

Analysis of the empirical complexity of Advanced Encryption Standards - 128 statistically

Animesh Kumar [†]

Prashant Pranav ^{*}

Sandip Dutta [§]

Department of Computer Science and Engineering

Birla Institute of Technology

Mesra

Ranchi 835215

Jharkhand

India

Soubhik Chakraborty [‡]

Department of Mathematics

Birla Institute of Technology

Mesra

Ranchi 835215

Jharkhand

India

Abstract

Algorithm analysis stands as a cornerstone in the field of theoretical computer science, captivating researchers with its focus on dissecting and understanding the operational complexity of various algorithms. This intricate process is pivotal for acquiring deeper insights into the practical performance of algorithms, beyond what is discernible from theoretical analysis alone. A relatively novel and increasingly relevant approach in this realm is the concept of empirical complexity. This method diverges from traditional theoretical analysis by placing emphasis on the practical execution of an algorithm across a spectrum of varying and escalating input sizes. Such a hands-on approach is instrumental in predicting the complexity of the algorithm in real-world scenarios. The empirical data thus obtained undergoes rigorous statistical analysis, enabling a more nuanced and concrete understanding of the algorithm's behavior and performance. This approach not only enriches

[†] E-mail: animeshkumarcse@gmail.com

^{*} E-mail: prashantpranav19@gmail.com (Corresponding Author)

[§] E-mail: sandipdutta@bitmesra.ac.in

[‡] E-mail: soubhikc@yahoo.co.in

our comprehension of algorithms but also plays a crucial role in optimizing and refining their application in practical computing environments.

- **Background:** In the realm of digital security, cryptographic algorithms are indispensable for protecting applications and safeguarding sensitive information, be it of national significance or personal confidentiality. The efficiency of these algorithms in contemporary, high-velocity computational environments is paramount. Such algorithms typically incorporate intricate mathematical operations. This study delves into the computational intricacies of AES-128, a predominant cryptographic algorithm, scrutinizing its computational complexity.
- **Method:** This study applies a thorough empirical complexity framework to establish the asymptotic limits of AES-128 accurately. It then conducts a detailed regression analysis with closed boundaries to measure AES-128's empirical complexity with precision.
- **Result:** The investigation reveals that the empirical complexity of AES-128 adheres to an $O(n)$ notation, where 'n' represents the size of the input data.
- **Conclusion:** The research establishes that AES-128's time complexity for both encryption and decryption processes aligns with $O(n)$, a finding supported through both empirical evidence and theoretical analysis. Furthermore, a statistical model has been crafted to accurately forecast the execution time based on any specified input size for AES-128's encryption and decryption activities. This advancement aids in predicting performance outcomes prior to the actual encoding process.

Subject Classification: 94A60, 94A62, 62M10, 11T71.

Keywords: Statistical analysis, Empirical complexity, Cryptography, AES-128, Algorithmic analysis.

1. Introduction

The main aim of analyzing an algorithm is to get an insight into the real working of the algorithm if deployed for use in any application. Such analysis is pivotal in determining the appropriateness of algorithms for various computational tasks, and it aids in contrasting them against their counterparts. More profoundly, this analytical process unveils deeper insights into the workings of algorithms, paving the way for optimization. This optimization often involves substituting complex components with more streamlined ones, thereby enhancing the overall efficacy and compactness of the algorithm.

Two primary methodologies are prevalent in computing complexity: the minimization of asymptotic execution time and the curtailment of operational requirements for specific methodologies. In practical scenarios, the complexity of an algorithm is typically gauged by counting the minimal number of operations necessary for its execution. While 'big O' notation delineates the worst-case scenarios as input sizes escalate,

empirical computational complexity emerges as a contemporary approach to ascertain real-time performance. This is achieved by executing the algorithm on computational systems and monitoring its runtime complexity across diverse input dimensions, thereby molding observed performance into a statistical model that forecasts performance relative to input size.

In the sphere of cryptographic algorithms, two aspects are paramount: their proficiency in securing confidential data and their efficiency in time management. The Advanced Encryption Standard (AES), known for its symmetric key architecture which utilizes a singular key for both encryption and decryption, stands as a paragon in data security algorithms. Subsequent sections will offer a concise overview of AES-128, delving into its computational and empirical computational complexities. The empirical analysis of AES is advantageous in several facets, which include but are not limited to aspects some of which are elaborated below:

- Providing tangible predictions regarding the algorithm's performance for longer message lengths.
- In scenarios where an algorithm operates in linear time, such as $O(n)$, it becomes imperative to scrutinize not just the dependent and independent variables but also the constants, as exemplified in the case of AES.
- The proposed statistical model estimates the runtime of AES – 128 prior to its implementation in a real-world application.

2. Related Work

An essential resource on this topic is detailed in a comprehensive volume [2], [3], which advocates for the application of linear regression techniques in the statistical measurement of algorithmic complexity. Complementing this, another study [4] undertakes a comparative analysis of two compounding operations to evaluate their statistical similarities and differences. A pioneering work [5] presents a groundbreaking method to establish that the complexity order of multiplying two dense $n \times n$ matrices is $O(n^2)$.

To gain a comprehensive grasp of statistical methods such as regression analysis, probability distributions, and correlation, it's beneficial to consult several pivotal texts, as cited in references [6], [7], [8], and [9]. Additionally, a crucial paper [1] thoroughly explores the topic of

empirical computational complexity within algorithms, while another significant publication [10] delves into methods for approximating empirical complexity through computer-based techniques.

The development and nuances of the Advanced Encryption Standard (AES) are thoroughly outlined in a publication by the National Institute of Standards and Technology (NIST) [11], with the initial proposal to NIST detailed in [12]. Within the field of cryptography, a novel algorithm named SSLC, which utilizes the Poisson probability distribution for managing large datasets with orderly and evenly distributed keys, is introduced in [13]. Another discussion [14] addresses the computational hurdles in achieving subjective utility maximization, emphasizing the importance of considering diverse objectives such as time and space efficiency alongside the adaptation to different hardware capabilities [15].

Moving beyond conventional discussions on computational complexity, a particular study [16] suggests that future theories of decision-making should account for the biological resource limitations of decision-makers caused by computational demands. Regarding feature extraction algorithms, a thorough theoretical and empirical analysis [17] evaluates the quality of features extracted by different algorithms, with a focus on their statistical relevance. Yet, there's a noted lack of information about the performance of various models before their practical application [18], highlighting a field in need of further.

3. Advanced Encryption Standards (AES)

Based on the number of key used, cryptographic algorithms are fundamentally categorized into two distinct types. Asymmetric-key algorithms, exemplified by RSA, utilize two separate keys for encryption and decoding operations. In contrast, symmetric key algorithms, such as the Data Encryption Standards (DES), Blowfish, and Advanced Encryption Standards (AES), utilize the same key for both encryption and decryption. AES specifically operates by encrypting data in 128-bit blocks and is available in three key length variations: 128 bits, 192 bits, and 256 bits, correspondingly leading to the AES-128, AES-192, and AES-256 standards. During its operational phases, AES treats all data uniformly across each encryption or decryption cycle, with AES-128 being the most prevalently employed variant in the current digital landscape.

The development of AES was anchored in three pivotal criteria:

- Resilience: The algorithm must be robust against all recognized forms of cryptographic attacks.
- Simplicity: The design of the algorithm should be straightforward and unambiguous.
- Efficiency: It should demonstrate high-speed performance and compactness in its code across various computing platforms.

Figure 1 illustrates the general operational mechanics of AES. It begins with a single 128-bit block, termed the input matrix, which is integral to both the encryption and decryption processes.

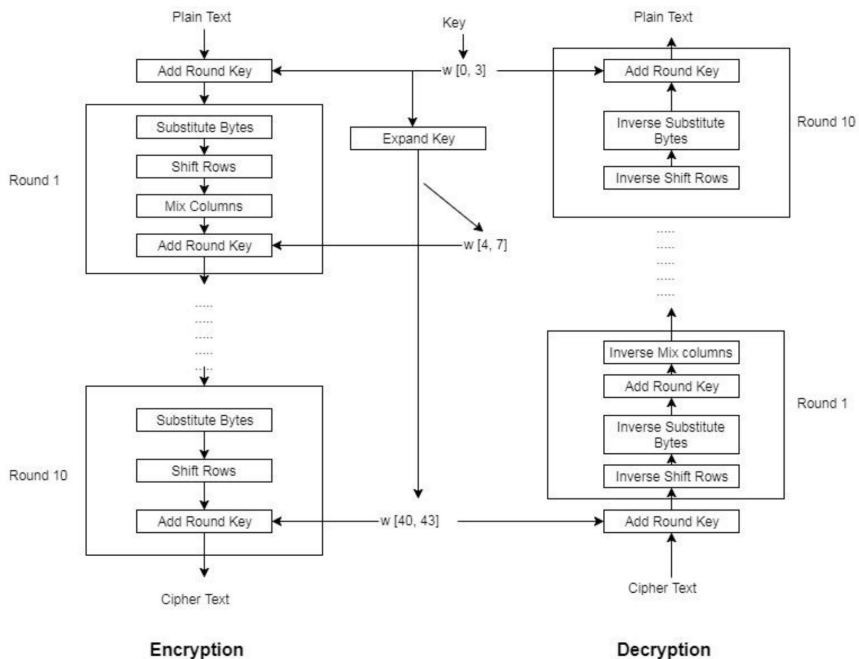


Figure 1
Outer Working of AES – 128

3.1 AES-128 Each Round Inner Working Mechanism

The AES-128 encryption algorithm initiates with an AddRoundKey stage and proceeds through a series of substages for encryption and decryption. During the first nine rounds, AES employs four substages,

whereas the tenth round comprises three substages. For decryption, the algorithm inverts each of these substages. The four substages for encryption are:

Substitute Bytes (SubBytes): This phase involves a table lookup using an S-box, a 16x16 array of bytes. The AES developers meticulously crafted the S-box to be resilient against all known cryptanalytic attacks, focusing on minimizing the correlation between input and output bits. This was achieved by carefully controlling the linear transformations and differential propagation probabilities. The inverse of this process for decryption utilizes a reverse S-box.

Shift Rows Transformation (ShiftRows): In this simple yet crucial stage, the first row of the state remains unchanged, while the second row is circularly shifted one byte to the left, the third row two bytes, and the fourth row three bytes. The decryption process involves the inverse operation, shifting rows in the opposite direction.

Mix Column: The MixColumns phase is an advanced substitution step that utilizes the arithmetic principles of the Galois Field, specifically $GF(2^8)$. In this phase, the algorithm views the state matrix in four distinct columns. It then remaps each byte within these columns to a new byte, using a predetermined function. This remapping involves intricate multiplications and additions within the confines of $GF(2^8)$, adding a layer of complexity to the process.

Add Round Key Transformation: In the concluding phase, a bitwise XOR operation is performed, merging the 128 bits of the current state with the 128 bits of the round key. This, along with extra procedures carried out in the ninth and tenth rounds, forms the entirety of the AES-128 encryption and decryption methodologies.

3.2 AES-128 Computational Complexity

Within the realm of block cipher algorithms like AES-128, which operate on a standard block size of 128 bits, the duration needed for both encryption and decryption tends to remain stable regardless of the size of the input. This aspect gives AES-128 a time complexity of $O(1)$ for its encryption and decryption activities, indicating consistent execution time. Nevertheless, for inputs that exceed 128 bits, it becomes necessary to divide these inputs into multiple blocks. These segments are then processed using operational modes such as CBC or CTR. In these instances, the total number of blocks that need to be encrypted or decrypted is proportional to $O(n)$, with 'n' indicating the size of the message. Therefore,

when considering the use of AES-128 in conjunction with any standard mode of operation, its overall time complexity transitions to $O(n)$, showcasing a linear relationship with the size of the message.

3.3 AES-128 Empirical Computational Complexity

Empirical complexity, or Empirical O , offers a real-time perspective on an algorithm's complexity, grounded in experimental evaluations of practical and significant bounds. Building on our preceding research [19], this study intensifies the examination of the empirical complexity of AES-128, focusing on each intermediate round to discern the most critical step.

4. Materials and Methods

Expanding upon the findings in [19], this investigation delves into the empirical complexity of the complete AES-128 algorithm. The methodology involves executing the AES-128 algorithm (in CBC mode) and fitting the results to a statistical model, thus gaining a pragmatic insight into its empirical complexity. This was conducted using MATLAB version 2015a on a system equipped with an Intel Core i3 processor, 4.00 GB of RAM, and running a 64-bit Windows 7 operating system.

In exploring how the algorithm performs with different sizes of input, this research conducted numerous tests for both the encryption and decryption phases, using messages of the same length but with different content. The purpose was to calculate an average runtime from these tests to pinpoint the algorithm's execution duration for each activity. This approach, involving a series of tests, was crucial to neutralize the impact that cache memory could have on the observed runtime. Specifically, executing the same message multiple times might lead to cached accesses, thus distorting the genuine runtime figures. The findings from this extensive testing are summarized in Tables 1 and 2, providing a reliable set of data for further examination.

The relationship between the length of a message (x) and its corresponding execution time (y) in the context of encryption or decryption using AES-128 is analyzed. The goal is to establish a function f such that $y = f(x)$. The empirical estimation of the runtime is approached through the application of the Fundamental Theorem of Finite Difference, specifically its inverse principle. This theorem posits that if the n^{th} difference for the tabulated function remains constant and higher order differences approach zero, while the function can be characterized as an n^{th} degree polynomial.

Table 1
AES-128 Encryption Process Runtime

Messages as Input (in bit)	Attempt 1	Attempt 2	Attempt 3	Attempt 4	Attempt 5	Execution Mean Time
16 bits	0.01610	0.0315	0.0332	0.0373	0.0338	0.03038
32 bits	0.04988	0.0645	0.0612	0.0652	0.0625	0.06065
48 bits	0.08384	0.0952	0.0925	0.0985	0.0925	0.09250
64 bits	0.11725	0.1474	0.1265	0.1269	0.1289	0.12939
80 bits	0.15155	0.1659	0.1895	0.1589	0.1520	0.16357
96 bits	0.18504	0.1805	0.1975	0.1689	0.1980	0.18598
112 bits	0.21814	0.1568	0.2345	0.2196	0.1848	0.20276
128 bits	0.25245	0.2578	0.2685	0.2398	0.2250	0.24871
144 bits	0.28635	0.2896	0.2895	0.2789	0.2358	0.27603
160 bits	0.32126	0.3545	0.2895	0.3125	0.4365	0.34285
176 bits	0.35465	0.3265	0.3688	0.3365	0.3789	0.35307
192 bits	0.38789	0.4688	0.3795	0.3456	0.4235	0.40105
208 bits	0.42145	0.4458	0.3875	0.4895	0.4895	0.44675
224 bits	0.45539	0.3589	0.5652	0.4156	0.5645	0.47191
240 bits	0.48978	0.5889	0.4785	0.4256	0.6297	0.52249
256 bits	0.52325	0.4580	0.5245	0.5369	0.4678	0.50209

Table 2
AES-128 Decryption Process Runtime

Messages as Input	Attempt 1	Attempt 2	Attempt 3	Attempt 4	Attempt 5	Execution mean Time
16 bits	0.17212	0.2456	0.2178	0.2158	0.1895	0.20816
32 bits	0.33666	0.3526	0.3695	0.3578	0.3562	0.35455
48 bits	0.50120	0.4912	0.4456	0.4856	0.4645	0.47762
64 bits	0.6662	0.7628	0.6956	0.6896	0.7215	0.70714
80 bits	0.83169	0.7615	0.7395	0.8510	0.7305	0.78283
96 bits	0.99744	0.8789	0.9894	0.8620	0.9754	0.94062
112 bits	1.16186	1.0568	1.0802	1.1582	1.1120	1.11381
128 bits	1.32472	1.3685	1.4365	1.3156	1.1897	1.32700
144 bits	1.49164	1.8862	1.7896	1.3931	1.3207	1.57624

Contd...

160 bits	1.65378	1.9689	1.5321	1.6098	1.8369	1.72029
176 bits	1.81386	1.6305	1.5902	1.5785	1.7473	1.67207
192 bits	1.98528	2.4123	1.7356	1.7532	2.3861	2.05449
208 bits	2.14972	2.0763	1.8702	2.1856	2.6209	2.18054
224 bits	2.31268	1.8892	2.1902	1.9621	2.5235	2.17553
240 bits	2.47912	2.6759	2.1136	2.0856	2.7741	2.42566
256 bits	2.64382	2.7319	3.1350	2.4269	2.2456	2.63664

Table 3
Encryption Time and Decryption Time (Empirical Estimate)

Estimation for Encryption			Estimation for Decryption		
x	y	Δy	X	y	Δy
16	0.03038	0.029 >	16	0.20816	0.16 >
32	0.06065		32	0.35455	
48	0.09250	0.036 >	48	0.47762	0.25 >
64	0.12939		64	0.70714	
80	0.16357	0.046 >	80	0.78283	0.25 >
96	0.18598		96	0.94062	
112	0.20276	0.040 >	112	1.11381	0.21 >
128	0.24871		128	1.32700	
144	0.27603	0.020 >	144	1.57624	0.14 >
160	0.34285		160	1.72029	
176	0.35307	0.045 >	176	1.67207	0.51 >
192	0.40105		192	2.05449	
208	0.44675	0.075 >	208	2.18054	0.09 >
224	0.47191		224	2.17553	
240	0.52249	0.061 >	240	2.42566	0.24 >
256	0.50209		256	2.63664	

Fundamental Theorem of Finite Difference

The theorem is expressed as follows:

$$\Delta^n P_n(x) = C = \text{Constant}$$

The n^{th} difference of a n^{th} degree of the polynomial is Constant.

$$\Delta^{n+r} P_n(x) = 0, r = 1, 2, 3, \dots$$

Indicating that higher differences (beyond the polynomial's degree) of a polynomial of degree n are zero.

This characteristic is distinctly observed in polynomial functions. By reviewing the data in Table 3, it was noted that the initial differences were almost consistent, leading to the inference that a linear (first-degree) polynomial model would closely estimate the execution durations for both encryption and decryption in AES-128. This insight effectively positions the empirical complexity of these AES-128 operations at a practical $O(n)$ level.

Building on this insight, a statistical model was designed to encapsulate this identified empirical complexity accurately. This model is aimed at reliably forecasting the execution times as a function of the message size, offering an invaluable mechanism for gauging AES-128's efficiency in practical applications.

5. Result and Discussion

The empirical methodology adopted in this study entails an analysis based on specific datasets, representing a subset of the broader universe of possible inputs. To extrapolate our findings to this wider universe and mitigate the risk of overfitting, a linear regression approach is utilized. This choice is particularly apt considering the nature of the relationship between the independent variable (the input's length or message size) and the runtime of AES-128, which is observed to be linear. It's vital to note that even though a transformation was applied to a predictor related to the independent variable (squared term), the functional form employed for fitting the model remains fundamentally linear.

5.1 *Statistical Model Calibration Based on Empirical Data for AES-128*

To achieve the most accurate fit for the empirical data gathered on AES-128, a first-degree polynomial of the form $y = a + bx'$ was employed, guided by the principle of least squares. This method seeks to minimize the discrepancies between observed values and those predicted by the model. The coefficients 'a' and 'b' in the polynomial are crucial for this model, and they are determined through a set of equations designed to align the model as closely as possible with the observed data. These coefficients are derived from the empirical data and are instrumental in formulating a model that effectively encapsulates the relationship between the message size and its execution time in AES-128 encryption and

decryption processes. The fitting of this statistical model to the empirical results serves as a robust tool for predicting the performance of AES-128, providing valuable insights for applications where encryption and decryption times are critical factors:

$$\Sigma y = na + b\Sigma x'$$

$$\Sigma x'y = a\Sigma x' + b\Sigma x'^2 \quad (1)$$

These equations are obtained by setting

$$\frac{\delta s}{\delta a} = 0 \text{ and } \frac{\delta s}{\delta b} = 0$$

Where $S = \Sigma(y - \hat{y})^2$ represents the estimated residual sum of squares:

$$\hat{y} = \hat{a} + \hat{b}x'$$

When solving equation 1 we get

$$\hat{a} = -0.01910 \text{ and } \hat{b} = 0.003221 \text{ (in case of encryption) and}$$

$$\hat{a} = 0.00868 \text{ and } \hat{b} = 0.01106 \text{ (in case of decryption)}$$

Now,

$$\hat{y} = -0.01910 + 0.003221 x' \text{ (in case of encryption) and}$$

$$\hat{y} = 0.00868 + 0.01106 x' \text{ (in case of decryption)} \quad (2)$$

The role of a residual table, presented as Table 4, in evaluating the fit of the statistical model developed for AES-128. The residual table is a critical tool in assessing the model's accuracy and effectiveness.

The evaluation of an algorithm's performance extends beyond merely understanding its complexity. It also involves comprehending the associated constants that are revealed through an empirical model. These constants, statistically fitted, contribute significantly to the empirical notation of the algorithm. In this context, for a competing algorithm addressing the same problem, such as Blowfish, the knowledge of its distinct associated constants enables a direct comparison. This comparative analysis is essential for understanding the relative efficiencies and performance metrics of different algorithms under similar conditions.

Moreover, the empirical model facilitates the prediction of runtime as a function of the message length. This predictive capability is a notable advantage, as it allows for the estimation of encryption or decryption times without the need to actually encrypt the message or run the

Table 4
Residual Table for Encryption and Decryption

Residuals during Encryption				Residuals during Decryption			
x	y	$\hat{y}$	$y - \hat{y}$ (Residuals)	x	y	$\hat{y}$	$y - \hat{y}$ (Residuals)
16	0.02912	0.015202	0.013918	16	0.23594	0.17313	0.06281
32	0.05924	0.048994	0.010246	32	0.38242	0.33777	0.04465
48	0.08768	0.082786	0.004894	48	0.48154	0.50241	-0.02087
64	0.12452	0.116578	0.007942	64	0.74878	0.66705	0.08173
80	0.14292	0.15037	-0.007450	80	0.77696	0.83169	-0.05473
96	0.17168	0.184162	-0.01248	96	0.93154	0.99633	-0.06479
112	0.20656	0.217954	-0.01139	112	1.10732	1.16097	-0.05365
128	0.2421	0.251746	-0.00965	128	1.32954	1.32561	0.00393
144	0.26446	0.285538	-0.02108	144	1.5378	1.49025	0.04755
160	0.32652	0.31933	0.00719	160	1.67366	1.65489	0.01877
176	0.32386	0.353122	-0.02926	176	1.62256	1.81953	-0.19697
192	0.39026	0.386914	0.003346	192	1.98286	1.98417	-0.00131
208	0.44104	0.420706	0.020334	208	2.15716	2.14881	0.00835
224	0.47304	0.454498	0.018542	224	2.2934	2.31345	-0.02005
240	0.50372	0.48829	0.01543	240	2.50796	2.47809	0.02987
256	0.5117	0.522082	-0.01038	256	2.76656	2.64273	0.12383

algorithm. Equations 1 and 2, as referenced earlier, demonstrate this predictive process. The ability to forecast runtime in this manner is particularly valuable in computational experiments, where efficiency and resource optimization are paramount. It allows for rapid, cost-effective assessments of algorithmic performance, a critical aspect in the design and implementation of cryptographic solutions. This approach aligns with the objective outlined in reference [20], emphasizing the importance of efficient and accurate predictions in computational research.

5.2 Statistical Model Testing

To evaluate how well the statistical model fits the data for AES-128, we primarily rely on analyzing the residual plot. This approach focuses on the discrepancies between the observed data points and the predictions made by our model, plotting these residuals against the predicted values.

Such an analysis is crucial for assessing the model’s precision in capturing the real-world complexity of AES-128’s execution time.

In examining the encryption’s dynamics, we present the analysis through graphical representations in Figures 2 and 3, crafted using the statistical package MINITAB. Figure 2, the fitted line plot, showcases the regression analysis outcomes, depicting a direct linear relationship between the input size and the execution time. This visual evidence supports the model’s assumption of a linear correlation, underscoring the linear nature of the encryption process’s complexity.

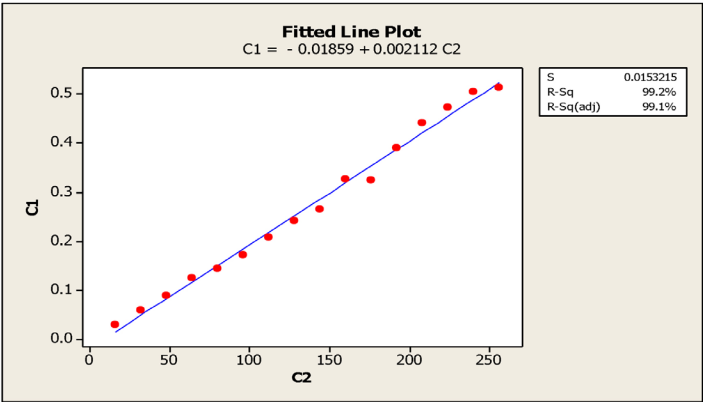


Figure 2

Regression Line Illustration for the Encryption Activity

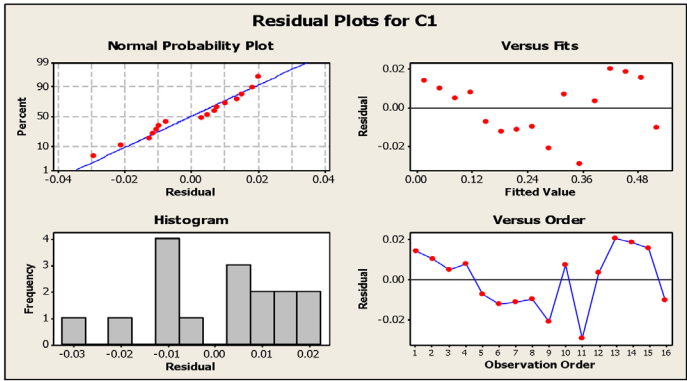


Figure 3

Differences in Predicted vs. Actual Values for the Encryption Activity

Figure 3, the residual plot, comprises four subplots, each providing unique insights:

Normal Probability Plot: This plot assesses the data distribution, specifically checking for normal distribution. The approximately straight line formed by the data points in our plot suggests that the data are normally distributed.

Residual Versus Fit: This scatter plot displays residuals on the y-axis against the fitted values on the x-axis. The lack of unusual data points in our plot implies a correlation between residuals and fitted values, supporting the validity of our linear model prediction.

Histogram: The histogram of residuals assists in identifying skewness or outliers in the data. Our histogram, showing no long tails or outliers, indicates an absence of skewness and outliers in the residuals.

Versus Order: This plot methodically exhibits the sequence of residual data collection, serving as a crucial tool for verifying the independence of residuals. In an optimal scenario, these residuals would be uniformly scattered around a normal line, a pattern that is notably evident in our encryption residual plot. Such a distribution bolsters the credibility of our analysis.

The ensuing analysis robustly supports the conclusion that AES-128's encryption empirical complexity, characterized as $O(n)$, is not merely speculative but firmly grounded in statistical evidence. This is visually articulated in the fitted line plot of Figure 2, where a clear linear relationship is delineated, with execution time plotted on the y-axis against the number of inputs on the x-axis. In a similar vein, the empirical analysis for the decryption process of AES-128 corroborates an $O(n)$ runtime. This finding underscores a consistent linear correlation between execution time and input length, validating this relationship for both the encryption and decryption facets of AES-128. Such insights are pivotal, as they provide a comprehensive understanding of the algorithm's performance under various operational scenarios.

6. Conclusion and Future Scope

This research underscores that theoretical computational complexity, while useful, can sometimes offer only a broad asymptotic approximation of an algorithm's performance. This limitation is particularly pronounced in scenarios involving compound operations where no single operation is overwhelmingly dominant. A more concrete and reliable methodology is empirical analysis, which entails running the algorithm with incrementally

increasing input sizes to accurately gauge its complexity. This practical, system-dependent approach enhances our understanding of the algorithm's real-world performance. Through integrating a statistical model with real-world data, we enhance our grasp of the estimated complexity in practice. Our detailed exploration of AES-128 utilized a blend of three methodologies: theoretical analysis, real-world experimentation, and statistical modeling. Remarkably, the theoretical expectation that AES-128's execution time should follow an $O(n)$ pattern, regardless of the operation mode, aligned perfectly with our real-world data. This alignment was further confirmed through rigorous statistical methods. This multifaceted approach provided a deep and reliable insight into AES-128's operational complexity. The formulas developed from our statistical modeling now serve as a reliable predictor for AES-128's execution time across various input sizes, confirming its practical complexity as $O(n)$.

Moving forward, our research agenda includes the examination of other cryptographic algorithms, such as RSA, Blowfish, and emerging trends in cryptographic technology. We aim to undertake a comparative analysis of these algorithms to evaluate their efficacy and suitability for real-time applications and security measures. Such a study will not only deepen our understanding of various cryptographic methods but also guide their practical application and development, ensuring enhanced data security in an increasingly digital world.

References

- [1] Goldsmith, S. F., Aiken, A. S., & Wilkerson, D. S. Measuring empirical computational complexity. In Proceedings of the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on the foundations of software engineering (pp. 395-404) (2007, September).
- [2] D.E. Knuth, "Fundamental Algorithms, Volume 1", Addison-Wisley, (1973).
- [3] S. Chakraborty, P.P. Choudhury, "A Statistical Analysis of an Algorithm's Complexity," *Applied Mathematics Letters* 13, Elsevier, pp-121-126 (2000).
- [4] S. Chakraborty, K.K. Sundararajan, "A Simple Empirical Formula for Categorizing Computing Operation", *Applied Mathematics and Computation* 187", Elsevier, pp-326-340 (2007).

- [5] S.K. Sourabh, S. Chakraborty, "Empirical $O(n^2)$ Complexity is Convincingly Gettable with Two Dense Matrix in $n \times n$ Matrix Multiplication", InterStat (2006).
- [6] S.C. Gupta, V.K. Kapoor, "Fundamental of Mathematical statistics", Sultan Chand & Sons
- [7] L. Wasserman, "All of Statistics: A Concise course in Statistical Inference", Springer, Oct. (2004).
- [8] G. James, D. Witten, T. Hastie, R. Tibshirani, "An Introduction to Statistical Learning with application in R", Springer, September (2017).
- [9] D. Wackerly, W. Mendenhall, R.L. Scheaffer, "Mathematical Statistics with Application", Thomson Brooks/Cole, 7th edition (2008).
- [10] S. Chakraborty, S. K. Sourabh, "A Computer Experiment Oriented Approach to Algorithmic Complexity", LAP LAMBERT Academic, (2010).
- [11] "Announcing the ADVANCED ENCRYPTION STANDARD (AES)" Federal Information Processing Standards Publication 197. United States National Institute of Standards and Technology (NIST). November 26, 2001. Retrieved October 2, (2012).
- [12] Daemen, Joan; Rijmen, Vincent (March 9, 2003). "AES Proposal: Rijndael".
- [13] Marcelo Vaz Netto and Sahudy Montenegro González. 2022. SSLC: A Search Algorithm Based on Linear Collisions and Poisson Probability Distribution. ACM J. Exp. Algorithmics 27, Article 1.4, 15 pages (December 2022). <https://doi.org/10.1145/3497876>.
- [14] Bossaerts, P., Yadav, N., & Murawski, C. Uncertainty and computational complexity. *Philosophical Transactions of the Royal Society B*, 374(1766), 20180138 (2019).
- [15] Athanasopoulos, D., & McEwen, M. Multi-objective empirical computational complexity of single-tenant service instances deployed at the Edge. *Journal of Systems and Software*, 111665 (2023).
- [16] Bossaerts, P., & Murawski, C. Computational complexity and human decision-making. *Trends in Cognitive Sciences*, 21(12), 917-929 (2017).
- [17] Anowar, F., Sadaoui, S., & Selim, B. Conceptual and empirical comparison of dimensionality reduction algorithms (pca, kpca, lda, mds, svd, lle, isomap, le, ica, t-sne). *Computer Science Review*, 40, 100378 (2021).

- [18] Alizadeh, R., Allen, J. K., & Mistree, F. Managing computational complexity using surrogate models: a critical review. *Research in Engineering Design*, 31, 275-298 (2020).
- [19] Prashant Pranav, Sandip Dutta and Soubhik Chakraborty, "Empirical and Statistical Comparison of Intermediate Steps of AES – 128 and RSA in Terms of Time Consumption", *Soft Computing Springer*, Volume – 25, pp. 13127 – 13145, August (2021).
- [20] Sacks, J., Welch, W. J., Mitchell, T. J., & Wynn, H. P. Design and analysis of computer experiments. *Statistical science*, 4(4), 409-423 (1989).

Received November, 2019