Advances in Education 教育进展, 2026, 16(5), 743-751
Published Online May 2026 in Hans. https://www.hanspub.org/journal/ae
https://doi.org/10.12677/ae.2026.165916

Hans汉斯

新工科背景下基于Python的《线性代数》教学模式探索与实践

罗　珊

广州应用科技学院计算机学院，广东　肇庆

收稿日期：2026年4月6日；录用日期：2026年5月8日；发布日期：2026年5月14日

摘　要

针对新工科背景下线性代数教学中“重计算、轻思想”及传统教学模式与学生主体性脱节的问题，本文旨在探索一种与Python深度融合的教学模式，将Python从辅助工具重塑为认知媒介。方法：以四阶行列式和逆矩阵为核心案例，设计递推法、多行展开验证、过程演示、伴随矩阵法及高斯－约当消元法等多种算法实现；并在两个学期共297名大一学生中开展教学实践，采用混合式教学(理论28学时 + 编程实践4学时)，通过大作业、学生反思报告和课堂观察进行质性评估。结果：超过80%的学生能成功将Python方法迁移至新应用场景；学生反思表明编程显著加深了对行列式唯一性、逆矩阵代数构成等概念的理解；课堂提问从“如何笔算”转向“如何选最优算法”。结论：该模式能有效促进学生从机械计算到计算思维的转变，为Python融入线性代数教学提供了可操作的实践路径。

关键词

新工科，线性代数，Python，计算思维

Exploration and Practice of a Python-Based Teaching Model for “Linear Algebra” in the Context of Emerging Engineering Education

Shan Luo

School of Computer Science, Guangzhou College of Applied Science and Technology, Zhaoqing Guangdong

Received: April 6, 2026; accepted: May 8, 2026; published: May 14, 2026

文章引用: 罗珊. 新工科背景下基于 Python 的《线性代数》教学模式探索与实践[J]. 教育进展, 2026, 16(5): 743-751.
DOI: 10.12677/ae.2026.165916

Abstract

To address the issues of "emphasizing computation over conceptual understanding" in linear algebra instruction and the disconnection between traditional teaching models and student-centered learning in the context of emerging engineering education, this paper aims to explore a teaching model deeply integrated with Python, repositioning Python from an auxiliary tool to a core cognitive medium. Methods: Taking fourth-order determinants and inverse matrices as typical cases, the study designs multiple algorithmic implementations, including recursive expansion, multi-row expansion verification, step-by-step process demonstration, adjugate matrix method, and Gauss-Jordan elimination. The model was implemented in two semesters involving 297 first-year students majoring in computer science and software engineering. A blended teaching approach (28 lecture hours + 4 Python practice hours) was adopted, and qualitative assessment was conducted through student projects, reflective reports, and classroom observations. Results: Over 80% of students successfully transferred the Python methods to new application scenarios. Student reflections revealed that programming significantly deepened their understanding of concepts such as the uniqueness of determinants and the algebraic structure of inverse matrices. Classroom questioning shifted from "how to compute manually" to "how to select optimal algorithms". Conclusion: The proposed model effectively facilitates students' transition from mechanical computation to computational thinking and from "tool usage" to "thinking construction", providing a practical pathway for integrating Python into linear algebra teaching.

Keywords

Emerging Engineering Education, Linear Algebra, Python, Computational Thinking

1. 引言

线性代数作为新工科人才培养体系中的重要数学基础，其理论和方法广泛应用于人工智能、数据科学等前沿领域。然而，传统的线性代数教学正面临双重挑战：其一，内容计算繁重，如四阶行列式、高阶逆矩阵及实对称矩阵特征值的求解过程繁琐，学生往往疲于精确的笔算，而忽略其作为“空间与变换的数学语言”的思维本质；其二，教学模式单一，以教师为中心的讲授模式难以调动学生的主观能动性，与“以学生为中心、成果为导向”(OBE)的教育理念相悖——OBE 要求教学目标可衡量、学习成果可评估、教学活动围绕学生达成预期成果而设计，而传统课堂中学生的角色主要是被动听讲与机械练习，缺乏主动建构知识的机会，也不利于复杂工程问题解决能力的培养。将 Python 等编程工具引入线性代数教学，已成为国内教学改革的重要趋势。现有研究主要围绕三个层面展开，呈现出从“工具应用”到“认知深化”的演进脉络。

1.1. 可视化与工具应用：作为化解抽象性的辅助手段

这是当前最主流的研究范式。大量研究将 Python 定位为实现抽象概念可视化和计算验证的实用工具。其核心目标是利用 Python 的绘图与计算库，将矩阵、特征向量等难以想象的对象转化为直观的图形或案例，以降低学习难度、激发学生兴趣。例如，有研究选取矩阵运算、特征值等典型知识点，结合人脸识别等应用案例，引入 Python 进行可视化教学设计，有效提升了学生的学习体验和成绩[1]。这类研究普

遍遵循“抽象概念-Python 可视化–实际应用”的路径，充分肯定了 Python 在增强教学直观性和联系工程实际方面的价值[2]。然而，其局限性也很明显：Python 主要作为“演示工具”呈现已知结论，学生仍然是被动的观看者；对于高阶计算(如四阶以上行列式的手工展开极易出错)和深层概念(如矩阵的秩与线性变换的几何意义)，简单的可视化难以揭示其内在算法逻辑，学生并未真正参与知识的生成过程。

1.2. 编程实践与教学改革：作为深化理解的实践环节

部分研究进一步将编程提升为教学过程中的一个实践环节，旨在通过“做中学”来深化理论理解。这类探索强调将编程实践(如使用 Python 或 MATLAB)有机融入课程，通过让学生动手实现算法、解决具体问题，来巩固和运用理论知识。有学者明确提出，在问题导向的教学中融入编程实践，能使教学过程更符合学生的认知规律，帮助学生理解理论背景与应用场景[3]。此类研究标志着从“被动观看”到“主动操作”的转变，Python 开始成为学生与数学知识交互的媒介。但深入分析可以发现，其根本目的仍是为了更好地服务传统教学目标的达成(如掌握算法步骤、验证定理结论)，编程任务往往是封闭的、验证性的(例如按照教材公式编写函数计算行列式)。学生虽然动手写了代码，但很少被引导去思考“为什么可以用不同方法计算行列式并得到相同结果”“消元法与伴随矩阵法在本质上反映了矩阵的哪些属性”等根本性问题。换言之，编程尚未成为重构教学内容与认知方式的核心载体。

1.3. 数学思想与建模融入：作为思想载体的初步探索

少数研究触及了更本质的层面，即关注数学思想方法本身或通过建模来培养数学能力。一方面，有研究专注于梳理和阐释线性代数中蕴含的核心数学思想(如化归思想、结构思想)，这为任何教学媒介的应用指明了深层次的内容目标[4]。另一方面，探索将数学建模思想融入课堂教学，通过从实际问题中提炼和解决线性代数模型，来加深学生对知识本质的理解[5]。这类研究为“通过编程阐释数学思想”提供了重要的思想基础和方法论借鉴——它表明，通过一种主动的认知活动(如建模)来建构理解是可行且有效的。然而，现有文献尚未将编程系统地、有意识地用于数学思想的辨析与阐释。例如，在逆矩阵的教学中，伴随矩阵法和高斯–约当消元法分别对应着“代数结构”与“行空间变换”两种不同的数学思想，但已有研究多分别介绍两种方法的编程实现，很少设计对比性任务让学生通过代码直接感受两种思想的内在联系与适用场景，从而错失了利用编程深化概念理解的宝贵机会。

综上所述，现有研究虽充分探索了 Python 在教学中的实用价值，但普遍存在以下具体缺陷：针对高阶计算负担(如四阶行列式、四阶逆矩阵的手工计算极易出错且耗时)，已有工具多直接调用现成库函数或仅给出最终数值结果，未能利用编程将计算过程分解为可理解的算法步骤，学生仍陷于“黑箱”操作；针对概念抽象问题(如行列式值的唯一性、逆矩阵的代数构成与行变换本质)，可视化或编程任务多为单向演示或封闭验证，学生缺少在“猜想–验证–调试–抽象”循环中主动建构概念的机会；缺乏对计算思维与数学思想深度融合的探索，编程与数学内容之间仍存在“两张皮”现象，学生学会的仅是孤立的技术操作，而非用编程语言去思考数学。

针对上述缺陷，本文提出并论证一种根本性的视角转换：将 Python 从教学的“辅助工具”重新定位为学习的“核心认知媒介”。本文的核心论点在于，编程不仅仅是一种技能或演示手段，更是一种独特的、强有力的思维语言和认知环境。通过精心设计的编程任务，学生得以在“猜想–验证–调试–抽象”的交互循环中，主动经历数学概念的生成过程，从而内化诸如“线性变换”“特征空间”“秩”等概念背后的数学思想。

同时，本教学模式天然契合 OBE 理念：

成果导向：教学目标不再是“学生会计算四阶行列式”，而是“学生能设计多种算法计算行列式并

解释其数学原理，能根据实际问题选择合适的方法求解逆矩阵”。大作业中的可运行代码、分析报告即为可量化、可评估的学习成果。

学生中心：教学活动从教师讲解转向学生自主编程、调试、对比验证。教师角色由“讲授者”转变为“引导者”和“任务设计者”，学生通过主动探索完成知识建构。

持续改进：学生在编程过程中遇到的错误与调试经历，成为其反思概念理解深度的宝贵反馈，教师可根据学生代码中暴露的共性问题调整教学重点，形成闭环改进。

1.4. 理论基础：Python 作为“认知媒介”的学理支撑

本文将 Python 定位为“核心认知媒介”而非“辅助工具”，其学理依据来自建构主义学习理论与分布式认知理论。

建构主义视角：知识由学习者在与环境的互动中主动建构。编程任务提供了四大要素——情境(具体问题)、协作(小组调试)、会话(解释代码逻辑)、意义建构(通过验证内化概念)。当学生亲手实现“不同行展开结果一致”时，行列式唯一性从被告知的定理转变为可操作的信念。

分布式认知视角：认知活动分布在大脑、工具与环境之中。Python 承担了外部记忆卸载(释放工作记忆用于策略思考)、表征变换(数学公式↔可执行代码↔运行输出)和即时反馈(猜想→验证→修正的短循环)三重功能。这种快速反馈机制符合高效学习认知规律。

本质区别：辅助工具(如现成库函数)替代认知劳动，加深“黑箱”感知；而认知媒介重构认知过程，迫使学习者将概念转化为算法、通过调试回溯理解、通过对比体悟思想。由此，编程从“技能训练”上升为“认知实践”。

2. 案例分析

2.1. 案例一：四阶行列式的 Python 实现与教学分析

以四阶行列式为例，展示三种 Python 实现方式及其教学侧重点：

$$\begin{vmatrix} 1 & 4 & 2 & 5 \\ 2 & 3 & 2 & 1 \\ 6 & 4 & 1 & 1 \\ 5 & 1 & 4 & 2 \end{vmatrix}$$

2.1.1. 方法一：递推法计算行列式

该方法按行列式的第一行进行展开，通过递归调用实现降阶计算，核心代码如下(仅截取了部分代码)，计算结果如图 1 所示：

```
for j in range(n):
    # 计算代数余子式
    minor = []
    for i in range(1, n):  # 跳过第一行
        row = []
        for k in range(n):
            if k != j:  # 跳过第j列
                row.append(matrix[i][k])
        minor.append(row)
```

```
递归法计算的行列式值：224
NumPy验证结果：223.99999999999994
```

Figure 1. Result of calculating the fourth-order determinant by the recursive method
图 1. 递推法计算四阶行列式结果

该方法按第一行展开行列式，逐步递推计算，结果为 224。教学重点在于展示行列式展开的基本思想

与递归逻辑，帮助学生理解“降阶”求解的基本路径。

2.1.2. 方法二：多行展开验证行列式的唯一性

为引导学生理解行列式的内在一致性，即其值与展开方式无关，我们设计了按不同行展开的验证方法，如图 2 所示。

```
if n == 1:
    return matrix[0][0]
elif n == 2:
    return matrix[0][0] * matrix[1][1] - matrix[0][1] * matrix[1][0]

det = 0
# 按指定行展开
for j in range(n):
    minor = get_minor(matrix, expand_row, j)
    sign = (-1) ** (expand_row + j)
    det += sign * matrix[expand_row][j] * determinant_by_expansion(minor)
```

```
=== 按不同行展开计算 ===
按第1行展开的结果：224
按第2行展开的结果：224
按第3行展开的结果：224
按第4行展开的结果：224
```

Figure 2. Comparison of calculation results by expanding different rows
图 2. 不同行展开计算结果对比

通过分别按四行展开计算，结果显示尽管展开路径不同，所得行列式值均为 224。该方式可引导学生理解行列式的内在一致性，强化“值与展开方式无关”的数学性质。

2.1.3. 方法三：行列式展开的详细过程演示

为在教学过程中对展开定理进行逐步剖析，设计了可输出每一步计算细节的演示方法，如图 3 所示。

```
det = 0
# 按第一行展开
for j in range(n):
    # 使用get_minor函数获取余子式
    minor = get_minor(matrix, 0, j)

    # 递归计算并累加
    sign = (-1) ** j
    det += sign * matrix[0][j] * determinant_recursive(minor)

return det
```

```
=== 按第一行展开的详细过程 ===

第1项: a₁,1 = 1
符号: (-1)¹⁺1 = 1
当前处理的元素:
* 1* ( 4) ( 2) ( 5)
( 2)   3    2    1
( 6)   4    1    1
( 5)   1    4    2

对应的3阶余子式 M₁,1:
[3, 2, 1]
[4, 1, 1]
[1, 4, 2]
余子式的行列式值: -5
当前项的值: 1 × 1 × -5 = -5
累加后行列式值: -5
```

Figure 3. Display of cofactors and the expansion process
图 3. 代数余子式与展开过程展示

本方法在计算过程中输出每一步的余子式与代数余子式，清晰展示其结构与符号变化规律，适用于教学过程中对展开定理的逐步剖析与可视化辅助理解。

三种方法由浅入深，从结果计算到过程演绎，形成教学闭环。学生不仅掌握 Python 编程技巧，更能通过对比理解行列式的数学本质，实现从“会算”到“懂理”的认知跃升。

2.2. 案例二：四阶逆矩阵的两种解法对比

以如下矩阵为例，展示两种逆矩阵求解方法，如图 4、图 5 所示：

$$A=\begin{pmatrix}1 & 4 & 2 & 5\\ 2 & 3 & 2 & 1\\ 6 & 4 & 1 & 1\\ 5 & 1 & 4 & 2\end{pmatrix}$$

2.2.1. 方法一：伴随矩阵法

伴随矩阵法求逆矩阵： $A^{-1}=\frac{1}{|A|}A^{*}$，其中 A^{*} 是 A 的伴随矩阵。

```
n = len(matrix)
inverse_matrix = []
for i in range(n):
    row = []
    for j in range(n):
        row.append(adjugate_matrix[i][j] / det)
    inverse_matrix.append(row)

return inverse_matrix, det, adjugate_matrix
```

```
行列式 det(A) = 224

伴随矩阵 adj(A):
[-5, -53, 40, 19]
[-1, 79, 8, -41]
[-21, 91, -56, 35]
[55, -89, 8, 15]

逆矩阵 A⁻¹:
[-0.0223, -0.2366, 0.1786, 0.0848]
[-0.0045, 0.3527, 0.0357, -0.183]
[-0.0938, 0.4062, -0.25, 0.1562]
[0.2455, -0.3973, 0.0357, 0.067]

验证: A × A⁻¹ =
[1.0, 0.0, -0.0, 0.0]
[-0.0, 1.0, 0.0, 0.0]
[0.0, 0.0, 1.0, 0.0]
[-0.0, 0.0, 0.0, 1.0]
PS C:\Users\罗珊>
```

Figure 4. Solving inverse matrix by adjugate matrix method
图 4. 伴随矩阵法求解逆矩阵

该方法基于公式 $A^{-1}=\frac{1}{|A|}A^{*}$，步骤清晰，理论完整，适合用于讲解逆矩阵的代数构成。尽管计算量大，但有助于学生理解行列式、余子式与逆矩阵之间的数学联系。

2.2.2. 方法二：高斯－约当消元法

高斯－约当消元法通过行变换将增广矩阵 $[A|I]$ 转化为 $[I|A^{-1}]$，计算效率高，数值稳定性好，是实际工程应用中的首选方法。

```
for col in range(n):
    # 寻找主元（当前列中绝对值最大的元素）
    pivot_row = col
    for r in range(col + 1, n):
        if abs(augmented[r][col]) > abs(augmented[pivot_row][col]):
            pivot_row = r
```

```
初始增广矩阵 [A|I]:
['  1.00', '  4.00', '  2.00', '  5.00', '  1.00', '  0.00', '  0.00', '  0.00']
['  2.00', '  3.00', '  2.00', '  1.00', '  0.00', '  1.00', '  0.00', '  0.00']
['  6.00', '  4.00', '  1.00', '  1.00', '  0.00', '  0.00', '  1.00', '  0.00']
['  5.00', '  1.00', '  4.00', '  2.00', '  0.00', '  0.00', '  0.00', '  1.00']

交换行 1 和行 3
['  6.00', '  4.00', '  1.00', '  1.00', '  0.00', '  0.00', '  1.00', '  0.00']
['  2.00', '  3.00', '  2.00', '  1.00', '  0.00', '  1.00', '  0.00', '  0.00']
['  1.00', '  4.00', '  2.00', '  5.00', '  1.00', '  0.00', '  0.00', '  0.00']
['  5.00', '  1.00', '  4.00', '  2.00', '  0.00', '  0.00', '  0.00', '  1.00']
```

Figure 5. Solving inverse matrix by Gauss-Jordan elimination method
图 5. 高斯－约当法求解逆矩阵

该方法的教学重点在于理解矩阵行变换的本质及其在求解线性方程组和逆矩阵中的应用。通过Python 实现，学生可以清晰地观察整个消元过程，理解“初等行变换不改变矩阵行空间”这一基本定理，并掌握一种在实际编程中高效、可靠的求逆算法。

对比这两种方法，伴随矩阵法强调理论构建，高斯－约当法侧重算法实现。两者结合，既可深化学生对逆矩阵结构的理解，也培养其根据实际需求选择合适方法的能力。

3. 教学实践与效果分析

为检验上述基于 Python 的融合教学模式的实际效果，本研究分别于 2023~2024 学年第一学期与2024~2025 第一学期在广州应用科技学院计算机科学与技术专业、软件工程专业的《线性代数》课程中进行了教学实践。实践对象为大学一年级，共计 297 名学生，为期两年。

3.1. 教学实施的具体策略

为保证教学模式的规范实施，本研究设计了以下具体策略。

3.1.1. 混合式教学安排

课程采用“理论讲授(线下) + Python 编程实践(线下实验)”的混合模式。其中：理论讲授：共 28 学时，主要用于讲解线性代数基本概念、定理推导及数学思想，Python 相关内容以伪代码和算法思路介绍为主。Python 编程实践：共 4 学时，安排在教室进行，学生每人自带电脑。其中 2 学时为教师演示与指导，2 学时为分组讨论与自主编程。

3.1.2. Python 教学的课时与内容分配

在总计 32 学时中，Python 相关内容贯穿于以下知识点(如表 1 所示)：

Table 1. Python teaching knowledge points and class hour allocation
表 1. Python 教学知识点与学时分配

知识点	Python 教学学时	主要内容
行列式	1 学时	递推法、多行展开、过程演示
逆矩阵	1 学时	伴随矩阵法、高斯－约当消元法
大作业综合实践	2 学时	选题指导、代码调试、报告撰写

3.1.3. 教师的指导方式

教师在教学过程中承担“任务设计者 + 引导者 + 反馈者”的角色：

课前：发布模板，包含必要的代码框架和注释说明，降低学生入门难度。课中：采用“5 分钟算法讲解 +20 分钟现场编程演示 +20 分钟学生模仿与调试”的节奏，鼓励学生边学边练；对于典型错误(如递归深度溢出、符号与数值计算混用)，教师统一讲解并提供修正思路。课后：通过线上平台进行代码审查，对提交的大作业给出文字反馈，并组织线下答疑；对共性问题在下一次课中进行补充讲解。

3.1.4. 应对编程能力差异的策略

针对约 30%学生零基础的现状，采用异质分组策略。异质分组：按“1 名较熟练 + 2 名有一定基础 + 1 名零基础”组建 4 人小组，角色轮换(驱动者、导航员、记录员、验证员)，避免代写，强调协作。

3.1.5. 4 学时编程实践课流程

采用“微任务驱动 + 即时反馈”循环：第 1 学时完成递推法与多行展开验证；第 2 学时分别实现伴

随矩阵法与高斯–约当法，并对比分析。教师巡视中穿插“预测–验证”“错误归因”式提问，引导学生关注数学本质而非单纯调试。

3.1.6. 引导数学思考的提问策略

课堂中可以使用三类问题：① 运行前“预测输出”，② 出错时“反思哪个数学概念理解有误”，③ 完成后“不运行代码，仅凭数学推导能否预判输出结果”。促使学生在调试中不断回溯数学定义，形成“代码是数学的可执行表达”的心智模型。

3.2. 控制变量说明

为确保两个学年实践结果的可比性，本研究对以下变量进行了控制：两个学年的授课教师均为本人，教学风格与经验无差异；教学大纲依据学校统一模板，均包含行列式、逆矩阵、线性方程组、特征值等章节，未作内容增删；考核方式均为平时成绩占 40% (其中出勤 10%、课堂练习 10%、课堂表现 20%)与大作业占 60%，大作业评分细则保持一致；Python 教学学时均为 4 学时；学生基础方面，两个学年的学生专业不同，但同属工科，入学数学成绩无统计学显著差异，且线性代数课程要求相同，因此两批学生的表现仍具有较好的可比性。

3.3. 实践过程与评估方法

在教学完行列式与逆矩阵的核心理论与 Python 实现方法后，教师布置了一项开放式的大作业。作业要求学生自选一个包含至少四阶矩阵的应用场景(如简单的图形变换等)，并运用课程所学的多种 Python 方法进行计算与分析，最终提交一份简短的报告，阐述其解题思路、代码实现及对相关数学概念的理解。

本研究的评估基于以下多元质性材料：

1) 学生作品分析：对所有提交的大作业代码与报告进行内容分析，评估其知识迁移与应用能力。
2) 学生反思反馈：在大作业报告中，提取学生对学习过程的自主反思与评价。
3) 课堂观察：记录学生在完成大作业过程中的课堂互动与讨论情况。

3.4. 知识迁移与应用能力的实现

通过对上述质性材料的分析，本教学模式的效果主要体现在以下三个方面：

对大作业的分析显示，超过八成的学生能够成功将课堂案例中的 Python 求解方法迁移至新的、简单的应用场景中。例如，部分学生构建了图形旋转矩阵并计算其行列式以判断变换特性，另有学生编写程序求解由线性方程组生成的矩阵的逆。这表明学生已初步掌握将抽象的矩阵知识转化为解决实际问题的工具，实现了计算思维的提升。

学生的作业报告为其概念理解提供了直接证据。多位学生在反思中提及：“通过自己编程实现不同行展开，我才真正相信行列式的值真的与展开方式无关。”“之前总觉得伴随矩阵很抽象，但当我看到代码一步步算出它并成功求出逆矩阵时，我突然明白了它们之间的内在联系。”此类反馈清晰地表明，Python 作为“认知工具”成功帮助学生跨越了抽象与直观之间的鸿沟，促成了其数学观念的深化。

课堂观察发现，在完成大作业期间，学生主动提问和小组讨论的频率与深度显著增加，问题多集中于“如何选择最优算法”而非“如何笔算”。这反映出学生的学习重心已从机械计算转向对数学思想与工程效率的探究，主观能动性得到有效调动。

4. 结论

本文针对新工科背景下线性代数教学中“重计算、轻思想”及传统教学模式与学生主体性脱节的问题，提出并实践了一种将 Python 作为核心认知媒介的融合教学模式。通过四阶行列式与逆矩阵两个典型

案例，设计了递推法、多行展开验证、过程演示、伴随矩阵法及高斯－约当消元法等多种算法实现，使学生在“猜想－验证－调试－抽象”的循环中主动建构数学概念。教学实践表明：超过八成的学生能够成功将 Python 方法迁移至新的应用场景；学生的反思报告证实编程显著加深了对行列式唯一性、逆矩阵代数构成等概念的理解；课堂提问重心从“如何笔算”转向“如何选择最优算法”。由此得出结论：该模式能有效促进学生从机械计算到计算思维的转变，实现从“工具使用”到“思维建构”的跨越，为 Python 融入线性代数教学提供了可操作的实践路径。

5. 反思与展望

本教学模式取得了积极成效，但也暴露出若干具体问题：学生编程基础参差不齐(约 30%零基础)，虽采用异质分组，但学时有限，未来将增设 Python 前置选修模块；部分抽象知识点(如矩阵的秩的几何意义)难以仅用代码直观呈现，后续将设计“可视化－代码－公式”三位一体的呈现模板；课堂时间分配存在矛盾，学生调试代码超时导致讨论环节被压缩，未来将采用微任务拆分和翻转课堂。综上，本模式可行有效，上述反思将为下一轮教学改进提供具体方向。

基金项目

广州应用科技学院 2025 年度科研规划项目(Gzyykjxy2025-07)。

参考文献

[1] 李清华, 王宝娟. 线性代数知识点的可视化教学设计探索与实践[J]. 大学数学, 2022, 38(2): 112-119.

[2] 偶世坤. 浅析线性代数混合式教学实践[J]. 教育进展, 2024, 14(1): 448-453.

[3] 朱佳俊, 李吉有, 张跃辉. 编程实践融入线性代数教学的探索[J]. 实验室研究与探索, 2019, 38(11): 231-234, 265.

[4] 张红燕, 何小娟. 线性代数混合式教学的探索与实践[J]. 当代教育实践与教学研究, 2022(12): 28-30.

[5] 刘熙娟, 刘云, 郭丽峰. 数学建模思想融入高校线性代数教学实践[J]. 创新教育研究, 2023, 11(7): 1701-1706.