La importancia de Python en la enseñanza de la matemática en las carreras de ingeniería

Ing. Gabriel Valenzuela, Dr. Ing. Orlando Micolini, Ing. Osvaldo Natali, Ing. Liliana Pastore, Ing. Luis Orlando Ventre, and Ing. Mauricio Ludemann

Laboratorio de Arquitectura de Computadoras, Departamenteo de Matemática, FCEFyN-Universidad Nacional de Córdoba Av. Velez Sarfield 1601, CP-5000, Córdoba, Argentina {gabriel.valenzuela, orlando.micolini, luis.ventre, onatali, liliana_pastore, mauri.ludemann}@unc.edu.ar

Resumen En este trabajo se analiza el creciente papel del lenguaje de programación Python en la enseñanza de materias relacionadas con las matemáticas dentro de los programas de ingeniería en computación en Argentina. Se prioriza el uso de Python frente a herramientas tradicionales como MATLAB y R, destacando ventajas (software libre, versatilidad y adopción generalizada) y desventajas (posible complejidad inicial, menor rendimiento en ciertos casos) respecto de su integración en cursos troncales como Análisis Matemático I, II, III y Álgebra Lineal. Basándose en la literatura académica y en experiencias recientes, el estudio analiza cómo Python facilita un aprendizaje más integrado entre las matemáticas y la informática, y su impacto en la formación académica y las competencias profesionales de los futuros ingenieros en computación. Los resultados indican que Python, debido a su simplicidad y legibilidad, puede igualar la funcionalidad de paquetes de software especializados, contribuyendo a una educación multidisciplinar más sólida y alineada con las demandas actuales de la industria tecnológica.

Keywords: Python , Educación , Matemática.

1. Introducción

En las últimas décadas, la integración de herramientas computacionales en la enseñanza de la matemática ha cobrado especial importancia en las carreras de ingeniería. A nivel global, Python se ha posicionado como uno de los lenguajes de programación más utilizados en contextos académicos y profesionales [1][4]. Diversos programas universitarios, particularmente en ciencias de la computación e ingeniería, han comenzado a requerir o alentar el aprendizaje de Python desde los primeros años de formación [1]. Esto obedece a la versatilidad de Python como lenguaje de propósito general y a su sintaxis relativamente simple, semejante al pseudocódigo, lo que facilita su adopción por parte de estudiantes que recién se inician en la programación [1]. En carreras de ingeniería en computación, donde los estudiantes cursan varias asignaturas matemáticas, como Análisis Matemático y Álgebra Lineal, junto con materias de programación, surge la oportunidad de complementar trasversalmente ambas áreas mediante un lenguaje común.

En Argentina, siguiendo tendencias internacionales, se observan iniciativas educativas que incorporan Python en la enseñanza de matemática. Por ejemplo, la Facultad de Ingeniería Química de la Universidad Nacional del Litoral desarrolló un curso de resolución de problemas matemáticos usando Python destinado a sus estudiantes de ingeniería [2]. Este tipo de experiencias reflejan un cambio de paradigma en la docencia: se pasa de utilizar software matemático especializado a emplear lenguajes de programación de uso extendido. La motivación principal es didáctica y práctica: al emplear Python, los estudiantes de ingeniería pueden conectar más directamente los conceptos matemáticos con su aplicación en la resolución computacional de problemas, adquiriendo simultáneamente habilidades de programación útiles en su desarrollo profesional.

El presente trabajo explora la importancia de Python como herramienta para la enseñanza de la matemática en carreras de ingeniería en Argentina , con énfasis en ingeniería en computación. En la siguientes secciones se describe el contexto actual de uso de Python en entornos educativos y se comparan sus características con las de otras herramientas tradicionales como MATLAB y R, se discuten las ventajas y desventajas de incorporar Python en materias matemáticas fundamentales, como lo es Análisis Matemático I, II y III y Álgebra Lineal, apoyándose en literatura existente para finalmente, en la conclusión, reflexionar sobre el impacto que tiene la formación en Python en la preparación académica y profesional de los futuros ingenieros, considerando las exigencias del ámbito laboral contemporáneo.

2. Uso de Python en la enseñanza de la matemática en comparación con MATLAB/R

Tradicionalmente, la enseñanza de asignaturas en el área de la matemática en ingeniería se apoyó en software especializados. Herramientas como **MATLAB (de licencia privativa)** han sido durante años estándar en asignaturas de cálculo numérico y álgebra lineal, mientras que en cursos de estadística es común el uso de **R** u otros paquetes estadísticos. En entornos académicos de computación, también se usaron sistemas de álgebra computacional como *Derive* o software matemático comercial como *Maple* o *Mathematica*. Sin embargo, este enfoque conlleva ciertas limitaciones. Un ejemplo ilustrativo se dio en la Universidad de las Ciencias Informáticas (Cuba), donde por mucho tiempo se emplearon Derive, MATLAB y su alternativa libre *Octave* en los laboratorios de matemática. Dichas herramientas presentaban inconvenientes: Derive quedó obsoleto tras su descontinuación en 2007, y MATLAB, además de ser de alto costo, resulta difícil de obtener para todos los estudiantes [3]. Si bien Octave es una opción libre compatible con MATLAB, carece de la totalidad de funcionalidades avanzadas que sí ofrece MATLAB [3]. Este panorama no es ajeno a las universidades latinoamericanas y argentinas, donde los altos costos de licencias y la necesidad de actualizar software privativo han impulsado la búsqueda de alternativas más accesibles. Otro factor relevante es la experiencia del estudiante de ingeniería en computación frente a estas herramientas tradicionales. Software como MATLAB,

Octave o paquetes estadísticos dedicados tienden a ser entornos especializados con sintaxis y enfoques distintos a los lenguajes de programación de propósito general. Como señalan Soto Gómez et al. (2021), estas herramientas, aunque poderosas en sus dominios, no suelen integrarse fácilmente en diversos ámbitos de aplicación, lo que las hace poco populares entre estudiantes de informática [3]. De hecho, su sintaxis y estilo difieren de los lenguajes generales, lo que puede generar desmotivación en estudiantes habituados a la programación, al percibir una desconexión entre lo aprendido en cursos de código y las herramientas usadas en matemática [3]. Esta desconexión dificulta la comprensión multidisciplinar y la integración de conocimientos entre materias de matemática y computación [3]. Por el contrario, emplear un lenguaje unificado para ambos campos podría evidenciar de forma más adecuada y motivadora la relación esencial que existe entre ellos[3].

En este contexto emerge **Python** como una alternativa superadora. Python es un lenguaje interpretado, de código abierto y multiplataforma, cuyas aplicaciones abarcan desde el desarrollo web y científico hasta la ciencia de datos e inteligencia artificial. Su uso en educación ha sido reconocido por diversos autores; por ejemplo, Parker, Beard y colegas (2019) destacaron las ventajas de Python en contextos educacionales, y comparativas previas ya subrayaban la simpleza e inteligibilidad de Python frente a MATLAB [3]. A diferencia de MATLAB, cuyo origen es el cálculo matricial numérico, Python nació como un lenguaje de propósito general con una sintaxis clara y cercana al lenguaje natural [1]. Esto le otorga una curva de aprendizaje suave para principiantes, comparable a la de R, pero con la ventaja de resultar más familiar a quienes conocen otros lenguajes de programación [1]. Además, Python es gratuito y de código abierto, lo que elimina barreras de acceso: cualquier estudiante o institución puede instalarlo sin costo en distintos sistemas operativos (Windows, Linux, macOS)[2].

Este carácter libre se alinea con políticas educativas que promueven software de libre distribución en varios países (como ocurrió en Cuba y también es tendencia en Argentina), evitando depender de licencias onerosas [3]. Comparado con R, Python ofrece un alcance más amplio. R se destaca en estadística y análisis de datos con una enorme cantidad de paquetes específicos, por lo que suele ser la herramienta preferida en cursos de probabilidad y estadística. No obstante, R fue concebido principalmente para ese dominio y su sintaxis, aunque poderosa para análisis estadísticos, puede resultar menos intuitiva fuera de ese contexto. Python, por su parte, cuenta con bibliotecas como **NumPy** y **SciPy** que extienden sus capacidades al cálculo numérico eficiente (similares a MATLAB en desempeño sobre arreglos) [3], así como **SymPy** para matemática simbólica (permitiendo realizar álgebra y cálculo simbólico directamente en Python) [3]. Esto significa que con Python es posible abarcar en un solo entorno tanto las necesidades de cálculo numérico de un curso de Análisis Matemático o Álgebra Lineal, como el análisis estadístico de un curso de Probabilidad, todo ello integrando también capacidades de programación. En suma, Python se presenta como un *"ecosistema"* unificado donde convergen la enseñanza de la programa-

ción y la enseñanza de la matemática, algo pertinente en carreras de ingeniería especialmente en computación.

3. Ventajas y desventajas de Python en asignaturas del área matemática en las carreras de ingeniería

3.1. Ventajas de utilizar Python en la enseñanza de la matemática

- **Visión integradora y aplicación multidisciplinar:** Al incorporar Python en materias como Análisis I, II y III y Álgebra, los estudiantes pueden verificar de forma práctica los conceptos teóricos mediante simulaciones, cálculos automatizados y visualizaciones. Esto promueve una comprensión más profunda, pues evidencia la conexión entre la teoría matemática y su implementación computacional[3]. Un entorno como Python permite, por ejemplo, graficar funciones y calcular derivadas e integrales en Análisis I,II y III, o resolver sistemas de ecuaciones y descomponer matrices en Álgebra Lineal, todo mediante código ejecutable. De esta manera, el estudiante desarrolla simultáneamente habilidades de modelado matemático y habilidades de programación, alineadas con el perfil del ingeniero moderno. En palabras de Fangohr (2004) y Colliau et al. (2017), la simplicidad y claridad de Python facilitan su uso para explorar problemas matemáticos, logrando resultados comparables a MATLAB pero con menor complejidad sintáctica[3].
- **Accesibilidad y costo nulo:** Python es software libre y gratuito, disponible para todas las plataformas, lo cual contrasta marcadamente con MATLAB, que requiere costosas licencias comerciales[3]. Esto implica que los estudiantes pueden instalar Python en sus equipos personales sin trámites ni restricciones, continuando sus prácticas fuera del aula. La eliminación de barreras económicas no solo beneficia a los estudiantes, sino también a las instituciones, permitiéndoles estandarizar un entorno de trabajo común sin incurrir en gastos adicionales. Además, la activa comunidad de Python a nivel global proporciona una gran cantidad de recursos de aprendizaje, documentación y soporte en línea, lo que significa que tanto docentes como estudiantes cuentan con ayuda accesible para resolver dudas o problemas. Adicionalmente, gracias a convenios con Google instituaciones con las universidades, es posible hacer uso de Colab, y poder ejecutar el intérprete desde un Saas (Software as a Service).
- **Versatilidad y preparación profesional:** Python se ha convertido en una herramienta ubicua en muchos campos de la industria y la investigación. Su incorporación en la currícula prepara a los estudiantes con competencias altamente valoradas en el mercado laboral [1]. A diferencia de herramientas especializadas (limitadas a ciertos dominios), Python es el "lenguaje por defecto" en múltiples áreas profesionales (desarrollo de software, ciencia de datos, inteligencia artificial, automatización, etc.)[5]. Un ingeniero en computación familiarizado con Python puede abordar problemas diversos, reutilizando sus conocimientos en nuevos contextos. Por ejemplo, un estudiante

que aprendió a usar librerías matemáticas de Python en Análisis Matemático III podrá luego aplicar esas mismas destrezas para resolver modelos de optimización en su trabajo, o para analizar datos experimentales, sin tener que aprender desde cero una herramienta distinta. En este sentido, la formación en Python confiere flexibilidad: incluso si el egresado no se dedica exclusivamente a programación, es muy probable que Python figure entre las herramientas que empleará en la práctica profesional cotidiana [1].

- **Amplio ecosistema científico:** La gran cantidad de bibliotecas de Python enfocadas en cómputo científico es un factor decisivo. Para cada tema típico de las asignaturas mencionadas hay paquetes disponibles: en cálculo y análisis matemático, SymPy permite realizar derivación, integración y álgebra simbólica, reforzando los contenidos de Análisis I, II y III con un enfoque CAS (Computer Algebra System) dentro de Python [3]. En análisis numérico y métodos aproximados, NumPy y SciPy brindan rutinas eficientes para cálculo matricial, resolución de sistemas lineales, interpolación, integración numérica y métodos iterativos [3]. En álgebra lineal, SciPy.linalg expande las capacidades de álgebra matricial más allá de las funciones básicas de NumPy [3], permitiendo, por ejemplo, calcular autovalores y autovectores de matrices grandes de forma sencilla. Adicionalmente, Matplotlib, Seaborn o bibliotecas afines facilitan la creación de gráficos y visualizaciones, útiles en todas estas asignaturas para ilustrar funciones, curvas, superficies o distribuciones de datos. Esta amplitud de herramientas convierte a Python en una plataforma única donde convergen capacidades equivalentes a las de MATLAB (cálculo numérico, gráficos) [1], a las de Mathematica/Maple (álgebra simbólica) y a las de R (análisis estadístico), todo interoperable en un solo lenguaje. Tal riqueza de funcionalidades, cuando es aprovechada pedagógicamente, enriquece la experiencia de aprendizaje de los estudiantes de ingeniería.

3.2. Desventajas o desafíos potenciales

- **Curva de aprendizaje inicial en programación:** Aunque Python es reconocido por su facilidad de uso, su introducción en una asignatura matemática requiere que el estudiante tenga nociones básicas de algoritmos y código. En carreras de ingeniería en computación esto suele estar contemplado (ya que los estudiantes cursan algoritmia/programación en paralelo), pero podría representar una carga cognitiva extra si la coordinación entre materias no es la ideal. En otras palabras, el docente de matemática debe destinar tiempo a familiarizar a los estudiantes con el entorno Python (sintaxis elemental, uso de notebooks o entornos como IDLE) antes de poder explotar la herramienta en profundidad. No obstante, experiencias educativas indican que esta inversión inicial rinde frutos rápidamente, dado que Python tiene una sintaxis bastante intuitiva y consistente con la notación matemática en muchos casos[2].
- **Inconsistencias y libertad excesiva:** Python, al ser un lenguaje de propósito general, no está diseñado específicamente para la enseñanza elemental.

Posee múltiples formas de alcanzar un mismo resultado y ciertos detalles de sintaxis que podrían confundir a principiantes. Por ejemplo, en Python existen diferentes estilos para iterar (bucles tradicionales con índices o comprensiones de listas), distintas maneras de importar bibliotecas, etc. Autores como Aguilera (2021) señalan que, a diferencia de lenguajes pedagógicos más simples, Python no ofrece un conjunto mínimo de instrucciones fácilmente acotable para novatos [2]. Asimismo, algunas incoherencias menores del lenguaje (como que print sea una función pero return una construcción sintáctica, o las diferencias entre métodos mutables como *.sort*() y funciones como *sorted*()) han sido criticadas en el ámbito educativo [2]. Estos aspectos requieren una planificación didáctica cuidadosa: el instructor debe decidir qué subset del lenguaje y qué buenas prácticas inculcar, para no abrumar al estudiante con opciones innecesarias. Aun así, muchos educadores consideran que los beneficios globales de enseñar un lenguaje "real" superan estos inconvenientes formales, y que los estudiantes adquieren criterios para manejar la flexibilidad de Python con adecuada guía.

- **Rendimiento y cálculo simbólico limitado:** Si bien para fines docentes el rendimiento de Python es más que suficiente, es sabido que en cálculos numéricos intensivos o complejos algebraicos, Python puro puede ser más lento que soluciones específicas. La biblioteca SymPy, por ejemplo, permite abordar problemas de cálculo simbólico pero no iguala en velocidad a sistemas especializados como Mathematica; de hecho, SymPy está escrita completamente en Python, lo que puede volverla más lenta en cálculos muy pesados [3]. Este no suele ser un problema en cursos de nivel universitario (donde los ejemplos manejados son de pequeña a mediana escala), pero conviene señalar que Python tiene como "costo" su menor eficiencia comparado con lenguajes compilados. En aplicaciones de análisis numérico con grandes volúmenes de datos o simulaciones complejas, puede requerirse optimizaciones adicionales (uso de bibliotecas en C o C++ integradas a Python, vectorización con NumPy, etc.). No obstante, en el contexto educativo estas limitaciones de rendimiento rara vez impactan negativamente; por el contrario, pueden aprovecharse para discutir con los estudiantes nociones de eficiencia algorítmica y uso de recursos computacionales. En cuanto al cálculo simbólico, SymPy cubre la mayoría de los contenidos típicos (derivadas, integrales, álgebra básica), pero algunos problemas muy avanzados podrían exceder sus capacidades o resultar en expresiones difíciles de simplificar automáticamente. En tales casos, combinar enfoques analíticos y numéricos en Python puede ser una solución enseñable, mostrando a los estudiantes las fortalezas y límites de las herramientas computacionales.
- **Capacitación docente y adaptación de materiales:** Un desafío práctico para la inclusión de Python en materias tradicionales de matemática es la preparación del cuerpo docente. Muchos profesores de matemática de las carreras de ingeniería quizás no tengan formación en Python, dado que su especialidad ha estado más ligada a las matemáticas puras o al uso de paquetes establecidos como MATLAB. La transición requiere capacitación y, sobre todo, la adaptación de los materiales didácticos (guías de ejercicios,

bibliografía, evaluaciones) para incorporar actividades computacionales significativas. Afortunadamente, la creciente comunidad educativa en torno a Python ha generado abundantes recursos: desde libros y apuntes en español enfocados en matemática con Python[2], hasta notebooks interactivos y foros de docentes compartiendo buenas prácticas. La colaboración entre departamentos de matemática e informática puede facilitar esta incorporación. En Argentina, se comienza a transitar este camino de forma gradual, y universidades que han implementado Python en algunas asignaturas reportan una buena recepción por parte de los estudiantes, quienes valoran el aprendizaje unificado y práctico. No obstante, es importante acompañar este cambio metodológico con evaluación continua de sus resultados, asegurando que el uso de Python potencie –y no opaque– la comprensión conceptual de la matemática. Esto constituye un trabajo a realizar o futuras investigaciones, para ver como perciben los docentes el abordaje de este nuevo concepto.

4. Consideraciones sobre las asignaturas clave

La inclusión de Python puede adecuarse a la naturaleza de cada asignatura matemática en las carreras de ingeniería especialmente en computación. En Análisis Matemático I (cálculo diferencial e integral en una variable real), Python permite ilustrar conceptos como derivadas e integrales a través de ejemplos computacionales: por ejemplo, comparando la derivada simbólica de una función usando SymPy con la aproximación numérica mediante diferencias finitas, o visualizando el cálculo de áreas bajo curvas mediante sumas de Riemann programadas por los propios estudiantes. En Análisis Matemático II (frecuentemente centrado en cálculo multivariable y ecuaciones diferenciales), herramientas como matplotlib son valiosas para graficar superficies y curvas de nivel, ayudando a los estudiantes a visualizar funciones de varias variables. Asimismo, Python puede emplearse para calcular desarrollos en serie y explorar su convergencia, reforzando la intuición sobre series de potencias con cálculos computacionales rápidos en Análisis Matemático III. En Álgebra Lineal, probablemente la materia donde MATLAB tradicionalmente era el aliado preferido, Python con NumPy/SciPy ofrece igual o mayor potencia: los estudiantes pueden resolver sistemas lineales grandes, calcular inversas, determinantes y descomposiciones **LU** o **QR**, además de experimentar con vectores propios y diagonalización de matrices de forma interactiva. Un ejercicio típico podría ser utilizar Python para iterar métodos numéricos de cálculo de autovalores (como la potencia), dándole al estudiante una comprensión dinámica de algoritmos que en clase se ven de forma teórica. La transformada de Fourier y otras transformadas integrales, que si son parte del temario, también pueden ilustrarse computacionalmente usando rutinas de FFT disponibles en *scipy.fftpack*[3], mostrando concretamente cómo la teoría se aplica en procesamiento de señales. En todos los casos mencionados, el uso de Python en estas materias debe plantearse como complemento de la enseñanza tradicional, no reemplazando los métodos analíticos sino proporcionándoles un refuerzo práctico. Cuando se implementa de este modo, los estudiantes tienden a

mostrar mayor motivación y entendimiento, al poder ver los resultados de procedimientos matemáticos y experimentar con variaciones de problemas en tiempo real.

5. Conclusiones

La adopción de Python en la enseñanza de la matemática para carreras las de ingeniería especialmente en computación en trabajo futuros. representa un avance significativo hacia la modernización y mejora de la formación académica. Desde el punto de vista educativo, el uso de un lenguaje de programación ampliamente difundido como Python logra tender puentes entre los conceptos matemáticos abstractos y las aplicaciones concretas en el mundo real[3]. Los estudiantes desarrollan una comprensión más sólida y multidisciplinar al aplicar inmediatamente las nociones teóricas en un entorno computacional que les es familiar y útil. Esto redunda en profesionales con mayores competencias desarrolladas para enfrentar problemas complejos, ya que han cultivado el hábito de combinar análisis formal con experimentación computacional. Las ventajas identificadas –versatilidad, gratuidad, amplia acogida comunitaria y relevancia industrial de Python– aportan un valor agregado indiscutible a la formación de ingenieros. Un graduado que a lo largo de su carrera universitaria ha utilizado Python en sus cursos de matemática y programación posee no solo los conocimientos teóricos de su disciplina, sino también una competencia práctica en una herramienta que es estándar de facto en múltiples industrias[5]. Dado que Python es actualmente uno de los lenguajes más populares en el mundo tecnológico [4], esta competencia incrementa la inserción laboral del egresado y su capacidad de adaptación a diversos roles profesionales. Estudios han mostrado que Python figura entre las competencias más demandadas por empleadores en campos vinculados a la ciencia de datos, desarrollo de software e incluso investigación científica [1]. Por ende, incorporar Python en la currícula no solo atiende a una necesidad pedagógica, sino también estratégica, al alinear la educación con las tendencias y exigencias del siglo XXI. No obstante, la implementación de Python en la enseñanza de la matemática conlleva desafíos que deben ser gestionados: la formación y soporte a los docentes, la elaboración de materiales adecuados y el equilibrio entre el aprendizaje de la herramienta y el contenido matemático. La experiencia temprana sugiere que estos retos son superables con capacitación y recursos, y que los beneficios obtenidos –en motivación estudiantil, en comprensión aplicada y en competencias adquiridas– justifican plenamente el esfuerzo de innovación educativa. En conclusión, la inclusión de Python en asignaturas como Análisis Matemático I, II y III y Álgebra Lineal en carreras de ingeniería en Argentina potencia la formación integral de los estudiantes. Esta integración les permite egresar no solo con sólidos conocimientos matemáticos, sino también con la habilidad de aplicarlos efectivamente mediante la programación, algo esencial en la práctica profesional moderna. Python, en definitiva, se erige como un catalizador que acerca la teoría a la práctica y prepara a una

nueva generación de ingenieros para liderar proyectos en un mundo cada vez más apoyado en la tecnología y el análisis de datos.

Referencias

1. Ozgur, C., Colliau, T., Rogers, G., & Hughes, Z. (2017). MatLab vs. Python vs. R. Journal of data Science, 15(3), 355-371.
2. N. Aguilera. (2021). Matemáticas y programación con Python para ingenierías. Olimpiada Matemática Argentina. [En línea]. Disponible: https://www.oma.org.ar/invydoc/docs-libro/apuntes.pdf
3. Rondón, A. A., Gómez, E. S., Dalmau, H. H., & Bony, M. M. (2021). Python en la enseñanza de las matemáticas en la carrera de Ingeniería en Ciencias Informáticas. Serie Científica de La Universidad de Las Ciencias Informáticas, 14(5), 181-202.
4. Índice TIOBE TIOBE, acceso en: 01 de Abril de 2025, [En línea]. Disponible:https://www.tiobe.com/tiobe-index/
5. Python Is TIOBE Index Language Of The Year 2024, Mike James, acceso en: 01 de Abril de 2025, [En línea]. Disponible:https://www.i-programmer.info/news/98-languages/17733-python-is-tiobe-index-language-of-the-year-2024.html