

De Wolfram Mathematica a Python: Visualización dinámica del crecimiento poblacional como estrategia de aprendizaje

From Wolfram Mathematica to Python: Dynamic visualization of population
growth as a learning strategy

—

María del Sol Muñoz-Mortera¹

munoz.maria@colpos.mx • ORCID: 0009-0004-8508-7172

Juan Valente Hidalgo-Contreras¹

jvhidalgo@colpos.mx • ORCID: 0000-0002-4300-7777

Paulino Pérez Rodríguez²

perpdgo@colpos.mx • ORCID: 0000-0002-3202-1784

1 COLEGIO DE POSTGRADUADOS (COLPOS), CAMPUS CÓRDOBA. POSGRADO EN
CIENCIAS EN INNOVACIÓN AGROALIMENTARIA SUSTENTABLE. AMATLÁN DE LOS
REYES, VERACRUZ, MÉXICO

2 COLEGIO DE POSTGRADUADOS (COLPOS), CAMPUS MONTECILLO. POSGRADO EN
ESTADÍSTICA Y CIENCIA DE DATOS. TEXCOCO, ESTADO DE MÉXICO

Para citar este artículo:

Muñoz Mortera, M. del S., Hidalgo Contreras, J. V., & Pérez Rodríguez, P. De Wolfram Mathematica a Python: Visualización dinámica del crecimiento poblacional como estrategia de aprendizaje. *Espacio I+D, Innovación más Desarrollo*, 15(45). <https://doi.org/10.31644/IMASD.45.2026.a04>

RESUMEN

El artículo presenta un análisis comparativo sobre cómo la simulación interactiva ayuda a entender el modelo logístico de crecimiento poblacional. Al enseñar únicamente con fórmulas estáticas, los estudiantes tienen dificultades para comprender conceptos abstractos como la tasa intrínseca de crecimiento y la capacidad de carga. Para resolver esto, la investigación evalúa dos softwares Wolfram Mathematica y Python. Metodológicamente, se implementaron simulaciones interactivas bajo un esquema didáctico de "predicción-observación-explicación", y los resultados se evaluaron en tres dimensiones clave: la curva de aprendizaje requerida, la flexibilidad para personalizar el entorno y la capacidad de la herramienta para fomentar un aprendizaje activo. Por un lado, Wolfram permite construir interfaces gráficas rápidamente, pero opera bajo un paradigma pedagógico de "caja negra" que oculta los procesos computacionales al estudiante y depende de costosas licencias comerciales. Por otro lado, Python, usando herramientas de código abierto como Jupyter Notebook, presenta una curva de aprendizaje inicial más exigente. No obstante, fomenta un aprendizaje activo al obligar a los alumnos a desglosar el modelo, programar las librerías matemáticas y comprender la lógica detrás de la simulación. Los resultados muestran que Python y Wolfram Mathematica ofrecen una flexibilidad, permitiendo integrar narrativas teóricas, código ejecutable y paneles interactivos complejos (como simulaciones multiespecie para observar el colapso poblacional) en un solo lugar. El estudio concluye que la visualización dinámica es mucho más efectiva que los métodos pedagógicos estáticos. Se recomienda la transición hacia Python, ya que no solo facilita la enseñanza de sistemas dinámicos de manera prácticamente gratuita, sino que también impulsa la alfabetización digital y democratiza el conocimiento científico, eliminando barreras económicas en la educación STEM por sus siglas en inglés (ciencia, tecnología, ingeniería y matemáticas).

Palabras Clave:

Modelización matemática; crecimiento logístico; software; aprendizaje; didáctica.

— Abstract —

The article presents a comparative analysis of how interactive simulation helps in understanding the logistic model of population growth. When teaching solely with static formulas, students face difficulties in grasping abstract concepts such as intrinsic growth rate and carrying capacity. To address this, the research evaluates two software packages: Wolfram Mathematica and Python. Methodologically, interactive simulations were implemented under a "predict-observe-explain" instructional framework, and the results were assessed across three key dimensions: the required learning curve, the flexibility to customize the environment, and the tool's ability to foster active learning. On one hand, Wolfram allows users to build graphical interfaces quickly, but it operates under a "black box" pedagogical paradigm that hides computational processes from students and relies on expensive commercial licenses. On the other hand, Python, using open-source tools like Jupyter Notebook, presents a steeper initial learning curve. Nonetheless, it promotes active learning by requiring students to break down the model, program mathematical libraries, and understand the logic behind the simulation. The results show that both Python and Wolfram Mathematica offer flexibility, allowing the integration of theoretical narratives, executable code, and complex interactive panels (such as multispecies simulations to observe population collapse) in a single environment. The study concludes that dynamic visualization is far more effective than static pedagogical methods. The transition to Python is recommended, as it not only facilitates the teaching of dynamic systems in a virtually cost-free manner, but also enhances digital literacy and democratizes scientific knowledge by removing economic barriers in STEM (an acronym for science, technology, engineering, and mathematics) education.

Keywords:

Mathematical modeling; Logistic growth; Software; Learning; Didactics.

La evolución del pensamiento en la dinámica de poblaciones ha transitado desde la simplicidad del crecimiento ilimitado hasta el reconocimiento de las restricciones ambientales. Thomas Malthus (1798) estableció el primer principio de la dinámica de poblaciones, postulado que, en ausencia de restricciones, las poblaciones crecen de forma exponencial. Sin embargo, el modelo es poco realista, ya que siempre existen restricciones que limitan el crecimiento, por ejemplo, disponibilidad de espacio, alimento, etc. La ecuación logística ofrece una interpretación más precisa del crecimiento poblacional y constituye una herramienta necesaria en la ecología de poblaciones y la dinámica de poblaciones. A diferencia del crecimiento exponencial maltusiano, que asume recursos ilimitados, el modelo logístico introduce restricciones biológicas realistas mediante la capacidad de carga (K), describiendo cómo las poblaciones se estabilizan al agotar los recursos de su entorno (Ulloa Ibarra & Rodríguez Carrillo, 2010). Este modelo no solo es crucial para entender dinámicas biológicas, sino sirve como una instrucción esencial a las ecuaciones diferenciales no lineales y al comportamiento de sistemas dinámicos complejos (Ulloa Ibarra & Rodríguez Carrillo, 2010).

A pesar de su importancia, los estudiantes a menudo enfrentan dificultades significativas para comprender conceptos abstractos como la tasa intrínseca de crecimiento (r), o la capacidad de carga (K), cuando se presentan únicamente a través de expresiones matemáticas. La falta de visualización inmediata, llevando de la conceptualización compleja a una representación casi física del fenómeno, es lo que impide que el alumno conecte el parámetro matemático con el fenómeno biológico, como la desaceleración del crecimiento o el colapso poblacional (Ulloa Ibarra & Rodríguez Carrillo, 2010). Además, el uso de lenguajes de programación complejos sin una guía visual adecuada puede generar confusión en niveles educativos medios y superiores (Contreras García & Hidalgo Castillo, 2024).

Históricamente, herramientas como Wolfram Mathematica han dominado la enseñanza debido a su potencia para realizar cálculo simbólico, una herramienta excepcionalmente buena que requiere del pago de una licencia para poder usarla. Sin embargo, existe una necesidad imperiosa de transitar hacia herramientas gratuitas de código abierto como Python lo que permite acceso a un mayor número de estudiantes. Esta transición no solo responde a una cuestión económica, sino a la democratización del conocimiento y la alfabetización de datos (Contreras García & Hidalgo Castillo, 2024). Python, es un lenguaje de programación de código abierto con miles de bibliotecas de funciones que permiten realizar tareas complejas en forma transparente para el usuario. El lenguaje puede usarse de forma amigable a través de entornos como Jupyter Notebook, permite una integración fluida de narrativa, código y visualización interactiva, fomentando un aprendizaje activo y reproducible que el software comercial restringe mediante licencias (Contreras García & Hidalgo Castillo, 2024).

La ecuación logística con límites o capacidad de carga (K), sirve como herramienta fundamental para poder predecir el crecimiento poblacional en ecosistemas con

recursos limitados modelando la relación entre el tamaño poblacional y disponibilidad de los recursos. La ecuación incorpora el concepto de capacidad de carga, que define el tamaño máximo de población que un ecosistema puede soportar de manera sostenible en base a los recursos disponibles como alimentos, agua, refugio (Hartvigsen, 2013). A medida que las poblaciones crecen, se acercan a esta capacidad de carga, lo que lleva a una desaceleración en las tasas de crecimiento debido a la limitación de recursos. Las dinámicas poblacionales, que abarcan los cambios en el tamaño de la población a lo largo del tiempo, son captadas efectivamente por el modelo logístico. Este modelo da cuenta de factores como las tasas de natalidad, las tasas de mortalidad y la migración, lo que permite a los investigadores predecir cómo fluctuarán las poblaciones en torno a un equilibrio estable (Gillman, 2005). En la ecuación logística también se integran de manera implícita dentro de los parámetros de la fórmula, factores ambientales, incluyendo el clima y la depredación, que pueden influir aún más en el crecimiento y la estabilidad de la población. Al aplicar este modelo, los científicos pueden evaluar los impactos de diferentes estrategias de manejo sobre la sustentabilidad de la población, informando con ello prácticas de manejo de ecosistemas dirigidas a conservar especies y mantener la salud de los ecosistemas (Breckler, 2006).

FUNDAMENTACIÓN TEÓRICA

Malthus (1798) estableció el modelo de crecimiento de poblaciones de forma exponencial, siendo esta ecuación diferencial:

$$\frac{dN}{dt} = rN,$$

donde N es el tamaño de la población, t es el tiempo y r es la constante de proporcionalidad. Lotka (1925) lo definió como la tasa intrínseca de aumento. Sin embargo, la realidad biológica impone límites. Fisher (1930) denominó a este parámetro r como el "parámetro maltusiano", mientras que Alfred J. Verhulst (1845) introdujo la capacidad de soporte o capacidad de carga (K), integrando un término de frenado ambiental que reconoce la competencia por recursos limitados como espacio y alimento.

La ley de Verhulst (1845), o ecuación logística se expresa como la ecuación diferencial:

$$\frac{dN}{dt} = rN\left(1 - \frac{N}{K}\right),$$

donde K es la capacidad de carga del ambiente y el resto de los términos ya fueron definidos anteriormente. Biológicamente, el término $\left(1 - \frac{N}{K}\right)$ actúa como un "freno" ambiental: a medida que la población se acerca a K , este término tiende a cero, deteniendo el crecimiento. Matemáticamente, esto genera una curva sigmoidea (en

forma de S) donde la velocidad máxima de crecimiento $\left(\frac{dN}{dt}\right)$ ocurre exactamente cuando la población alcanza $\frac{K}{2}$.

- Si N es muy pequeño (pocos individuos), el término es casi 1, y la población crece rápido (como si fuera exponencial).
- Si N se acerca a K (casa llena), el término se acerca a 0, y el crecimiento se detiene.

En este contexto, la tasa de crecimiento per cápita instantánea se define como $\frac{x'(t)}{x(t)}$, siendo:

- $x'(t)$: Es simplemente otra notación matemática para la derivada $\frac{dx}{dt}$. Representa la velocidad total a la que cambia o crece la población en un momento específico t .
- $x(t)$: Es el tamaño total de la población en ese mismo momento t .

Mientras que en modelo de Malthus esta tasa es constante, en el modelo de Verhulst depende de la densidad de población, decreciendo linealmente a medida que N se aproxima a K .

VISUALIZACIÓN DINÁMICA COMO ESTRATEGIA PEDAGÓGICA

La visualización de datos es una habilidad esencial en la educación para la comprensión profunda de variables (Contreras García & Hidalgo Castillo, 2024).

En el ámbito de la investigación, aplicar un modelo determinista sirve como una estrategia fundamental de reduccionismo metodológico. Este enfoque facilita la identificación y análisis de los factores clave de un sistema, lo que ayuda a comprenderlo antes de enfrentarse a la complejidad de sus propiedades emergentes. La capacidad de manipular celdas de código y observar cambios inmediatos en las gráficas fomenta el pensamiento crítico y permite identificar patrones y anomalías que pasarían desapercibidos en un análisis estático.

El objetivo de este artículo es analizar cómo la visualización dinámica de modelos mediante Python y Wolfram Mathematica facilita una comprensión más profunda de la estabilidad del sistema y del comportamiento de las soluciones durante su transición al estado estacionario. Estas herramientas permiten representar de forma rigurosa los fenómenos observados en un entorno controlado.

METODOLOGÍA

Para este estudio comparativo y pedagógico, se desarrollaron simulaciones interactivas del modelo logístico utilizando dos entornos distintos y/o implementaciones

distintas: Wolfram Mathematica y Python. El enfoque técnico se sustenta en las fases de modelización propuestas por Cervantes Gómez (2015):



Figura 1. Enfoque técnico general en la modelización

En la fase de elaboración del modelo, destaca el paso de traducción del fenómeno biológico, ecológico, químico o según sea el caso, donde el investigador traduce el lenguaje del especialista biólogo o ecólogo a variables matemáticas identificables. En este contexto, para este estudio comparativo y pedagógico, se desarrollaron simulaciones interactivas del modelo logístico utilizando dos entornos distintos.

ENTORNO WOLFRAM MATHEMATICA, VERSIÓN 13

En este entorno se aprovecha la capacidad de cálculo simbólico y la facilidad de crear interfaces gráficas rápidas. Mathematica como también se le conoce, destaca por su sintaxis simbólica natural. Se utilizó la función `Dsolve` o `NDSolve` para obtener la solución numérica de la ecuación diferencial. La interactividad se logró mediante el comando `Manipulate`, que permite parametrizar mediante controles deslizantes expresiones matemáticas que dependen de parámetros, como en el caso que nos ocupa r , K y N_0 . Este enfoque permite al estudiante interactuar con el modelo como una “caja negra”, donde el foco está en el resultado visual más que en el algoritmo subyacente que permite resolver la ecuación diferencial. En este entorno la declaración de las variables se realiza de forma directa y cohesiva:

```
(* Declaración de parámetros en Mathematica *)
r = 0.5; K = 100; n0 = 10;
logisticaN[t_] := K / (1 + ((K - n0)/n0)*Exp[-r*t]);
```

Figura 2. Ejemplo de declaración de variables en Wolfram Mathematica

ENTORNO PYTHON, VERSIÓN 3.7

En el ecosistema de código abierto, Python se implementó un script en un entorno Jupyter Notebook/Google Colab, estructurado en bloques funcionales.

1. **Cálculo Numérico:** Se utilizó la librería NumPy para vectorizar la fórmula integrada del modelo logístico $N(t) = \frac{K}{1 + ((K - N_0)/N_0)e^{-rt}}$ y generar arreglos con información de tiempo en un solo paso.
2. **Visualización:** Se empleó la biblioteca de funciones Matplotlib para graficar tanto la curva de crecimiento poblacional (N vs t) como la parábola de velocidad de crecimiento $\frac{dN}{dt}$ vs N .
3. **Interactividad:** Se utilizó la biblioteca de funciones ipywidgets (específicamente interact, FloatSlider, IntSlider) para crear controles deslizantes que permiten al usuario modificar la tasa de crecimiento y la capacidad de carga en tiempo real, actualizando los gráficos instantáneamente sin necesidad de re-ejecutar el código manualmente.

En este entorno la declaración de las variables se realiza de la siguiente forma:

```
# Declaración de parámetros en Python
import numpy as np
r = 0.5
K = 100
n0 = 10
# Definición de la función logística integrada
def logistica_N(t, r, K, n0):
    return K / (1 + ((K - n0) / n0) * np.exp(-r * t))
```

Figura 3. Ejemplo de declaración de variables en Python

ENFOQUE PEDAGÓGICO

La aplicación en el aula se diseñó bajo el esquema de "predicción-observación-explicación". El alumno utiliza los controles deslizantes para responder preguntas guía: ¿Qué sucede con la forma de la curva S si aumentamos r ? o ¿Cómo afecta la población inicial N_0 al tiempo necesario para alcanzar el equilibrio? El script en Python incluye paneles dinámicos que alertan visualmente cuando la población colapsa o cuando los recursos se agotan, conectando el código con conceptos ecológicos reales como el colapso de la población "overshoot".

A continuación, en el Cuadro 1 se muestra una comparativa de las herramientas utilizadas (Wolfram Mathematica y Python), así como sus características.

Cuadro 1

Comparativa de las características de las herramientas de Wolfram Mathematica y Python

Característica	Wolfram Mathematica, V.13	PythonV.3.7 (Jupyter + ipywidgets)
Estructura de Código	Alta cohesión (todo en uno).	Modular (requiere bibliotecas de funciones de terceros).
Facilidad de Implementación	Inmediata (Sintaxis nativa).	Media (Requiere manejo de decoradores).
Sintaxis Interactiva	Manipulate[]	@interact / widgets.interactive
Precisión Matemática	Simbólica y Numérica de alta precisión y complejidad.	Numérica robusta (NumPy/SciPy).
Licencias	Propietario / Académico.	Código Abierto (Gratuito).

COMPARATIVA Y EVALUACIÓN DE ENTORNOS PEDAGÓGICOS (WOLFRAM MATHEMATICA VS. PYTHON)

Para efectuar la comparación pedagógica entre los entornos Wolfram Mathematica y Python, se implementó el modelo de crecimiento logístico en ambas plataformas, observando de manera práctica tanto el proceso de construcción como la interacción obtenida. La evaluación se organizó en torno a tres dimensiones principales:

- La curva de aprendizaje requerida para programar el modelo.
- La flexibilidad del software para personalizar el entorno.
- La capacidad de fomentar un aprendizaje activo.

El objetivo de esta evaluación fue determinar qué herramienta facilita de manera más efectiva la “alfabetización” de problemas matemáticos, estadísticos, etc., donde se vuelve complejo llevar lo intangible a la realidad. Es decir, qué entorno permite al estudiante transitar de ser un receptor pasivo de fórmulas a un investigador que puede experimentar, modificar y comprender críticamente el comportamiento de los sistemas dinámicos.

Autores como Contreras e Hidalgo (2024), validan que el uso de cuadernos interactivos (como Jupyter para Python) permite a los estudiantes manipular celdas de código en tiempo real y observar los efectos de manera inmediata, lo cual materializa conceptos abstractos y promueve una comprensión integral profunda. La evaluación y percepción práctica demostró que Wolfram ofrece interfaces interactivas sumamente rápidas mediante comandos de alto nivel como Manipulate, opera pedagógicamente como una forma más ágil y precisa, sin tanto código. En contraste, Python exige al estudiante integrar librerías como NumPy, Matplotlib e ipywidgets, lo que, si bien requiere un uso más guiado inicialmente para evitar confusiones sintácticas, abre el algoritmo matemático, permitiendo una interactividad más profunda. Esta metodología, similar a la empleada con simuladores

de sistemas complejos, respalda el uso pedagógico de lenguajes de código abierto al convertir la computadora en un laboratorio donde el estudiante controla todas las variables biológicas y matemáticas. Posteriormente, se presentan los resultados obtenidos de la modelación de la ecuación en sus diferentes softwares.

RESULTADOS

La visualización dinámica permite a los estudiantes descomponer el modelo en capas analíticas, observando la sensibilidad de los parámetros en tiempo real. Se presenta en el repositorio GitHub para su consulta:

- Script en Wolfram Mathematica y Python está alojado en el siguiente repositorio: <https://github.com/MariaDelSolMM13/De-Wolfram-Mathematica-a-Python-Visualizacion-dinamica-del-crecimiento-poblacional.git>

A continuación, se presentan los resultados obtenidos de cada visualización.

1. Curva sigmoidea, variación de tasa de crecimiento

Los hallazgos visuales de r (tasa de crecimiento), muestran que al desplazar el control deslizante para r se observa que valores bajos (0.2) producen una curva sigmoide suave, mientras que valores altos (1.0) fuerzan una transición agresiva hacia K . En otras palabras, en valores altos de r se muestra cómo la población crece exponencialmente al principio y se estabiliza al llegar a la asíntota de K . A continuación, en las Figuras 4 y 5, se presenta la visualización de la evaluación de los valores de la tasa de crecimiento ($r \in \{0.2, 0.5, 1.0\}$) de cada uno de los programas utilizados (Wolfram Mathematica y Python).

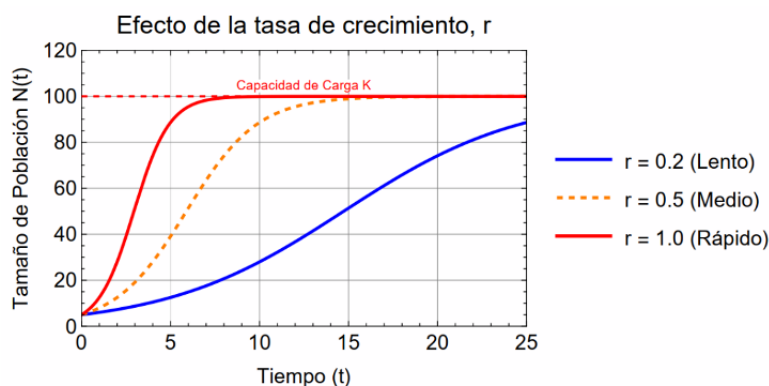


Figura 4. Evaluación de la tasa de crecimiento, con Wolfram Mathematica

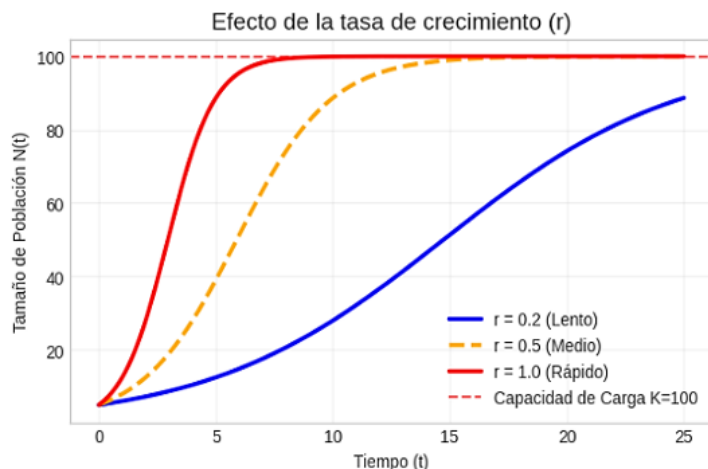


Figura 5. Evaluación de la tasa de crecimiento, con Python

La generación de la curva Sigmoidal en los ambientes como Wolfram Mathematica resulta eficiente y directo gracias al comando Manipulate, este devuelve la función Plot y genera automáticamente los controles deslizantes sin que el usuario deba de programar lógica interna de actualización de la interfaz. Por su parte Python requiere una estructura de código más explícita y fragmentada: exigiendo separar la definición matemática de la función logística, generar vectores numéricos con librerías como NumPy (`np.linspace`), y configurar los sliders y sus vinculaciones usando ipywidgets. Aunque Wolfram Mathematica parece ofrecer una inmediatez superior para construir el prototipo visual, Python procesa cálculo de manera extremadamente eficiente gracias a los arreglos vectorizados actualizando gráficas fluidas en tiempo real al variar r dentro de cuadernos interactivos.

2. Parábola de velocidad, punto de máximo crecimiento $\frac{K}{2}$.

Los hallazgos visuales del punto máximo de crecimiento $\frac{K}{2}$, muestran la velocidad de crecimiento de $\frac{dN}{dt}$ la cuál describe una parábola cuando se grafica frente a N . El punto máximo ocurre exactamente en $N = \frac{K}{2}$. Este es el punto de inflexión en la sigmoide, donde la población pasa de una fase de aceleración (predominio del crecimiento exponencial) a una de desaceleración (predominio del freno ambiental). A continuación, en las Figuras 6 y 7, se presenta de manera visual el punto máximo de crecimiento $\frac{K}{2}$. En ambas figuras se observa que cuando la población (N) es reducida, la velocidad de crecimiento $\frac{dN}{dt}$ se acelera rápidamente gracias a la abundancia de espacio y alimento, alcanzando su punto de mayor rapidez exactamente a la mitad de la capacidad del ecosistema $\frac{K}{2}$, correspondiente a 50 individuos). Pasando ese pico, la competencia y la escasez natural provocan que el crecimiento se frene de forma progresiva hasta detenerse por completo al llegar a la capacidad

de carga máxima ($N=K$) equivalente a 100 individuos, donde si por alguna razón la población supera los 100 individuos, la curva cae a números negativos. Esto indica que habría más muertes que nacimientos debido a la falta de recursos y la población decaería hasta volver al equilibrio en 100.

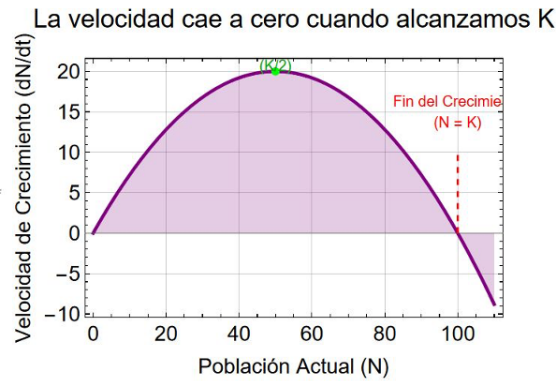


Figura 6. Punto máximo de crecimiento, con Wolfram Mathematica

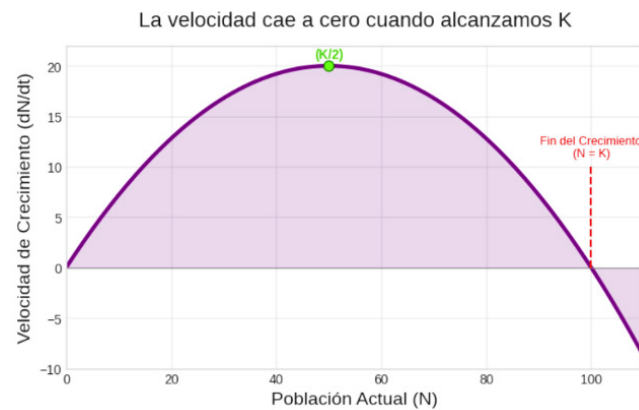


Figura 7. Punto máximo de crecimiento, con Python

Para realizar la gráfica de la tasa de cambio $\frac{dN}{dt}$ vs N y el momento crítico donde la población alcanza la mitad de su capacidad de carga $\frac{K}{2}$, ambos entornos permiten incrustar en las visualizaciones elementos dinámicos, pero difieren en el control sobre capas. Wolfram Mathematica maneja la interactividad del punto mediante el comando Epilog, lo que resulta muy funcional y requiere pocas líneas de código para señalar la posición de la curva. Sin embargo, Python demuestra eficiencia en la personalización visual de Matplotlib. Esta librería permite al estudiante programar capas secuenciales detalladas, como rellenar el área bajo la parábola usando fill_between, o definir colores, marcadores y alertas condicionales según el estado de la población con un nivel de granularidad alta. Esto hace que en Python la gráfica deje de ser un simple dibujo y se convierta en un lienzo adaptable para fines pedagógicos.

3. INTEGRACIÓN DINÁMICA DE PANEL INTERACTIVO, EVALUACIÓN DE CRECIMIENTO POBLACIÓN Y VELOCIDAD

Este panel interactivo permite analizar de forma dinámica los parámetros biológicos como la tasa de crecimiento r , la capacidad máxima de crecimiento K y la población inicial. Así, es posible observar cómo estos parámetros cambian a lo largo del tiempo y evaluar su comportamiento al modificar alguno de ellos, véanse las Figuras 8 y 9.

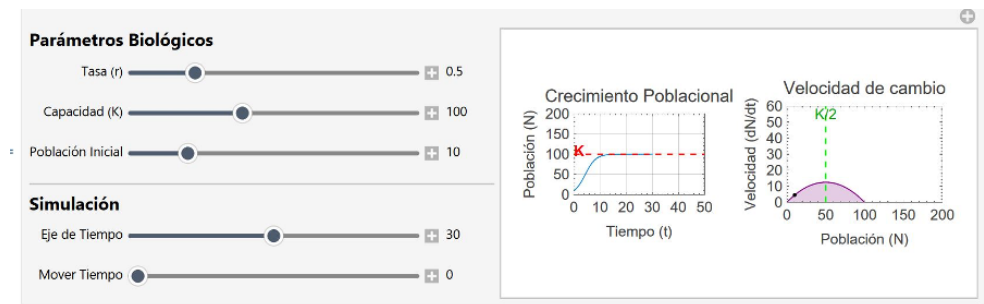


Figura 8. Evaluación de los parámetros biológicos, con Wolfram Mathematica

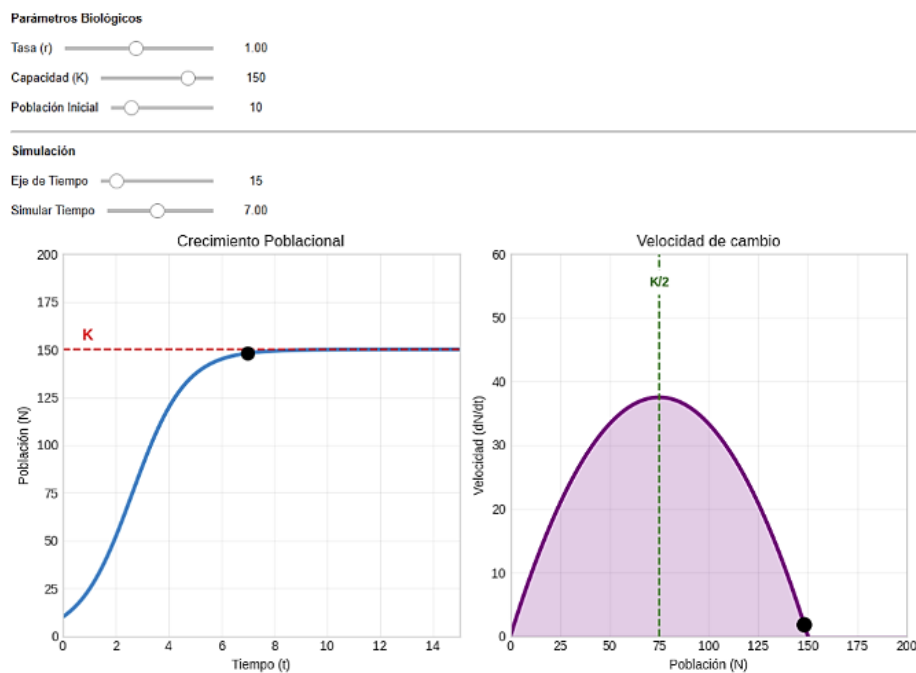


Figura 9. Evaluación de los parámetros biológicos, con Python

Para evaluar la capacidad de integrar múltiples representaciones visuales en un solo panel, por ejemplo, graficar simultáneamente la curva poblacional $N(t)$ y la

parábola de velocidad $\frac{dN}{dt}$, se observan diferencias arquitectónicas clave. En Wolfram Mathematica la creación de un panel unificado es sumamente rápida, ya que se pueden agrupar múltiples gráficos usando comandos como Grid o GraphicsRow dentro del mismo bloque Manipulate. Sin embargo, a medida que se añaden elementos, la interfaz puede volverse rígida. Por otro lado, Python demuestra una eficiencia superior en la modularidad del diseño de interfaces (UI). Utilizando matplotlib.pyplot.subplots (1, 2), Python permite renderizar ambos gráficos lado a lado con un control milimétrico sobre los ejes, colores y sombreado. Además, la combinación de ipywidgets con contenedores como VBox y HBox separa eficientemente la lógica del cálculo matemático de la interfaz gráfica. Desde una perspectiva pedagógica, el enfoque de Python es más robusto, ya que permite al estudiante observar con fluidez web¹ cómo el movimiento de un solo deslizador altera instantáneamente dos dominios matemáticos distintos (el temporal y el de fases), operando como un verdadero tablero de control analítico.

4. PANEL INTERACTIVO SOBRE EL CRECIMIENTO EXCESIVO Y EL COLAPSO DE LA POBLACIÓN (OVERSHOOT)

La evaluación del crecimiento excesivo y el colapso de la población (overshoot), se presenta de manera visual a través de un panel interactivo multiespecie que evalúa tres organismos: bacteria, ballenas y humano, cada uno con sus respectivos parámetros. En estos tres escenarios se desea plasmar la importancia de las variables, a lo largo del tiempo, en este caso para el script se agregó en el panel una alerta visual donde al sobrepasar alguno de los parámetros se mostrará, “recursos limitados”, “sé sobre colapsó”, etc. De esta forma se puede utilizar este último para reflexionar cómo evoluciona la población en diferentes escenarios donde existe una limitación de recursos, de medio ambiente, entre otros. Esto puede servir, para crear interés entre los alumnos, y puedan jugar con los parámetros, y por ejemplo encontrar, en qué escenario, la población llegaría a su punto de equilibrio o donde los decesos son mayores a los nacimientos, o los recursos como el alimento y/o espacio se acabaron, entre otros. En las Figuras 10 y 11, se presenta de manera visual, el panel multiespecie.

1 La **fluidez web** en aplicaciones visuales combina la fluidez en el procesamiento perceptivo y la fluidez en la interfaz digital. Es la facilidad con la que los usuarios interpretan los diseños web y la perfecta capacidad de respuesta de las interfaces gráficas de usuario. Resource

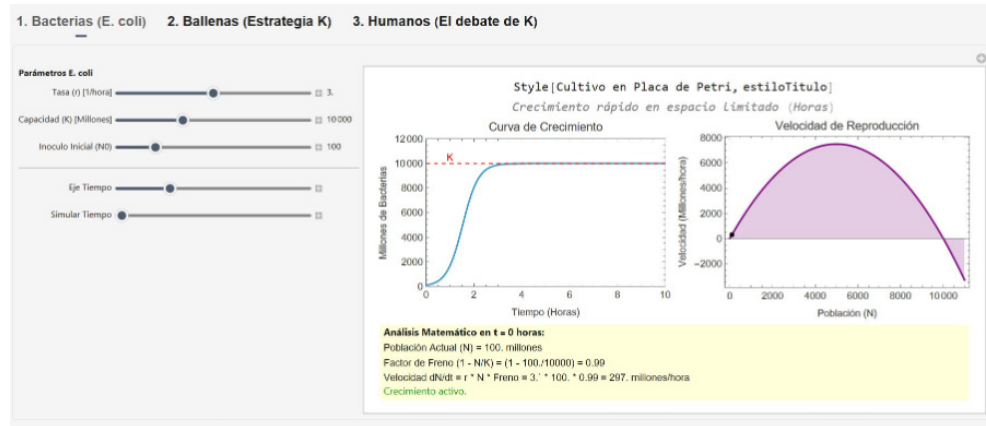


Figura 10. Panel multispecie (bacterias, ballenas, humanos), en Wolfram Mathematica

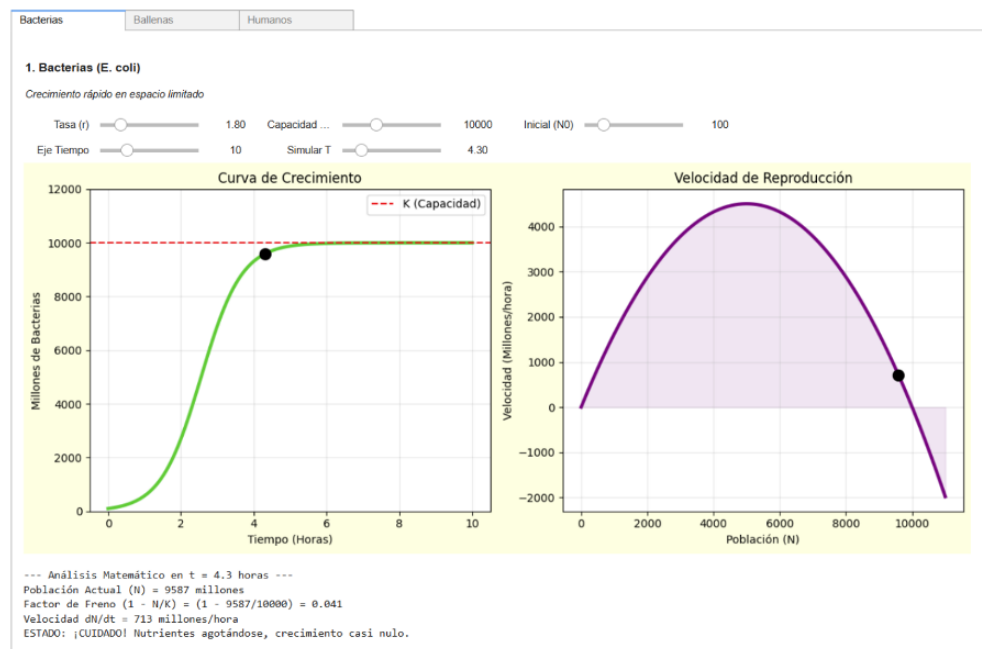


Figura 11. Panel multispecie (bacterias, ballenas, humanos) en Python

Para simular escenarios complejos y no lineales como el "overshoot" (cuando una población supera su capacidad de carga y agota los recursos), Python presenta una clara ventaja en la construcción de narrativas interactivas. En el entorno de Jupyter/Colab, Python permite el uso de organizadores como TabView, lo que facilita incrustar múltiples escenarios ecológicos (bacterias, ballenas, humanos) en un solo cuaderno sin saturar la pantalla. Python para manejar la lógica condicional integrada con la visualización lo hace a través de sentencias if. Los scripts de Python y Wolfram Mathematica evalúan en tiempo real métricas críticas (como el factor de freno ambiental) y generan alertas textuales dinámicas que cambian de color,

por ejemplo: "*¡CUIDADO! Nutrientes agotándose*" en rojo si el espacio disponible cae bajo un umbral, o "*Escenario Pesimista: Estamos chocando con el límite*" si se simula un colapso poblacional humano. Si bien Wolfram Mathematica puede procesar estas sentencias lógicas internamente, la integración de narrativa, código y alertas condicionales visuales en Python resulta pedagógicamente mucho más inmersiva, aunque el código en Python será mayor al tratar de indicar cada una de las sentencias. En este sentido para el alumno ambas herramientas se transforman en una herramienta más allá de una simple calculadora en un instrumento donde el alumno visualiza en tiempo real las consecuencias del agotamiento de recursos.

RESULTADOS DE LA COMPARATIVA Y EVALUACIÓN DE ENTORNOS PEDAGÓGICOS (WOLFRAM MATHEMATICA VS. PYTHON)

De manera general ambas herramientas son potentes, y casi se puede decir que su uso sería prácticamente por la preferencia del usuario, sin embargo, al comparar la eficiencia en el proceso de enseñanza-aprendizaje, Wolfram Mathematica presenta una barrera mucho menor para lograr simulaciones funcionales. No obstante, opera bajo un paradigma de "caja negra" que oculta los procesos de renderizado y cálculo iterativo. Por el contrario, aunque Python exige a los alumnos una mayor alfabetización inicial en programación (importar librerías, comprender la sintaxis de variables, y diseñar la disposición con elementos como VBox o interactive_output) es pedagógicamente más eficiente a largo plazo. Desglosar el modelo en Python obliga al estudiante a entender cómo las matemáticas se traducen en lógica computacional, logrando una asimilación profunda de los conceptos que trasciende el simple uso pasivo de una interfaz.

En el aspecto de la flexibilidad general, Python supera ampliamente a Wolfram Mathematica, mientras que Mathematica es potente, se encuentra en un ecosistema propietario cerrado; Python es un lenguaje de código abierto, gratuito, altamente versátil y popular. Al utilizar plataformas como Jupyter Notebook o Google Colab, Python permite integrar explicaciones detalladas (incluso en formato LaTeX), código de ejecución y visualizaciones interactivas complejas tipo "dashboard" (como incluir múltiples pestañas para comparar bacterias, ballenas y humanos en un solo bloque). Esta arquitectura abierta y modular confirma que Python es el entorno más eficiente para democratizar el conocimiento analítico, ya que no depende de costosas licencias comerciales para que instituciones y alumnos desarrollen simulaciones de calidad científica.

DISCUSIÓN

Este apartado es una sección de confrontación con las opiniones o hallazgos de otros investigadores, donde nuestros resultados sobre la implementación interactiva del modelo logístico adquieren un contexto analítico más amplio. Por un lado,

nuestra observación respecto a que la programación en Python presenta una curva de aprendizaje inicial más empinada concuerda directamente con Thompson et al. (2020), quienes advierten que el uso de la programación en el aula de matemáticas requiere acompañamiento guiado para que los estudiantes puedan aprovechar la exploración de datos sin frustrarse por la sintaxis. No obstante, una vez superada esta barrera técnica, nuestros resultados validan firmemente la postura de Contreras y Castillo (2024), quienes sostienen que los cuadernos interactivos (Jupyter Notebooks) son herramientas invaluable porque permiten combinar explicaciones teóricas con código ejecutable, promoviendo así un aprendizaje verdaderamente activo frente a la pasividad del software de "caja negra". Esta transición metodológica transforma a la computadora en un "potente laboratorio" virtual que da vida a conceptos abstractos, una idea fuertemente respaldada por Arrieta y Riveros et al. (2013) y Riveros et al. (2011), al concluir que las Tecnologías de la Información y Comunicación (TIC) favorecen el desarrollo del pensamiento crítico al permitir al estudiante observar en tiempo real cómo varían las gráficas al modificar parámetros específicos. Finalmente, la efectividad de aplicar estas herramientas al crecimiento poblacional coincide con las conclusiones de Ulloa y Rodríguez (2010), quienes determinan que la simulación computacional interactiva del modelo logístico es indispensable no solo para que el alumno internalice dinámicas no lineales (como la capacidad de carga ambiental), sino para dotarlo de bases sólidas en la toma de decisiones ecológicas y científicas.

RECOMENDACIONES

Para un investigador más experimentado debe de tomar en consideración que el modelo de Verhulst es una simplificación inicial. Y que, para una formación avanzada, se debe introducir, el "Efecto Alle", donde la tasa de crecimiento intrínseca es máxima a densidades intermedias, modelando la dificultad de encontrar pareja en poblaciones bajas. Así como también la "Ley de Gompertz", que es utilizada en oncología clínica para tumores donde la tasa de crecimiento decrece logarítmicamente con la masa: $\frac{dx}{dt} = -Kx \ln(x)$, cada término comparte significado con los términos de la función logarítmica, mencionados en el artículo.

CONCLUSIONES

La visualización dinámica se presenta como una estrategia pedagógica superior a los métodos estáticos. Permite a los estudiantes internalizar conceptos abstractos como la "estabilidad asintótica" observando directamente cómo las curvas convergen a K independientemente de la condición inicial. Por lo que el uso de los scripts en Python y Wolfram Mathematica cumplen con el objetivo de dotar de "vida" a la ecuación de Verhulst (1845), permitiendo al estudiante reflexionar y aprender con mayor facilidad estos fenómenos de la vida real.

Se destaca la importancia de la democratización del software, mientras Wolfram Mathematica es una herramienta potente pero comercial y costosa, Python es gratuito y de código abierto. Introducir Python en el aula no solo enseña matemáticas o biología, si no que alfabetiza digitalmente a los estudiantes en una herramienta estándar en la ciencia, en particular el análisis y visualización de datos. Esto facilita una educación STEM de calidad y acceso igualitario a herramientas avanzadas, eliminando barreras económicas.

REFERENCIAS

- Breckler, M. E. (2006). *The Evolving Face of Ecosystem Management* [Honors thesis, The Ohio State University]. <http://hdl.handle.net/1811/6009>
- Cervantes Gómez, L. (Ed.). (2015). *Modelización matemática: Principios y aplicaciones*. Benemérita Universidad Autónoma de Puebla. Facultad de Ciencias Físico Matemáticas.
- Contreras García, J. M., & Hidalgo Castillo, C. (2024). Exploración y visualización de datos en Educación Matemática con Jupyter y Python. *Épsilon – Revista de Educación Matemática*, 118, 81–89.
- Elias Arrieta, J. (2013). *Las TIC y las matemáticas, avanzando hacia el futuro*. <https://repositorio.unican.es/xmlui/handle/10902/3012>
- Fisher, R. A. (1930). *The Genetical Theory of Natural Selection*. Clarendon Press. <https://doi.org/10.5962/bhl.title.27468>
- Gillman, M. P. (2005). Population Dynamics: Introduction. In *Encyclopedia of Life Sciences*. John Wiley & Sons. <https://doi.org/10.1038/npg.els.0003164>
- Hartvigsen, G. (2013). Carrying Capacity, Concept of. In S. A. Levin (Ed.), *Encyclopedia of Biodiversity* (2nd ed., pp. 695–701). Academic Press. <https://doi.org/10.1016/B978-0-12-384719-5.00020-4>
- Lotka, A. J. (1925). *Elements of Physical Biology*. Williams & Wilkins.
- Malthus, T. R. (1798). *An Essay on the Principle of Population, as It Affects the Future Improvement of Society, with Remarks on the Speculations of Mr. Godwin, M. Condorcet, and Other Writers*. J. Johnson.
- Riveros, V., Mendoza, M. I., & Castro, R. (2011). Las tecnologías de la información y la comunicación en el proceso de instrucción de la matemática. *Quórum Académico*, 8(15). <https://www.produccioncientificaluz.org/index.php/quorum/article/view/29272>
- Thompson, J., Wu, S. P. W., & Mills, J. (2020, June 22). *The Use of Computer Programming in a Secondary Mathematics Class*. 2020 ASEE Virtual Annual Conference Content Access. <https://peer.asee.org/the-use-of-computer-programming-in-a-secondary-mathematics-class>
- Ulloa Ibarra, J. T., & Rodríguez Carrillo, J. A. (2010). El modelo logístico: Una alternativa para el estudio del crecimiento poblacional de organismos. *REDVET. Revista Electrónica de Veterinaria*, 11(3). <https://www.redalyc.org/articulo.oa?id=63613123024>
- Verhulst, P.-F. (1845). Recherches mathématiques sur la loi d'accroissement de la population. *Nouveaux Mémoires de l'Académie Royale Des Sciences et Belles-Lettres de Bruxelles*, 18, 1–45.