074127 2.29074129 2.29074129 2.29074129 2.29074129 2.29074129 2.29074129 2.29074128 2.29074094 2.29074094 2.29074125 2.29074129 2.29074129 2.29074128 2.29074084 2.29074129 2.29074128 2.29074129 2.29074129 2.29074129 2.29074129 2.29074129 2.29074129 2.29074129 2.29074129 2.29074129]. The value of the likelihood function calculated with these optimized values is: [-26.52401456]

Fig. 4. Results reported by the PSO program and maximum likelihood in Python from Bing AI chat. In Jupiter notebook. Anaconda

From Figure 4, we have that parameters obtained by PSO algorithm are: p: 2.2907, α1: 0.5106, β1: -0.2712, α2: -0.1680, β2: 2.4006.

Additionally, program reports density function value f(x) evaluated at each x value, as well as objective function value of -26.5240

Upon reviewing previous results, it was observed that p value obtained by program is outside values that probabilistically define such parameters (p can only take values between 0 and 1), but this information was not included in algorithm so it was necessary to make modifications, asking Bing AI chat to add instructions to add such restrictions. Final modified program appears in Figure 5.

```
import numpy as np
import matplotlib.pyplot as plt
import pyswarms as ps
# Definition of the x vector with the data
nd = 32
x=
np.array([56.8,40,81.8,43,100,36.2,60,52.5,65,108.5,42.6,42.1,118.5,40.2,30.5,30.5,35,52.6,70,40,30,60,40,45,60,65,65,65,50,45,80,65
])
# Definition of the objective function
def f(params):
    p, alpha1, alpha2, beta1, beta2 = params
    y1 = alpha1 * (x - beta1)
    y2 = alpha2 * (x - beta2)
    return p * np.exp(-np.exp(-y1)) + (1 - p) * np.exp(-np.exp(-y2))
# Definition of the cumulative distribution function F(x)
def F(params):
    return np.cumsum(f(params))
# Definition of the likelihood function
def likelihood(swarm):
    n_particles = swarm.shape[0]
    return np.array([-np.sum(np.log(f(swarm[i]))) for i in range(n_particles)])
# Optimizer configuration
options = {'c1': 0.5, 'c2': 0.3, 'w':0.9}
# Definition of particle boundaries
bounds = (np.array([0,-1000,-1000,-1000,-1000]), np.array([1,1000,1000,1000,1000]))
optimizer = ps.single.GlobalBestPSO(n_particles=10,dimensions=5,options=options,bounds=bounds)
# Execution of the optimizer and storage of the likelihood function at each iteration
iters = 20
likelihood_values = []
for i in range(iters):
    cost,pos = optimizer.optimize(likelihood,iters=1)
    likelihood_values.append(cost)
# Plotting the results
plt.plot(range(iters), likelihood_values,'-')  # Calculated data with lines
plt.xlabel('Iteration number')  # Label for the horizontal axis
plt.ylabel('Likelihood')  # Label for the vertical axis
plt.show()
# Print the optimal parameters
print("The optimal parameters found with maximum likelihood are:")
print(f"p: {pos[0]}, alpha1: {pos[1]}, alpha2: {pos[2]}, beta1: {pos[3]}, beta2: {pos[4]}")
# Calculate and print the value of the function f(x) with optimized values
fx_opt = f(pos)
print(f"The value of the function f(x) calculated with these optimized values is: {fx_opt}")
# Calculate and print the value of the likelihood function with optimized values
likelihood_opt = likelihood(np.array([pos]))
print(f"The value of the likelihood function calculated with these optimized values is: {likelihood_opt}")
```

Fig. 5. Modified code of the maximum likelihood optimization program using PSO generated by Bing AI Chat

Results obtained in this case appear in Figures 6 and 7.

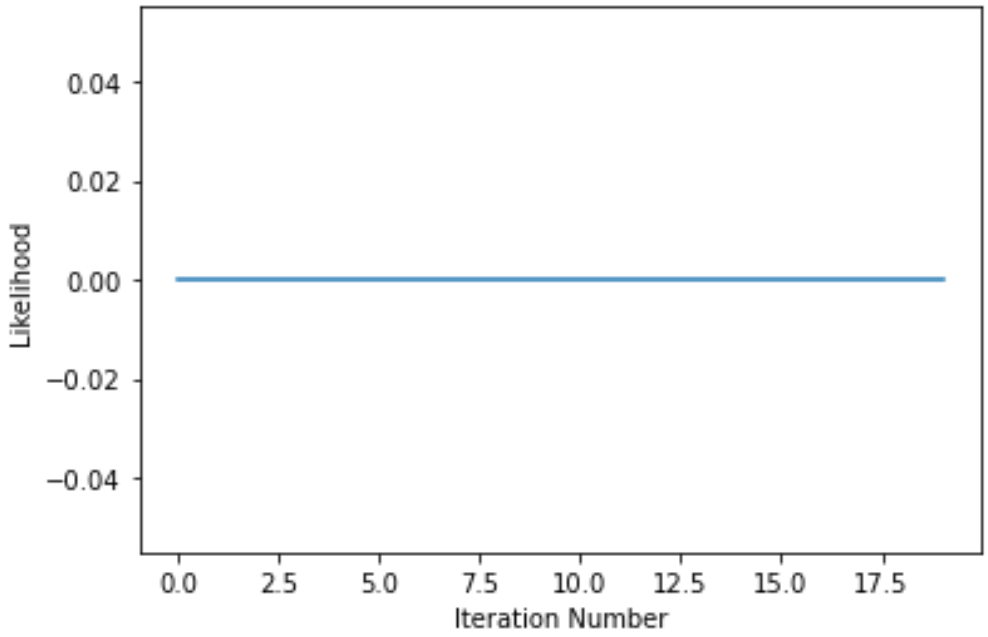

Fig. 6. Iteration number Graph vs optimized likelihood function value of. Source: PSO program and maximum likelihood in Python from Bing AI chat. In Jupiter notebook. Anaconda

Optimal parameters found with maximum likelihood are: p: 0.06229598442521622, alpha1: 431.53226191262684, alpha2: -587.9153262587531, beta1: -163.58414373459, beta2: 173.92428544406835 The value of the function f(x) calculated with these optimized values is: [1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.] The value of the likelihood function calculated with these optimized values is: [-0.]

Fig. 7. Results reported by modified PSO program and maximum likelihood in Python from Bing AI chat. In Jupiter notebook. Anaconda

From Figure 7, parameters obtained by PSO algorithm are: p=0.062296, α1= 431.5323, β1= -163.5841, α2= -587.9154, β2=173.9243, with a number of iterations equal to 20.
To check measured and calculated data, AX program [8] was used as support to obtain double Gumbel function parameters; this program performs an automatic adjustment and for this it uses an optimization by least squares; parameters reported by AX were: p= 0.895 α1= 0.082958 β1= 44.3403 α2 =0.072066 β2=103.7846 standard fit [8] of 3.18.
On the other hand, double Gumbel function parameters obtained with PSO and maximum likelihood from AI program were manually entered into AX. Standard fit error in this case is 129.282, but in this case with AX it is not possible to calculate the x data values to make results comparison.
By increasing iterations number at program to 100 and changing search parameters interval to bounds = (np.array([0.7,0.05,0.05,30,80]), np.array([1,0.1,0.1,50,120])), the results of Figures 8 and 9 are obtained.

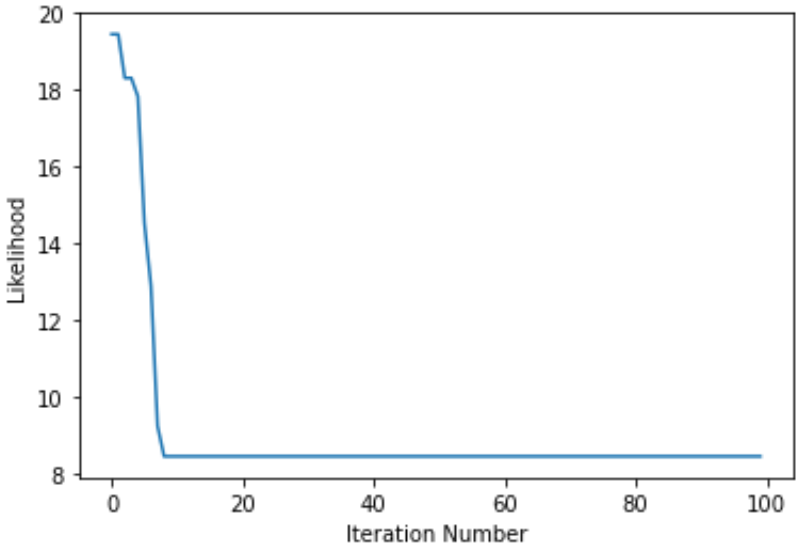

Fig. 8. Graph of iteration number vs value of optimized likelihood function. Source: PSO program and maximum likelihood in Python from Bing AI chat. In Jupiter notebook. Anaconda

Optimal parameters found with maximum likelihood are: p: 0.9044526127596921, alpha1: 0.06894210209217676, alpha2: 0.0692385601018597, beta1: 30.760016258389406, beta2: 97.08696390089152 The value of the function f(x) calculated with these optimized values is: [0.76604752 0.53297039 0.88339593 0.58832433 0.93903588 0.45489691 0.79165459 0.72337798 0.82301594 0.96090501 0.58129238 0.57234923 0.97846066 0.53684426 0.32676531 0.32676531 0.42871629 0.72448911 0.84610593 0.53297039 0.31530353 0.79165459 0.53297039 0.62183152 0.79165459 0.82301594 0.82301594 0.82301594 0.69361062 0.62183152 0.8782631 0.82301594] The value of the likelihood function calculated with these optimized values is: [13.61567589]

Fig. 9. Results reported by modified pso program and maximum likelihood in Python from Bing AI chat. In Jupiter notebook. Anaconda

From Figure 9, parameters obtained by pso algorithm are: p=0.9045, α1= 0.0689, β1= 30.7600, α2= 0.0692, β2=97.0870, with a number of iterations equal to 40. Upon manually inserting these parameters into AX, a standard fitting error of 13.858 is obtained.
Figure 10 compares results obtained with pso AI maximum likelihood and AX program.

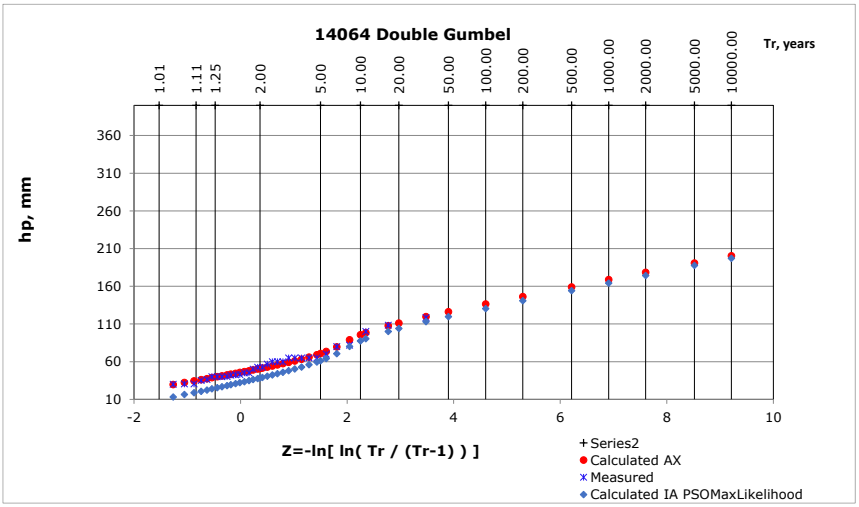

Fig. 10. Probabilistic extrapolation comparison using IA PSO Max Likelihood and AX program

Figure 10 illustrates how as AI is oriented, an improved optimization program was obtained that uses a combined evolutionary computing technique and Maximum Likelihood method as support in frequency analysis of a maximum annual rainfall series data to estimate Double Gumbel function parameters to adjust data.

4. Conclusions

Double Gumbel function optimization through PSO and maximum likelihood for data analysis is a promising strategy that combines classic techniques with advances in field of machine learning. By using PSI, we can explore search space more efficiently and find optimal solutions in a shorter time. In addition, by applying the maximum likelihood approach, we can obtain more accurate estimates double Gumbel function parameters

By leveraging the power of AI, we can further enhance this process and obtain more accurate and reliable results. The use of supervised machine learning algorithms allows us to train models that can predict the optimal parameters of the double Gumbel function from previously observed data, which accelerates the optimization process and improves efficiency.

This approach leverages classic techniques strengths such as PSO and maximum likelihood, and combines them with the advances in the field of machine learning. By using supervised machine learning algorithms, we can train models that can predict optimal parameters, thereby accelerating the optimization process and improving efficiency. Overall, this strategy holds great promise for enhancing data analysis and achieving more accurate results.

References

[1] González-Villarreal, Fernando. *Contribution to the analysis of frequencies of extreme values of maximum flows in a river. / Contribución al análisis de frecuencias de valores extremos de los gastos máximos en un río*. Blue Series / Serie Azul, Engineering Institute / Instituto de Ingeniería, UNAM, 1970.

[2] Fuentes Mariles, Óscar A., M. L. Arganis Juárez, R. Domínguez Mora, G. E. Fuentes Mariles, and K. Rodríguez Vázquez. "Maximization of the Likelihood Function of Probability Distributions Using Genetic Algorithms." / "Maximización de la función de Verosimilitud de Distribuciones de Probabilidad usando Algoritmos Genéticos." *Ingeniería Del Agua* 19, no. 1 (January 2015): 17–29.

[3] Rossi, Fabio, Mauro Fiorentino, and Pasquale Versace. "Two-Component Extreme Value Distribution for Flood Frequency Analysis." *Water Resources Research* 20, no. 7 (July 1984): 847-856.

[4] Eberhart, Russel C., and J. Kennedy. "A New Optimizer Using Particle Swarm Theory." Paper presented at the Sixth International Symposium on Micro Machine and Human Science MHS'95, Nagoya, Japan, October 4-6, 1995.

[5] Bing. "Chat Bing AI." 2023. [Available online]

[6] Anaconda. "Jupyter Notebook in Anaconda" / "Jupiter notebook en Anaconda." 2023. [Available online]

[7] Domínguez Mora, Ramón, E. Carrizosa Elizondo, and M.L. Arganis Juárez. *Study to regionalize the expenses generated by maximum floods, as a basis for the preparation of danger maps for river floods in all the basins of the Mexican Republic. Volume II. Regional statistical analysis of maximum annual rainfall. / Estudio para regionalizar los gastos generados por avenidas máximas, como base para la elaboración de mapas de peligro por inundaciones fluviales en todas las cuencas de la república Mexicana. Tomo II. Análisis estadístico regional de las precipitaciones máximas anuales.* IISCONV 103 2016 Project, UNAM, INSTITUTE OF ENGINEERING / Proyecto IISCONV 103 2016, UNAM, INSTITUTO DE INGENIERÍA. Mexico, March 2017.

[8] Jiménez-Espinoza, M. "Ax Program. Hydrometeorological Hazards Area."/ "Programa Ax. Área De Riesgos Hidrometeorológicos." National Disaster Prevention Center / Centro Nacional de Prevención de Desastres. Mexico, 1996.