

An Integrated One-Click Python Pipeline for Bacterial Codon-Usage Analysis and Proteome-Wide Characterization

Ujwal Dahal

ujwalbosc@gmail.com

Lovely Professional University Department of Bioengineering and Biosciences <https://orcid.org/0009-0005-1111-0955>

Research Article

Keywords: Bacterial codon usage bias, Relative synonymous codon usage (RSCU), Codon adaptation index (CAI), Effective number of codons (ENC), Nucleotide composition, Protein physicochemical properties, Python pipeline, Computational validation

Posted Date: October 28th, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-11517937/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.
[Read Full License](#)

Additional Declarations: The authors declare no competing interests.

An Integrated One-Click Python Pipeline for Bacterial Codon-Usage Analysis and Proteome-Wide Characterization

Abstract

Background: Codon-usage analysis provides insights into synonymous codon preferences, nucleotide composition, and potential evolutionary constraints, while protein physicochemical indices offer complementary information about proteome characteristics. Together, these sequence- and protein-level measures can provide a broader computational view of genomic and proteomic variation across bacterial genomes providing insights into crucial functional mechanisms like pathogenicity and antibiotic resistance. However, integrating these analyses into a streamlined and reproducible workflow can be challenging. **Methods:** We developed an integrated one-click Python pipeline for bacterial codon-usage analysis and proteome-wide characterization. The workflow processes FASTA sequences, applies sequence-quality filters, calculates codon-usage and nucleotide-composition indices, performs parity analysis, and conducts principal component analysis (PCA) using relative synonymous codon usage (RSCU) and relative amino-acid usage (RAAU). It also estimates protein-level characteristics, including theoretical isoelectric point, molecular weight, grand average of hydropathicity, aromaticity, and sequence-length measures. Results are organized into two worksheets within a single Excel output. **Results:** The pipeline was applied to coding sequences from *Acinetobacter baumannii*, *Pseudomonas aeruginosa*, and *Elizabethkingia anophelis*. Comparisons with previously reported values showed broadly similar estimates for selected indices, including GC content, GC3s, GRAVY, and aromaticity. However, differences were observed in third-position nucleotide composition, codon adaptation index (CAI), frequency of optimal codons (Fop), and sequence-length-related measures. These discrepancies highlight the influence of computational definitions, reference sets, and sequence-processing criteria on the resulting estimates. **Conclusion:** The pipeline integrates codon-level and protein-level analyses into an accessible workflow that generates structured outputs. Further benchmarking against established tools using matched sequences and standardized definitions is needed to establish its numerical agreement, robustness, and broader applicability. Future development could expand reference-set customization and validation across diverse bacterial genomes.

Keywords: Bacterial codon usage bias; Relative synonymous codon usage (RSCU); Codon adaptation index (CAI); Effective number of codons (ENC); Nucleotide composition; Protein physicochemical properties; Python pipeline; Computational validation

1. Introduction

Codon usage is a fundamental feature of molecular evolution, reflecting the non-uniform distribution of synonymous codons across genes, genomes, and organisms [1–3]. Although synonymous codons encode the same amino acid, their usage frequencies can differ substantially among coding sequences, influenced by nucleotide composition, mutational biases, natural selection, genetic drift, gene expression, and interactions between codons and the translational machinery. In bacteria, these patterns provide a useful framework for investigating genome composition, evolutionary relationships, translational adaptation, and variation among genes with different biological functions as well as antibiotic resistance [1, 4]. Codon-usage analysis has consequently become an established component of comparative genomics, with indices such as relative synonymous codon usage (RSCU), the effective number of codons (ENC), and the codon adaptation index (CAI) used to characterize distinct aspects of codon preference and bias [5–8]. However, these indices capture different properties of coding sequences and should be interpreted alongside nucleotide composition, sequence characteristics, and the assumptions underlying their calculation.

A range of computational tools has been developed to quantify codon usage and related molecular characteristics, including CodonW (<https://sourceforge.net/projects/codonw/>), DAMBE [9], EMBOSS [10], and specialized codon-analysis packages [11, 12]. These resources provide complementary capabilities for calculating codon frequencies, synonymous codon usage, codon-bias indices, and other sequence-derived parameters. Nevertheless, comprehensive bacterial sequence characterization often requires multiple analytical steps, including input validation, removal of unsuitable sequences, treatment of terminal and internal stop codons, translation, calculation of nucleotide and amino-acid composition, and estimation of protein physicochemical properties. When these tasks are performed using separate tools or custom pipelines, differences in input handling, reference-gene selection, calculation conventions, and output formats can complicate comparisons between datasets and analytical workflows [1]. Furthermore, similarly named indices are not necessarily calculated identically across implementations, making explicit definitions and reproducible computational procedures essential for interpreting numerical results. These considerations highlight the need for an integrated workflow that combines sequence preprocessing, codon-usage analysis, and protein-level characterization within a consistent computational framework.

The reliability of codon-usage analyses therefore depends not only on the mathematical definitions of individual indices but also on the consistency of sequence processing and the reproducibility of their implementation. Differences in coding-sequence boundaries, incomplete codons, ambiguous nucleotides, stop-codon treatment, or genetic-code assignment can affect the number of analyzed codons and consequently influence downstream estimates [1, 12]. Similarly, CAI depends on the composition of its reference set, while protein molecular-weight and isoelectric-point estimates depend on the calculation conventions and physicochemical assumptions used [6]. Existing codon-usage research has demonstrated the value of combining complementary indices to characterize coding-sequence variation, but comparisons between analytical workflows require careful distinction between differences arising from biological datasets and those arising from computational methods [5, 6, 8, 13]. Independent comparison with established software and previously generated results can help identify discrepancies, clarify implementation-specific behavior, and establish the extent to which a computational workflow reproduces expected values.

In this study, we developed an integrated one-click Python pipeline for bacterial codon-usage analysis and proteome-wide characterization. The workflow combines automated FASTA sequence processing and quality control, nucleotide-composition analysis, calculation of codon-usage and codon-bias indices, parity analysis, principal component analysis (PCA) of relative synonymous codon usage (RSCU) and relative amino-acid usage (RAAU), and estimation of protein physicochemical properties, including theoretical isoelectric point, molecular weight, grand average of hydropathicity, aromaticity, and sequence-length measures. The pipeline organizes the resulting measurements into two complementary worksheets, separating codon-usage indices from protein-level characteristics within a single Excel output. The pipeline was designed to provide a streamlined and reproducible workflow for analyzing bacterial coding sequences while integrating codon-level measurements with protein-level characteristics. Its implementation and outputs were examined to characterize the calculated indices, assess the consistency of the generated results, and identify limitations associated with the computational definitions and assumptions used. This integrated approach provides a practical framework for applying codon-usage and proteome-wide analyses to bacterial coding-sequence datasets.

2. Methodology

2.1. Pipeline overview and validation framework

A Python-based workflow was used for bacterial coding-sequence (CDS) codon-usage analysis and proteome-wide protein characterization. The pipeline integrates sequence quality control, nucleotide-composition analysis, synonymous codon-usage indices, codon-adaptation measures, parity analysis, principal component analysis (PCA), and protein physicochemical characterization. It was designed to process FASTA-formatted CDS records and export the resulting measurements into a standardized Excel workbook containing two worksheets: 1_Codon_Usage and 2_Protein_Indices.

The pipeline was evaluated using a comparative framework incorporating independent codon-level analysis in previously reported computational results from previous published studies that includes *Acinetobacter Baumannii*, *Pseudomonas aeruginosa* and *Elizabethkingia anopheles* [14–17]. The corresponding species from published data has been retrieved in [Table 1]. The objective was to examine numerical correspondence across codon counts, nucleotide-composition indices, codon-bias measures, and protein-level characteristics across the same bacterial sequences mentioned in [Table 1].

Table 1. Bacterial species, genome accession numbers, coding-sequence counts, and reference studies used for comparison with the present pipeline.

Bacterial Species	Assession Number	Coding Genes	Gene transcripts	Reference studies
<i>Pseudomonas aeruginosa</i>	GCA_003325605	8715	8770	[14]
<i>Acinetobacter. baumannii</i>	GCA_000580515	4,425	4,572	[15, 17]
<i>Elizabethkingia anopheles</i>	GCA_000447375	3,636	3673	[16]

Comparisons were restricted to parameters for which corresponding reference results were available. Direct comparisons using identical sequences were distinguished from comparisons with previously published aggregate

statistics. Differences in sequence selection, filtering, genetic-code assignment, parameter definitions, reference-gene selection, and aggregation procedures were considered when interpreting discrepancies. Published values were treated as comparative references rather than as an automatically equivalent computational gold standard.

2.2. Input sequences and quality control

The pipeline accepted nucleotide sequences in FASTA format. When executed without manual modification, it searched the directory containing the pipeline for files with the extensions. fa. If multiple eligible files were present, the most recently modified file was selected. The default output was Codon_Usage_Analysis.xlsx, saved in the same directory.

Sequences were converted to uppercase, whitespace was removed, and uracil (U), when present, was converted to thymine (T). Records containing 300 bp or fewer were excluded; therefore, the minimum retained sequence length was 301 bp. Sequences containing ambiguous or otherwise unsupported nucleotide characters after normalization were excluded. The sequences whose lengths were not divisible by three, were also excluded. The bacterial genetic code (NCBI translation table 11) was used for translation [18, 19].

Terminal stop codons were permitted at the 3' end but were automatically trimmed to ensure accurate codon statistics and protein length calculations. Sequences containing premature internal stop codons were excluded. Furthermore, an in-silico translation-level check was implemented to reject any sequence yielding internal stop codons within the translated amino acid sequence. The analyzed CDS therefore comprised complete sense codons, excluding an accepted terminal stop codon and any incomplete 3' fragment. Protein length was measured in amino-acid residues, excluding the terminal stop. Rejected records and their exclusion reasons were written to a separate CSV file (Codon_Usage_Analysis_rejected_records) when applicable.

2.3. Codon counting and relative synonymous codon usage

Codon counts were obtained by dividing each accepted coding region into consecutive, non-overlapping triplets in the original reading frame. Counts were calculated for the 61 sense codons specified by the selected genetic code; stop codons were handled separately [6, 15]. For each sense codon, the pipeline calculated its observed count and frequency per 1,000 analyzed codons:

$$F_{c,1000} = \frac{N_c}{N_{codons}} \times 1000$$

where N_c is the observed count of codon c , and N_{codons} is the total number of analyzed sense codons in the corresponding CDS.

Relative synonymous codon usage (RSCU) was calculated as:

$$RSCU_{ij} = \frac{n_i X_{ij}}{\sum_{j=1}^{n_i} X_{ij}}$$

where X_{ij} is the observed count of codon j encoding amino acid i , and n_i is the number of synonymous codons encoding that amino acid. An RSCU value of 1 indicates usage consistent with equal use of synonymous codons

within the corresponding amino-acid family; values greater than 1 indicate above-expected usage, whereas values below 1 indicate below-expected usage. When no codon in a synonymous family was observed, its RSCU values were undefined and were represented as missing values [5] .

The codon counts, frequencies per 1,000 codons, and RSCU values were calculated internally for each CDS. These codon-level values were used in the downstream analyses, although the final workbook displayed only the selected summary indices rather than all individual codon columns.

2.4. Nucleotide composition and synonymous-site indices

Overall GC content was calculated from the complete analyzed coding region as:

$$GC = \frac{N_G + N_C}{N_A + N_T + N_G + N_C}$$

where NA, NT, NG, and NC denote the nucleotide counts within the analyzed CDS.

The GC content of the first and second codon positions was calculated using all complete analyzed codons [8] . GC12 was defined as the mean GC fraction at these two positions:

$$GC12 = \frac{GC_1 + GC_2}{2}$$

where GC1 and GC2 are the GC fractions at the first and second codon positions, respectively.

Third-position synonymous-site composition was calculated using codons encoding amino acids with more than one synonymous codon. Methionine and tryptophan codons were excluded because these amino acids have no synonymous alternatives under the selected genetic code [8] . The proportions of thymine, cytosine, adenine, and guanine at these synonymous third positions were calculated as:

$$T_{3s} = \frac{N_{T,3s}}{N_{3s}}, \quad C_{3s} = \frac{N_{C,3s}}{N_{3s}}$$

$$A_{3s} = \frac{N_{A,3s}}{N_{3s}}, \quad G_{3s} = \frac{N_{G,3s}}{N_{3s}}$$

where N3s is the number of third positions contributed by codons belonging to synonymous amino-acid families [8] . Synonymous third-position GC content was calculated as:

$$GC_{3s} = G_{3s} + C_{3s}$$

These quantities are synonymous-site proportions and were not treated as equivalent to third-position nucleotide proportions calculated across all codons. If no eligible synonymous third positions were present, the corresponding measures were undefined.

The distinction between GC and GC3s was maintained throughout the analysis: GC represents the complete analyzed coding region, whereas GC3s represents the G+C proportion at synonymous third-codon positions.

2.5. Codon-bias and codon-adaptation indices

2.5.1. Effective number of codons

The effective number of codons (N_c), also commonly termed the effective number of codons or ENC, was estimated using a Wright-style calculation based on synonymous-family codon homozygosity [8]. Synonymous families containing two, three, four, or six codons were considered. For each family containing at least two observations, codon frequencies were calculated as:

$$p_j = \frac{X_j}{N_i}$$

where X_j is the count of synonymous codon j , and N_i is the total count for the corresponding amino-acid family.

A finite-sample-corrected homozygosity estimate was calculated as:

$$F_i = \frac{\sum_{j=1}^{k_i} p_j^2 - \frac{1}{N_i}}{1 - \frac{1}{N_i}}$$

where k_i is the number of synonymous codons in the family. The estimate was constrained to the interval from $1/k_i$ to 1 to maintain numerical stability [8]. Mean family-specific homozygosity values were then combined using the coefficients implemented in the pipeline:

$$N_c = 2 + \frac{9}{F_2} + \frac{1}{F_3} + \frac{5}{F_4} + \frac{3}{F_6}$$

where F_2 , F_3 , F_4 , and F_6 denote the mean corrected homozygosity estimates for families containing two, three, four, and six synonymous codons, respectively. Terms were included only when the corresponding family estimate was available. If no eligible family estimates were available, N_c was reported as undefined.

N_c conventionally ranges from approximately 20, indicating strong synonymous codon-usage bias, to 61, indicating little or no bias. Because the implementation uses a finite-sample-corrected, Wright-style estimate, its output was interpreted as an implementation-specific [8] N_c estimate rather than assumed to be numerically identical to every published implementation.

2.5.2. Codon adaptation index

The codon adaptation index (CAI) was calculated relative to a reference set of ribosomal-protein coding sequences identified using a predefined, case-insensitive FASTA-header pattern [6]. The pattern included terms associated with ribosomal proteins, ribosomes, 30S and 50S subunits, and common *rps* and *rpl* gene identifiers. Only accepted CDS records identified by this procedure contributed to the reference set.

For each synonymous amino-acid family, the observed count of each codon in the reference set was divided by the highest count in that family to obtain relative-adaptiveness weights:

$$w_c = \frac{n_c}{\max(n_1, \dots, n_k)}$$

where n_c is the reference-set count of codon c , and the denominator is the maximum reference-set count among synonymous codons encoding the same amino acid.

CAI was calculated as the geometric mean of the weights for observed codons in synonymous families:

$$CAI = \exp\left(\frac{\sum_c X_c \ln(w_c)}{\sum_c X_c}\right)$$

where X_c is the count of codon c in the analyzed CDS. The denominator includes observed codons belonging to eligible synonymous families. Methionine and tryptophan were excluded because they lack synonymous alternatives [6].

The implementation required reference weights for all synonymous codons in any family represented in the analyzed CDS. If a required family lacked usable reference weights, CAI was left undefined rather than calculated by silently omitting the affected codons. An observed codon with a zero relative-adaptiveness weight resulted in a CAI of zero under the implemented calculation. If no usable synonymous codons were available, CAI was reported as undefined. When no ribosomal reference CDS were detected, the pipeline issued a warning and reference-dependent outputs could not be interpreted as valid reference-based measurements.

2.5.3. Frequency of optimal codons

The frequency of optimal codons (Fop) was calculated using a reference-defined operational criterion. Within each synonymous amino-acid family, codons having the highest observed count in the accepted ribosomal-protein reference set were classified as preferred; tied codons were all retained in the preferred set [20, 21].

Fop was calculated as:

$$Fop = \frac{N_{preferred, syn}}{N_{syn}}$$

where $N_{preferred, syn}$ is the number of codons in the analyzed CDS belonging to the reference-defined preferred synonymous set, and N_{syn} is the total number of codons belonging to synonymous amino-acid families.

Methionine and tryptophan were excluded from the denominator [20, 21] r. Fop was undefined when no synonymous codons were available or when a usable preferred-codon set could not be established.

This definition represents the pipeline's reference-based implementation and should not be assumed equivalent to Fop estimates generated using experimentally established optimal codons or alternative preference criteria.

2.6. Parity indices and P2

Third-position nucleotide parity was characterized using the following ratios:

$$Parity_{AT} = \frac{A_{3s}}{A_{3s} + T_{3s}}$$

$$Parity_{GC} = \frac{G_{3s}}{G_{3s} + C_{3s}}$$

where the component quantities represent the nucleotide proportions at synonymous third-codon positions. Because each ratio uses the same synonymous-site denominator for its component nucleotides, the ratios are equivalent to the corresponding nucleotide-count ratios [22, 23]. Values of 0.5 indicate equal proportions within the relevant nucleotide pair, whereas departures from 0.5 indicate compositional asymmetry. These indices were interpreted descriptively and were not considered sufficient on their own to identify the evolutionary mechanisms responsible for codon-usage patterns. A ratio was undefined if its denominator was zero.

The P2 index was calculated from in-frame codons using the conventional nucleotide classes W (A or T), S (G or C), and Y (C or T):

$$P2 = \frac{WWC + SST}{WWY + SSY}$$

Here, WWC denotes codons whose first two nucleotides are W and whose third nucleotide is C; SST denotes codons whose first two nucleotides are S and whose third nucleotide is T. WWY and SSY represent the corresponding codon categories with third-position nucleotides C or T [24]. The implementation counted these categories directly from complete analyzed codons rather than from adjacent codon-pair junctions. P2 was reported as undefined when its denominator was zero.

2.7. Protein characterization and physicochemical indices

Accepted CDS records were translated using the same bacterial genetic code applied during sequence processing. Protein length (Laa) was defined as the number of translated amino-acid residues, excluding a terminal stop codon [15].

2.7.1. Molecular weight

Protein molecular weight was estimated from amino-acid residue masses using the residue-mass constants encoded in the pipeline. The mass of one water molecule was added to represent the terminal groups of a complete polypeptide chain [25] :

$$MW = \sum_{i=1}^{20} N_i M_i + 18.01528$$

where N_i is the number of residues of amino-acid type i , M_i is its residue mass in daltons, and 18.01528 Da is the mass of one water molecule [25] . The resulting mass was divided by 1,000 to express molecular weight in kilodaltons (kDa). This calculation represents an estimated sequence-derived mass and does not account for post-translational modifications or other chemical alterations.

2.7.2. Theoretical isoelectric point

The theoretical isoelectric point (pI) was estimated using a charge-balance model based on the Henderson–Hasselbalch equation. The model incorporated ionizable side chains of lysine, arginine, histidine, aspartate, glutamate, cysteine, and tyrosine, together with the N- and C-terminal groups. The pKa constants were fixed in the pipeline at 7.50 for the N-terminus, 3.55 for the C-terminus, 10.54 for lysine, 12.48 for arginine, 6.04 for histidine, 3.90 for aspartate, 4.07 for glutamate, 8.18 for cysteine, and 10.46 for tyrosine [26, 27] .

At each pH, the estimated positive and negative charges were calculated from the corresponding ionization fractions. The theoretical pI was obtained by numerically locating the pH at which the estimated net charge approached zero. A binary-search procedure was applied over pH 0–14 for 100 iterations [26, 27] . The resulting pI is a theoretical estimate dependent on the selected pKa constants and charge model.

2.7.3. Grand average of hydropathicity

The grand average of hydropathicity (GRAVY) was calculated as the arithmetic mean of the Kyte–Doolittle hydropathy scores of the amino-acid residues [28] :

$$GRAVY = \frac{1}{L_{aa}} \sum_{k=1}^{L_{aa}} h_k$$

where h_k is the Kyte–Doolittle hydropathy score of residue k , and L_{aa} is the protein length. Positive values indicate greater average hydrophobicity, whereas negative values indicate greater average hydrophilicity [28] .

2.7.4. Aromaticity

Aromaticity (Aromo) was calculated as the fraction of phenylalanine (F), tryptophan (W), and tyrosine (Y) residues [29] :

$$Aromo = \frac{N_F + N_W + N_Y}{L_{aa}}$$

where NF, NW, and NY denote the respective residue counts. The index represents the proportion of these three aromatic amino acids in the protein sequence [29] .

2.7.5. Synonymous-codon length measure

The index L_{sym} was calculated as the number of codons in the analyzed CDS that encode amino acids with more than one synonymous codon under the selected genetic code:

$$L_{sym} = \sum_{c \in CDS} I(kaa(c)_{>1})$$

where I is an indicator function equal to 1 when the amino acid encoded by codon c has more than one synonymous codon and 0 otherwise [8] . Accordingly, L_{sym} is a count of codons belonging to synonymous amino-acid families, not the total number of synonymous codon types or the total protein length.

2.8. Principal component analysis of codon and amino-acid composition

PCA was applied separately to per-CDS RSCU profiles and amino-acid-composition profiles [30] . The first analysis generated the RSCU axes, whereas the second generated the RAAU axes reported in the protein-characterization worksheet. In this implementation, the RAAU analysis used the relative frequencies of the 20 standard amino acids within each translated protein; a separate amino-acid-frequency worksheet was not generated [29–31] .

For RSCU-based PCA, the input matrix contained one row per accepted CDS and one column per sense codon. Undefined RSCU entries were first replaced by the corresponding column means; any remaining missing values were replaced by zero. For RAAU-based PCA, each row contained the relative frequency of each of the 20 standard amino acids, calculated by dividing its count by the total protein length.

Before PCA, each input matrix was centred by subtracting the column mean and standardized by dividing each column by its standard deviation. Variables with zero standard deviation were assigned a scaling factor of one. Singular value decomposition was then used to obtain the first two component-score coordinates. If only one axis could be obtained, the second axis was set to zero; where the PCA could not be calculated, the corresponding scores were left undefined.

The resulting Axis 1 and Axis 2 values are dataset-dependent ordination coordinates. They are not interchangeable with correspondence-analysis axes or PCA coordinates generated using different input matrices, missing-value procedures, or scaling conventions. Because the matrices were centred before decomposition, the mean component scores across the observations used to fit each PCA are expected to be approximately zero, subject to numerical precision. The summary-row PCA values therefore represent arithmetic means of per-CDS scores and should not be interpreted as independent biological measurements.

2.9. Dataset-level aggregation and output structure

The workbook retained one row per accepted CDS in each worksheet and appended one final summary row. For the summary row, all exported metrics were calculated as the arithmetic mean of the corresponding values across

all accepted CDS records. Thus, the final row represented the average value across the complete set of analyzed CDSs rather than a value recalculated from pooled nucleotide or codon counts.

In 1_Codon_Usage, the summary row therefore contained the arithmetic means of the individual-CDS values for T3s, C3s, A3s, G3s, Nc, GC3s, GC, RSCU Axis 1, RSCU Axis 2, GC12, Parity_AT, Parity_GC, P2, CAI, and Fop. Similarly, in 2_Protein_Indices, the summary row contained the arithmetic means of the corresponding protein-level values, including theoretical pI, molecular weight, GRAVY, aromaticity, L_sym, L_aa, RAAU Axis 1, and RAAU Axis 2. Consequently, the final summary row in both worksheets represented the unweighted arithmetic mean across all accepted CDS records, with each CDS contributing equally regardless of sequence length.

The 1_Codon_Usage worksheet displayed the gene or sequence identifier, T3s, C3s, A3s, G3s, Nc, GC3s, GC, RSCU Axis 1 and Axis 2, GC12, the two parity indices, P2, CAI, and Fop. The 2_Protein_Indices worksheet displayed the gene or sequence identifier, theoretical pI, molecular weight in kDa, RAAU Axis 1 and Axis 2, GRAVY, aromaticity, L_sym, and L_aa. Individual sense-codon counts, per-1,000-codon frequencies, RSCU values, and amino-acid frequencies were calculated internally where required, but were not exported as separate columns in the final two-sheet workbook.

2.10. Comparative evaluation and numerical agreement

The pipeline's results were compared with previously reported values from the previous studies by the authors [14–17], where corresponding measurements were available. The comparisons used the same species and strain represented in the respective published analyses, with corresponding sequence data used for the computational evaluation. The purpose of this comparison was to assess the numerical correspondence between the newly generated pipeline outputs and previously reported results.

For each parameter for which a corresponding published value was available, the pipeline-derived value was compared directly with the previously reported value. No additional difference, percentage-difference, or statistical agreement metric was calculated. The comparison was therefore based on the reported numerical values and their correspondence rather than on a predefined numerical acceptance threshold.

Exact codon counts and directly sequence-derived measurements were expected to correspond when the same sequence boundaries, reading frame, genetic code, and codon definitions were applied. For derived indices, correspondence was considered in relation to their mathematical definitions and implementation-specific conventions, including sequence preprocessing, stop-codon handling, synonymous-codon classification, and reference-data requirements.

For CAI and Fop, particular attention was given to the reference-set composition, synonymous-family weighting, and definition of preferred codons because these parameters depend on the reference dataset and its implementation. Protein-level comparisons considered sequence boundaries, amino-acid inclusion criteria, residue-mass conventions, terminal-group contributions, and pKa assumptions. Differences, where present, were reported descriptively and were not interpreted as evidence that either computational approach was inherently incorrect.

2.11. Software implementation and reproducibility

The pipeline was implemented in Python and used Biopython for FASTA parsing and translation, NumPy for numerical calculations and matrix operations, pandas for tabular processing and aggregation, and openpyxl for Excel workbook generation and formatting [32–34] . The genetic code, minimum sequence-length threshold, ribosomal-header pattern, physicochemical constants, filtering criteria, and calculation conventions were specified in the pipeline. The pipeline is freely accessible at [<https://github.com/Drujwal/Bacterial-Codon-and-Amino-Acid-Usage-Bias-Analysis-Pipeline>]. The workflow generated a formatted Excel workbook with frozen header rows, autofilters, and separate worksheets for codon-usage and protein-level results. Records that failed quality-control criteria were documented in an external rejection log where applicable.

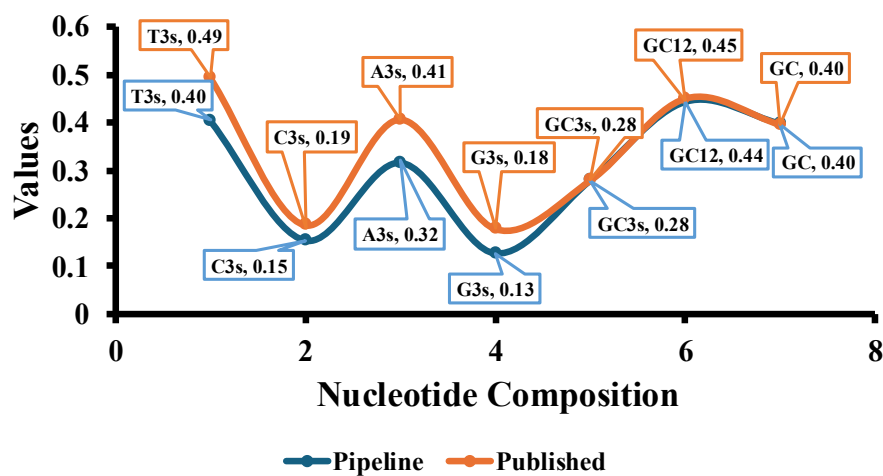
The validation framework distinguished direct comparisons with identical input sequences from broader cross-study comparisons using published summary values. Where historical processing details or original sequence-level data were unavailable, the resulting comparisons were interpreted as assessments of numerical correspondence rather than proof of exact computational equivalence. This approach provides a transparent basis for evaluating the calculation of the pipeline, reproducibility, and practical application to bacterial codon-usage and protein characterization.

3. Results

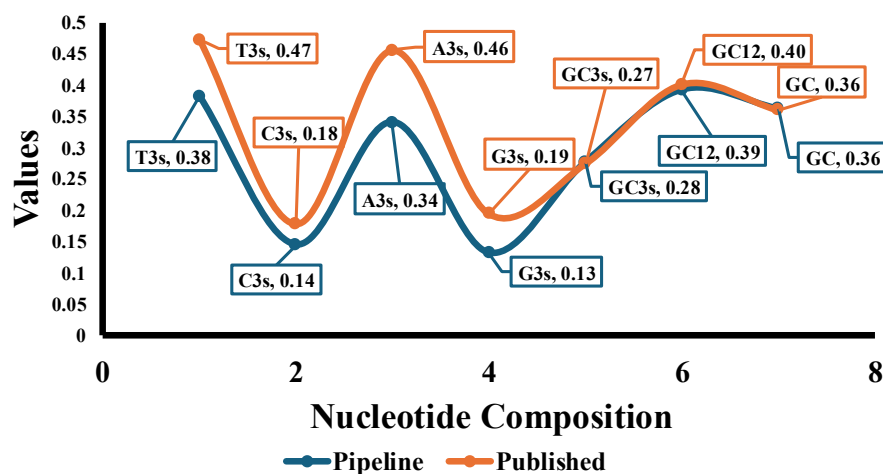
3.1. Comparison of codon-usage and nucleotide-composition indices

Comparison of the finalized Python pipeline with previously published results revealed variable numerical agreement across the three bacterial datasets [**Figure 1, Supplementary Information S1**]. The degree of correspondence differed among indices, with relatively close values for several genome-wide measures and larger differences for selected nucleotide-composition and codon-adaptation indices.

(A) *A. baumannii*



(B) *E. anopheles*



(C) *P. aeruginosa*

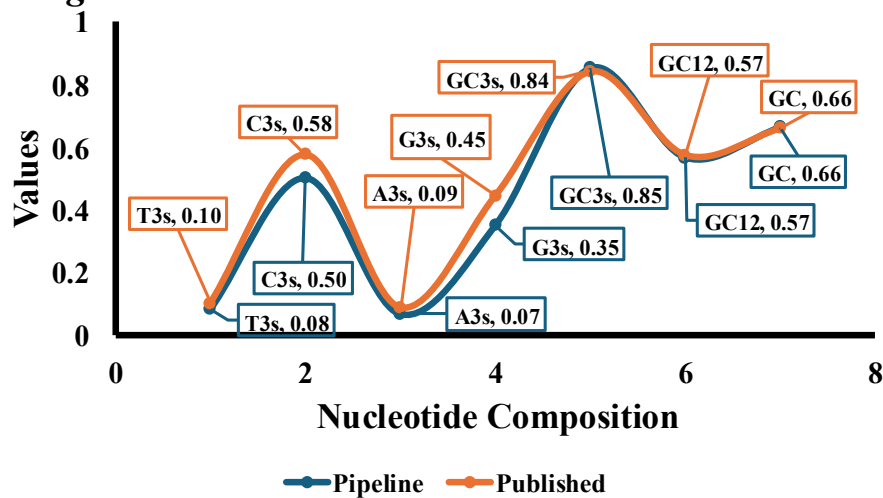


Figure 1. Comparison of nucleotide-composition indices generated by the present pipeline and previously published results.

Note: Scatter plots (A–C) show the correspondence between pipeline-derived and previously reported values for A3s, C3s, G3s, T3s, GC12, GC3s, and overall GC composition for the indicated bacterial species. For graphical representation, values are displayed to two decimal places, whereas the corresponding supplementary data retain the directly generated values to six decimal places or more; therefore, minor differences may occur between the displayed plot values and the full-precision supplementary values because of rounding. Composition values are expressed as proportions of the total (0–1 scale), with values of 0.10 and 0.50 corresponding to 10% and 50%, respectively. The A3s, C3s, G3s, and T3s values generated by the present pipeline sum to 1 because they were calculated from the defined synonymous third-codon positions. Differences between these values and the corresponding published values should therefore be interpreted in the context of the nucleotide-position definitions and methodological assumptions used in the respective analyses, including the treatment of synonymous positions in CodonW, rather than as evidence of an error in the present pipeline.

The effective number of codons (ENC) showed relatively close agreement between the pipeline-derived and published values. For *Acinetobacter baumannii*, the respective values were 44.63 and 45.31; for *Pseudomonas aeruginosa*, 33.74 and 34.44; and for *Elizabethkingia anopheles*, 45.88 and 46.46 [Supplementary Information S1]. GC content also showed close numerical correspondence, with pipeline-derived versus published values of 0.39 versus 0.39, 0.66 versus 0.66, and 0.36 versus 0.36, respectively. Similarly, GC3s values were 0.27, 0.85, and 0.27 in the pipeline, compared with 0.27, 0.84, and 0.27 in the published datasets [Figure 1, Supplementary Information S1].

The third-position nucleotide-composition indices exhibited larger differences. For *A. baumannii*, the pipeline-derived T3s, C3s, A3s, and G3s values were 0.40, 0.15, 0.31, and 0.12, respectively, compared with published values of 0.49, 0.18, 0.40, and 0.17. For *P. aeruginosa*, the corresponding values were 0.08, 0.50, 0.06, and 0.35 in the pipeline, compared with 0.10, 0.57, 0.08, and 0.44 in the published dataset. For *E. anophelis*, the pipeline-derived values were 0.38, 0.14, 0.34, and 0.13, compared with 0.47, 0.17, 0.45, and 0.19, respectively [Figure 1, Supplementary Information S1].

GC12 values were 0.44 versus 0.45 for *A. baumannii*, 0.56 versus 0.57 for *P. aeruginosa*, and 0.39 versus 0.40 for *E. anophelis*. The parity indices also exhibited variable numerical correspondence. The pipeline-derived A3s/(A3s+T3s) values were 0.43, 0.46, and 0.47 for *A. baumannii*, *P. aeruginosa*, and *E. anophelis*, respectively, compared with published values of 0.45, 0.47, and 0.49. The corresponding G3s/(G3s+C3s) values were 0.453, 0.41, and 0.47 in the pipeline, compared with 0.48, 0.43, and 0.52 in the published datasets. P2 values were 0.44, 0.40, and 0.42 in the pipeline, compared with 0.44, 0.39, and 0.42, respectively [Figure 1, Supplementary Information S1].

3.2. Codon-adaptation and optimal-codon indices

The pipeline-derived codon adaptation index (CAI) values were 0.60 for *A. baumannii*, 0.64 for *P. aeruginosa*, and 0.65 for *E. anophelis*. The corresponding published values were 0.54, 0.63, and 0.57, respectively [Figure 2, Supplementary Information S1]. The pipeline-derived values were higher in all three comparisons, although the magnitude of the difference varied among datasets.

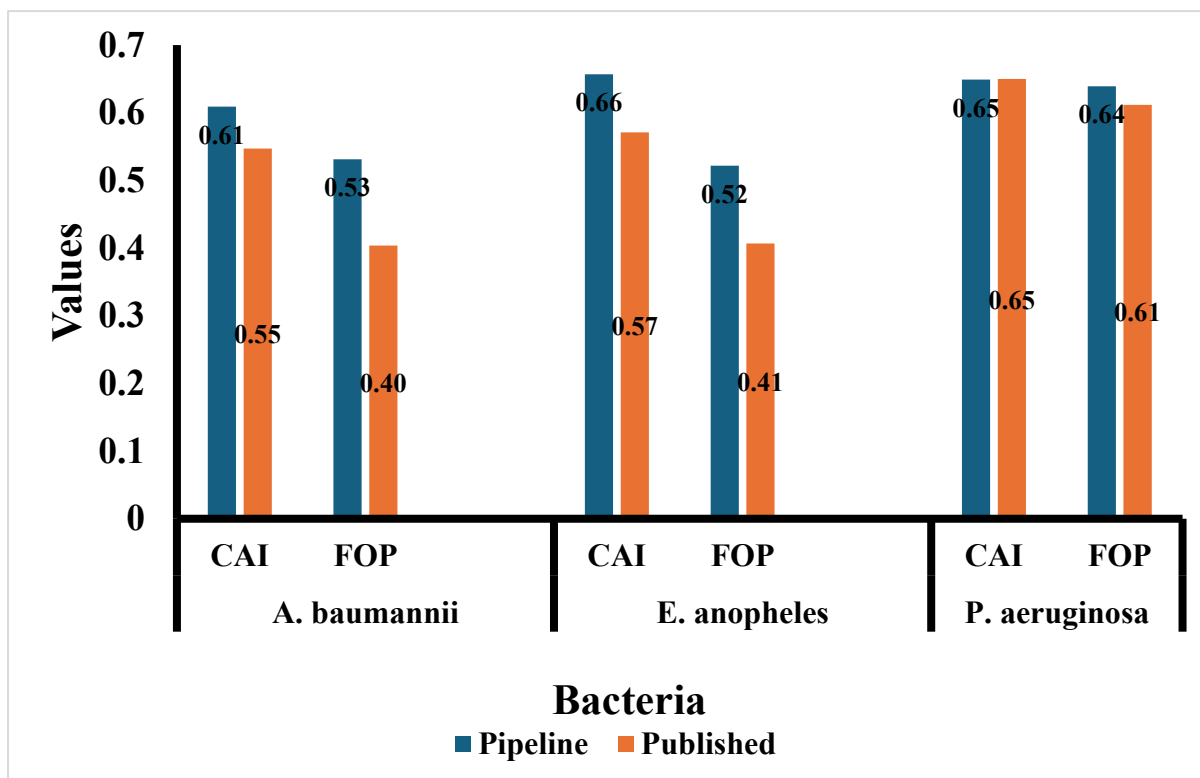


Figure 2. Comparison of codon adaptation and optimal codon usage indices across bacterial species.

Note :Bar plots show the CAI and Fop values for each indicated species, comparing the values obtained from the present pipeline with the corresponding previously reported values. For graphical representation, values are rounded to two decimal places; the directly generated full-precision values are provided in the Supplementary Information. Accordingly, minor differences between values displayed in the figure and those reported in the supplementary data may occur because of rounding during graphical representation. CAI and Fop values were calculated according to the respective methodological definitions and reference-set criteria described in the Methods.

The frequency of optimal codons (Fop) values were 0.53, 0.65, and 0.52 in the pipeline, compared with published values of 0.40, 0.612, and 0.40 for the same organisms [Figure 2, Supplementary Information S1]. These results provide a comparative overview of the codon-adaptation-related outputs generated by the pipeline and their numerical correspondence with previously reported values.

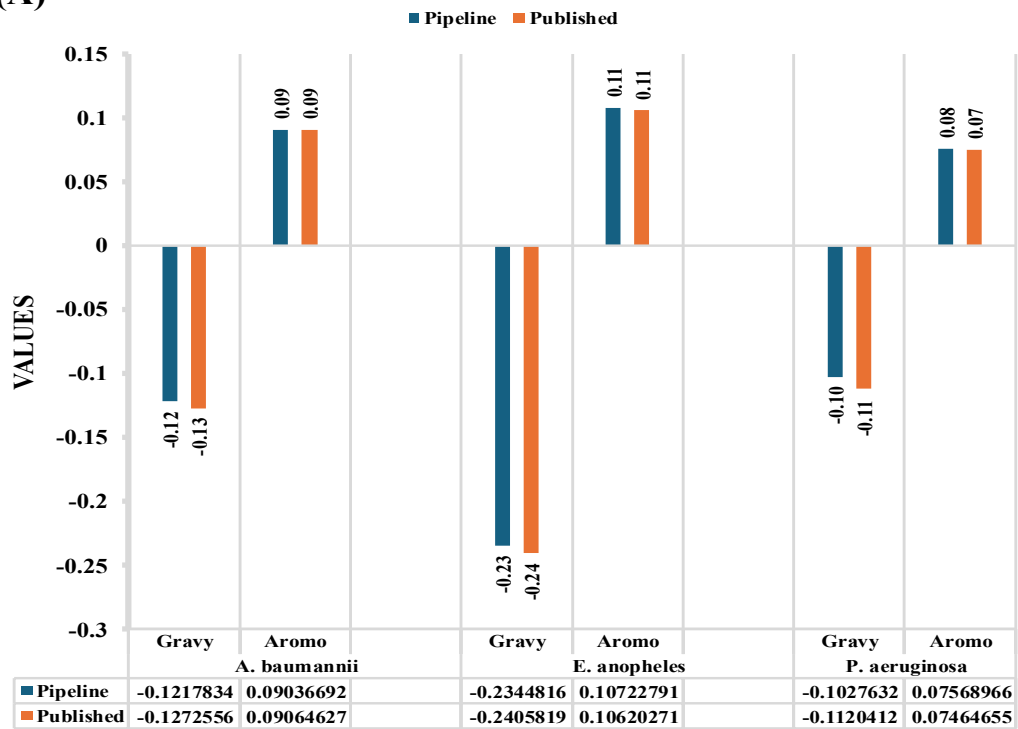
3.3. Protein-related physicochemical indices

The pipeline-derived mean isoelectric points were 6.84 for *A. baumannii*, 6.86 for *P. aeruginosa*, and 7.357 for *E. anophelis*. A published value of 6.913 was available for *A. baumannii*, while published values were not available for the remaining bacterial species. The corresponding mean molecular weights generated by the pipeline were 35.99, 28.984, and 39.25 kDa, respectively; a published value of 32.178 kDa was available for *A. baumannii* [Supplementary Information S1].

The pipeline-derived grand mean GRAVY values were -0.12 , -0.10 , and -0.23 for *A. baumannii*, *P. aeruginosa*, and *E. anophelis*, respectively. These values were numerically close to the published values of -0.12 , -0.11 , and -0.24 . Aromaticity values also showed close correspondence, with pipeline-derived values of 0.09, 0.07, and 0.10

compared with published values of 0.09, 0.07, and 0.10, respectively [Figure 3, Supplementary Information S1].

(A)



(B)

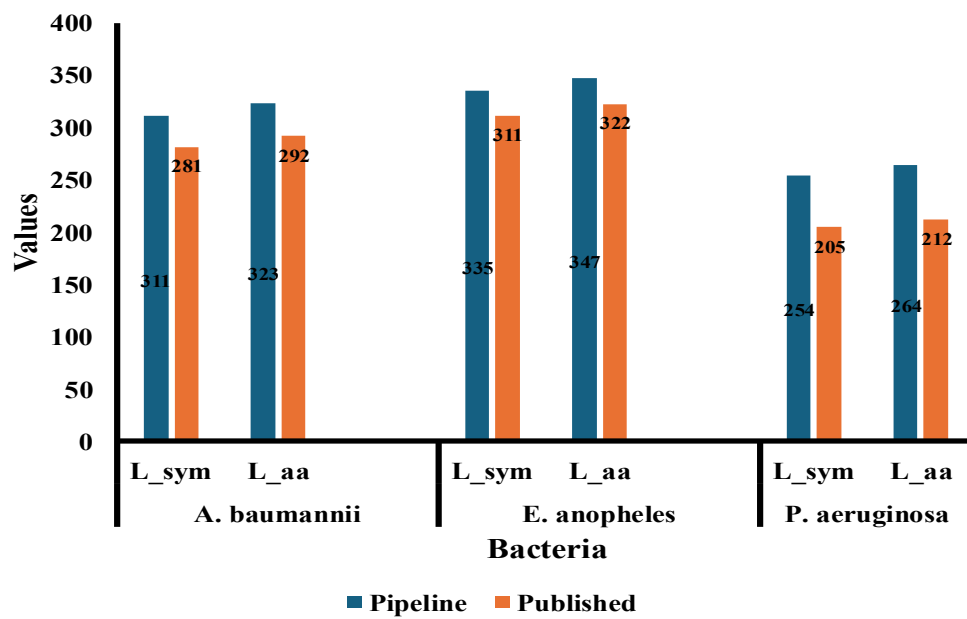


Figure 3. Comparison of protein compositional and sequence-derived indices across bacterial species.

(A) GRAVY and aromaticity values and (B) L_{sym} and protein length (L_{aa}) values for the indicated bacterial species generated by the present pipeline vs published results.

Note: Values are displayed ignoring the decimals for graphical representation, whereas the directly generated full-precision values are provided in the Supplementary Information. Therefore, minor differences between values shown in the figure and the corresponding supplementary values may occur because of rounding during graphical representation.

The pipeline-derived L_{sym} values were 311, 254, and 335, compared with published values of 281, 204, and 311. Similarly, L_{aa} values were 322, 263, and 346 in the pipeline, compared with 291, 212, and 322 in the published datasets [Figure 3, Supplementary Information S1]. Both length-related measures were consistently higher in the pipeline-derived results across the three comparisons.

3.4. Principal-component-derived indices

The genome-wide Axis 1 and Axis 2 summaries generated from RSCU as well as RAAU based analyses were numerically close to zero across all three bacterial datasets. The corresponding published values were also small in magnitude, although they did not consistently match the pipeline-derived values [Table 2]. These summaries represent PCA-derived outputs whose numerical interpretation depends on the input data, preprocessing, component loadings, and method used to aggregate scores.

Table 2: RSCU- and RAAU-based principal component axis scores generated by the pipeline and corresponding previously published values for the analyzed bacterial species.

Bacteria	RSCU(Axis1)		RSCU(Axis2)		RAAU(Axis1)		RAAU(Axis2)	
	Pipeline	Published	Pipeline	Published	Pipeline	Published	Pipeline	Published
<i>A. baumannii</i>	-7.6E-17	-0.00102	5.16E-16	-0.00105	-1.7E-16	0.007874	-2E-16	0.020937
<i>E. anopheles</i>	8.15E-17	-0.00077	-1.2E-16	0.001065	-2.6E-16	-0.02855	-4.4E-16	0.012406
<i>P. aeruginosa</i>	5.58E-16	-0.00168	1.47E-16	-0.00026	-9.4E-17	0.006288	3.67E-17	0.005443

Note: Values expressed in the form E-n represent scientific (exponential) notation, where E-n denotes $\times 10^{-n}$. Values approaching zero (e.g., E-16 or E-17) reflect very small numerical values arising from computational precision and should not be interpreted as biologically meaningful differences from zero.

4. Discussion

4.1. Cross-study correspondence of codon-usage indices

The comparison of three bacterial datasets illustrates the range of codon-usage and protein-related measurements generated by the finalized Python pipeline. Several indices, including ENC, GC content, GC3s, GRAVY, and aromaticity, showed relatively close numerical correspondence with previously published results. Other measures,

particularly third-position nucleotide composition, CAI, Fop, and protein-length-related indices, exhibited larger differences. This index-specific pattern provides a basis for documenting the pipeline's outputs and identifying the calculation settings and tool-specific implementations that govern their interpretation.

The close numerical correspondence observed for several genome-wide measures supports their use as comparative reference points. However, comparisons across studies can be influenced by sequence selection, filtering criteria, index definitions, software-specific normalization procedures, and aggregation procedures [5, 6, 8, 35, 36]. Consequently, numerical agreement should be interpreted in the context of the methods and tools used to generate each value. This distinction is important because cross-study correspondence does not, by itself, establish equivalence between computational implementations.

4.2. Nucleotide composition, codon adaptation, and parity measures

The larger differences observed in T3s, C3s, G3s and A3s highlight the importance of clearly defining nucleotide-composition measures and accounting for software-specific calculation procedures. In the pipeline-derived datasets, these four values sum to approximately 1.0 for each organism, whereas the corresponding published values sum to approximately 1.263 for *A. baumannii*, 1.210 for *P. aeruginosa*, and 1.303 for *E. anophelis*. These discrepancies stem from fundamental differences in index definitions and normalization conventions: CodonW reports T3s, C3s, G3s and A3s using nucleotide-specific denominators for synonymous third-position usage rather than overall third-position proportions; consequently, those published values are not expected to sum to 1 (or 100%). This reflects CodonW's internal normalization procedures rather than a computational error, underlining why third-position measures must be compared directly only after confirming that their underlying denominators are equivalent [16, 36].

CAI and Fop also differed between the pipeline-derived and published datasets due to methodological differences in reference selection and optimal-codon identification. The published studies calculated CAI using DAMBE, whereas our Python pipeline derives relative synonymous-codon adaptiveness weights from an automatically identified ribosomal-protein reference set. Differences in reference-set selection and weight calculation directly contribute to the observed discrepancies in CAI values [6, 9, 35]. Similarly, for Fop, the published studies relied on CodonW, whereas our pipeline identifies the most frequent codon(s) within each synonymous family in the detected ribosomal reference set as preferred codons. These distinct conventions for defining optimal codons and reference sets naturally produce different CAI and Fop estimates, meaning numerical variations reflect methodological settings rather than differences in biological adaptation.

For RSCU and RAAU measures, RSCU in published studies was calculated using CodonW, while our pipeline determines RSCU directly from synonymous codon counts in Python. Where input sequences and mathematical definitions match, results align closely; slight discrepancies reflect sequence preprocessing, implementation details, or aggregation methods. Similarly, RAAU calculations in our pipeline derive relative frequencies of the 20 standard amino acids for PCA, whereas published comparisons relied on CodonW's RAAU routines [36].

The parity indices and P2 provide additional depipelineors of codon-position composition and codon-usage patterns. Their variable numerical correspondence reinforces the importance of specifying the nucleotide groups and denominators used in each calculation. These indices are useful for characterising compositional patterns, although their values alone do not establish the evolutionary mechanisms underlying those patterns [22, 23] .

4.3. Protein-related characteristics and length measures

GRAVY and aromaticity showed close numerical correspondence between the pipeline-derived and published values across the three bacterial datasets. These results provide useful comparative evidence for the protein-related outputs generated by the pipeline. In contrast, L_{sym} and L_{aa} were consistently higher in the pipeline-derived results. Their interpretation depends on the sequence set analysed and on the precise definitions and aggregation procedures used. Consistent treatment of incomplete coding sequences, stop codons, protein inclusion criteria, and the distinction between mean lengths and other summary measures is therefore important when comparing results across studies [28, 29] .

The pipeline also generates isoelectric-point and molecular-weight summaries, extending the analysis beyond nucleotide and codon composition to protein-level physicochemical characteristics. Direct cross-study comparison of these measures was limited by the availability of published reference values, which were supplied for *A. baumannii* but not for the other two datasets. Their inclusion provides complementary depipelineive outputs within the workflow, although broader cross-organism comparisons would require additional reference data.

4.4. Principal-component-derived summaries

The near-zero genome-wide PCA-axis summaries should be interpreted according to the construction of the principal-component analyses. When PCA scores are centred, their mean across the observations used to fit the analysis is expected to be approximately zero. Therefore, near-zero mean coordinates may reflect a mathematical property of the analysis rather than a distinct biological pattern [30, 37, 38] . In addition, component scores from separately fitted PCAs are not directly interchangeable because their loadings, orientations, and preprocessing depend on the data analysed and whether RSCU or RAAU values serve as the underlying matrix. These outputs should consequently be documented as PCA-derived summaries, with comparisons made only when the underlying procedures and interpretation of the coordinates are equivalent [30, 37, 38] .

4.5. Contribution and practical significance of the pipeline

The practical contribution of the pipeline lies in providing a consolidated Python workflow for bacterial codon-usage analysis and proteome-wide physicochemical characterisation. It brings together nucleotide-composition measures, codon-usage indices, codon-adaptation and optimal-codon measures, parity analyses, PCA-derived summaries, and protein-level properties within an organised computational workflow. Its two-sheet output separates codon-usage results from protein-characterisation results, facilitating access to complementary measurements without requiring users to assemble separate output files manually. By integrating codon-usage, nucleotide-composition, and protein-level characteristics, the workflow can also provide a computational basis

for investigating sequence-level patterns associated with bacterial evolution, including patterns relevant to adaptation, antibiotic resistance, and pathogenicity [15–17, 39, 40] .

The application of the pipeline to *A. baumannii*, *P. aeruginosa*, and *E. anophelis* demonstrates its use across three bacterial taxa and provides a structured basis for comparative sequence analysis. However, integration alone does not establish methodological novelty, particularly because existing software packages (such as CodonW and DAMBE) already implement overlapping codon-usage analyses [9, 36] . The pipeline's contribution should therefore be evaluated through its specific analytical coverage, output organisation, reproducibility, and performance relative to existing tools. A feature-by-feature comparison using documented software versions and matched input sequences would clarify which functions are shared with established packages and whether the present implementation offers a distinct practical advantage.

The cross-study comparisons provide an initial assessment of numerical correspondence, but they represent only one aspect of software evaluation. Reproducibility also depends on explicit calculation definitions, documented sequence-selection criteria, and clearly specified reference settings. In particular, nucleotide-composition denominators, ribosomal reference data for CAI, frequency of optimal codon definitions, protein-length summaries, and PCA score aggregation should be documented so that users can interpret and reproduce the reported outputs. These details are essential for distinguishing the convenience of an integrated workflow from any demonstrable improvement in analytical coverage or computational performance.

4.6. Limitations

This assessment is based on three bacterial datasets and previously published genome-wide summary values. The available comparisons do not establish that all studies used identically filtered sequence sets, calculation settings, or aggregation procedures. In addition, published summary values do not constitute an independent reference standard unless the underlying definitions, software tools (e.g., CodonW vs. DAMBE), and analytical procedures are sufficiently comparable. The results should therefore have been interpreted as a cross-study assessment of numerical correspondence rather than as comprehensive validation of every pipeline output [9, 36] .

The present comparison also does not, on its own, establish superiority over existing codon-usage software. Such a conclusion would require a systematic comparison of available functions, matched-input outputs, reproducibility, and relevant practical features. The pipeline's contribution is therefore most appropriately framed around its consolidated workflow and structured output, with any additional methodological or performance advantage established through direct comparative evidence. Overall, it's a single click python pipeline for the mentioned analysis.

Despite these limitations, the analysis demonstrates the application of the workflow to three bacterial datasets and identifies both measures showing close numerical correspondence and parameters for which calculation conventions require particular attention. Therefore, this work frames the contribution around integration of multiple indices together with transparent outputs, and reproducible analysis providing an appropriately scoped foundation for evaluating the pipeline as a practical resource for bacterial codon-usage and proteome-wide

characterisation. When interpreted together with appropriate genomic, phenotypic, and functional evidence, these integrated sequence- and protein-level measurements further support investigations of evolutionary processes underlying bacterial adaptation, antibiotic resistance, and pathogenicity [16, 17, 39, 40] .

Reference

1. Dahal U, Bansal A (2025) Unravelling Prokaryotic Codon Usage: Insights from Phylogeny, Influencing Factors and Pathogenicity. *Curr Genomics* 26:81–94. <https://doi.org/10.2174/0113892029325491240919151045>
2. Kaleem S, Dahal U, Devi S, et al (2025) Codon Usage Evolution in Viruses: Implications for Survival and Pathogenicity. *J Mol Evol* 93:553–580. <https://doi.org/10.1007/s00239-025-10263-7>
3. Dahal U, Shakya R, Kour B, et al (2026) Evolution and Functional Implications of Codon Usage Bias in Eukaryotes. *J Mol Evol* 94:286–303. <https://doi.org/10.1007/s00239-026-10305-8>
4. Dahal U, Bansal A (2025) Codon usage and antibiotic resistance: A hidden evolutionary mechanism. *Biochimie* 238:112–124. <https://doi.org/10.1016/j.biochi.2025.07.027>
5. Sharp PM, Li W-H (1986) An evolutionary perspective on synonymous codon usage in unicellular organisms. *J Mol Evol* 24:28–38. <https://doi.org/10.1007/BF02099948>
6. Sharp PM, Li W-H (1987) The codon adaptation index-a measure of directional synonymous codon usage bias, and its potential applications. *Nucleic Acids Res* 15:1281–1295. <https://doi.org/10.1093/nar/15.3.1281>
7. Johnson JS, Spakowicz DJ, Hong B-Y, et al (2019) Evaluation of 16S rRNA gene sequencing for species and strain-level microbiome analysis. *Nat Commun* 10:5029. <https://doi.org/10.1038/s41467-019-13036-1>
8. Wright F (1990) The ‘effective number of codons’ used in a gene. *Gene* 87:23–29. [https://doi.org/10.1016/0378-1119\(90\)90491-9](https://doi.org/10.1016/0378-1119(90)90491-9)
9. Xia X (2018) DAMBE7: New and Improved Tools for Data Analysis in Molecular Biology and Evolution. *Mol Biol Evol* 35:1550–1552. <https://doi.org/10.1093/molbev/msy073>
10. Rice P, Longden I, Bleasby A (2000) EMBOSS: The European Molecular Biology Open Software Suite. *Trends in Genetics* 16:276–277. [https://doi.org/10.1016/S0168-9525\(00\)00204-2](https://doi.org/10.1016/S0168-9525(00)00204-2)
11. Choudhuri S, Sau K (2024) CodonU: A Python Package for Codon Usage Analysis. *IEEE/ACM Trans Comput Biol Bioinform* 21:36–44. <https://doi.org/10.1109/TCBB.2023.3335823>
12. Parvathy ST, Udayasuriyan V, Bhadana V (2022) Codon usage bias. *Mol Biol Rep* 49:539–565. <https://doi.org/10.1007/s11033-021-06749-4>

13. Guan D-L, Ma L-B, Khan MS, et al (2018) Analysis of codon usage patterns in *Hirudinaria manillensis* reveals a preference for GC-ending codons caused by dominant selection constraints. *BMC Genomics* 19:542. <https://doi.org/10.1186/s12864-018-4937-x>
14. Dahal U, Bansal A, Chitara D (2024) Decoding Multidrug Resistance: Genetic Architecture and Codon Usage Patterns in ESKAPE Pathogens. *Curr Pharmacogenomics Person Med* 21:179–198. <https://doi.org/10.2174/0118756921344687241015063919>
15. Dahal U, Sharma A, Paul K, et al (2025) Distinct codon usage signatures reflecting evolutionary and pathogenic adaptation in the *Acinetobacter baumannii* complex. *European Journal of Clinical Microbiology & Infectious Diseases* 44:2803–2828. <https://doi.org/10.1007/s10096-025-05233-2>
16. Dahal U, Bansal A, Kour B, et al (2026) Mapping the evolutionary and translational landscape of antibiotic resistance genes in *Elizabethkingia anopheles*. *Molecular Genetics and Genomics* 301:67. <https://doi.org/10.1007/s00438-026-02394-3>
17. Dahal U, Gautam A, Arora P, et al (2026) Isoelectric point stabilization and protein mass compression are associated with antimicrobial resistance in the genus *Acinetobacter*. *World J Microbiol Biotechnol* 42:568. <https://doi.org/10.1007/s11274-026-05295-0>
18. Jukes TH, Osawa S (1993) Evolutionary changes in the genetic code. *Comparative Biochemistry and Physiology Part B: Comparative Biochemistry* 106:489–494. [https://doi.org/10.1016/0305-0491\(93\)90122-L](https://doi.org/10.1016/0305-0491(93)90122-L)
19. Osawa S, Jukes TH, Watanabe K, Muto A (1992) Recent evidence for evolution of the genetic code. *Microbiol Rev* 56:229–264. <https://doi.org/10.1128/mr.56.1.229-264.1992>
20. (1985) Codon usage and tRNA content in unicellular and multicellular organisms. *Mol Biol Evol.* <https://doi.org/10.1093/oxfordjournals.molbev.a040335>
21. Ikemura T (1981) Correlation between the abundance of *Escherichia coli* transfer RNAs and the occurrence of the respective codons in its protein genes. *J Mol Biol* 146:1–21. [https://doi.org/10.1016/0022-2836\(81\)90363-6](https://doi.org/10.1016/0022-2836(81)90363-6)
22. Sueoka N (1995) Intrastrand parity rules of DNA base composition and usage biases of synonymous codons. *J Mol Evol* 40:318–325. <https://doi.org/10.1007/BF00163236>
23. Sueoka N (1999) Translation-coupled violation of Parity Rule 2 in human genes is not the cause of heterogeneity of the DNA G+C content of third codon position. *Gene* 238:53–58. [https://doi.org/10.1016/S0378-1119\(99\)00320-0](https://doi.org/10.1016/S0378-1119(99)00320-0)
24. Gouy M, Gautier C (1982) Codon usage in bacteria: correlation with gene expressivity. *Nucleic Acids Res* 10:7055–7074. <https://doi.org/10.1093/nar/10.22.7055>
25. Gasteiger E, Hoogland C, Gattiker A, et al (2005) Protein Identification and Analysis Tools on the ExPASy Server. In: *The Proteomics Protocols Handbook*. Humana Press, Totowa, NJ, pp 571–607

26. Bjellqvist B, Hughes GJ, Pasquali C, et al (1993) The focusing positions of polypeptides in immobilized pH gradients can be predicted from their amino acid sequences. *Electrophoresis* 14:1023–1031. <https://doi.org/10.1002/elps.11501401163>
27. Kozłowski LP (2016) IPC – Isoelectric Point Calculator. *Biol Direct* 11:55. <https://doi.org/10.1186/s13062-016-0159-9>
28. Kyte J, Doolittle RF (1982) A simple method for displaying the hydropathic character of a protein. *J Mol Biol* 157:105–132. [https://doi.org/10.1016/0022-2836\(82\)90515-0](https://doi.org/10.1016/0022-2836(82)90515-0)
29. Lobry JR, Gautier C (1994) Hydrophobicity, expressivity and aromaticity are the major trends of amino-acid usage in 999 *Escherichia coli* chromosome-encoded genes. *Nucleic Acids Res* 22:3174–3180. <https://doi.org/10.1093/nar/22.15.3174>
30. Jolliffe IT, Cadima J (2016) Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 374:20150202. <https://doi.org/10.1098/rsta.2015.0202>
31. Grantham R, Gautier C, Gouy M, et al (1980) Codon catalog usage and the genome hypothesis. *Nucleic Acids Res* 8:197–197. <https://doi.org/10.1093/nar/8.1.197-c>
32. Cock PJA, Antao T, Chang JT, et al (2009) Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics* 25:1422–1423. <https://doi.org/10.1093/bioinformatics/btp163>
33. Harris CR, Millman KJ, van der Walt SJ, et al (2020) Array programming with NumPy. *Nature* 585:357–362. <https://doi.org/10.1038/s41586-020-2649-2>
34. McKinney W (2010) Data Structures for Statistical Computing in Python. pp 56–61
35. Puigbò P, Bravo IG, Garcia-Vallve S (2008) CAIcal: A combined set of tools to assess codon usage adaptation. *Biol Direct* 3:38. <https://doi.org/10.1186/1745-6150-3-38>
36. Peden JF (1999) Analysis of codon usage (Doctoral dissertation, University of Nottingham)
37. Ringnér M (2008) What is principal component analysis? *Nat Biotechnol* 26:303–304. <https://doi.org/10.1038/nbt0308-303>
38. Ma S, Dai Y (2011) Principal component analysis based methods in bioinformatics studies. *Brief Bioinform* 12:714–722. <https://doi.org/10.1093/bib/bbq090>
39. Arora P, Mukhopadhyay CS, Kaur S (2024) Comparative genome wise analysis of codon usage of *Staphylococcus* Genus. *Curr Genet* 70:10. <https://doi.org/10.1007/s00294-024-01297-3>
40. Sharma A, Gupta S, Paul K (2023) Codon usage behavior distinguishes pathogenic *Clostridium* species from the non-pathogenic species. *Gene* 873:147394. <https://doi.org/10.1016/j.gene.2023.147394>