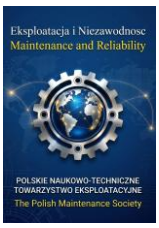

Eksploatacja i Niezawodnosc - Maintenance and Reliability

Volume 29 (2027), Issue 1

https://ein.org.pl

The Polish Maintenance Society

Software system reliability modeling with component dependence based on copula function and multi-index fusion

Yanchao Wang[1], Jianfeng Yang[1,2,*]

[1] School of Mathematics and Statistics, Nanning Normal University, China
[2] Center for Applied Mathematics of Guangxi, Nanning Normal University, China
* Corresponding Author: yangjianfeng@nnnu.edu.cn

Received: 24 June 2026
Revised: 3 July 2026
Accepted: 21 July 2026
Online: 13 August 2026

Abstract

Traditional software system reliability models typically assume that failures between subsystems are independent of each other, ignoring the potential fault dependencies that may exist in the actual system, this results in a biased assessment of the overall reliability of the system. In response to this issue, this paper proposes a dependent system reliability modeling method based on the Copula function and multi-index fusion. Firstly, by fitting the subsystem failure interval time with multiple life distribution models, and integrating multi-dimensional evaluation indicators by combining the rank-sum method, the optimal life distribution for each subsystem is determined. Secondly, the maximum information coefficient is introduced to quantify the fault dependency strength between subsystems, and identify pairs of subsystems that exhibit significant dependencies. On this basis, the Archimedean Copula function family is employed to model the dependence structure, parameters are obtained through maximum likelihood estimation, and the optimal Copula function is selected based on a comprehensive comparison of fitting evaluation indices. Empirical research based on real software system failure data shows that, compared to traditional independent assumption models, the proposed method improves the accuracy of system reliability assessment throughout the entire observation period, confirming that fault dependency modeling has a positive corrective effect on reliability assessment. Based on actual data, this study has constructed a comprehensive reliability evaluation framework encompassing life distribution fitting, dependency identification, and optimal Copula model selection. This framework provides a more practical reliability analysis tool for software systems exhibiting fault dependency characteristics, it has reference value for system maintenance decision-making and risk control.

Keywords: *software reliability model, failure dependency, Copula function, maximum information coefficient*

Article citation:
Wang Y, Yang J, Software system reliability modeling with component dependence based on copula function and multi-index fusion, Eksploatacja i Niezawodnosc – Maintenance and Reliability 2027: 29(1) http://doi.org/10.17531/ein/226580

Highlights

- A dependent software system reliability modeling based on Copula function and multi-index fusion.
- Optimal marginal lifetime distributions for each subsystem are determined using the rank-sum method.
- The optimal Copula function is selected via multiple goodness-of-fit criteria.
- The maximal information coefficient is employed to quantify the strength of failure dependence among subsystems.

1. Introduction

With the acceleration of global digital transformation and the comprehensive penetration of information technology, software systems have become an indispensable core pillar for social infrastructure operations and industrial iteration and upgrading. Their quality and reliability not only affect enterprise operational efficiency and service quality, but also directly relate to key infrastructure security, people's livelihood security, and even national security strategies. Software reliability, as a key quality attribute for evaluating the ability of software to consistently and correctly perform its intended functions under specific time and environmental conditions, is a core indicator for measuring software maturity and stability. Especially in safety-critical fields such as finance [1], communications [2], aerospace [3], and intelligent connected vehicles [4,5], software failures can lead to catastrophic consequences, causing huge economic losses and even casualties [6]. Against this backdrop, employing scientific methods to model, analyze, and predict software reliability has become a cutting-edge topic in the field of software engineering. By constructing precise mathematical models, not only can resource allocation be optimized and

potential defects be identified [4] during the development and testing stages, but also system behavior can be predicted and preventive maintenance can be guided during the operation and maintenance stages [7,8]. Thereby systematically enhancing the credibility and lifecycle value of software products, which holds profound theoretical significance and meets urgent practical demands.

Software reliability modeling research has evolved over decades, forming a rich ecosystem [3] centered on statistical models and data-driven methods. Early models primarily viewed software systems as a whole and established macro reliability growth curves based on their failure time data. Among these, Non-Homogeneous Poisson Process (NHPP) models and their numerous variants [9], such as the Goel-Okumoto model and the S-shaped growth model [6], are widely used due to their concise form and ease of interpretation, they describe the stochastic yet trending characteristics exhibited by the software fault detection process as testing and debugging proceed. To more accurately reflect the imperfections in the debugging process (such as the introduction of new defects), to further extend the NHPP model, scholars have proposed a reliability growth model that considers imperfect debugging [10,11].With the dramatic increase in system complexity, to depict the dynamic evolutionary process under uncertain environments, the modeling method based on stochastic differential equations (SDE) has been introduced, a framework for handling the effects of random fluctuations and external noise is provided [12]. Meanwhile, non-parametric methods, represented by local polynomial regression, it exhibits good adaptability due to not presupposing a specific function form [13]. In recent years, with the rise of big data and artificial intelligence technology, machine learning and deep learning models have also been attempted to learn from historical failure data and predict reliability trends [14]. However, such holistic models generally suffer from a fundamental limitation: they treat software as a "black box" and ignore the architectural characteristics of the system, which are composed of multiple cooperating components or subsystems. Therefore, researchers have proposed the idea of component-level reliability modeling, which involves first modeling the failure data of each subsystem separately and then integrating the reliability based on the system architecture, in order to obtain more refined evaluation results [15,16]. Regrettably, in order to reduce model complexity and computational difficulty, the vast majority of existing component-level models [17,18] are built upon the idealized assumption that "component failures are mutually independent", this is severely inconsistent with the reality of inter-component interactions and dependencies that are prevalent in modern complex software systems [7,19–21].

In fact, in modern software systems where modularization, servitization, and microservices architecture are increasingly prevalent, components or subsystems are tightly coupled through complex interface call chains, shared data states, concurrent resource competition, common runtime platforms, and other means. This coupling relationship may cause the failure of one component to trigger or accelerate the failure of other components, implying a propagation and correlation effect among faults. Ignoring such interdependence may lead to significant deviations in the estimation of overall system reliability (either underestimating risks or overestimating capabilities), thereby misleading testing strategies and maintenance decisions [7,22,23].To scientifically characterize this complex dependency, Copula theory is introduced into the field of system reliability engineering. The core advantage of the Copula function lies in its ability to separate the marginal distributions of multiple random variables from their rich joint dependence structure for modeling, thereby enabling it to flexibly describe various dependence patterns such as nonlinear, asymmetric, and tail correlations among variables. This tool has been successfully validated and applied in the reliability analysis of numerous engineering systems, for example: describing the correlation among multiple failure modes in mechanical systems [4,5,8,24–26]; Characterize the interaction between hardware faults and software faults in embedded systems with software and hardware cooperation [22]; In vehicle systems, correlate the degradation across the two dimensions of mileage and service time [27]; In civil structures, deal with the combined effects of multiple failure modes and loads [7,28–30]; In power systems, encompassing critical stages from power generation and transmission to distribution [23,26,31], and in manufacturing systems, analyzing the interdependent performance between different production stages [5]. In addition, the research has also extended to the following areas: by utilizing the survival feature analysis

method based on Copula functions, extending the theory of survival characteristics to systems with component interdependencies [32,33]; In specific fields such as battery safety [3], maritime safety [1], bridge engineering [29], the transitional Markov chain Monte Carlo algorithm [34], and building risk [35], Bayesian networks and Copula functions have significantly enhanced the quantification level of uncertainty in complex systems and the accuracy of risk prediction; Handling masked data [36]; Multivariate control charts [1]; Redundancy allocation optimization [37] and dynamic reliability assessment [8], among other more complex scenarios. The fruitful achievements in these fields have fully demonstrated the powerful capability and value of Copula functions in capturing and quantifying the internal dependence within complex systems [38–40].

Although the Copula approach has demonstrated great potential in general systems engineering, within the specific context of software system reliability modeling, existing research still has several gaps to be filled. Firstly, from a modeling perspective, a large body of reliability research on software systems still follows or defaults to the component independence assumption, lacking a standardized, operable, and data-driven complete analysis framework to systematically identify, quantify, and ultimately assess the actual extent of the impact of dependence on the overall reliability of the software system. Secondly, from the perspective of methodological application depth, the few existing software reliability studies that incorporate Copulas have primarily focused on demonstrating that introducing a dependence structure into the model can improve goodness-of-fit. However, the basis for selecting among different types of dependence structures (such as the Clayton, Gumbel, and Frank Copula families), as well as the mechanistic analysis of how varying dependence strength parameters quantitatively propagate and affect system-level reliability indicators, has not been sufficiently explored. This leads to insufficient model interpretability, making it difficult for engineers to select the most suitable dependence model according to the specific system architecture and failure characteristics.

Therefore, this paper aims to systematically address the aforementioned limitations by constructing a reliability assessment framework for software systems that starts from real failure data, features clear steps, and is applicable to software systems, the core objective is to quantitatively analyze the specific impact of failure dependence on system reliability. This paper will based on historical failure data from a real-world software system comprising multiple interacting subsystems as an empirical foundation, follow a logically rigorous four-stage analytical process to conduct the research: Phase 1 conducts optimal selection of component-level life distributions, determining the best-fitting marginal distribution for each subsystem; Phase 2 is dedicated to identifying and measuring dependencies, employing statistical methods to test and quantify the correlations among component failures; Phase 3 undertakes Copula function screening and comparison, selecting from mainstream Copula families the function that best characterizes the system's dependence structure and estimating its parameters; Ultimately, Phase 4 conducts a comprehensive assessment of system reliability and quantification of impact, comparing the differences in the overall system reliability function under the independence assumption versus the dependence model, and analyzing the influence of dependence on reliability. This study not only for processing software systems with dependent characteristics provides a set of from data to decision-making complete methodology, but also reveals, through empirical analysis, the specific mechanism by which dependence influences software system failures. Thereby providing more precise theoretical foundations and practical guidance for the architectural design of high-reliability software systems, determining the focus of integration testing, and risk management and control of operation and maintenance.

2. The review of Copula model

Traditional software reliability growth models are typically based on a core assumption: the failure processes of individual components or modules within a system are independent of each other. Under this assumption, the overall reliability of the system can be obtained by simple accumulation or superposition failure data of each independent component or reliability function. However, there is a significant gap between this idealized independence assumption and the operational mechanisms of most real-world software systems. In real-world complex systems, subsystems or components often interact by

sharing hardware resources, invoking common service libraries, coexisting in the same operating environment, or exhibiting tight logical and data coupling. This creates a situation where the failure of one component may trigger, accelerate, or expose defects in other components, thereby forming complex failure dependencies. Ignoring this inherent dependence will lead to systematic bias in the assessment of overall system reliability, underestimating the risk of cascading failures under stress; It may also result in misjudging reliability growth trends, thereby affecting the optimal allocation of testing resources and the accuracy of software release decisions.

To overcome the fundamental limitation imposed by the independence assumption, there is an urgent need for a mathematical tool capable of rigorously quantifying and modeling failure dependencies. Therefore, the introduction of Copula functions to handle fault dependencies among subsystems becomes critical.

2.1. Copula function

Copula function [41] is a model that connects one-dimensional marginal distribution functions with a multivariate joint distribution function to describe the correlation between variables. That is, for an n dimensional random vector $x = (x_1, x_2, \cdots, x_n)$, its n dimensional joint distribution function is denoted as:

$$F(x_1, x_2, \cdots, x_n) = C\big(F_1(x_1), F_2(x_2), \cdots, F_n(x_n)\big) \quad (1)$$

where, $F_1(x_1), F_2(x_2), \cdots, F_n(x_n)$ are the marginal distribution functions, C is the Copula function corresponding to F.

Let $u_1 = F_1(x_1), u_2 = F_2(x_2), \cdots, u_n = F_n(x_n)$, differentiating both sides of equation (1) yields the probability density function of the random vector x:

$$f(x_1, x_2, \cdots, x_n) = c\big(F_1(x_1), F_2(x_2), \cdots, F_n(x_n)\big) \prod_{i=1}^{n} f_i(x_i) \quad (2)$$

where, c is the probability density function of the copula function C:

$$c(u_1, u_2, \cdots, u_n) = \frac{\partial^n C(u_1, u_2, \cdots, u_n)}{\partial u_1 \partial u_2 \cdots \partial u_n} \quad (3)$$

2.2. Commonly used Copula functions

In the reliability analysis of software systems, the copula function is an important mathematical tool for quantifying and modeling the failure dependence among subsystems. In complex software systems, in particular, the failure processes of subsystems or components are often not independent but exhibit complex dependence structures, due to shared resources, a common operating environment, or logical invocation relationships. Among the various copula functions, the Archimedean copula has become a powerful choice for analyzing reliability problems in such software systems due to its clear construction principles, concise mathematical form, and flexibility in characterizing various dependence patterns. The three commonly used Archimedean copula functions are given as follows (Table 1).

Table 1. Some common Archimedean Copula functions.

Archimedean Copula	$C(u, v; \theta)$	Parameter θ
Clayton Copula	$\left(u^{-\theta} + v^{-\theta} - 1\right)^{\frac{-1}{\theta}}$	$\theta \in (0, +\infty)$
Gumbel Copula	$exp\left\{-\left[(-\ln u)^{\theta} + (-\ln v)^{\theta}\right]^{\frac{1}{\theta}}\right\}$	$\theta \in [1, +\infty)$
Frank Copula	$-\frac{1}{\theta} ln\left[1 + \frac{\left(e^{-\theta u} - 1\right)\left(e^{-\theta v} - 1\right)}{e^{-\theta} - 1}\right]$	$\theta \in (-\infty, +\infty)\backslash\{0\}$

To facilitate a more intuitive identification of the characteristics of different types of Copula functions, the definition of the tail dependence coefficient is provided as follows.

Let the marginal distribution functions of two random variables X and Y be $F(\cdot)$ and $G(\cdot)$, respectively. The upper-tail dependence coefficient λ_U and lower-tail dependence coefficient λ_L are then defined as follows:

$$\lambda_U = \lim_{u \to 1^-} P[Y > G^{-1}(u) | X > F^{-1}(u)], \ \lambda_U \in [0,1] \quad (4)$$

$$\lambda_L = \lim_{u \to 0^+} P[Y \leq G^{-1}(u) | X \leq F^{-1}(u)], \ \lambda_L \in [0,1] \quad (5)$$

Based on the tail dependence coefficient, the commonly used Archimedean Copula functions and elliptical Copula functions can be distinguished as follows:

Copula functions with equal upper- and lower-tail dependence coefficients: the Gaussian and t Copulas from the elliptical family, as well as the Frank Copula from the

Archimedean family. Among them, the Gaussian Copula and the Frank Copula have upper- and lower-tail dependence coefficients equal to 0, while the t Copula has equal upper- and lower-tail dependence coefficients.

Copula functions with unequal upper- and lower-tail dependence coefficients: the Gumbel and Clayton Copulas from the Archimedean family. Among them, the Gumbel Copula captures upper-tail dependence with its lower-tail dependence coefficient equal to 0, while the Clayton Copula captures lower-tail dependence with its upper-tail dependence coefficient equal to 0.

3. Software system reliability modeling with component dependence based on Copula functions

The proposed system reliability evaluation framework is illustrated in Figure 1, which consists of four core stages. First, marginal distribution fitting is performed for each subsystem individually, and goodness-of-fit metrics including AIC, BIC, the KS test, and its p-value are calculated. Distributions that do not meet the criteria are excluded based on the p-value, and the remaining candidate distributions are then comprehensively compared using the rank-sum method. Finally, the optimal distribution for each subsystem is selected. Second, the MIC method is employed to identify the fault dependencies among subsystems, and the candidate Copula models are determined based on the tail dependence characteristics of the dependent subsystem pairs. Then, the parameters of the candidate Copulas are estimated using maximum likelihood estimation, and the optimal Copula model is selected based on the AIC and BIC criteria. Finally, the joint reliability of the system is calculated and compared against the independent assumption as a baseline. The model is then validated by means of MAE, RMSE, and Bootstrap confidence bands.

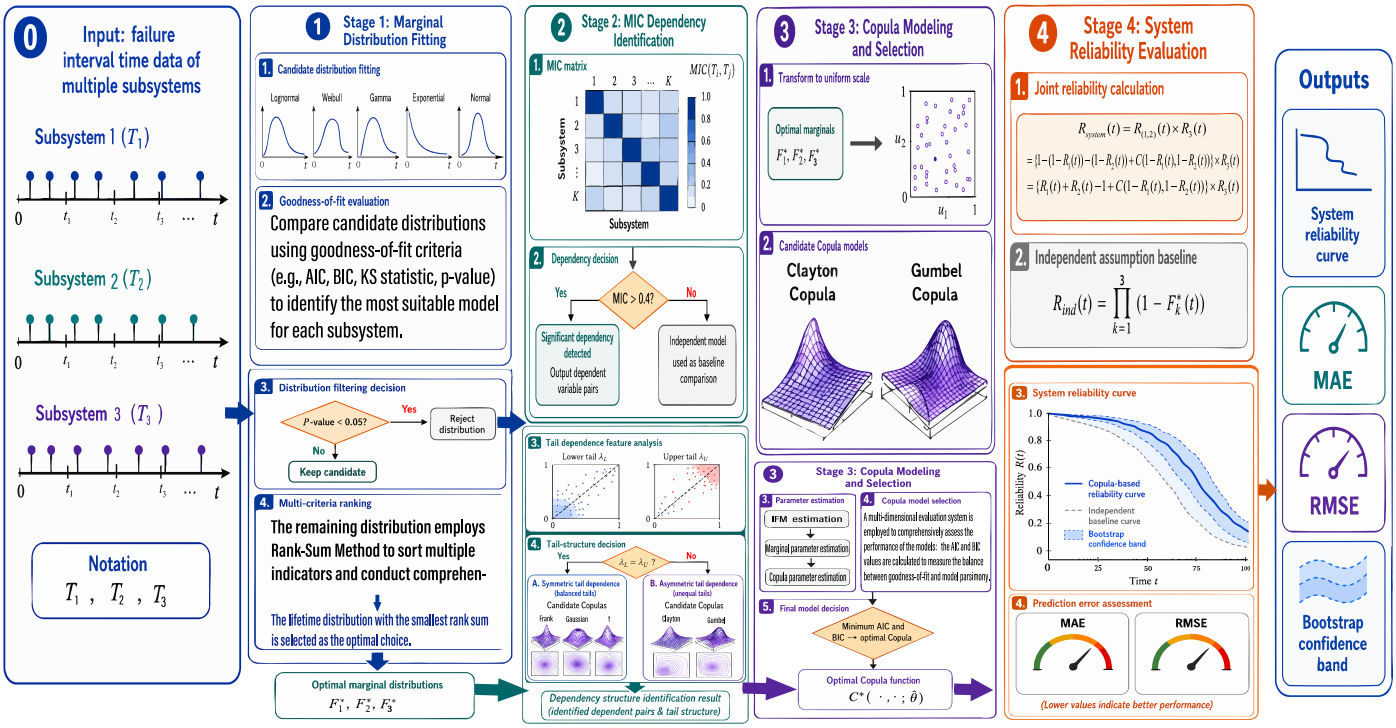

Figure 1. Flowchart of the system reliability evaluation.

The software system studied in this paper is composed of n subsystems, and each subsystem and the whole system obey the following basic assumptions.

Basic assumptions:

A. There are only two states for each subsystem and the entire software system: normal and failure.

B. Subsystems i and j, which are highly correlated in the software system, have failure dependence ($i \neq j$, $i,j = 1,2,\cdots,n$), while the failures of the remaining subsystems are independent of each other and do not affect one another.

C. The failure dependence between subsystems can be described using copula functions.

3.1. Software reliability modeling with failure dependence

Assume that system X consists of n subsystems connected in series, the failure-dependent subsystem is denoted as subsystem

X_1, the remaining subsystems, which are independent of each other, are considered as subsystem X_2, the lifetime of subsystem X_2 is denoted by T, $T_1, T_2, \cdots, T_n$ are the lifetimes of the n subsystems, respectively, $F_1, F_2, \cdots, F_n$ are the distribution functions of $T_1, T_2, \cdots, T_n$, respectively.

When subsystems i and j in a software system exhibit failure dependence and the failures of the remaining subsystems are independent of each other, the reliability function of the failure-dependent subsystem X_1 is denoted by $R_{i,j}(t)$:

$$R_{i,j}(t) = P(X_i > t, X_j > t) = 1 - P(X_i \le t) - P(X_j \le t) + P(X_i \le t, X_j \le t) \quad (6)$$

The above expression can be expressed using a copula function as:

$$R_{i,j}(t) = 1 - F_i - F_j + C(F_i, F_j) \quad (7)$$

while the failure processes of the remaining subsystems $F_l(l = 1, \cdots, n, l \neq i, j)$ are independent of each other, and its reliability function can be expressed as $R_l(t)$:

$$R_l(t) = P(T > t) = P(T_1 > t, T_2 > t, \cdots, T_n > t) = \prod_{l=1, l\neq i,j}^{n} P(T_l > t) = \prod_{l=1, l\neq i,j}^{n} (1 - F_l) \quad (8)$$

In summary, the system reliability function of the software system with a dependent structure, which consisting of n subsystems connected in series, can be expressed as:

$$R(t) = R_{i,j}(t) \times R_l(t) \quad (9)$$

3.2. Subsystem correlation evaluation with multi-index fusion

The reliability analysis of software system with multiple subsystems in this paper is based on the real failure data of software system, which belongs to the reliability evaluation driven by empirical research. The real fault data can effectively characterize the behavior of the system under complex working conditions, coupling logic and hidden fault conduction, the failure dependencies between components often present nonlinear, non-monotonic and even non-functional complex correlations. The failure dependence among components manifests directly as the correlation between their failure data, the strength of this correlation can be calculated based on the metrics in Table 2.

Commonly used metrics for correlation analysis include Pearson, Spearman, Kendall, and MIC. As shown in the following table (Table 2).

Table 2. Metrics for correlation analysis.

Metrics	Relationship Type	Data Requirements	Main Advantages	Main Limitations
Pearson	Linear relationship	Continuous, normally distributed data	Standard measure for linear relationships; intuitive interpretation.	Detects only linear relationships; sensitive to outliers.
Spearman	Monotonic relationship	Ordered data	Robust, normal distribution is not required.	Detects only monotonic relationships.
Kendall	Monotonic relationship	Ordered data	Better interpretation for small samples; intuitive measure of concordance.	Detects only monotonic relationships.
MIC	Wide range of functional relationships	No specific distributional requirements	Capable of detecting complex relationships.	Potentially prone to overfitting.

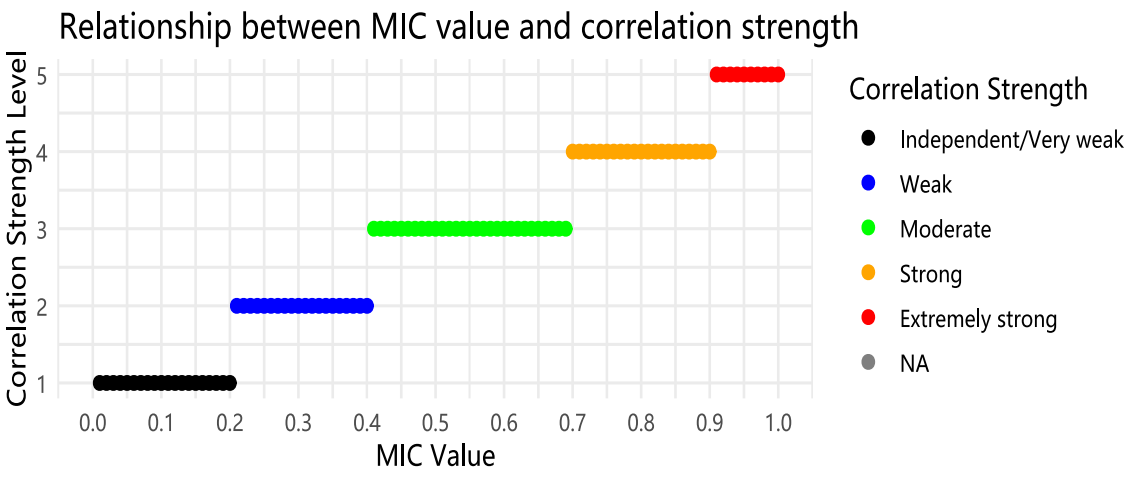

Figure 2. Criteria for determining correlation strength based on MIC values.

To accurately characterize the potentially complex nonlinear or non-monotonic failure dependence among subsystems in software system reliability analysis, traditional linear or monotonic correlation metrics (such as Pearson, Spearman, and Kendall) have limitations. The Maximum Information Coefficient (MIC) [42], due to its ability to capture a wide range of functional relationships (including both linear and nonlinear) and its flexible requirements regarding data distribution, has become a more suitable measure for such contexts.

Therefore, this paper selects MIC as the core metric for quantifying failure dependence. The calculation is implemented using R, and the resulting values serve as a key basis for determining whether subsystems are mutually independent or exhibit interdependence. If the failure dependence among software subsystems has been explicitly specified, it can also be mutually validated with the results of the MIC-based data-driven analysis, jointly supporting the subsequent copula-based dependence modeling.

The Maximum Information Coefficient (MIC) [43] is a statistical measure widely applied across various fields, with its core concept rooted in mutual information. Unlike traditional correlation coefficients, MIC is committed to detecting and quantifying a wide range of functional correlations between variables (including linear, nonlinear, non-monotonic, etc.), and providing a standardized correlation strength measure.

4. Parameter estimation methods and model evaluation criteria

In the modeling of interdependence, The parameter estimation of Copula function is a key step to quantify the failure correlation strength between components. Maximum Likelihood Estimation (MLE) is one of the most commonly used methods for this type of problem, which obtains the optimal parameter values by maximizing the joint likelihood function of the sample. For bivariate copula models, due to the relatively low dimensionality of the parameters, maximum likelihood estimation generally exhibits good feasibility and statistical efficiency. Therefore, this paper employs maximum likelihood estimation to estimate the parameters of the bivariate copula function that characterizes the failure dependence between subsystems, aiming to accurately capture the underlying dependence pattern.

4.1. Parameter estimation

4.1.1. Parameter estimation for marginal distributions

For independent identically distributed samples$x_1, x_2, \cdots, x_n$, its probability density function is$f(x;\theta)$, where θ is an unknown parameter.

Construct the likelihood function:

$$L(\theta; x_1, x_2, \cdots, x_n) = \prod_{i=1}^{n} f(x_i;\theta) \tag{10}$$

Take the logarithm:

$$l(\theta) = ln\, L\,(\theta) = \sum_{i=1}^{n} ln\, f\,(x_i;\theta) \tag{11}$$

Differentiate the above expression to obtain the derivative of the log-likelihood function:

$$u(\theta) = \frac{\partial l(\theta)}{\partial \theta} \tag{12}$$

By setting $u(\theta) = 0$, $\hat{\theta}$ can be obtained.

4.1.2. Parameter estimation of Copula functions

Let the joint distribution function of random variables (X, Y) be $F(x, y)$, with the corresponding joint probability density function $f(x, y)$. Let $F_1(X)$ and $F_2(Y)$ be the marginal distribution functions of X and Y, respectively, with corresponding marginal density functions $f_1(x)$ and $f_2(y)$. If the copula function is denoted by $C(u, v)$, its density function is $c(u, v)$.

The joint probability density function can be obtained as:

$$f(x, y) = c\big(F_1(X), F_2(Y)\big) f_1(x) f_2(y) \tag{13}$$

where, $c(u, v) = \frac{\partial^2 C(u,v)}{\partial u \partial v}$.

Let the sample of random variables (X, Y) be $\{(X_i, Y_i), i = 1,2, \cdots, n\}$, then, the likelihood function can be expressed as:

$$L(\theta) = \prod_{i=1}^{n} c\big(F_1(X_i), F_2(Y_i)\big)\, f_1(x_i) f_2(y_i) \tag{14}$$

where n is the sample size.

Taking the logarithm of Equation (12), the log-likelihood function is obtained as:

$$\ln L(\theta) = \sum_{i=1}^{n} \ln c\big(F_1(X_i), F_2(Y_i)\big) + \sum_{i=1}^{n} \ln f_1(x_i) + \sum_{i=1}^{n} \ln f_2(y_i) \tag{15}$$

Taking the partial derivative of the log-likelihood function in Equation (13) with respect to θ and setting it equal to zero:

$$\frac{\partial\, ln\, L(\theta)}{\partial \theta} = 0 \tag{16}$$

Solving this equation finally yields the maximum likelihood estimate of the unknown parameter θ:

$$\hat{\theta} = argmax\, L\,(\theta) \tag{17}$$

4.2. Model evaluation metrics

A. AIC（Akaike Information Criterion）

Its formula is given by:

$$AIC = 2K - 2\,ln(L) \tag{18}$$

where K is the number of parameters in the model; L is the maximum likelihood value of the model.

A smaller AIC value indicates a better balance between goodness-of-fit and model complexity. When fitting copula functions, the model with the smallest AIC value is selected, as it is considered the best for capturing the underlying dependence structure in the data.

B. BIC（Bayesian Information Criterion）

Its formula is given by:

$$BIC = K\,ln(n) - 2\,ln(L) \tag{19}$$

where K is the number of parameters in the model, n is the sample size, and L is the maximum likelihood value.

In the selection of copula functions, a smaller BIC value indicates a better model. The model with the smallest BIC is chosen as the optimal one.

C. KS Test（Kolmogorov–Smirnov Test）

The formula for the KS test is given by:

$$D_n = \sup_x |F_n(x) - F(x)| \tag{20}$$

where $F_n(x)$ is the empirical distribution function of the sample, $F(x)$ is the cumulative distribution function of the theoretical distribution, and sup denotes the supremum.

The closer the value of D_n is to 0, the closer the empirical distribution is to the theoretical distribution; The larger the value of D_n, the greater the discrepancy between them.

4.3. Rank-sum method for lifetime distribution selection

Assume that the software system has n subsystems denoted by $F_1, F_2, \cdots, F_n$, with candidate lifetime distributions to be fitted given by $A_1, A_2, \cdots, A_k$, and the evaluation metrics for fitting denoted by $B_1, B_2, \cdots, B_m$.

To fit the lifetime distributions $A_1, A_2, \cdots, A_k$ to each subsystem $F_1, F_2, \cdots, F_n$, and to obtain their values under the m evaluation metrics $B_1, B_2, \cdots, B_m$, the procedure is as follows.

For each subsystem $F_i(i = 1, \cdots, n)$ and each candidate lifetime distribution $A_j(j = 1, \cdots, k)$, calculate the rankings for every metric $B_q(q = 1, \cdots, m)$ sequentially. The best-performing distribution under a given metric receives a rank of 1, the second-best receives a rank of 2, and so on, until the worst-performing distribution receives a rank of j. The ranks of each lifetime distribution across all m metrics are then summed to obtain its comprehensive rank sum:

$$S_{rank} = R_{B_1} + R_{B_2} + \cdots + R_{B_m} \tag{21}$$

where R_{B_1}、R_{B_2}、…、R_{B_m} represent the ranks of the lifetime distribution under the corresponding metrics, respectively.

Therefore, for each metric $B_q(q = 1, \cdots, m)$, a rank is assigned to each lifetime distribution $A_j(j = 1, \cdots, k)$. The ranks of each distribution across all m metrics are then summed, yielding the rank sum $S_{rank} = \sum_{p=1}^{m} R_p$. The lifetime distribution with the smallest rank sum is selected as the optimal choice.

5. Case study

5.1. Real software failure data

This paper utilizes real-world failure data from a software system consisting of multiple components. The data were obtained from Apache Tomcat, an open-source software project. Tomcat is a core project within the "Jakarta" project of the Apache Software Foundation and serves as a free, open-source web application server. The data are sourced from https://bz.apache.org/bugzilla/query.cgi; Specifically, failure data were collected from the Tomcat 7 software system, spanning from February 2010 to September 2020 (recorded on a daily basis). Tomcat 7 contains three subsystems, denoted as F_1 for the Documentation subsystem, F_2 for the Packaging subsystem, and F_3 for the Manager subsystem. Among them, subsystem F_1 has 73 failure records, subsystem F_2 has 51 failure records, and subsystem F_3 has 59 failure records.

5.2. Fitting results of subsystem lifetime distributions

For each of the subsystems-Documentation, Packaging, and Manager-this paper calculates the values of three evaluation metrics, namely AIC, BIC, and the KS test, obtained by fitting five candidate lifetime distributions: the Log-normal distribution, Weibull distribution, Gamma distribution, exponential distribution, and Normal distribution.

For each metric, a rank is assigned to every fitted lifetime distribution. The ranks of each distribution across all three metrics are then summed, yielding the rank sum $S_{rank} = \sum_{i=1}^{3} R_i$. The lifetime distribution with the smallest rank sum is

selected as the optimal choice.

Table 3. Failure data of subsystem documentation.

NO.	Time between failures	NO.	Time between failures	NO.	Time between failures	NO.	Time between failures
1	71.36	19	4.47	37	6.62	55	20.71
2	8.68	20	17.18	38	7.00	56	5.52
3	86.86	21	3.93	39	28.80	57	45.61
4	42.79	22	62.52	40	1.98	58	52.48
5	16.68	23	11.57	41	106.76	59	11.93
6	23.66	24	42.05	42	21.92	60	3.92
7	17.21	25	1.94	43	22.26	61	25.76
8	13.02	26	13.00	44	14.15	62	437.80
9	7.99	27	23.38	45	19.68	63	33.29
10	47.86	28	42.88	46	0.92	64	194.13
11	1.43	29	9.82	47	2.19	65	4.17
12	8.73	30	5.60	48	65.72	66	48.09
13	55.89	31	7.03	49	23.22	67	88.18
14	20.37	32	14.80	50	12.85	68	4.29
15	48.61	33	4.54	51	23.20	69	318.99
16	2.15	34	12.00	52	11.29	70	673.46
17	13.84	35	1.18	53	36.99	71	521.92
18	41.54	36	1.46	54	82.72	72	220.63

Table 4. Failure data of subsystem packaging.

NO.	Time between failures	NO.	Time between failures	NO.	Time between failures	NO.	Time between failures
1	140.10	14	64.04	27	16.19	40	6.81
2	127.06	15	14.26	28	31.40	41	203.89
3	20.33	16	8.60	29	89.91	42	39.02
4	15.12	17	53.25	30	57.20	43	192.08
5	38.75	18	2.77	31	42.11	44	939.34
6	162.07	19	5.81	32	174.90	45	84.66
7	17.95	20	2.49	33	36.02	46	103.05
8	18.84	21	0.95	34	1.28	47	0.66
9	2.19	22	25.93	35	89.54	48	149.96
10	25.92	23	6.99	36	69.22	49	329.52
11	94.43	24	19.85	37	10.92	50	58.85
12	23.52	25	6.33	38	5.10		
13	20.27	26	55.94	39	12.11		

Table 5. Failure data of subsystem manager.

NO.	Time between failures	NO.	Time between failures	NO.	Time between failures	NO.	Time between failures
1	36.70	16	30.60	31	13.10	46	12.12
2	6.04	17	32.19	32	21.37	47	56.25
3	1.29	18	170.05	33	39.08	48	144.81
4	53.07	19	44.01	34	9.71	49	61.93
5	53.99	20	2.02	35	21.25	50	184.20
6	15.83	21	7.53	36	46.51	51	269.98
7	10.23	22	87.51	37	43.26	52	133.90
8	10.57	23	49.76	38	11.94	53	329.96
9	1.26	24	65.08	39	6.24	54	1.33
10	70.83	25	152.64	40	17.63	55	62.59
11	13.99	26	22.23	41	15.30	56	182.23
12	42.20	27	67.91	42	9.90	57	124.44
13	49.82	28	29.02	43	175.15	58	536.13
14	5.28	29	8.87	44	40.10		
15	10.26	30	41.94	45	0.49		

Table 6. Fitting of lifetime distributions for subsystem documentation.

Distribution Type	Parameters	Log-likelihood	AIC	BIC	KS	P-value (KS)
Log-normal	μ =2.9181,σ = 1.4457	-338.8102	681.6204	686.1737	0.0575	0.9601
Weibull	k = 0.6701,λ = 38.7038	-346.5348	697.0695	701.6229	0.1122	0.3022
Gamma	α = 0.5660,β = 0.0102	-351.5227	707.0453	711.5986	0.1641	0.0367
Exponential	λ = 0.0180	-361.3814	724.7629	727.0396	0.2787	0
Normal	μ = 55.6558,σ = 115.6640	-444.2132	892.4265	896.9798	0.3187	0

5.2.1. Fitting results of lifetime distributions

From the table below, the fitting results of lifetime distributions for subsystem Documentation can be obtained. It can be observed from the p-values of the KS test that only the log-normal distribution and the Weibull distribution have p-values greater than 0.05. Therefore, these distributions can be considered to fit the failure interval data of subsystem Documentation well, indicating no significant difference between the data and the distributional assumptions, and thus

the fitting performance is satisfactory.

In the following analysis, only the log-normal distribution and the Weibull distribution are evaluated using the rank-sum method.

According to the ranking results in the table below, the log-normal distribution has a comprehensive rank of 4, and the Weibull distribution has a comprehensive rank of 5. The log-normal distribution has the smallest comprehensive rank, therefore, it is ultimately selected as the lifetime distribution for subsystem Documentation, with parameters μ =2.9181, σ = 1.4457.

Table 7. Rank sum for subsystem documentation.

Distribution Type	R_{AIC}	R_{BIC}	R_{KS}	S_{rank}
Log-normal	2	1	1	4
Weibull	1	2	2	5

From the table below, the fitting results of lifetime distributions for subsystem Packaging can be obtained. It can be observed from the p-values of the KS test that only the log-normal distribution, Weibull distribution, and Gamma distribution have p-values greater than 0.05. Therefore, these distributions can be considered to fit the failure interval data of subsystem Packaging well, indicating no significant difference between the data and the distributional assumptions, and thus the fitting performance is satisfactory.

In the following analysis, only the log-normal distribution, Weibull distribution, and Gamma distribution are evaluated using the rank-sum method.

According to the ranking results in the table below, the log-normal distribution has a comprehensive rank of 4, the Weibull distribution has a comprehensive rank of 5, and the Gamma distribution has a comprehensive rank of 9. The log-normal distribution has the smallest comprehensive rank, therefore, it is ultimately selected as the lifetime distribution for subsystem Packaging, with parameters μ =3.2956, σ =1.5508.

Table 8. Fitting of lifetime distributions for subsystem packaging.

Distribution Type	Parameters	Log-likelihood	AIC	BIC	KS	P-value (KS)
Log-normal	μ =3.2956,σ =1.5508	-257.6641	519.3282	523.1522	0.0694	0.9561
Weibull	k =0.7077,λ =57.0940	-258.5809	521.1618	524.9858	0.0644	0.9770
Gamma	α =0.6086,β =0.0082	-260.4079	524.8185	528.6398	0.0976	0.6909
Exponential	λ = 0.0134	-265.4389	532.8778	534.7898	0.2056	0.0249
Normal	μ =74.3495,σ =140.3320	-318.1475	640.2949	644.1190	0.2997	0.0002

Table 9. Rank sum for subsystem packaging.

Distribution Type	R_{AIC}	R_{BIC}	R_{KS}	S_{rank}
Log-normal	1	1	2	4
Weibull	2	2	1	5
Gamma	3	3	3	9

Table 10. Fitting of lifetime distributions for subsystem manager.

Distribution Type	Parameters	Log-likelihood	AIC	BIC	KS	P-value (KS)
Log-normal	μ =3.3278,σ = 1.4561	-297.1041	598.2082	602.3291	0.0922	0.6737
Weibull	k =0.7856,λ = 55.7736	-296.6131	597.2262	601.3471	0.0923	0.6716
Gamma	α =0.7137,β = 0.0110	-297.5149	599.0387	603.1596	0.1114	0.4362
Exponential	λ = 0.0154	-300.0168	602.0335	604.0940	0.1455	0.1550
Normal	μ =64.8906,σ =92.1903	-344.6820	693.3640	697.4849	0.2674	0.0004

From the table below, the fitting results of lifetime distributions for subsystem Manager can be obtained. It can be observed from the p-values of the KS test that the log-normal distribution, Weibull distribution, Gamma distribution, and exponential distribution all have p-values greater than 0.05. Therefore, these distributions can be considered to fit the failure interval data of subsystem Manager well, indicating no significant difference between the data and the distributional assumptions, and thus the fitting performance is satisfactory.

In the following analysis, only the log-normal distribution, Weibull distribution, Gamma distribution, and exponential distribution are evaluated using the rank-sum method.

According to the ranking results in the table below, the log-normal distribution has a comprehensive rank of 5, the Weibull distribution has a comprehensive rank of 4, the Gamma distribution has a comprehensive rank of 9, and the exponential distribution has a comprehensive rank of 12. The Weibull distribution has the smallest comprehensive rank, therefore, it is ultimately selected as the lifetime distribution for subsystem Manager, with parameters$k$ =0.7856, λ = 55.7736.

Table 11. Rank sum for subsystem manager.

Distribution Type	R_{AIC}	R_{BIC}	R_{KS}	S_{rank}
Log-normal	2	2	1	5
Weibull	1	1	2	4
Gamma	3	3	3	9
Exponential	4	4	4	12

5.3. Correlation analysis and optimal Copula function

Using R software, the MIC values between each pair of subsystems in Table 12 are calculated. The MIC value ranges from 0 to 1, with larger values indicating stronger failure dependence (see Figure 1). From the MIC results, it can be found that F_1 and F_2 exhibit a relatively high correlation, with an MIC value above 0.5. The correlation between F_2 and F_3 is moderate, while that between F_1 and F_3 is the lowest, with MIC values around 0.33 for both F_3 with F_1 and F_2. Therefore, it can be preliminarily concluded that, in this failure dataset, subsystem F_1 (Documentation) and subsystem F_2 (Packaging) exhibit failure dependence. Based on this judgment, the failure dataset can be further modeled and analyzed for reliability using a corresponding failure-dependent software reliability model.

Table 12. MIC value for each pair of subsystems.

pair of subsystems	MIC Value
(F_1, F_2)	0.5060
(F_1, F_3)	0.3371
(F_2, F_3)	0.3397

After calculating the MIC values for each pair of subsystems and preliminarily determining the failure dependence between subsystems, this study further calculates the tail dependence coefficient between the dependent subsystem pair, Documentation and Packaging. The results show that the upper-tail dependence coefficient is 0.2, while the lower-tail dependence coefficient approaches 0, indicating a significant asymmetric tail dependence structure.

Based on this characteristic, elliptical Copulas such as the Gaussian Copula (with no tail dependence) and the t-Copula (with symmetric tail dependence) are theoretically incapable of capturing asymmetric tail dependence structures. Therefore, Copula functions with symmetric tail dependence are excluded, and Copulas with asymmetric tail dependence characteristics are selected as candidate models. However, although the Frank Copula has both upper- and lower-tail dependence coefficients equal to 0, as a member of the Archimedean family, it is capable of capturing the overall correlation between variables. In this study, it is retained as a reference for the case with no tail dependence tendency, so as to more comprehensively compare the effects of different dependence structures on system reliability.

Accordingly, this paper selects three types of Copula functions (Clayton, Gumbel, and Frank) based on the characteristics of the data to determine the optimal Copula function for the dependent subsystems. First, the maximum likelihood estimation method is used to fit these three Copula functions, obtaining the corresponding parameter estimates θ. Subsequently, a multi-dimensional evaluation system is employed to comprehensively assess the performance of the models: the AIC and BIC values are calculated to measure the balance between goodness-of-fit and model parsimony.

From the Copula fitting results in the table below, it can be seen that the Gumbel Copula function has the smallest AIC and BIC values among the three Copulas. Therefore, the Gumbel Copula function is selected as the optimal Copula model for subsystems Documentation and Packaging.

Table 13. Copula Fitting Results for (F_1, F_2).

Copula Type	Parameterθ	Log-likelihood	AIC	BIC
Gumbel	1.1617	0.7812	0.4376	2.3496
Frank	1.1030	0.6950	0.6100	2.5220
Clayton	0.2812	0.6481	0.7037	2.6158

5.4. System reliability analysis

5.4.1. System reliability function

From Equation (5), the reliability of the dependent subsystems is given by $R_{(1,2)}(t) = 1 - F_1 - F_2 + C(F_1, F_2)$. Then, as indicated by Equation (7), when subsystem F_1 and subsystem F_2 exhibit failure dependence, the reliability function of the overall system $R_{system}(t)$ is:

$$R_{system}(t) = R_{(1,2)}(t) \times R_3(t) = \{1 - F_1 - F_2 + C(F_1, F_2)\} \times R_3(t) = \{1 - (1 - R_1(t)) - (1 - R_2(t)) + C(1 - R_1(t), 1 - R_2(t))\} \times R_3(t) = \{R_1(t) + R_2(t) - 1 + C(1 - R_1(t), 1 - R_2(t))\} \times R_3(t) \quad (22)$$

When the failures of the subsystems are considered to be independent of each other, the system reliability function is given by:

$$R(t) = R_1(t)R_2(t)R_3(t) \quad (23)$$

where:

$F_1(t) = \Phi\left(\frac{ln(t)-\mu_1}{\sigma_1}\right)$,$R_1(t) = 1 - F_1(t) = 1 - \Phi\left(\frac{ln(t)-\mu_1}{\sigma_1}\right)$,$\mu_1$=2.9181,$\sigma_1$=1.4457.

$F_2(t) = \Phi\left(\frac{ln(t)-\mu_2}{\sigma_2}\right), R_2(t) = 1 - F_2(t) = 1 - \Phi\left(\frac{ln(t)-\mu_2}{\sigma_2}\right)$, μ_2=3.2956, σ_2=1.5508.

$F_3(t) = 1 - exp\left[-\left(\frac{t}{\lambda}\right)^k\right], R_3(t) = exp\left[-\left(\frac{t}{\lambda}\right)^k\right]$, k =0.7856, λ =55.7736.

The copula function is given by:

$$C(u, v; \theta) = exp\left\{-\left[(-\ln u)^\theta + (-\ln v)^\theta\right]^{\frac{1}{\theta}}\right\}, \theta = 1.1617.$$

5.4.2. Analysis of results

Based on the reliability function model constructed in Section 5.4.1, this section presents a comparative analysis of the software system's reliability evaluation results under two different assumptions. Through calculation and visualization, the reliability functions of the overall system and its individual subsystems over time under the Gumbel Copula model（θ = 1.1617）are obtained, as shown in Figure 3.

As can be observed from Figure 3, by comparing the subsystems, it is evident that the reliability of F_1 and F_2 are the clear weak points of the system. Moreover, their combined reliability decays more rapidly due to the failure dependence between them, which directly dominates the overall system failure. Therefore, the weak points of the system are mainly concentrated in F_1 and F_2 . When conducting preventive maintenance, redundancy design, or quality improvement, resources should be allocated preferentially to F_1 and F_2, as the benefits will be far greater than optimizing F_3, whose reliability is already high. A differentiated maintenance strategy should be adopted, implementing high-frequency, preventive monitoring and maintenance for F_1 and F_2, while for F_3, condition-based maintenance or a longer inspection cycle can be applied, thereby allocating operational and maintenance resources more effectively.

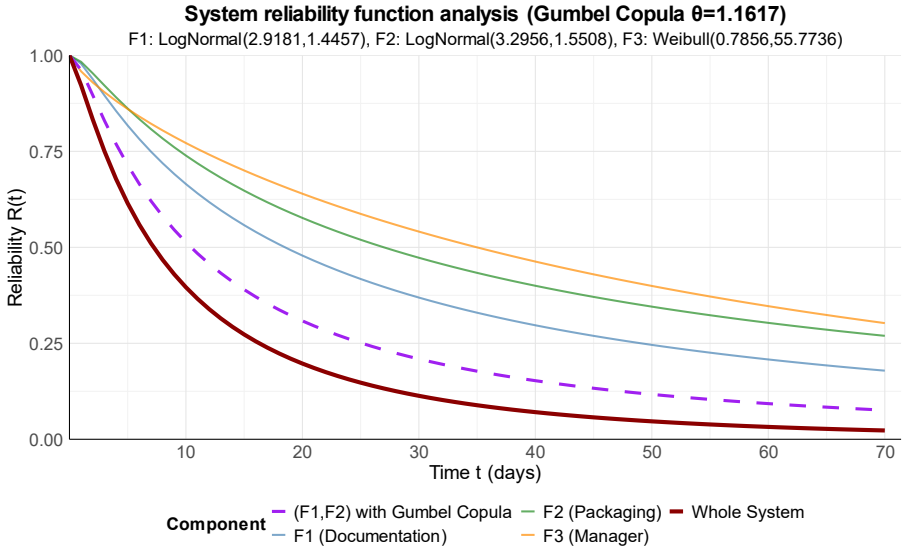

Figure 3. Reliability function curves of the system and its components.

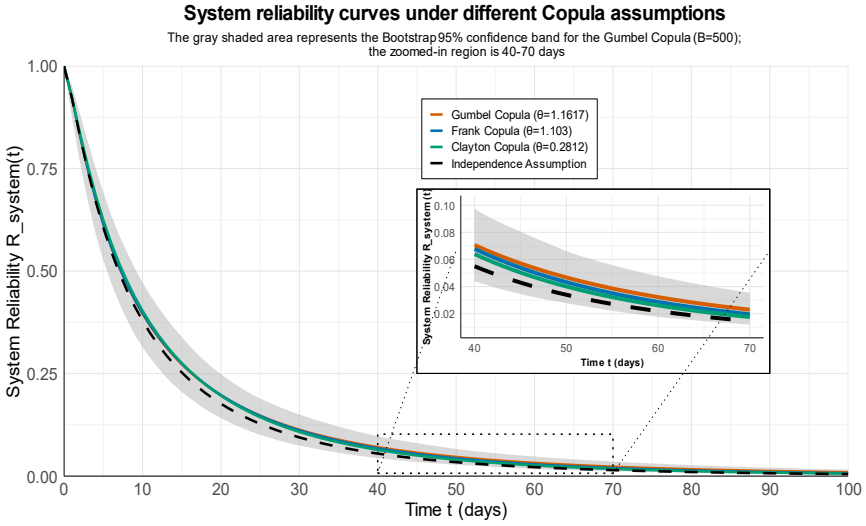

Figure 4. Comparison of system reliability under different Copula assumptions and the independence assumption.

The overall system reliability drops below 0.25 by day 40, indicating that after a very short "useful life period", the system quickly enters a wear-out phase with a high failure rate. The design of this system may not be suitable for production environments requiring long-term continuous operation. The expected service life of the system should be clearly defined (e.g., 30 days), and mandatory version updates, service restarts, or system replacement processes should be planned around this cycle, rather than attempting to maintain its operation indefinitely. For example, by day 10, the reliability of the software system has already dropped below 0.9, suggesting that software engineers should promptly repair system faults; otherwise, the software may fail and become unavailable.

Figure. 4 intuitively presents the differences in system reliability evaluation under four assumptions. Among them, the orange solid line represents the system reliability curve based on the Gumbel Copula considering the upper-tail dependence among subsystems, the blue solid line represents the reliability curve under the Frank Copula assumption, the green solid line represents the reliability curve under the Clayton Copula assumption, and the black dashed line represents the system reliability curve under the assumption that all subsystem failures are completely independent of each other.

As shown in Figure. 4, throughout the entire observed time domain, the ranking of the four curves remains stable: the Gumbel Copula curve is always at the top, followed by the Frank Copula curve, the Clayton Copula curve lies below the first two, and the independence assumption curve is always at the bottom. Among them, the differences between Gumbel, Frank, Clayton and the independence assumption decrease successively, indicating that the selection of Copula functions has a significant impact on the system reliability evaluation results.

It is particularly noteworthy that the difference between the Gumbel Copula and the independence assumption is the most significant, and the area enclosed between them clearly characterizes the deviation in reliability evaluation results between the two extreme scenarios of "considering upper-tail dependence" and "completely ignoring dependence". This graphical relationship strongly demonstrates that, in this case, when the Gumbel Copula model is introduced to characterize the failure dependence among subsystems, the evaluated overall system reliability is not lower than that obtained under the traditional independence assumption at any time, and is significantly higher than the latter during most of the time period.

The reason for this result is that the Gumbel Copula can accurately capture the dependence and upper-tail dependence among subsystems. Although the Frank Copula can describe the overall correlation, its tail dependence coefficient is 0, making it unable to capture extreme tail co-movement. The Clayton Copula, despite having tail dependence, exhibits lower-tail dependence, which is inconsistent with the upper-tail dependence characteristics of the present data. The ranking of the goodness-of-fit of the three Copulas is completely consistent with the ranking of the reliability curves, further verifying the rationality of the Copula selection.

To quantitatively verify the necessity of considering dependence, this paper calculates the mean absolute error (MAE) and root mean square error (RMSE) of the Gumbel Copula model and the traditional independent model in predicting the cumulative number of failures. As shown in Table 14, the MAE and RMSE of the Gumbel Copula model are 2.9608 and 3.3693, respectively, both significantly lower than those of the independent model (3.0275 and 3.4473). This result indicates that considering the failure dependence among subsystems can effectively improve the fitting accuracy of system reliability evaluation, quantitatively verifying the superiority of the proposed dependence model.

This dependence structure reflects a common scenario in real-world engineering: due to shared operating environments, common loads, or potential common-cause failure modes, the health status of one component has a positive influence on that of another. The traditional independence assumption completely ignores the inherent statistical associations among subsystems, which may lead to an overly pessimistic estimation of the system state when computing the joint system reliability. In contrast, the Gumbel Copula incorporates this positive correlation into the model through its mathematical structure, thereby yielding a more optimistic and generally more realistic reliability evaluation that closely reflects the actual operational mechanisms of complex systems.

Table 14. Comparison of prediction accuracy for the cumulative number of system failures under different models.

Model	MAE	RMSE
Gumbel Copula	2.9608	3.3693
Independence Model	3.0275	3.4473

6. Conclusion

This study establishes a reliability assessment framework for software systems based on actual failure data. To address the limitations of the traditional independence assumption, a reliability analysis method integrating optimal lifetime distribution selection via multiple metrics and Copula-based failure dependence modeling is proposed. Empirical results indicate that, for the software system analyzed in this study, the optimal lifetime distributions for the failure behavior of subsystem are the Weibull distribution and the log-normal distribution, which better capture the time-varying characteristics of their failure rates. When characterizing the failure dependence between subsystems, the optimal Copula function is identified as the Gumbel Copula, with its parameter =1.1617 effectively quantifying the “upper-tail dependence” risk between the two subsystems. Compared to the traditional independence model, the advantage of the established failure dependence model lies in its ability to significantly correct the overoptimistic estimation of system reliability by the independent assumption model, thereby providing early warnings of more credible failure risks. It is recommended that preventive maintenance or inspections be implemented for this software system with a cycle shorter than 30 days to effectively control the risks.

This study provides a comprehensive analytical framework for the case of binary failure dependence, yet there remains room for further extension and refinement. First, high-dimensional dependence modeling represents an important direction for future research. This paper only addresses the dependence between two subsystems; For complex software systems involving multiple interrelated subsystems, it is necessary to introduce structures such as multivariate copulas or vine copulas to finely characterize the multivariate failure dependence network. Second, the dynamic adaptability of model parameters and maintenance strategies needs to be enhanced. Future work could explore the integration of time-varying copulas with methods such as reinforcement learning, enabling the model to adaptively adjust based on online monitoring data and dynamically optimize maintenance scheduling, thereby striking a precise balance between reliability assurance and operational cost.

Funding

This work was supported by Guangxi Natural Science Foundation (2025GXNSFAA069686, 2025GXNSFBA069559), National Natural Science Foundation of China (72361008), and Guangxi University Young and Middle-aged Teachers' Research Basic Ability Improvement Project of Nanning Normal University (No. KY26ZQN028).

References

1. Wang J, Fan H, Wu Z, Wang N, Chang Z, Lyu J. Navigating straits/canals safety: a data-driven Copula Bayesian network risk assessment framework. *Reliability Engineering & System Safety* 2026; 270: 112157. https://doi.org/10.1016/J.RESS.2025.112157.
2. Huang J, Yang J, Zheng Z, Qiu Z. A novel probabilistic modeling for multilateral random attacks in cyber-physical system reliability analysis. *Quality and Reliability Engineering International* 2024; 40(5): 2620-2637. https://doi.org/10.1002/qre.3533.
3. Meng H, Hu M. Data-driven copula Bayesian network for risk analysis of lithium-ion battery accidents. *Reliability Engineering & System Safety* 2026; 271: 112074. https://doi.org/10.1016/J.RESS.2025.112074.
4. Oszczypała M, Konwerski J, Ziółkowski J, Małachowski J. Copula-based reliability analysis of vehicles based on censored failures data using reliability importance measures. *IEEE Access* 2024; 12: 154119-154137. https://doi.org/10.1109/ACCESS.2024.3450076.
5. Maihulla A S., Yusuf I., Khoo M B C. Reliability and performance analysis on stages to control manufacturing system using Copula technique. *Life Cycle Reliability and Safety Engineering* 2023; 12(3): 197-205. https://doi.org/10.1007/S41872-023-00224-8.
6. Zheng Z, Yang J, Hu Y, Wang X. Open-source Software Reliability Modeling with Stochastic Impulsive Differential Equations. *Eksploatacja i Niezawodność – Maintenance and Reliability* 2023; 25(2): 166342. https://doi.org/10.17531/EIN/166342.
7. Liu W, Li A, Chen E J, Luo H, Wang Y. A novel reliability analysis method for a dependent system by copula model: a case study in

operation tunnels maintenance. *Journal of Civil Structural Health Monitoring* 2022; 12(5): 1133-1155. https://doi.org/10.1007/S13349-022-00581-5.

8. Gao S, Li J, Zhang Y, Li T. Dynamic reliability modeling of gear transmission system considering multiple failure modes based on copula functions. *Journal of Mechanical Science and Technology* 2025; 39(9): 5131-5142. https://doi.org/10.1007/S12206-025-0818-9.
9. Lai R, Garg M. A Detailed Study of NHPP Software Reliability Models *Journal of Software* 2012; 7(6): 1296-1306. https://doi.org/10.4304/jsw.7.6.1296-1306.
10. Wang J, Wu Z. Study of the nonlinear imperfect software debugging model. *Reliability Engineering & System Safety* 2016; 153: 180-192. https://doi.org/10.1016/j.ress.2016.05.003.
11. Huang Y S, Chiu K C, Chen W M. A software reliability growth model for imperfect debugging. *Journal of Systems and Software* 2022; 188: 111267. https://doi.org/10.1016/J.JSS.2022.111267.
12. Yang J, Ding M, He M, Zheng Z, Yang N. SDE-based software reliability additive models with masked data using ELS algorithm. *Journal of King Saud University - Computer and Information Sciences* 2024; 36(3): 101978. https://doi.org/10.1016/J.JKSUCI.2024.101978.
13. Zhou W, Yang J, Chen J, Huang J. The Additive Reliability Model of Cyber Physical Systems Based on Local Polynomial Regression with Masked Failure Data. *Eksploatacja i Niezawodność – Maintenance and Reliability* 2026; 28(1): 1-16. https://doi.org/10.17531/ein/207794.
14. Yang R, Sun L, Li H, Yang Y. A neural network copula function approach for solving joint basic probability assignment in structural reliability analysis. *Quality and Reliability Engineering International* 2024; 40(6): 3096-3119. https://doi.org/10.1002/QRE.3568.
15. Xie M, Goh T N. System reliability growth analysis using component failure data. *International Journal of Reliability, Quality and Safety Engineering* 1994;1(1): 71-83. https://doi.org/10.1142/S0218539394000076.
16. Fiondella L, Rajasekaran S, Gokhale S S. Efficient software reliability analysis with correlated component failures. *IEEE Transactions on Reliability* 2013; 62(1): 244-255. https://doi.org/10.1109/TR.2013.2241131.
17. Haque M A, Ahmad N. A Software Reliability Growth Model Considering Mutual Fault Dependency. *Reliability: Theory & Applications* 2021; 16(2): 222-229. https://doi.org/10.24412/1932-2321-2021-262-222-229.
18. Zhao M, Yang J, Zhao B, Wu Z. Copula-based reliability modelling of wireless sensor networks with dependent failures. *International Journal of Sensor Networks* 2019; 31(2): 90-98. https://doi.org/10.1504/IJSNET.2019.102185.
19. Zhang L, Yan R. Copula-based reliability estimation of parallel-series system in the multicomponent stress–strength dependent model. *Probability in the Engineering and Informational Sciences* 2025; 39(3): 309-325. https://doi.org/10.1017/S0269964824000263.
20. Li X, Zhan K, Xiong X, Vu T Q. A copula-based reliability model for phased mission systems with dependent components. *Quality and Reliability Engineering International* 2023; 39(5): 1533-1547. https://doi.org/10.1002/qre.3342.
21. Li Y, Dong Y, Guo H. Copula-based multivariate renewal model for life-cycle analysis of civil infrastructure considering multiple dependent deterioration processes. *Reliability Engineering & System Safety* 2023; 231: 108992. https://doi.org/10.1016/J.RESS.2022.108992.
22. Zheng Z, Yang J, Huang J. Software-hardware embedded system reliability modeling with failure dependency and masked data. *Computers & Industrial Engineering* 2023; 186: 109746. https://doi.org/10.1016/J.CIE.2023.109746.
23. Wen J, Li X, Xue J. Feasibility evaluation of Copula theory for substation equipment with multiple nonlinear-related seismic response indexes. *Reliability Engineering & System Safety* 2024; 247: 110132. https://doi.org/10.1016/J.RESS.2024.110132.
24. He C, Wu W, Meng Q. A Copula-based Mechanical System Reliability Model and Its Application. *Acta Armamentarii* 2012; 33(3): 379-384. https://dx.doi.org/10.3969/j.issn.1000-1093.2012.03.022.
25. Deng Z, Huang T, Qian H, Zeng Y, Huang H. Reliability Analysis of Gear Under Multiple Failure Modes Based on Time-Dependent Mixed Copula Function. *Quality and Reliability Engineering International* 2025; 41(8): 3415-3429. https://doi.org/10.1002/qre.70061.
26. Wang Y, Yan Z. Statistical inference on accelerated life testing with dependent competing failure model under progressively type-II censored data based on copula theory. *Quality and Reliability Engineering International* 2020; 37(4): 1396-1408. https://doi.org/10.1002/qre.2803.
27. Gu Y, Fan C, Liang Q, Zhang J. Reliability calculation method based on the Copula function for mechanical systems with dependent failure . *Annals of Operations Research* 2022; 99-116. https://doi.org/10.1007/s10479-019-03202-5.
28. Li Q, Zhang T. Research on the Reliability of Bridge Structure Construction Process System Based on Copula Theory. *Applied Sciences*

2022; 12(16): 8137. https://doi.org/10.3390/app12168137.

29. Sheng X, Lin C, Zheng W, Yang Y, Zhu Z. Joint values of temperature and wind actions on long–span bridges based on mixed copula with Bayesian selection approach. *Reliability Engineering & System Safety* 2026; 265: 111626. https://doi.org/10.1016/j.ress.2025.111626.
30. Xu Y, Liu L. Copula-based conditional reliability analysis of slopes in spatially variable soils. *Reliability Engineering & System Safety* 2026; 265: 111522. https://doi.org/10.1016/j.ress.2025.111522.
31. Meng X, Tian L, Li C, Liu J. Copula-based wind-induced failure prediction of overhead transmission line considering multiple temperature factors . *Reliability Engineering & System Safety* 2024; 247: 110138. https://doi.org/10.1016/j.ress.2024.110138.
32. Zheng Y, Zhang Y. Reliability analysis for system with dependent components based on survival signature and copula theory. *Reliability Engineering & System Safety* 2023; 238: 109402. https://doi.org/10.1016/j.ress.2023.109402.
33. Vineshkumar B, Nair N U. Inferring association from reliability functions: An approach based on copulas. *Brazilian Journal of Probability and Statistics* 2021; 35(3): 484-498. https://doi.org/10.1016/10.1214/20-BJPS491.
34. Ma P, Zhang Y, Cai E, Luo M, Guo J, Guo T.A copula-based transitional markov chain monte carlo method for bayesian model updating. *Reliability Engineering & System Safety* 2026; 265: 111572. https://doi.org/10.1016/j.ress.2025.111572.
35. Zheng X W, Li H N, Gardoni P. Hybrid Bayesian-Copula-based risk assessment for tall buildings subject to wind loads considering various uncertainties. *Reliability Engineering & System Safety* 2023; 233: 109100. https://doi.org/10.1016/j.ress.2023.109100.
36. Yang J, Zhao M, Chen J. ELS algorithm for estimating open source software reliability with masked data considering both fault detection and correction processes. *Communications in Statistics - Theory and Methods* 2022; 51(19): 6792-6817. https://doi.org/10.1080/03610926.2020.1866610.
37. Li Z, Coit D W. Reliability Modeling and Redundancy Allocation for Systems of Dependent Components Using Copula Functions . *IISE Annual Conference Proceedings* 2024; 1-6. https://doi.org/10.21872/2024IISE_8375.
38. Jia X, Wang L, Wei C. Reliability Research of Dependent Failure Systems Using Copula. *Communications in Statistics - Simulation and Computation* 2014; 43(8): 1838-1851. https://doi.org/10.1080/03610918.2013.800879.
39. Elshoubary E E, Radwan T. Studying the Efficiency of the Apache Kafka System Using the Reduction Method, and Its Effectiveness in Terms of Reliability Metrics Subject to a Copula Approach. *Applied Sciences* 2024; 14(15): 6758. https://doi.org/10.3390/app14156758.
40. Yang J, Ding M, Chen J, Gu Z. Reliability Assessment Modeling for Star Wireless Sensor Network Based on Copula Function and k/n(G) Model . *Quality and Reliability Engineering International* 2026; 42(4): 1537-1548. https://doi.org/10.1002/qre.70152.
41. Nelsen R B. *An Introduction to Copulas*. Springer Series in Statistics. New York: Springer; 1999.
42. Cao D, Chen Y, Chen J, Zhang H, Yuan Z. An improved algorithm for the maxi mal information coefficient and its application. *Royal Society Open Science* 2021; 8(2): 201424. https://doi.org/10.1098/rsos.201424.
43. Reshef D N, Reshef Y A, Finucane H K, Grossman S R, McVean G, Turnbaugh P J, et al. Detecting Novel Associations in Large Data Sets. *Science* 2011; 334(6062): 1518-1524. https://doi.org/10.1126/science.1205438.