Spectral Stat Analysis in Genshin Impact Character Build Optimization Using Eigenvalue

Bernhard Aprillio Pramana 13524074[1,2]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13524074@std.stei.itb.ac.id, [2]piyips96@gmail.com

Abstract*—Character build optimization in Genshin Impact involves many synergic statistics that make it difficult to determine which attributes are the most important and strongly affect build effectiveness. This paper applies eigenvalue-based spectral analysis to study some character builds in Genshin Impact through linear algebra approach. Each build is represented as a vector in a real vector space, and a covariance matrix for multiple character build. Eigenvalue decomposition is used to identify dominant directions of different stat combinations. The results show that most builds can be explained by eigenvalues indicating redundancy of the build itself. This study demonstrates how eigenvalues and eigenvectors can provide meaningful insight into complex game mechanics without trial-and-error.

***Keywords*—Eigenvalue Analysis, Spectral Analysis, Linear Algebra, Genshin Impact build**

I. Introduction

Genshin Impact is an action role-playing game that features a complex character progression system. Each character's performance is determined by multiple statistics, including Attack (ATK), Defense (DEF), Hit Points (HP), Elemental Mastery (EM), Critical Rate (CR), Critical Damage (CD), Energy Recharge (ER), and Elemental or Physical Damage Bonus. Players improve these statistics through artifacts, weapons, talent upgrades, and character leveling. Optimizing a character build is challenging because these statistics interact with each other meaning increasing one attribute often reduces the opportunity to improve another. Furthermore, different characters scale on different primary stats. For example character like Hu Tao and Furina scale strongly on HP, while other character like Albedo and Itto scale strongly on DEF. Some character also benefit most with stat like EM. As a result, identifying which statistics are most influential is not straightforward.

In practice, many players rely on damage formulas, online calculators, or trial-and-error experimentation. While effective, these approaches often focus on final damage output and do not clearly reveal structural relationships between statistics.

From a mathematical perspective, especially linear algebra, a character build can be represented as a vector in a real vector space, where each dimension shows specific character statistics. A collection of builds then forms a dataset that can be analyzed using matrix-based methods. Eigenvalue-based spectral analysis shows variation in builds is distributed across different combinations of statistics.

This paper applies eigenvalue analysis to character stat data from *Genshin Impact*. The objective is not to calculate exact damage values, but to understand how character builds vary and which combinations of statistics dominate that variation.

II. Theoretical Background

A. Vector Spaces and Euclidean Vectors

In linear algebra, a vector space is a mathematical structure consisting of a set of elements called vectors, together with two operations, vector addition and scalar multiplication. These operations must satisfy closure, associativity, distributivity, and the existence of a zero vector and additive inverses [1], [2].

One of the most important examples is the real vector space $\mathbb{R}^n$, where each vector is an ordered tuple of real numbers:

$$x = (x1, x2, ..., xn),\ xi \in R$$

This representation is useful for modeling systems with multiple numerical attributes because each component of the vector corresponds to one independent variable [1]. In the context of this paper, each coordinate corresponds to one character statistic. Vector spaces provide a way to analyze multidimensional data because linear combinations of vectors remain within the same space.

B. Euclidean Space and Inner Product

A Euclidean space is a real vector space equipped with an inner product. The inner product introduces geometric concepts such as length, angle, and orthogonality [1]. For vectors x, y $\in \mathbb{R}^n$, the standard inner product is defined as:

$$\langle x, y\rangle = x_1y_1 + x_2y_2 + \ldots + x\square y\square$$

Using the inner product, the norm (or length) of a vector is defined as:

$$\|x\| = \sqrt{\langle x, x \rangle}$$

The norm measures the magnitude of a vector, while the inner product allows the measurement of angles between vectors. Two vectors are said to be orthogonal if their inner product is zero. Inner products allow vectors to be compared geometrically and enable projections onto specific directions. In eigenvalue analysis, projections are used to measure how much variance in a particular eigenvector direction [3].

C. Matrices and Linear Transformations

Matrices are used to represent linear transformations between vector spaces. Matrix A of size m × n, a linear transformation T from R^n to R^m can be written in matrix form as [2]:

$$T(x) = Ax$$

A transformation is linear if it satisfies the following properties for all vectors x, y and scalar c:

$$T(x + y) = T(x) + T(y)$$
$$T(c\,x) = c\,T(x)$$

Linear transformations preserve the structure of vector spaces and describe how changes in one vector affect another. Many important properties of a system can be understood by studying the matrix that represents the transformation [2]. In this paper, the covariance matrix acts as a linear transformation applied to character build vectors.

D. Symmetric Matrices and Covariance Matrices

A square matrix A is called symmetric if it is equal to its transpose:

$$A = A^T$$

Symmetric matrices have several important theoretical properties. Most notably, all eigenvalues of a symmetric matrix are real, and eigenvectors corresponding to different eigenvalues are orthogonal [2], [3].

These properties are critical for spectral analysis because they guarantee that the eigenvalue decomposition is stable. The orthogonality of eigenvectors allows the data to be decomposed into independent directions of variation.

E. Covariance Matrix

The covariance matrix is a symmetric matrix that describes how different variables vary together. Given a data matrix X consisting of m observations, the covariance matrix is defined as:

$$C = ((X - \bar{X})T(X - \bar{X}))/(m - 1)$$

where $\bar{X}$ is the mean vector of the dataset [4].

Each entry Cij represents the covariance between variables i and j. A positive value indicates that the two variables tend to increase together, while a negative value indicates an inverse relationship. Each entry Cij represents the covariance between variables i and j.

F. Eigenvalues and Eigenvectors

Eigenvalues and eigenvectors describe how a linear transformation behaves along specific directions. For a square matrix A, a non-zero vector v is an eigenvector if:

$$Av = \lambda v$$

where λ is the corresponding eigenvalue [1][2].

This equation means that applying the transformation A to v only scales the vector without changing its direction. In the context of the covariance matrix, the eigenvalue λ represents the variance of the data along the eigenvector direction v.

For symmetric matrices, eigenvectors related to different eigenvalues are orthogonal, and eigenvalues are real [2][3]. Larger eigenvalues indicate directions that capture more variance in the data. This property forms the basis of spectral analysis and principal component analysis.

G. Spectral Interpretation

Spectral analysis refers to the study of a matrix through its eigenvalues and eigenvectors. For symmetric matrices such as covariance matrices, eigenvalue decomposition provides an orthogonal basis that reveals the internal structure of the data [2][3].

The eigenvectors of a covariance matrix represent directions, while eigenvalues measure the magnitude of that variation [2]. By ordering eigenvalues from largest to smallest, it is possible to identify a reduced set of dominant directions that approximate the original data with minimal loss of information [3].

In this study, spectral analysis is used to identify dominant combinations of character statistics. Instead of analyzing each statistic separately, eigenvectors provide linear combinations of attributes that represent meaningful build patterns across different characters. Eigenvalues then quantify how strongly each pattern contributes to overall build variation.

III. Character Build Modeling

A. Mathematical Representation of Character Builds

In *Genshin Impact*, each character build is defined by multiple attributes that describe combat performance. From a linear algebra perspective, these attributes can be treated as components of a vector in a real vector space.

In this study, a character build is represented as an

element of the vector space $\mathbb{R}^8$:

X = (ATK, HP, DEF, CR, CD, EM, ER, DMG)
where:

- ATK denotes Attack,
- HP denotes Hit Points,
- DEF denotes Defense,
- CR denotes Critical Rate,
- CD denotes Critical Damage,
- EM denotes Elemental Mastery,
- ER denotes Energy Recharge,
- DMG denotes Elemental or Physical Damage Bonus.

Each component corresponds to one dimension of the vector space. This representation allows different character builds to be compared and analyzed using vector and matrix operations, independent of the specific damage formulas used in the game.

B. Selection of Statistics

The selected statistics are chosen because they directly influence character performance and are commonly used in build optimization. Importantly, the model does not assume that all characters rely on the same primary scaling stat.

- ATK is dominant for most damage dealers.
- HP is relevant for HP-scaling characters such as Hu Tao and Furina.
- DEF is important for DEF-scaling characters such as Albedo.
- CR and CD describe critical damage behavior.
- EM affects elemental reaction damage.
- ER controls skill and burst uptime.
- DMG represents elemental or physical damage bonuses.

By including all these attributes, the model can represent a wide range of character roles within a single vector space.

C. Dataset Construction

The dataset is constructed using character level 90 base statistics combined with realistic optimized build values commonly used by players. The values are obtained from a random number generator based on real game interval statistics. Based on the selected characters, a dataset is constructed using realistic build values commonly observed in actual gameplay. The resulting dataset is shown conceptually as a data matrix:

$$X = \begin{bmatrix} \text{ATK}_1 & \text{HP}_1 & \text{DEF}_1 & \cdots & \text{DMG}_1 \\ \text{ATK}_2 & \text{HP}_2 & \text{DEF}_2 & \cdots & \text{DMG}_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \text{ATK}_n & \text{HP}_n & \text{DEF}_n & \cdots & \text{DMG}_n \end{bmatrix}$$

where each row corresponds to one character build and each column corresponds to one statistic.

Because different statistics use different units and numerical scales for example HP base is way higher than ATK or DEF, and critical rate, critical damage, energy recharge and damage bonus based on percentage, the raw data cannot be analyzed directly without bias.

D. Data Normalization

To ensure that all statistics make a fair contribution to the analysis, z-score normalization is applied independently to each statistic using this formula:

$$z_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j}$$

- xij is the value of statistic j for character i,
- μj is the mean of statistic j,
- σ j is the standard deviation of statistic j.

Normalization ensures that the analysis focuses on relative variation rather than absolute magnitude. This step is crucial because eigenvalue analysis is sensitive to scale differences across dimensions.

E. Covariance Matrix Construction

After normalization, the covariance matrix is computed from the normalized data matrix Z:

$$C = \frac{1}{n-1} Z^T Z$$

The covariance matrix captures how different statistics vary together across character builds. Since the covariance matrix is symmetric and positive semi-definite, it satisfies the theoretical requirements for eigenvalue decomposition.

Each entry Cij represents the degree to which statistic i and statistic j vary jointly. High covariance indicates that two statistics tend to increase or decrease together across different builds.

F. Purpose of the Model

The goal of this modeling approach is not to evaluate damage directly, but to analyze the structural relationships between statistics. By studying the covariance matrix and its spectral decomposition, dominant patterns in character build design can be identified.

This framework allows eigenvalues and eigenvectors to be interpreted as eigenvalue and eigenvector. Eigenvalues are the amount of variation explained by a statistical direction, while eigenvectors are the combinations of statistics that characterize dominant build patterns.

IV. Experimental Analysis and Implementation

A. Experimental Objective

The purpose of this experiment is to apply eigenvalue-based spectral analysis to character build statistics in *Genshin Impact* and to examine the structural relationships between different attributes. Unlike approaches that rely on damage formulas, this study focuses on how statistics interact within a vector space framework.

All explanations related to eigenvalues, explained variance, and principal components are intentionally consolidated in this chapter to provide an interpretation of the experimental results. The goal is to demonstrate how abstract concepts from linear algebra can be meaningfully interpreted within the context of real in-game mechanics, such as artifact limitations, energy requirements, critical optimization, and reaction-based damage.

B. Dataset Construction

The analysis is performed on a dataset generated using realistic base character statistics, artifact main-stat rules, and a discrete 20-roll substat simulation that reflects in-game artifact enhancement mechanics. In addition to reporting numerical results, this chapter also demonstrates how principal component analysis (PCA) can be used to generate interpretable and gameplay-relevant build recommendations.

The dataset consists of six characters chosen to represent distinct damage scaling mechanisms. For each character, 100 build variations were generated under intervals that reflect actual gameplay statistics. Each build is represented as an eight-dimensional vector containing ATK, HP, DEF, CR, CD, EM, ER, and DMG. There are six characters that this paper tested:

- Mavuika: ATK scaling DPS that does not require Energy Recharge (ER)
- Ganyu: conventional ATK-scaling DPS
- Flins: Reaction-based DPS using ATK dan EM as implicit damage multiplier
- Furina: HP-scaling character
- Hu Tao: HP-to-ATK conversion scaling
- Albedo: DEF-scaling character

C. Data preprocessing

Before performing eigenvalue decomposition, the raw character build data must be prepared to ensure that the analysis reflects meaningful statistical structure. Without preprocessing, attributes with larger numeric ranges would dominate the covariance matrix and distort the eigenvalue analysis. Therefore, preprocessing is required to ensure that each statistic contributes proportionally to the analysis.

To address this issue, z-score normalization is applied to each attribute using python code

```
features =
["ATK","HP","DEF","CR","CD","EM","E
R","DMG"]
columns=columns)
X =
StandardScaler().fit_transform(df[f
eatures])
```

This transformation centers each attribute around zero and scales it to unit variance. As a result, all attributes become dimensionless and comparable within the same vector space. After normalization, each character build is represented as a vector in an 8-dimensional standardized space:

$$z = (zATK, zHP, zDEF, zCR, zCD, zEM, zER, zDMG)$$

From the normalized dataset, a covariance matrix is constructed to capture the relationships between different attributes

```
cov_matrix = np.cov(X,
rowvar=False)
```

By standardizing the data before constructing the covariance matrix, the eigenvalues obtained from the decomposition reflect true structural variance rather than dominance by large-magnitude attributes. Large eigenvalues correspond to dominant combinations of statistics and small eigenvalues indicate redundancy or weak variation. This preprocessing step ensures that the eigenvectors extracted in the next stage represent meaningful directions in the character build space.

After the covariance matrix C is obtained from the standardized dataset, eigenvalue decomposition is applied to extract the principal statistical structures of character builds.

```
cov = np.cov(X, rowvar=False)
eigval, eigvec =
np.linalg.eigh(cov)
```

This experiment focuses on six characters representing different stat scaling mechanisms: ATK-based, HP-based, DEF-based, hybrid, and resource-independent characters.

Using a python program and importing numpy and pandas to demonstrate linear algebra.

```
import numpy as np
```

```
import pandas as pd
from sklearn.preprocessing import
StandardScaler

np.random.seed(67)
# ==============================
# 1. DATA GENERATION
# ==============================
N = 100  # builds per character

characters = {
    "Mavuika":
{"HP":12552,"ATK":359,"DEF":792,"CR
":0.05,"CD":1.384,"ER":1.0,"EM":0,"
DMG":0,"scale":"ATK"},
    "Ganyu":
{"HP":9797,"ATK":335,"DEF":630,"CR"
:0.05,"CD":1.384,"ER":1.0,"EM":0,"D
MG":0,"scale":"ATK"},
    "HuTao":
{"HP":15552,"ATK":106,"DEF":876,"CR
":0.05,"CD":1.384,"ER":1.0,"EM":0,"
DMG":0,"scale":"HP"},
    "Furina":
{"HP":15307,"ATK":244,"DEF":696,"CR
":0.292,"CD":0.5,"ER":1.0,"EM":0,"D
MG":0,"scale":"HP"},
    "Flins":
{"HP":12491,"ATK":352,"DEF":809,"CR
":0.05,"CD":1.384,"ER":1.0,"EM":0,"
DMG":0,"scale":"ATK"},
    "Albedo":
{"HP":13226,"ATK":251,"DEF":876,"CR
":0.05,"CD":0.5,"ER":1.0,"EM":0,"DM
G":0.288,"scale":"DEF"}
}

SUBSTAT_POOL = {
    "HP_flat":  [209, 239, 269],
    "ATK_flat": [14, 16, 18],
    "DEF_flat": [16, 19, 21],
    "HP_pct":   [0.041, 0.047,
0.053],
    "ATK_pct":  [0.041, 0.047,
0.053],
    "DEF_pct":  [0.051, 0.058,
0.066],
    "EM":       [16, 19, 23],
    "ER":       [0.045, 0.052,
0.058],
    "CR":       [0.027, 0.031,
0.035],
    "CD":       [0.054, 0.062,
0.070]
}

def random_substats_20rolls():
    result = {k: 0.0 for k in
SUBSTAT_POOL}

    rolls =
np.random.choice(list(SUBSTAT_POOL.
keys()), size=20)

    for stat in rolls:
        result[stat] +=
np.random.choice(SUBSTAT_POOL[stat]
)

    return result

data = []

for char, base in
characters.items():
    for _ in range(N):
        HP, ATK, DEF = base["HP"],
base["ATK"], base["DEF"]
        CR, CD, ER = base["CR"],
base["CD"], base["ER"]
        EM, DMG = base["EM"],
base["DMG"]
        HP1, ATK1, DEF1 = 1.0, 1.0,
1.0

        # ======================
```

```
        # SUBSTATS FIRST
        # ======================
        sub =
random_substats_20rolls()

        HP  += sub["HP_flat"]
        ATK += sub["ATK_flat"]
        DEF += sub["DEF_flat"]

        HP1  = (1 + sub["HP_pct"])
        ATK1 = (1 + sub["ATK_pct"])
        DEF1 = (1 + sub["DEF_pct"])

        EM += sub["EM"]
        ER += sub["ER"]
        CR += sub["CR"]
        CD += sub["CD"]

        # Flower & Feather
        HP += 4780
        ATK += 311

        # Sands
        if base["scale"] == "HP":
            HP1 += 0.466
        elif base["scale"] ==
"DEF":
            DEF1 += 0.466
        elif base["scale"] ==
"ATK":
            ATK1 += 0.466
        # Goblet
        if np.random.rand() < 0.5:
            # scaling main stat
            if base["scale"] ==
"HP":
                HP *= 1.466
            elif base["scale"] ==
"DEF":
                DEF *= 1.466
            elif base["scale"] in
["ATK", "ATK_EM"]:
                ATK *= 1.466
        else:
            # DMG bonus
            if char == "Flins":
                EM += 187
            else:
                DMG += 0.466

        # Circlet (balance CR/CD)
        if CR < 0.5:
            CR += 0.311
        else:
            CD += 0.622

        HP *= HP1
        ATK *= ATK1
        DEF *= DEF1

        # ======================
        # STATS SECOND
        # ======================

        data.append([char, ATK, HP,
DEF, CR, CD, EM, ER, DMG])

features =
["ATK","HP","DEF","CR","CD","EM","E
R","DMG"]

# ==============================
# 2. PREPROCESSING
# ==============================
X =
StandardScaler().fit_transform(df[f
eatures])

# ==============================
# 3. COVARIANCE MATRIX
# ==============================
cov_matrix = np.cov(X,
rowvar=False)

# ==============================
# 4. EIGEN DECOMPOSITION
```

```
# ==============================
cov = np.cov(X, rowvar=False)
eigval, eigvec =
np.linalg.eigh(cov)

# Sort descending
idx = np.argsort(eigval)[::-1]
eigval = eigval[idx]
eigvec = eigvec[:, idx]

explained_var = eigval /
eigval.sum() * 100

# ==============================
# 5. OUTPUT RESULTS
# ==============================

# Eigenvalues table
eigen_df = pd.DataFrame({
    "Eigenvalue": eigval,
    "Explained Variance (%)":
explained_var
})

# Eigenvectors table (first 3
components)
eigvec_df = pd.DataFrame(
    eigvec[:, :3],
    index=columns,
    columns=["Eigenvector 1",
"Eigenvector 2", "Eigenvector 3"]
)

print("\n=== Eigenvalues and
Explained Variance ===\n")
print(eigen_df.round(3))

print("\n=== Dominant Eigenvectors
(Top 3) ===\n")
print(eigvec_df.round(3))
```

With this implementation of random simulation data, resulting following eigenvalues and explained variance ratios

```
=== Eigenvalues and Explained Variance ===

   Eigenvalue  Explained Variance (%)
0       1.448                  18.076
1       1.294                  16.153
2       1.123                  14.016
3       1.058                  13.208
4       0.963                  12.022
5       0.862                  10.756
6       0.754                   9.405
7       0.510                   6.363

=== Dominant Eigenvectors (Top 3) ===

     Eigenvector 1  Eigenvector 2  Eigenvector 3
ATK         -0.229          0.610         -0.331
HP          -0.181         -0.413         -0.526
DEF         -0.285         -0.299          0.059
CR           0.026          0.310          0.549
CD          -0.169         -0.133          0.486
EM          -0.527         -0.052          0.193
ER          -0.288         -0.406          0.163
DMG          0.667         -0.293          0.093
```

The largest eigenvalue explains approximately 18.08% of the total variance, followed by 16.15% and 14.02% for the second and third eigenvalues, respectively. Collectively, the first three principal components explain nearly 48% of the total variance. This indicates that although character builds are influenced by multiple interacting statistics, a relatively low-dimensional structure captures most of the meaningful variation.

Unlike problems where a single dominant eigenvalue exists, the gradual decay of eigenvalues here reflects the inherently multi-objective nature of character optimization in *Genshin Impact*. No single statistic fully determines build effectiveness; instead, multiple trade-offs coexist.

From a linear algebra perspective, this result suggests that the build space is not strongly low-dimensional, and multiple independent factors are required to describe build differences.

The first principal component (Eigenvector 1) is primarily characterized by a strong positive loading on DMG Bonus and a strong negative loading on Elemental Mastery (EM), with secondary contributions from ATK, HP, and DEF. This component represents a damage expression axis, distinguishing builds that rely on elemental or bonus-based damage amplification from those that emphasize raw stat scaling.

The second principal component (Eigenvector 2) shows large positive contributions from ATK and Critical Rate, and negative contributions from HP, DEF, and Energy Recharge (ER). This component reflects an energy constraint and offensive trade-off axis, separating builds that prioritize offensive uptime from those constrained by burst energy requirements.

The third principal component (Eigenvector 3) is dominated by Critical Rate and Critical Damage, with a smaller contribution from HP. This axis captures critical investment intensity, indicating how aggressively a build prioritizes critical optimization relative to other statistics.

Lower-order components continue to capture finer

variations in stat allocation but contribute less to the overall variance. Their presence reinforces the idea that character builds cannot be reduced to a small set of simple rules, as multiple layers of trade-offs coexist within the system.

Negative coefficients in eigenvectors do not imply negative importance. Instead, they indicate directional trade-offs meaning increasing one group of statistics naturally reduces emphasis on another under fixed resource constraints.

To connect the spectral structure with gameplay interpretation, each build is projected onto the principal component space. For each character, the mean values of PC1, PC2, and PC3 across 100 simulated builds are computed.

Mavuika exhibits near-neutral PC2 values, confirming that Energy Recharge is not a meaningful constraint for her gameplay. Her builds distribute more evenly across damage and critical axes.

Ganyu shows moderate values across PC1 and PC2, indicating flexible goblet choices and situational ER requirements depending on rotation style.

Hu Tao and Furina display strongly negative PC2 values, reflecting that their rotations are not ER-intensive, while scaling efficiency is driven by HP-related mechanics.

Flins has extremely high PC1 and PC2 values, indicating that EM-based damage replacement and strict energy requirements dominate his build decisions.

Albedo shows a strongly negative PC1, consistent with DEF-scaling dominance rather than damage bonus reliance.

Beyond interpretation, the principal component results are used to generate automated build recommendations. Unlike rule-based systems that rely on predefined formulas, the recommendations are derived directly from the relative positions of each character in principal component space.

```
    # ====================
    # Goblet Decision (PC1)
    # ====================
    if pc1 <= PC1_LOW:
        rec.append("Prefer main-scaling Goblet (HP/ATK/DEF)")
    elif pc1 >= PC1_HIGH:
        if char == "Flins":
            rec.append("EM Goblet preferred (reaction scaling)")
        else:
            rec.append("DMG% Goblet preferred")
    else:
        rec.append("Goblet choice flexible depending on substats")

    # ====================
    # Energy Constraint (PC2)
    # ====================
    if needs_er:
        if pc2 >= PC2_HIGH:
            rec.append("ER is a gameplay constraint (burst every rotation)")
        else:
            rec.append("Moderate ER sufficient")
    else:
        rec.append("ER investment unnecessary")

    # ====================
    # Critical Stats (PC3)
    # ====================
    rec.append("CR-CD optimization remains a core priority")

    return ", ".join(rec)


summary_df["Build Recommendation"] = [
    recommend_character_build(
        summary_df.loc[c, "PC1_mean"],
        summary_df.loc[c, "PC2_mean"],
        summary_df.loc[c, "PC3_mean"],
        c
    )
    for c in summary_df.index
]
pd.set_option("display.max_colwidth", None)
```

```
print("\n=== Corrected Final Build Recommendations ===\n")
print(summary_df[["Build Recommendation"]])
```

The resulting recommendations are summarized below:

Mavuika: Flexible goblet choice depending on substats; ER investment unnecessary; CR–CD optimization remains a core priority.

Ganyu: Flexible goblet choice; ER is a gameplay constraint for consistent burst rotations; CR–CD optimization remains essential.

Hu Tao: DMG% goblet preferred; moderate ER sufficient; CR–CD optimization remains a core priority.

Furina: Main HP-scaling goblet preferred; moderate ER sufficient; CR–CD optimization remains important.

Flins: EM goblet preferred as a damage replacement; ER is a strict gameplay constraint; CR–CD optimization remains relevant.

Albedo: Main DEF-scaling goblet preferred; moderate ER sufficient; CR–CD optimization remains a core priority.

V. Conclusion

This paper demonstrates that eigenvalue-based spectral analysis can be effectively used to study character lineup optimization in Genshin Impact from a linear algebra perspective. By representing character lineups as vectors and analyzing their covariance structure, the method identifies dominant statistical trade-offs without relying on explicit damage formulas. The results indicate that lineup variation is arranged by several interacting factors, including damage expression, energy constraints, and critical statistical investment, reflecting the actual game mechanics. This confirms that eigenvalues and eigenvectors provide a meaningful mathematical framework for understanding and optimizing complex multi-parameter systems in games.

VI. Appendix

Source Code : https://colab.research.google.com/drive/1v3L6awmGqM4aazZRznhOVj8r2h5cg7Y7?usp=sharing

VII. Acknowledgment

The author would like to thank Sir Ir. Rila Mandala, M.Eng., Ph.D and other teaching staff of IF2123 the Linear Algebra and Geometry course for their guidance and learning materials, which provided the theoretical foundation for this work. Appreciation is also extended to the *Genshin Impact* player community for publicly available data and discussions that helped inform realistic modeling assumptions used in this study. The author would also give an appreciation to *Genshin Impact Characters,* especially Mavuika, Ganyu, Hutao, Furina, Albedo, Flins for accompanied author during the work of this paper.

References

[1] R. Munir, *Aljabar Geometri (IF2123)*, Institut Teknologi Bandung, 2025–2026. https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2025-2026/algeo25-26.htm (17 Desember 2025 00.10)

[2] G. Strang, *Linear Algebra and Its Applications*, 4th ed. Belmont, CA, USA: Brooks/Cole, 2006. https://vf-tropi.com/linearalgebraanditsapplications.pdf (20 Desember 2025 00.20)

[3] D. C. Lay, S. R. Lay, and J. J. McDonald, *Linear Algebra and Its Applications*, 5th ed. Boston, MA, USA: Pearson, 2016. https://api.pageplace.de/preview/DT0400.9781292092249_A26575335/preview-9781292092249_A26575335.pdf (20 Desember 2025 00.23)

[4] *Positive Semi-Definiteness of the Covariance Matrix*, The Book of Statistical Proofs. https://statproofbook.github.io/P/covmat-psd.html (23 Desember 2025 21.00)\

[5] https://game8.co/ (24 Desember 2025 16.00)

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 24 Desember 2025

Bernhard Aprillio Pramana 13524074