Monografía IB Física

Inteligencia Artificial como Herramienta para Predecir Leyes Astrofísicas

¿Cuál es la ley que determina la masa del halo de materia oscura de una galaxia en función de su masa estelar y su tasa de formación estelar?

Palabras: 3995

Tabla de Contenidos

1. Introducción 4

2. Fundamento Teórico 4

2.1. Conceptos de Astrofísica 5

2.1.1. Unidades 5

2.1.2. Parámetros Cosmológicos 5

2.1.3. Redshift 5

2.1.4. Galaxias 6

2.1.5. Simulaciones Cosmológicas *N-body* 7

2.2. Regresión con Inteligencia Artificial 8

2.2.1. Modelos de Regresión *Black-Box* 8

2.2.2. Regresión Simbólica 8

3. Métodos 9

3.1. Metodología 9

3.2. El Catálogo *Uchuu-UM* 10

3.3. *AI-Feynman* 11

4. Análisis de los Datos 13

4.1. Masa del Halo de Materia Oscura frente a Masa Estelar 13

4.1.1. Función de las Medianas de la Masa del Halo de Materia Oscura 13

4.1.2. Función de Densidad de Probabilidad 16

4.2. Masa del Halo de Materia Oscura frente a Masa Estelar y Tasa de Formación Estelar 17

5. Resultados 18

5.1. Masa del Halo de Materia Oscura frente a Masa Estelar 18

5.1.1. Función de las Medianas de la Masa del Halo de Materia Oscura 18

5.1.2. Función de Densidad de Probabilidad 21

5.2. Masa del Halo de Materia Oscura frente a Masa Estelar y Tasa de Formación Estelar 23

5.2.1. Función de las Medianas de la Masa del Halo de Materia Oscura 23

5.2.2. Función de Densidad de Probabilidad 25

6. Conclusión 26

7. Bibliografía 28

Anexos 30

A. Código 30

1. Introducción

En el siglo XVII, Johannes Kepler empleó datos empíricos de planetas del sistema solar para encontrar relaciones entre magnitudes astrofísicas, como la tercera ley de Kepler, que enuncia

$$T^2 \propto a^3 \tag{1}$$

donde T es el período orbital de un planeta y a la distancia media del planeta al Sol.

El trabajo de Kepler con su tercera ley es un ejemplo de regresión simbólica, pues buscó en el espacio de expresiones matemáticas una que se ajustara a los datos. Kepler no empleó nada más que un papel y datos para enunciar sus leyes y, a medida que la física avanzó, se pudo comprobar que eran ciertas. Actualmente existen herramientas que ayudan a automatizar este proceso utilizando la inteligencia artificial (IA). Esta investigación utilizará un modelo reciente de IA especializado en encontrar expresiones físicas, denominado *AI-Feynman* [1], para comprobar su utilidad en el avance científico.

Se investigará la dependencia de la masa del halo de materia oscura de una galaxia, M_{halo}, con su masa estelar, M_*, y su tasa de formación estelar, SFR, pues estas variables están relacionadas [2], aunque no existe una expresión matemática que lo describa. Para ello, se emplearán los datos del catálogo de galaxias *Uchuu-UM* [3].

Por tanto, la pregunta de investigación que se plantea es ¿cuál es la ley que determina la masa del halo de materia oscura de una galaxia en función de su masa estelar y su tasa de formación estelar?

2. Fundamento Teórico

La astrofísica es el campo principal de esta investigación, pues se pretende hallar una ley física que relacione distintas propiedades de una galaxia. Por otro lado, un algoritmo de inteligencia artificial analizará estos datos y obtendrá resultados. En este paso, también es esencial el conocimiento de la astrofísica, pues la IA no comprende el sentido físico de las

magnitudes físicas usadas; solo las analiza estadísticamente, por lo que un humano tendrá que decidir si finalmente se puede aceptar una ley o no.

2.1. Conceptos de Astrofísica

2.1.1. Unidades

La Tabla 1 muestra las unidades empleadas en esta investigación.

Unidad	Símbolo	Magnitud	Valor aproximado	Descripción
Masa Solar	$M_\odot$	Masa	$2{,}00 \cdot 10^{30} kg$	Masa del Sol
Unidad Astronómica	UA	Longitud	$1{,}50 \cdot 10^{11} m$	Distancia media entre la Tierra y el Sol
Parsec	pc	Longitud	$3{,}09 \cdot 10^{16} m$	Distancia a la que una UA subtiende un ángulo de 1"

Tabla 1. Unidades astrofísicas más relevantes en esta investigación.

2.1.2. Parámetros Cosmológicos

No conocemos con seguridad los valores de los parámetros fundamentales que determinan el funcionamiento del universo, por lo que se suele asumir un modelo cosmológico que los especifique.

Los principales parámetros cosmológicos son los de densidad, que determinan la densidad de materia (Ω_M) y energía oscura (Ω_Λ) del universo, y su geometría (Ω_K). Los tres cumplen [4] para el modelo cosmológico estándar

$$\Omega_M + \Omega_\Lambda + \Omega_\mathrm{K} = 1 \tag{2}$$

2.1.3. Redshift

A medida que un cuerpo en el espacio se aleja de un punto de referencia, la frecuencia de las ondas de luz emitidas por este irá disminuyendo debido al efecto Doppler. El *redshift*, z, se define de la siguiente forma [4]

$$z \equiv \frac{\nu_e}{\nu_o} - 1 \tag{3}$$

donde ν_o es la frecuencia observada en el punto de referencia y ν_e la emitida originalmente por el cuerpo en movimiento.

El *redshift, z,* es una magnitud que proporciona el tiempo para un modelo cosmológico, y es muy importante en esta investigación, pues a medida que aumenta, el número de galaxias y sus propiedades cambian. Se ha optado por hacer el estudio propuesto a $z = 0$, que corresponde a la edad actual del universo.

2.1.4. Galaxias

Una galaxia es un sistema de estrellas, remanentes estelares, gas interestelar, polvo y materia oscura unidos por la fuerza de la gravedad. Estas van evolucionando con el tiempo, por lo que para comparar propiedades entre distintas galaxias, deben estar en el mismo *redshift*.

Las galaxias están formadas por estrellas, y su masa estelar, M_*, mide la suma de la masa de todos estos astros, que puede variar con fenómenos como la formación y evolución de cada una de las estrellas. Otra propiedad es la tasa de formación estelar, SFR, que indica cuánta masa de estrellas se forma por unidad de tiempo, por lo que está relacionada con la masa estelar, M_*.

Las galaxias residen en una región, el halo galáctico, que está formada principalmente por materia oscura. Este halo de materia oscura tiene masa, M_{halo}, y publicaciones [2] indican una relación entre esta magnitud y M_* y SFR de la galaxia que reside en el halo de materia oscura (ver Fig. 1).

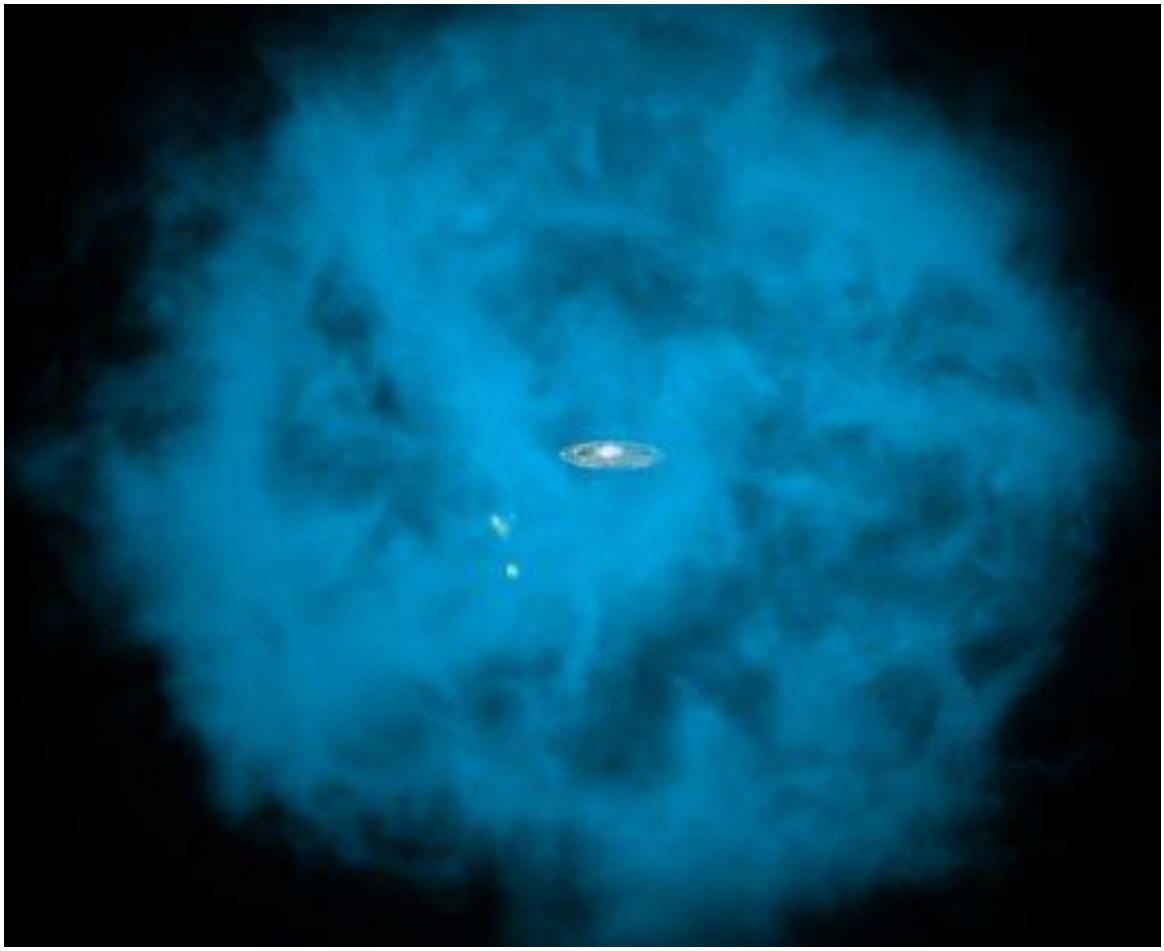

Fig. 1. Representación de una galaxia con su correspondiente halo de materia oscura [5].

La Tabla 2 muestra las propiedades de las galaxias relevantes en esta investigación.

Propiedad	Símbolo	Unidades
Masa Estelar	M_*	$M_\odot$
Tasa de Formación Estelar	SFR	$M_\odot yr^{-1}$
Masa del Halo de Materia Oscura	M_{halo}	$M_\odot$

Tabla 2. Propiedades de las galaxias empleadas en esta investigación.

2.1.5. Simulaciones Cosmológicas *N-body*

Las simulaciones cosmológicas *N-body* modelizan cómo se comportarían partículas de materia oscura en el universo al estar sometidas a interacciones exclusivamente gravitatorias, y permiten estudiar la formación de estructuras y halos de materia oscura

para analizar las propiedades estadísticas de la distribución de galaxias en el universo. Para realizar estas simulaciones, se debe fijar un modelo cosmológico.

2.2. Regresión con Inteligencia Artificial

Los modelos de regresión pretenden predecir una variable dependiente a partir de una o más variables independientes. En esta investigación, se empleará un modelo de regresión con inteligencia artificial para hallar la ley que calcula M_{halo} a partir de M_* y SFR.

2.2.1. Modelos de Regresión *Black-Box*

Muchas inteligencias artificiales emplean redes neuronales. No obstante, estos algoritmos tan potentes no son aptos para esta investigación, pues son modelos *black-box* que dado un conjunto de datos de entrada (*inputs*) producen un resultado (*output*) mediante un algoritmo desconocido para el usuario, ya que sacrifican la interpretabilidad del modelo a favor de mejores resultados, y en esta investigación es importante precisamente este elemento que permanece oculto. Sin embargo, estos modelos *black-box* pueden complementar otros algoritmos (ver sección 3.3).

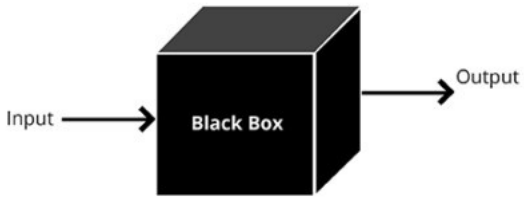

Fig. 2. Diagrama simple de modelos de inteligencia artificial *black-box* [6].

2.2.2. Regresión Simbólica

En el otro lado del espectro está la regresión simbólica [7], una técnica que busca en el espacio de fórmulas matemáticas la que mejor se ajuste al problema a partir de operaciones básicas.

Este método fue popularizado por Eureqa [8], un software que empleaba un algoritmo genético[1] para buscar estas expresiones matemáticas. Aunque estos algoritmos son útiles para expresiones matemáticas sencillas, no cuentan con ningún mecanismo especializado para tratar expresiones físicas, que cumplen propiedades como la homogeneidad dimensional. Esto ha hecho que nazcan alternativas más potentes a estos algoritmos. La más avanzada actualmente es *AI-Feynman*.

3. Métodos

Para hallar la relación matemática entre M_{halo} en función de M_*y SFR, se usan los datos de la simulación *Uchuu-UM* para entrenar la inteligencia artificial *AI-Feynman*, que predice la ecuación buscada.

3.1. Metodología

La idea general de este método consiste en encontrar una expresión f que relacione una variable dependiente y con un conjunto de variables independientes $\vec{x} = \{x_1, x_2, \dots, x_n\}$, de forma que

$$y = f(\vec{x}) \tag{4}$$

Las variables independientes son $\vec{x} = \{M_*, SFR\}$[2] y la variable dependiente $y = M_{halo}$. Para comprobar que los resultados son correctos, se debe entrenar el modelo varias veces con ligeras modificaciones en los datos y ver que no existen diferencias notables en las funciones obtenidas.

Durante las primeras pruebas, se observó que la variable M_{halo} presenta una gran dispersión con respecto a M_* y SFR (hay muchos valores de M_{halo} para cada M_* y SFR). Por

[1] Los algoritmos genéticos funcionan de forma similar a la selección natural, pero con expresiones matemáticas que sufren variaciones para constituir la siguiente población.

[2] Existen más variables que podrían tener relación con M_{halo} pero investigaciones indican que sus factores de dependencia son bajos [2].

ello, los datos se agruparán en intervalos y se obtendrá la ecuación con estos valores. Estos intervalos tendrán un solo valor de M_{halo} (la mediana de M_{halo} en el intervalo), de M_* (la media pesada de M_* en el intervalo) y de SFR (la media pesada de SFR en el intervalo).

Posteriormente, se buscará una función de densidad de probabilidad que determine la probabilidad de cierto valor de M_{halo} en función de la mediana, $\tilde{M}_{halo}$, en un intervalo de M_* y SFR. Por tanto, las expresiones que se obtendrán son

$$\tilde{y} = f(\vec{x}) \tag{5}$$

$$P(y) = p_y(\tilde{y}, \vec{x}) \tag{6}$$

donde $\tilde{y}$ representa la mediana de la variable dependiente y p_y es la función de densidad de probabilidad de y con parámetros $\tilde{y}$ y $\vec{x}$.

Además, como prueba inicial del modelo de regresión simbólica empleado, se hallarán primero estas dos funciones para una sola variable independiente, $\vec{x} = \{M_*\}$. Esto producirá resultados aceptables, pues existe una relación entre las dos variables independientes M_* y SFR, debido a que ambas magnitudes fueron generadas a partir de la masa del halo, M_{halo}, de la galaxia (ver sección 3.2).

Finalmente, es importante destacar que se trabajará con los logaritmos en base 10, $\log_{10} x$, de las variables, pues estas toman valores muy grandes que pueden dar problemas a *AI-Feynman*. Esto, además, reduce la dispersión de los datos y muestra relaciones entre variables de forma más sencilla.

3.2. El Catálogo *Uchuu-UM*

Uchuu-UniverseMachine [3] es un catálogo de galaxias con sus propiedades desde $z = 0$ hasta $z = 10$, generado mediante el algoritmo *UniverseMachine* [9] y la simulación cosmológica *N-body Uchuu* [10]. *UniverseMachine* asigna galaxias y sus propiedades

(incluyendo M_* y SFR) a los halos de la simulación *Uchuu* mediante observaciones realizadas sobre la relación galaxia-halo.

La Tabla 3 detalla los valores de los parámetros cosmológicos relevantes de *Uchuu*.

Parámetro	Valor
$\boldsymbol{\Omega_M}$	0,3089
$\boldsymbol{\Omega_\Lambda}$	0,6911
$\boldsymbol{\Omega_K}$	0,0000

Tabla 3. Parámetros cosmológicos relevantes de *Uchuu*, los cuales corresponden a los del modelo cosmológico estándar [11].

3.3. *AI-Feynman*

El modelo de regresión simbólica que se empleará será *AI-Feynman*[3], publicado en 2020 por investigadores del MIT ([1] y [14]), pues está especializado en encontrar leyes físicas usando técnicas como el análisis dimensional para encontrar mejores resultados.

Los datos de entrada de *AI-Feynman* son un fichero en el que cada línea contiene las variables independientes $\vec{x}$ y la variable dependiente y. La cantidad de puntos necesarios varía desde 50 para funciones sencillas hasta millones para las más complejas.

El funcionamiento de *AI-Feynman* se divide en 5 pasos (ver Fig. 3):

1. **Análisis Dimensional:** Se simplifica la expresión a una adimensional con menos variables mediante sustituciones. No se hará uso de este paso porque se tomarán logaritmos de las magnitudes y son adimensionales.
2. **Ajuste Polinomial:** Se intenta ajustar la ecuación a un polinomio de grado no muy alto determinado por un parámetro modificable.
3. **Fuerza Bruta:** Se prueban todas las combinaciones posibles de operaciones hasta que la expresión tenga un error menor a un parámetro modificable o pase un tiempo especificado por otro parámetro.

[3] Se probaron otros modelos ([12] y [13]), pero los resultados de AI-Feynman fueron superiores.

4. **Red Neural y Transformaciones:** Se intenta explotar propiedades comunes en leyes físicas como la simetría traslacional o separabilidad de algunas variables. Para comprobar estas propiedades suele ser necesario usar puntos determinados que no están en los datos de entrada, por lo que se entrena una red neuronal para hallar estos nuevos puntos. Una vez aplicadas las transformaciones se repite el proceso desde el inicio, y cuando no queden más se continúa al siguiente paso.
5. **Otras Transformaciones:** Se aplican distintas funciones como $\sqrt{x}$ o e^x a las variables y se vuelve al inicio del algoritmo para ver si se obtienen mejores resultados. Cuando no hay más funciones que aplicar, termina el algoritmo.

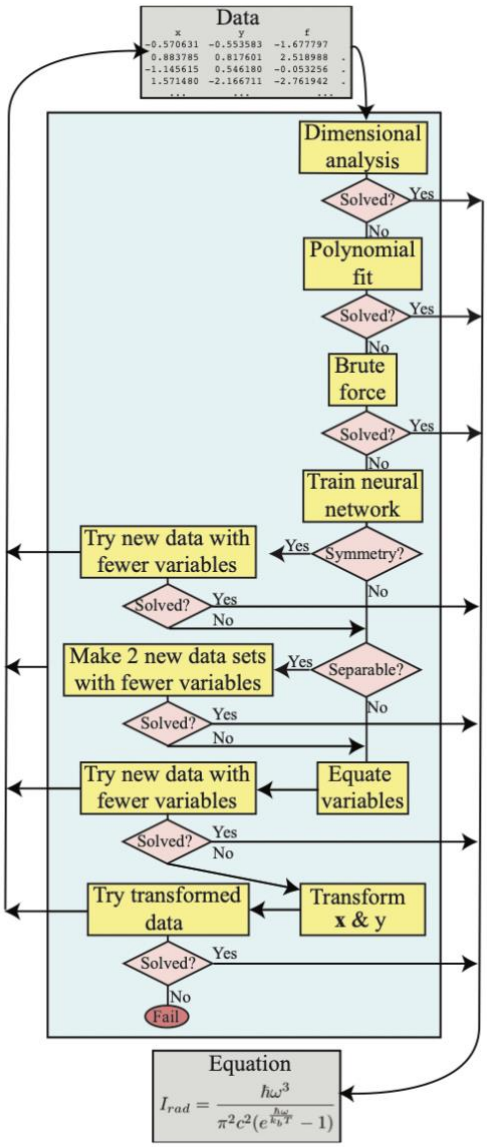

Fig. 3. Diagrama detallando el funcionamiento de *AI-Feynman* [1].

Además, pueden especificarse parámetros para controlar el entrenamiento del modelo (ver Tabla 4).

Parámetros	Descripción
Tiempo Máximo en la Fase de Fuerza Bruta	Esta fase terminará si el error de la expresión obtenida es suficientemente pequeño o se ha excedido este tiempo.
Operaciones en la Fase de Fuerza Bruta	Lista de operaciones posibles durante la fase de fuerza bruta.
Grado Máximo en el Ajuste Polinomial	Grado máximo del polinomio durante el ajuste polinomial. Se probarán polinomios de grado menor o igual al seleccionado.
Número de Épocas en el Entrenamiento de la Red Neuronal	Número de épocas para entrenar la red neuronal. Puede entenderse como el número de veces que la red neuronal verá un mismo dato individual.

Tabla 4. Parámetros principales del entrenamiento del modelo *AI-Feynman* en esta investigación.

El resultado del entrenamiento es una lista de funciones junto a sus errores y complejidades, y un humano podrá seleccionar la que considere mejor según estas métricas o su propia intuición física.

4. Análisis de los Datos

Antes de comenzar con el entrenamiento de *AI-Feynman*, es preciso conocer los datos con los que se trabaja y cómo se relacionan.

4.1. Masa del Halo de Materia Oscura frente a Masa Estelar

4.1.1. Función de las Medianas de la Masa del Halo de Materia Oscura

La Fig. 4 muestra el mapa de densidad de M_{halo} frente a M_*. Cuanto más amarillo sea el color en una región, encontramos mayor densidad de galaxias con esa M_{halo}.

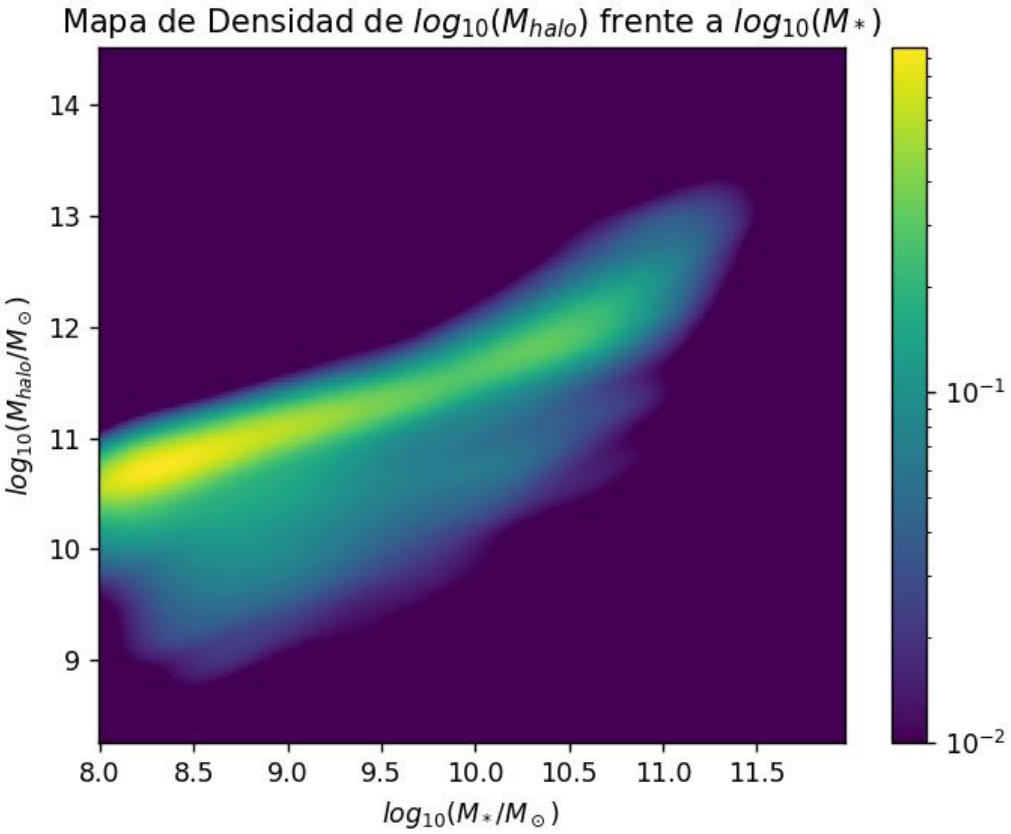

Fig. 4. Mapa de densidad de M_{halo} frente a M_*.

En la Fig. 4, gran parte de las galaxias tienen una M_{halo} que sigue una relación observable creciente, lo cual tiene sentido físico, pues si aumenta M_*, la galaxia sería más masiva y aumentaría la masa de su halo, M_{halo}. Por tanto, al agrupar los datos en intervalos de M_* y tomar las medianas[4] de M_{halo} en estos intervalos, se obtendrán los puntos que se pretenden ajustar a una expresión matemática.

El número de intervalos para agrupar los datos es muy importante en el entrenamiento de *AI-Feynman*, pues pocos intervalos resultarían en escasos datos para ajustar una curva y obtener una ley física, mientras que muchos intervalos resultarían en una dispersión con respecto a la tendencia que dificultaría este ajuste (ver Fig. 5).

[4] Es mejor emplear la mediana y no la media porque esta última es muy sensible a valores atípicos y daría un valor inferior a los que se encuentran en la franja amarilla de la Fig. 4.

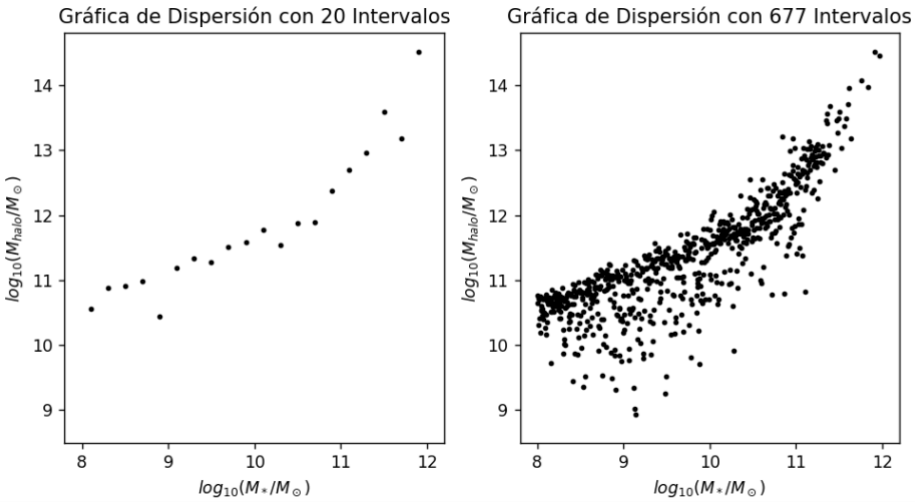

Fig. 5. A la izquierda, una gráfica con un número pequeño de intervalos resulta en una curva poco fiable. A la derecha, un gran número dificulta el ajuste de la curva y la obtención de una ley física debido a la dispersión intrínseca.

Tras entrenar el modelo con distintos números de intervalos, se observó que los mejores resultados se dan con 77 intervalos, pues ofrece un buen balance entre cantidad de puntos y dispersión de estos (ver Fig. 6).

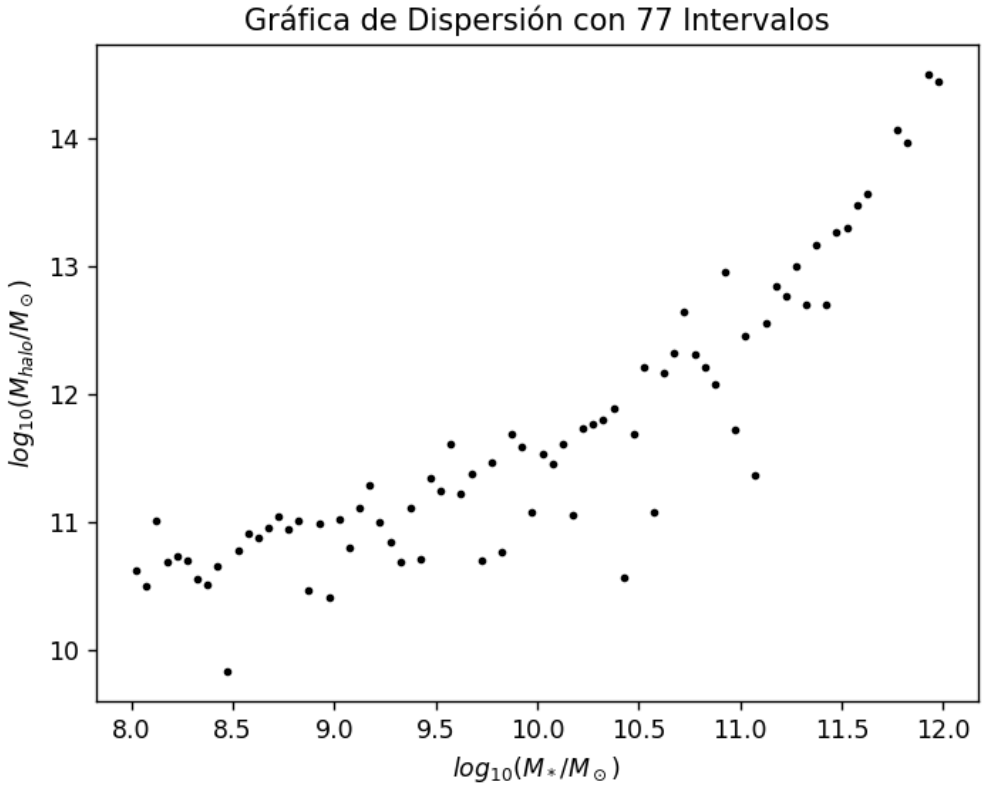

Fig. 6. Gráfica de dispersión de las medianas de M_{halo} en cada uno de los 77 intervalos de M_*.

4.1.2. Función de Densidad de Probabilidad

Para hacer la gráfica de la densidad de probabilidad de M_{halo}, se debe seleccionar un intervalo de M_* y representar las veces que aparece cada valor de M_{halo} en el intervalo frente a M_{halo}. Si se tomara cada valor individual de M_{halo} por separado, es probable que no se repitieran mucho, por lo que se agruparán en intervalos. Además, para que sea una función de densidad de probabilidad, habrá que normalizarla para que su área sea 1,

$$\int P(y) = 1 \tag{7}$$

La Fig. 7 muestra cuatro gráficas de densidad de probabilidad de M_{halo} centradas en distintos M_*. Se agruparon los datos de M_{halo} en 750 intervalos, pues era un valor que producía buenos resultados al entrenar a *AI-Feynman*.

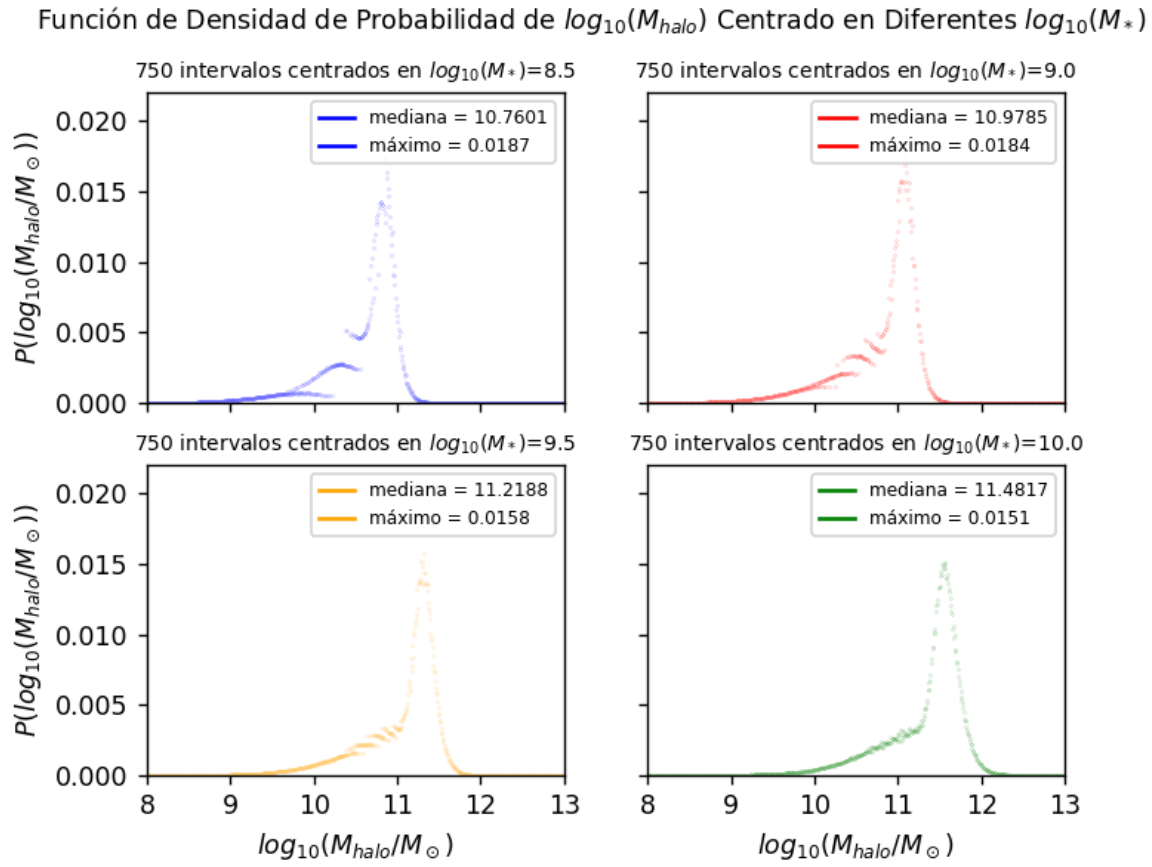

Fig. 7. Gráficas de densidades de probabilidad de M_{halo} centradas en distintos intervalos de M_*. Coinciden con la Fig.4, pues presentan un máximo de probabilidad (franja amarilla) en un punto y una pequeña probabilidad de encontrar valores menores solo en un lado (franja verde).

En la Fig. 7 se observa que la forma de la función de densidad de probabilidad es muy parecida en las cuatro gráficas a pesar de estar centradas en distintos intervalos de M_* (solo cambia la mediana, $\widetilde{M}_{halo}$, y el valor máximo, A). Esto permite hacer la suposición de que la función de densidad de probabilidad solo depende de estos dos parámetros, además de la propia M_{halo}, así que podremos ahorrar al modelo el parámetro $\vec{x} = \{M_*\}$ y usar solo $\widetilde{M}_{halo}$ y A.

$$P(M_{halo}) = p_{M_{halo}}(\widetilde{M}_{halo}, A) \tag{8}$$

Además, como es una función de densidad de probabilidad, puede predecirse que será exponencial.

4.2. Masa del Halo de Materia Oscura frente a Masa Estelar y Tasa de Formación Estelar

Para hacerse una idea de la dependencia de M_{halo} con M_* y SFR, puede ignorarse una variable independiente y ver cómo cambia la otra. En la Fig. 4 puede verse la relación entre M_{halo} y M_* y en la Fig. 8 entre M_{halo} y SFR.

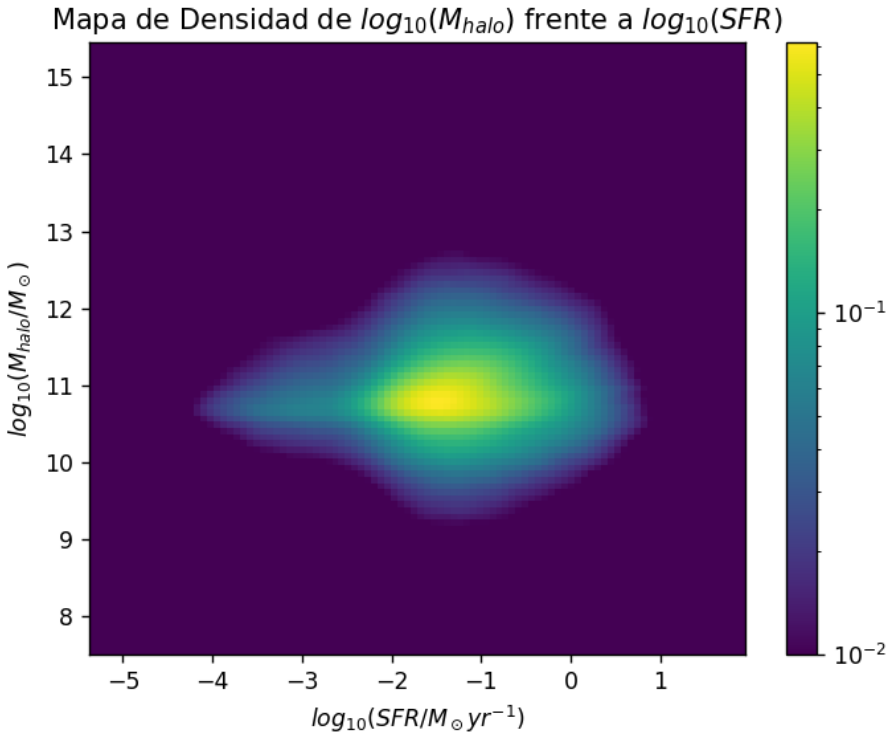

Fig. 8. Mapa de densidad de M_{halo} frente a SFR.

Esta gráfica muestra que la mayoría de las galaxias tienen un M_{halo} de $10^{11} M_{\odot}$, con las demás siguiendo una distribución circular levemente arqueada hacia arriba.

A continuación, en la Fig. 9, se muestran tres perspectivas de la gráfica de M_{halo} frente a M_* y SFR.

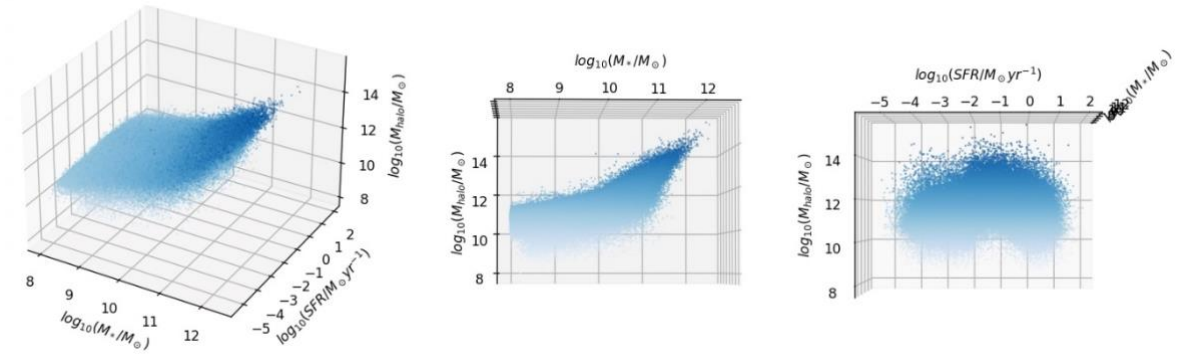

Fig. 9. Gráfica de la dispersión de M_{halo} frente a M_* y SFR desde tres perspectivas distintas.

En la Fig. 9 se han agrupado los puntos en 750x750 intervalos de M_* y SFR. Puede comprobarse que la gráfica desde la perspectiva del eje M_* (la segunda) coincide con la Fig. 4.

Por otra parte, se dejará fuera de esta investigación la función de densidad de probabilidad de M_{halo} con M_* y SFR, pues los resultados obtenidos con una sola variable no han sido buenos (ver sección 5.1.2).

5. Resultados

En esta sección se discuten los resultados obtenidos al entrenar el modelo de regresión *AI-Feynman*.

5.1. Masa del Halo de Materia Oscura frente a Masa Estelar

5.1.1. Función de las Medianas de la Masa del Halo de Materia Oscura

Para obtener la expresión matemática que relaciona $\widetilde{M}_{halo}$ con M_*, se entrenó el modelo con variable independiente $\vec{x} = \{M_*\}$ y variable dependiente $y = \widetilde{M}_{halo}$. Los datos se

agruparon en 77 intervalos (ver sección 4.1.1). El entrenamiento con mejores resultados tuvo los parámetros de la Tabla 5.

Parámetros	Valor
Tiempo Máximo en la Fase de Fuerza Bruta	$7200s$
Operaciones en la Fase de Fuerza Bruta	$+, \times, -, \div, +1, -1, -x, \frac{1}{x}, \sqrt{x}, \pi, ln(x), e^x$
Grado Máximo en el Ajuste Polinomial	4
Número de Épocas en el Entrenamiento de la Red Neuronal	4000

Tabla 5. Parámetros del entrenamiento para la obtención de la relación entre M_* y $\widetilde{M}_{halo}$.

Durante la fase de fuerza bruta se omitieron las funciones trigonométricas, ya que no tiene sentido físico que aparezcan en estas leyes, pues se tratan variables astrofísicas que no presentan relación aparente con la periodicidad.

Tras muchos entrenamientos, la expresión con mejor relación complejidad-error es la de la Tabla 6.

Expresión	Complejidad	Error
$\mathbf{\log_{10} \widetilde{M}_{halo} = 7,712912048928 + \sqrt{\log_{10} M_* + 0,00020084339 e^{\log_{10} M_*}}}$	9,47	26,88

Tabla 6. Expresión resultante del entrenamiento del modelo para la relación entre M_{halo} y $\widetilde{M}_{halo}$.

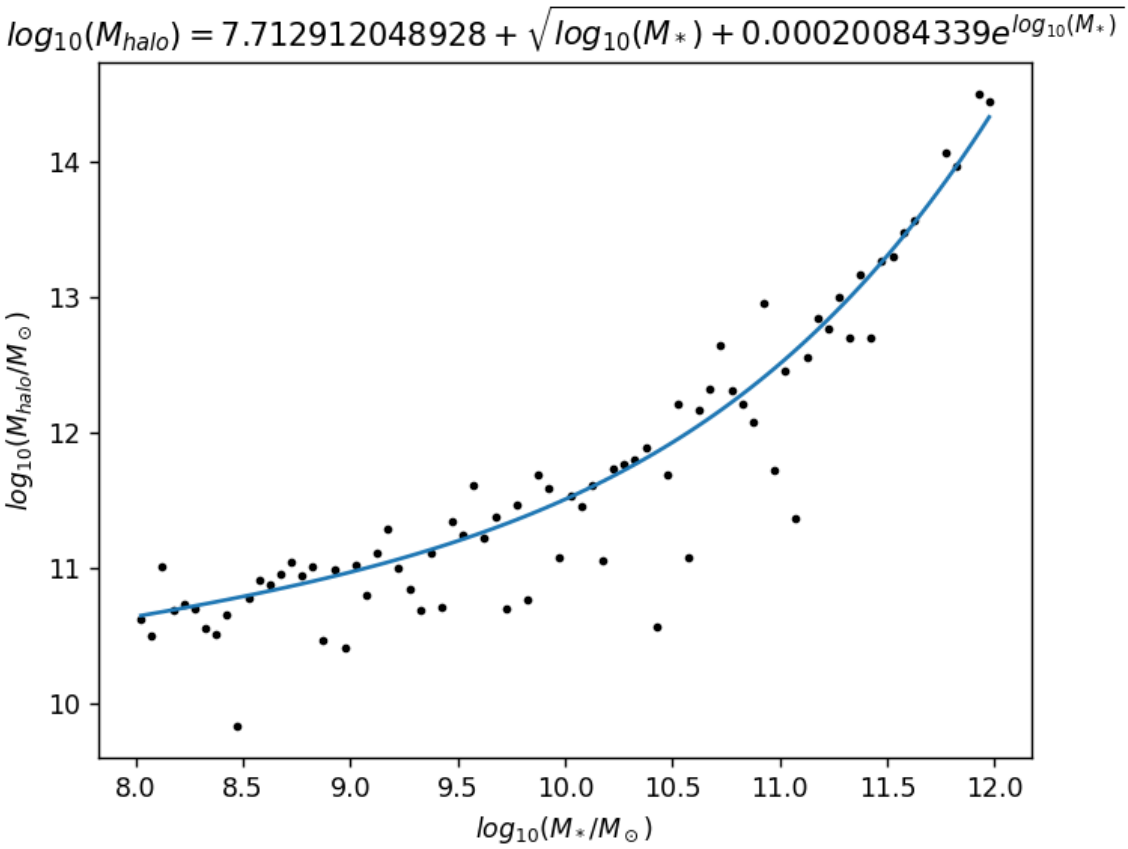

Fig. 10. Ley empírica que relaciona la mediana de M_{halo} con M_*.

La ecuación se ajusta bien a los datos representados (ver Fig. 10) y tiene sentido físico, pues se trata de una ley de la forma

$$y = y_{min} + g(x) \tag{9}$$

donde y_{min} es el valor mínimo de $\log_{10}(\widetilde{M}_{halo})$ y $g(x)$ es una expresión que depende únicamente de x y que es aproximadamente 0 cuando $\log_{10}(M_*)$ es mínimo. En este caso, $y_{min} = 7.712912048928$ y es un valor que se acerca a este mínimo en la Fig. 4.

No obstante, esta ley empírica podría mejorarse, pues *AI-Feynman* usa como operador e^x y no 10^x, y los datos de entrada son $\log_{10}(M_*)$. Esto lleva a que la expresión tenga un termino $e^{\log_{10}(M_*)}$ que podría simplificarse a $k \cdot M_*$, siendo k una constante si se hubiera tomado $\ln M_*$ en lugar de $\log_{10}(M_*)$.

Por tanto, la ley empírica simplificada es la siguiente.

$$\log_{10} \widetilde{M}_{halo} = 7{,}71 + \sqrt{\log_{10} M_* + \frac{{M_*}^{0{,}43}}{5000}} \tag{10}$$

5.1.2. Función de Densidad de Probabilidad

Para obtener la función de densidad de probabilidad de M_{halo} se entrenó a *AI-Feynman* con los datos de la Fig. 7. Las variables independientes fueron $\vec{x} = \{M_{halo}, \widetilde{M}_{halo}, A\}$ y la variable dependiente $y = P(M_{halo})$. El entrenamiento con mejores resultados tuvo los parámetros de la Tabla 7.

Parámetros	**Valor**
Tiempo Máximo en la Fase de Fuerza Bruta	$21600s$
Operaciones en la Fase de Fuerza Bruta	$+, \times, -, \div, +1, -1, -x, \frac{1}{x}, \sqrt{x}, \pi, ln(x), e^x$
Grado Máximo en el Ajuste Polinomial	3
Número de Épocas en el Entrenamiento de la Red Neuronal	4000

Tabla 7. Parámetros del entrenamiento para la obtención de la relación entre M_* y M_{halo}.

Nuevamente, se omitieron las funciones trigonométricas. En el parámetro del grado máximo del ajuste polinomial se empleó un valor pequeño, pues una función de densidad de probabilidad no debería tener polinomios de grados altos. Además, la complejidad de la expresión implica un tiempo elevado en la fase de fuerza bruta.

Esta función precisó de varias semanas de entrenamiento, pues los resultados no eran buenos. Se intentó desde fijar un solo valor de M_* para los intervalos de M_{halo} hasta entrenar la función por partes. La expresión que tuvo mejor relación complejidad-error y, además, tenía sentido matemático al ser una función de densidad de probabilidad, es la de la Tabla 8.

Expresión	Complejidad	Error
$P(\log_{10} M_{halo}) = Ae^{-\log_{10} M_{halo} \cdot e^{\frac{1}{\log_{10} M_{halo} - \log_{10} \tilde{M}_{halo}}}}$	22,65	30,35

Tabla 8. Expresión resultante del entrenamiento del modelo para la función de densidad de probabilidad de M_{halo} en la gráfica de M_{halo} frente a M_*.

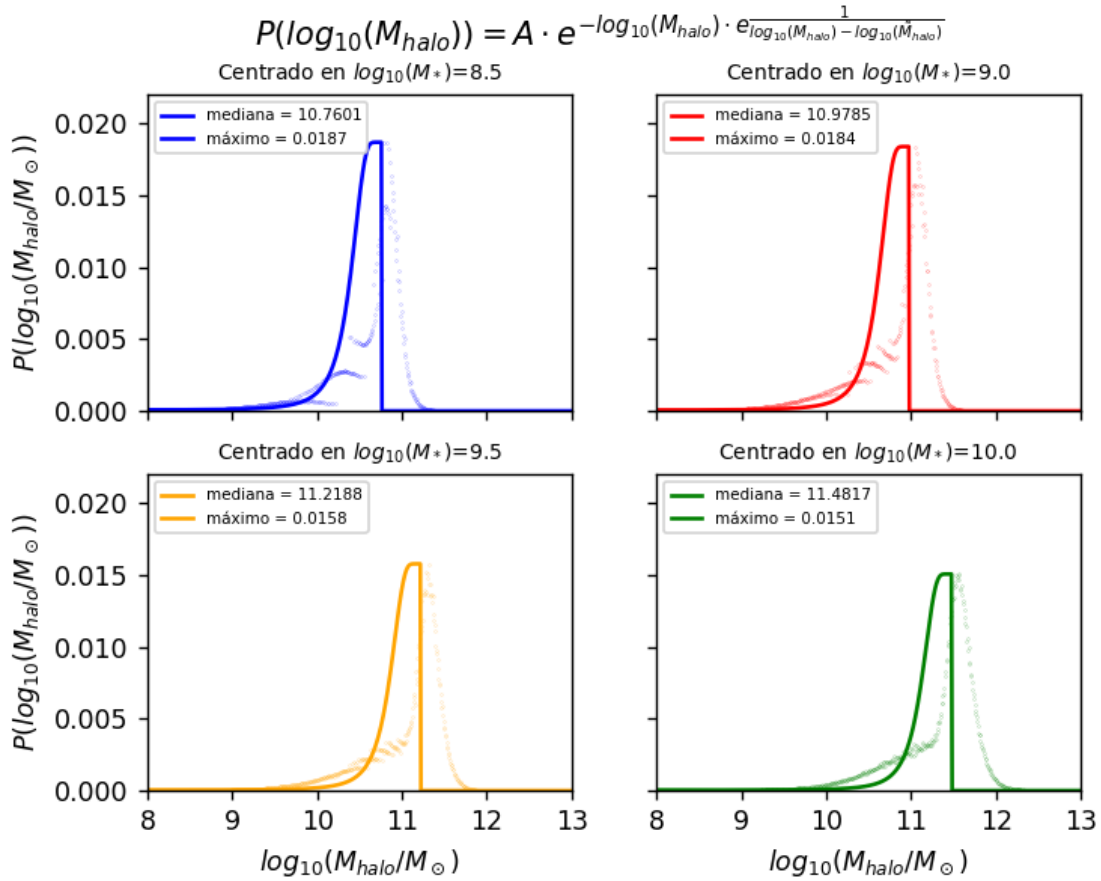

Fig. 11. Función de densidad de probabilidad de M_{halo} en la gráfica de M_{halo} frente a M_*.

La función obtenida no se ajusta demasiado bien a los datos (ver Fig. 11), aunque al menos es una exponencial con el máximo correcto. El principal problema es que hay una discontinuidad en la mediana, $\tilde{M}_{halo}$. Esto no es correcto, pues la función de densidad de probabilidad debería ser continua (ver Fig. 7). Este problema se debe a que la bajada es tan abrupta que no tiene suficientes puntos en comparación con la subida, por lo que el error matemático en la aproximación de la bajada a cero no es tan grande y el modelo acepta este resultado, si bien un humano podría identificar inmediatamente el error. En futuras investigaciones podría duplicarse el número de puntos en la bajada para que, aunque estén

repetidos, el error al ajustar mal esta sección fuera mayor y obligara a *AI-Feynman* a producir mejores resultados.

Otro problema es que la función no se ajusta correctamente a la subida porque existe un punto alrededor de $\log_{10} M_{halo} = 11{,}0$ (se aprecia mejor en la gráfica de la Fig. 11 centrada en $\log_{10} M_* = 10{,}0$) en todas las gráficas que presenta un cambio repentino de pendiente, haciendo que sea complicado para el modelo ajustarse a ambas partes. Una posible solución sería entrenar la función de densidad de probabilidad por partes, aunque esta solución no es nada elegante.

No obstante, el ajuste tiene sus puntos fuertes, pues *AI-Feynman* ha identificado que se trata de una función del tipo ecuación (11) (donde $g(y)$ es una función cualquiera e $\tilde{y}$ la mediana de la distribución), lo cual es muy similar a otras funciones de densidad de probabilidad como la gaussiana (ver ecuación (12) donde σ es la desviación típica y μ la media de la distribución).

$$P(y) = Ae^{-g(y-\tilde{y})} \tag{11}$$

$$G(y) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(y-\mu)^2}{2\sigma^2}} \tag{12}$$

5.2. Masa del Halo de Materia Oscura frente a Masa Estelar y Tasa de Formación Estelar

5.2.1. Función de las Medianas de la Masa del Halo de Materia Oscura

Para encontrar la expresión de $\widetilde{M}_{halo}$ en función de M_* y SFR, se entrenó a *AI-Feynman* con variables independientes $\vec{x} = \{M_*, SFR\}$ y variable dependiente $y = \widetilde{M}_{halo}$. El entrenamiento con mejores resultados tuvo los parámetros de la Tabla 9.

Parámetros	Valor
Tiempo Máximo en la Fase de Fuerza Bruta	$21600s$
Operaciones en la Fase de Fuerza Bruta	$+, \times, -, \div, +1, -1, -x, \frac{1}{x}, \sqrt{x}, \pi, ln(x), e^x$
Grado Máximo en el Ajuste Polinomial	5
Número de Épocas en el Entrenamiento de la Red Neuronal	4000

Tabla 9. Parámetros del entrenamiento del modelo para la obtención de la relación de $\widetilde{M}_{halo}$ con M_* y SFR.

Como en casos anteriores, se omitieron las funciones trigonométricas y se empleó un tiempo elevado para la fase de fuerza bruta.

Las dos mejores expresiones son las de la Tabla 10.

Expresión	Complejidad	Error
$\mathbf{\log_{10}(\widetilde{M}_{halo}) = 12,651821110273 + \sqrt{\log_{10}(SFR) + e^{\sqrt{e^{\sqrt{\log_{10}(M_*)-1}}}}} - \log_{10}(M_*)}$	77,82	26,92
$\mathbf{\log_{10}(\widetilde{M}_{halo}) = 8,134471514136 + e^{\frac{\sqrt{\log_{10}(SFR)+\sqrt{e^{\log_{10}(M_*)}}}}{\log_{10}(M_*)-1}}}$	75,18	26,96

Tabla 10. Expresiones resultantes del entrenamiento del modelo para la relación de $\widetilde{M}_{halo}$ con M_* y SFR.

El modelo calificaba la primera expresión como mejor, pues consideraba que la disminución en el error con respecto a la segunda era lo suficientemente buena como para compensar el aumento de complejidad. Sin embargo, como humanos, podemos ver que la segunda expresión es más simple y no merece la pena sustituirla por una diferencia de error de $0{,}04$. Además, *AI-Feynman* simplemente se guía por estadística, pero nosotros tenemos una ligera idea de la forma general de la expresión final, y la segunda parece mejor, pues la constante $8{,}134471514136$ se aproxima por abajo al valor mínimo de $\log_{10} M_{halo}$ (lo cual es mejor, pues la tendencia que sigue la gráfica es creciente) a diferencia de la primera expresión, en la que se aproxima por arriba.

Por tanto, la mejor expresión, simplificada, es la ecuación (13).

$$\log_{10}(\widetilde{M}_{halo}) = 8{,}13 + e^{\frac{\sqrt{\log_{10}(SFR)+{M_*}^{0{,}22}}}{\log_{10}(M_*)-1}} \tag{13}$$

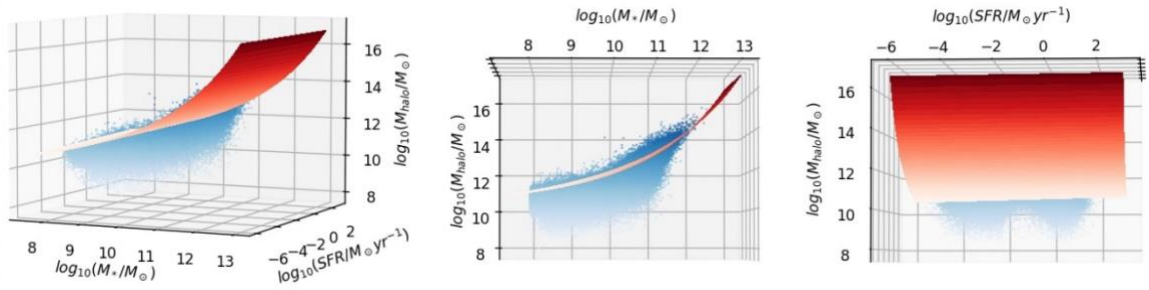

Fig. 12. Ley empírica para obtener la mediana de M_{halo} en función de M_* y SFR desde tres perspectivas distintas.

La función logra ajustarse a la tendencia creciente de $\widetilde{M}_{halo}$ con M_* y, de hecho, observándolo desde la segunda perspectiva de la Fig. 12, la función se parece mucho a la obtenida en la sección 5.1.1. No obstante, la función no se ajusta bien a la relación entre $\widetilde{M}_{halo}$ y SFR, pues no presenta el arco que puede verse en la Fig. 8. Como en la sección 5.1.2, debajo del arco hay pocos puntos en comparación con los que hay arriba, por lo que una función similar a un plano no tendría un error muy significativo matemáticamente, aunque un humano sí lo notaría. Además, desde un punto de vista físico, es extraño que la función sea exponencial, pues no suelen ajustarse bien a curvas como el arco de la tercera perspectiva.

Por otra parte, como en la sección 5.1.1, la ley empírica presenta un término $e^{\log_{10}(M_*)}$ que podría simplificarse a $k \cdot M_*$ si se hubiera tomado $\ln M_*$ en lugar de $\log_{10}(M_*)$.

5.2.2. Función de Densidad de Probabilidad

Debido a las dificultades que tuvo *AI-Feynman* con la función de densidad de probabilidad con una sola variable independiente, M_*, es razonable pensar que los resultados al añadir una variable más, SFR, no serían aceptables. Por tanto, se presenta esta parte de la

investigación como una línea de investigación futura cuando los modelos de regresión simbólica hayan mejorado.

6. Conclusión

Esta investigación ha mostrado el estado actual de la regresión simbólica en la física mediante el entrenamiento del modelo *AI-Feynman* para predecir la relación entre tres variables astrofísicas características de las galaxias.

En cuanto a la astrofísica, los resultados indican una clara relación creciente entre M_{halo}, M_* y SFR. Esto es acertado en el caso de la relación entre M_{halo} y M_*, donde se cumplen las observaciones iniciales, pero no en el caso de M_{halo} y SFR, pues en la Fig. 8 se pudo ver que esta relación no era del todo creciente, sino que primero crecía y luego decrecía.

En cuanto al rendimiento del modelo de regresión simbólica *AI-Feynman*, los resultados obtenidos sobre la relación entre M_{halo} y M_* son buenos y podrían usarse en futuras investigaciones, pues la expresión no es demasiado compleja, se ajusta bien a la gráfica y tiene sentido físico. Sin embargo, la función de densidad de probabilidad no se ajusta bien a la gráfica y presenta una discontinuidad, lo cual constituye un error, pues debería ser prácticamente continua. No obstante, a pesar de que la función no se ajusta bien a los datos, esta tiene sentido matemático, pues se trata de una función exponencial que contiene el término $y - \tilde{y}$ (donde y es la variable y $\tilde{y}$ la mediana), que es bastante común en las funciones de densidad de probabilidad. Finalmente, la función que relaciona M_{halo}, M_* y SFR no presenta ningún error tan grave como el anterior y, de hecho, se ajusta bien al parámetro M_*. Sin embargo, falla al tratar de relacionar M_{halo} y SFR, pues muestra una relación creciente entre estas dos variables cuando realmente no lo es. Además, la función obtenida resulta demasiado compleja.

En estos tres casos, el mayor problema que ha tenido *AI-Feynman* ha sido tratar con puntos que venían previamente de una gráfica de dispersión donde hay varios valores de la variable dependiente para cada valor de la variable independiente. En la astrofísica, esto es muy común, pues esta dispersión es inevitable debido a los procesos físicos que contribuyen a la

formación de galaxias a lo largo del tiempo. No obstante, *AI-Feynman* fue diseñada para obtener funciones sin estas dispersiones tan grandes. Por ejemplo, el modelo sería capaz de hallar la ecuación del movimiento rectilíneo uniformemente acelerado con suficientes puntos medidos experimentalmente, a pesar de que estas medidas presentan un pequeño error aleatorio, pero cuando esta dispersión pasa de ser un error a una propiedad intrínseca del objeto de estudio (las galaxias), la variación es más grande y hace que el modelo no pueda producir resultados tan buenos. Por ello, se ha recurrido a la compresión de información (se tomaron las medianas y logaritmos de las variables en lugar de los datos brutos) para obtener resultados coherentes, lo cual hace que se pierda información en el proceso.

Como futuras líneas de trabajo, podrían investigarse alternativas a *AI-Feynman* capaces de trabajar con variables con dispersiones intrínsecas o incluso desarrollar una mejora al algoritmo de *AI-Feynman* capaz de tratar estos casos siguiendo una metodología parecida a la usada en esta investigación (hallar las funciones de las medianas y de densidad de probabilidad por separado), pues es un campo poco investigado en la actualidad.

7. Bibliografía

[1] Udrescu, S. M., & Tegmark, M. (2020). AI Feynman: A physics-inspired method for symbolic regression. *Science Advances*, 6(16). https://doi.org/10.1126/SCIADV.AAY2631

[2] Kusakabe, H., Shimasaku, K., Ouchi, M., Nakajima, K., Goto, R., Hashimoto, T., Konno, A., Harikane, Y., Silverman, J. D., & Capak, P. L. (2018). The stellar mass, star formation rate and dark matter halo properties of LAEs at z ∼ 2. *Publications of the Astronomical Society of Japan*, 70(1), 4–5. https://doi.org/10.1093/PASJ/PSX148

[3] Aung, H., Nagai, D., Klypin, A., Behroozi, P., Abdullah, M. H., Ishiyama, T., Prada, F., Pérez, E., López Cacheiro, J., & Ruedas, J. (2022). The Uchuu-universe machine data set: galaxies in and around clusters. *Monthly Notices of the Royal Astronomical Society*, 519(2), 1648–1656. https://doi.org/10.1093/MNRAS/STAC3514

[4] Hogg, D. W. (2000). *Distance measures in cosmology*. https://doi.org/10.48550/arXiv.astro-ph/9905116

[5] Universitam. (2017, September). *Todas las galaxias se ubican dentro de un halo de materia oscura de libre interacción que conforma el andamiaje de la gravedad – UNIVERSITAM. Universy*. https://universitam.com/academicos/noticias/todas-las-galaxias-se-ubican-dentro-de-un-halo-de-materia-oscura-de-libre-interaccion-que-conforma-el-andamiaje-de-la-gravedad/

[6] Lotus QA. (2020, March). *Black-box Test Design Techniques - Lotus QA - Leading IT Outsourcing Company In Vietnam*. Lotus QA. https://www.lotus-qa.com/black-box-test-design-techniques/

[7] Ruggiero, R. (2020, November). *Symbolic Regression: The Forgotten Machine Learning Method | by Rafael Ruggiero | Towards Data Science*. Towards Data Science. https://towardsdatascience.com/symbolic-regression-the-forgotten-machine-learning-method-ac50365a7d95

[8] Eureqa. (2012). *Eureqa - Creative Machines Lab - Columbia University*. https://www.creativemachineslab.com/eureqa.html

[9] Behroozi, P., Wechsler, R. H., Hearin, A. P., & Conroy, C. (2019). UniverseMachine: The correlation between galaxy growth and dark matter halo assembly from z = 0−10. *Monthly Notices of the Royal Astronomical Society*, 488(3), 3143–3194. https://doi.org/10.1093/MNRAS/STZ1182

[10] Ishiyama, T., Prada, F., Klypin, A. A., Sinha, M., Metcalf, R. B., Jullo, E., Altieri, B., Cora, S. A., Croton, D., De La Torre, S., Millán-Calero, D. E., Oogi, T., Ruedas, J., & Vega-Martínez, C. A. (2021). The Uchuu simulations: Data Release 1 and dark matter halo concentrations. *Monthly Notices of the Royal Astronomical Society*, 506(3), 4210–4231. https://doi.org/10.1093/MNRAS/STAB1755

[11] Aghanim, N., Akrami, Y., Ashdown, M., Aumont, J., Baccigalupi, C., Ballardini, M., Banday, A. J., Barreiro, R. B., Bartolo, N., Basak, S., Battye, R., Benabed, K., Bernard, J. P., Bersanelli, M., Bielewicz, P., Bock, J. J., Bond, J. R., Borrill, J., Bouchet, F. R., … Zonca, A. (2020). Planck 2018 results - VI. Cosmological parameters. *Astronomy & Astrophysics*, 641, A6. https://doi.org/10.1051/0004-6361/201833910

[12] Martius, G., & Lampert, C. H. (2016). Extrapolation and learning equations. *29th Conference on Neural Information Processing Systems (NIPS 2016)*. http://phys.csail.mit.edu/papers/12.pdf

[13] *PySR: High-Performance Symbolic Regression in Python*. (2022). PySR. Retrieved March 19, 2023, from https://astroautomata.com/PySR/

[14] Udrescu, S. M. (2020). *SJ001/AI Feynman*. Github; Neural information processing systems foundation. https://github.com/SJ001/AI-Feynman

Anexos

A. Código

En este anexo se muestra el código desarrollado durante esta investigación, tanto para las gráficas como para el entrenamiento del modelo *AI-Feynman*.

Imports

```
import sys
import numpy as np
import math
import matplotlib
import matplotlib.pyplot as plt
from tqdm.notebook import trange, tqdm
import aifeynman
import scipy
from scipy.stats import kde
import h5py
```

Clases

```
class Galaxy:
    def __init__(self, M, SM, SFR, log=False):
        if not log:
            self.M = M
            self.SM = SM
            self.SFR = SFR
        else:
            self.M = math.log10(M)
            self.SM = math.log10(SM)
            self.SFR = math.log10(SFR+0.00000000001)

    def __repr__(self):
        return f"<Galaxy M:{self.M} SM:{self.SM} SFR:{self.SFR}>"

class Redshift:
    def __init__(self, path, Ngalaxies=-1, load=True, log=False):
        self.h = h5py.File(path, 'r')

        self.Box = int(self.h.attrs["Box"])
        if Ngalaxies >= 0:
            self.Ngalaxies = min(int(self.h.attrs["Ngalaxies"]), Ngalaxies
)
        else:
            self.Ngalaxies = int(self.h.attrs["Ngalaxies"])
        self.Redshift = self.h.attrs["Redshift"]

        self.galaxies = []

        if load:
```

```
            self.load_galaxies(log)

    def load_galaxies(self, log=False):
        M = np.real(self.h.get('Mvir'))
        SM = np.real(self.h.get('StellarMass'))
        SFR = np.real(self.h.get('StarFormationRate'))

        self.galaxies = []

        for i in trange(self.Ngalaxies):
            if SFR[i] == 0: continue
            g = Galaxy(M[i], SM[i], SFR[i], log)
            self.galaxies.append(g)

        self.Ngalaxies = len(self.galaxies)

    def __repr__(self):
        return f"<Redshift Box:{self.Box} Ngalaxies:{self.Ngalaxies} Redsh
ift:{self.Redshift}>"
```

Procesamiento de Datos

Rutas

```
ROOT = sys.prefix[:-4]
DATA_DIR = ROOT + 'data/Uchuu-UM/'

FILES = [
            'Sample.01.Redshift.0.000.h5',
            'Sample.01.Redshift.1.032.h5',
            'Sample.01.Redshift.2.029.h5',
            'Sample.01.Redshift.3.129.h5',
            'Sample.01.Redshift.4.269.h5',
            'Sample.01.Redshift.5.155.h5'
        ]

PATHS = [DATA_DIR + f for f in FILES]
```

Constantes

```
# Constantes de selección de datos
Z = 0
N_GALAXIES = -1

# Constantes de M_halo frente a M_*
STEP = 0.05

# Constantes de la distribución de probabilidad
DISTRIBUTION_CENTERS = [8.5, 9.0, 9.5, 10.0]
DISTRIBUTION_WIDTH = 0.05
DISTRIBUTION_BINS = 750
DISTRIBUTION_COLORS = ['blue', 'red', 'orange', 'green']
```

```
# Constantes de M_halo frente a M_* y SFR
SM_BINS = 750
SFR_BINS = 750
```

Cargar Redshift

```
redshift = Redshift(PATHS[Z], Ngalaxies=N_GALAXIES, log=True)

SM = [g.SM for g in redshift.galaxies]
SFR = [g.SFR for g in redshift.galaxies]
M = [g.M for g in redshift.galaxies]
```

Masa del Halo de Materia Oscura frente a Masa Estelar

Función de las Medianas de la Masa del Halo de Materia Oscura

Mapa de Densidad

```
%matplotlib notebook

X = np.array(SM)
y = np.array(M)

nbins=300
k = kde.gaussian_kde([X,y])
xi, yi = np.mgrid[X.min():X.max():nbins*1j, y.min():y.max():nbins*1j]
zi = k(np.vstack([xi.flatten(), yi.flatten()]))

plt.title("Mapa de Densidad de $log_{10}(M_{halo})$ frente a $log_{10}(M_*
)$")
plt.xlabel(r"$log_{10}(M_*/M_\odot)$")
plt.ylabel(r"$log_{10}(M_{halo}/M_\odot)$")

plt.pcolormesh(xi, yi, zi.reshape(xi.shape), shading='auto', norm=matplotl
ib.colors.LogNorm(vmin=10**(-2), vmax=zi.max()))
plt.colorbar()
plt.show()
```

Agrupación en Intervalos

```
def median_galaxy(gs):
    SM = STEP*(gs[0].SM//STEP)+STEP/2

    M = 0
    if len(gs) % 2 == 0:
        M = (gs[len(gs)//2-1].M+gs[len(gs)//2].M)/2
    else:
        M = gs[len(gs)//2].M

    return Galaxy(M, SM, 0)

redshift.galaxies.sort(key=lambda x: x.SM)

r = Redshift(PATHS[0], False)
```

```
curr = 0
tmp = []
for g in redshift.galaxies:
    region = g.SM//STEP

    if curr == region:
        tmp.append(g)
    else:
        if len(tmp):
            r.galaxies.append(median_galaxy(tmp))
        curr = region
        tmp = [g]

if len(tmp):
    r.galaxies.append(median_galaxy(tmp))

r.Ngalaxies = len(r.galaxies)

SM = [g.SM for g in r.galaxies]
M = [g.M for g in r.galaxies]
```

Gráficas con Intervalos

```
%matplotlib notebook

fig, axs = plt.subplots(1, 2)

STEPS = [0.2, 0.005]

for i in range(2):
    STEP = STEPS[i]
    r = Redshift(PATHS[0], False)

    curr = 0
    tmp = []
    for g in redshift.galaxies:
        region = g.SM//STEP

        if curr == region:
            tmp.append(g)
        else:
            if len(tmp):
                r.galaxies.append(median_galaxy(tmp))
            curr = region
            tmp = [g]

    if len(tmp):
        r.galaxies.append(median_galaxy(tmp))

    r.Ngalaxies = len(r.galaxies)

    SM = [g.SM for g in r.galaxies]
    M = [g.M for g in r.galaxies]
```

```
    axs[i].set_title(f"Gráfica de Dispersión con {len(SM)} Intervalos")
    axs[i].set(xlabel=r"$log_{10}(M_*/M_\odot)$", ylabel=r"$log_{10}(M_{ha
lo}/M_\odot)$")

    axs[i].set_xlim([7.8, 12.2])
    axs[i].set_ylim([8.5, 14.8])

    axs[i].scatter(SM, M, s=5, c ='black')
```

```
%matplotlib notebook

plt.title(f"Gráfica de Dispersión con {len(SM)} Intervalos")
plt.xlabel(r"$log_{10}(M_*/M_\odot)$")
plt.ylabel(r"$log_{10}(M_{halo}/M_\odot)$")

plt.scatter(SM, M, s=5, c ='black')
```

AI-Feynman

```
with open(f"data/redshift{Z}.txt", "w") as f:
    for g in r.galaxies:
        f.write(f"{g.SM} {g.M}\n")
```

```
aifeynman.run_aifeynman("./data/", f"redshift{0}.txt", 7200, ROOT+"aifeynm
an/ops.txt", polyfit_deg=4, NN_epochs=4000)
```

```
%matplotlib notebook

ax = plt.subplot()

x = np.linspace(min(SM), max(SM), 100)
y = 7.712912048928+np.sqrt((x+0.00020084339*np.exp(x)))

ax.scatter(SM, M, s=5, c ='black')
ax.set_title("$log_{10}(M_{halo}) = 7.712912048928+\sqrt{log_{10}(M_*)+0.0
0020084339e^{log_{10}(M_*)}}$")
ax.plot(x, y)
ax.set(xlabel='$log_{10}(M_*/M_\odot)$', ylabel='$log_{10}(M_{halo}/M_\odo
t)$')
```

Función de Densidad de Probabilidad

Utilidades

```
def normalized_distribution(var, bins):
    distribution_counts, distribution_bins = np.histogram(var, bins, densi
ty=True)
    bin_width = distribution_bins[1]-distribution_bins[0]

    X = np.array([])
    y = distribution_counts*bin_width
    for j in range(len(distribution_counts)):
            X = np.append(X, (distribution_bins[j+1]+distribution_bins[j])
/2)
```

```
    return X, y, bin_width
```

Gráficas con Intervalos

```
%matplotlib notebook

fig, axs = plt.subplots(2, 2)
fig.suptitle("Función de Densidad de Probabilidad de $log_{10}(M_{halo})$
Centrado en Diferentes $log_{10}(M_*)$", size=10)

redshift.galaxies.sort(key=lambda x: x.SM)

for i, ax in enumerate(axs.flat):
    ax.set_title(f'{DISTRIBUTION_BINS} intervalos centrados en $log_{{{10}
}}(M_*)$={DISTRIBUTION_CENTERS[i]}', fontsize=8)
    ax.set_xlim(8, 13)
    ax.set_ylim(0, 0.022)
    ax.set(xlabel='$log_{10}(M_{halo}/M_\odot)$', ylabel='$P(log_{10}(M_{h
alo}/M_\odot))$')

    var = []
    for g in redshift.galaxies:
        if g.SM >= DISTRIBUTION_CENTERS[i]-DISTRIBUTION_WIDTH and g.SM <=
DISTRIBUTION_CENTERS[i]+DISTRIBUTION_WIDTH:
            var.append(g.M)

    X, y, bin_width = normalized_distribution(var, DISTRIBUTION_BINS)

    ax.scatter(X, y, bin_width, color=DISTRIBUTION_COLORS[i])

    median = np.median(var)
    mx = np.max(y)
    ax.plot([], [], color=DISTRIBUTION_COLORS[i], label=f"mediana = {media
n:.4f}")
    ax.plot([], [], color=DISTRIBUTION_COLORS[i], label=f"máximo = {mx:.4f
}")
    ax.legend(loc="upper right", fontsize=7)

for ax in axs.flat:
    ax.label_outer()

plt.show()
```

AI-Feynman

```
K = 100000

for i in range(len(DISTRIBUTION_CENTERS)):
    var = []
    for g in redshift.galaxies:
        if g.SM >= DISTRIBUTION_CENTERS[i]-DISTRIBUTION_WIDTH and g.SM <=
DISTRIBUTION_CENTERS[i]+DISTRIBUTION_WIDTH:
            var.append(g.M)
```

```
    X, y, bin_width = normalized_distribution(var, DISTRIBUTION_BINS)
    y *= K
    median = np.median(var)
    mx = np.max(y)

    mode = "w" if i == 0 else "a"
    with open(f"data/redshift{Z}.txt", mode) as f:
        for i in range(len(X)):
            f.write(f"{X[i]} {median} {mx} {y[i]}\n")

aifeynman.run_aifeynman("./data/", f"redshift{Z}.txt", 21600, ROOT+"aifeyn
man/ops.txt", polyfit_deg=3, NN_epochs=4000)

%matplotlib notebook

fig, axs = plt.subplots(2, 2)
fig.suptitle("$P(log_{10}(M_{halo}))=A\cdot e^{-log_{10}(M_{halo})\cdot e^
{\dfrac{1}{log_{10}(M_{halo})-log_{10}(\\tilde{M}_{halo})}}}$")

redshift.galaxies.sort(key=lambda x: x.SM)

for i, ax in enumerate(axs.flat):
    ax.set_title(f'Centrado en $log_{{{10}}}(M_*)$={DISTRIBUTION_CENTERS[i
]}', fontsize=8)
    ax.set_xlim(8, 13)
    ax.set_ylim(0, 0.022)
    ax.set(xlabel='$log_{10}(M_{halo}/M_\odot)$', ylabel='$P(log_{10}(M_{h
alo}/M_\odot))$')

    var = []
    for g in redshift.galaxies:
        if g.SM >= DISTRIBUTION_CENTERS[i]-DISTRIBUTION_WIDTH and g.SM <=
DISTRIBUTION_CENTERS[i]+DISTRIBUTION_WIDTH:
            var.append(g.M)

    X, y, bin_width = normalized_distribution(var, DISTRIBUTION_BINS)

    ax.scatter(X, y, bin_width, color=DISTRIBUTION_COLORS[i])

    median = np.median(var)
    mx = np.max(y)

    y_pred = (mx/np.exp((X*np.exp(((X-median))**(-1)))))
    ax.plot(X, y_pred, color=DISTRIBUTION_COLORS[i])

    ax.plot([], [], color=DISTRIBUTION_COLORS[i], label=f"mediana = {media
n:.4f}")
    ax.plot([], [], color=DISTRIBUTION_COLORS[i], label=f"máximo = {mx:.4f
}")
    ax.legend(loc="upper left", fontsize=6)

for ax in axs.flat:
    ax.label_outer()
```

```
plt.show()
```

Masa del Halo de Materia Oscura frente a Masa Estelar y Tasa de Formación Estelar

Masa del Halo de Materia Oscura frente a Tasa de Formación Estelar

```
%matplotlib notebook

X = np.array(SFR)
y = np.array(M)

nbins=100
k = kde.gaussian_kde([X,y])
xi, yi = np.mgrid[X.min():X.max():nbins*1j, y.min():y.max():nbins*1j]
zi = k(np.vstack([xi.flatten(), yi.flatten()]))

plt.title("Mapa de Densidad de $log_{10}(M_{halo})$ frente a $log_{10}(SFR
)$")
plt.xlabel(r"$log_{10}(SFR/M_\odot yr^{-1})$")
plt.ylabel(r"$log_{10}(M_{halo}/M_\odot)$")

plt.pcolormesh(xi, yi, zi.reshape(xi.shape), shading='auto', norm=matplotl
ib.colors.LogNorm(vmin=10**(-2), vmax=zi.max()))
plt.colorbar()
plt.show()
```

Agrupación en Intervalos

```
median, x_edge, y_edge, _ = scipy.stats.binned_statistic_2d(SM, SFR, M, st
atistic='max', bins=[SM_BINS, SFR_BINS])
median = np.nan_to_num(median)

X = np.array([])
Y = np.array([])
z = np.array([])
for i in range(SM_BINS):
    for j in range(SFR_BINS):
        if median[i][j] == 0: continue
        X = np.append(X, (x_edge[i]+x_edge[i+1])/2)
        Y = np.append(Y, (y_edge[j]+y_edge[j+1])/2)
        z = np.append(z, median[i][j])
```

Gráficas con Intervalos

```
%matplotlib notebook

ax = plt.axes(projection='3d')
ax.set(xlabel='$log_{10}(M_*/M_\odot)$', ylabel='$log_{10}(SFR/M_\odot yr^
{-1})$', zlabel='$log_{10}(M_{halo}/M_\odot)$')
ax.scatter3D(X, Y, z, c=z, cmap='Blues', s=0.1);
plt.show()
```

AI-Feynman

```
with open(f"data/redshift{Z}.txt", 'w') as f:
        for i in range(len(X)):
            f.write(f"{X[i]} {Y[i]} {z[i]}\n")

aifeynman.run_aifeynman("./data/", f"redshift{Z}.txt", 21600, ROOT+"aifeyn
man/ops.txt", polyfit_deg=5)

%matplotlib notebook

ax = plt.axes(projection='3d')
ax.set(xlabel='$log_{10}(M_*/M_\odot)$', ylabel='$log_{10}(SFR/M_\odot yr^
{-1})$', zlabel='$log_{10}(M_{halo}/M_\odot)$')
ax.scatter3D(X, Y, z, c=z, cmap='Blues', s=0.1);

def f(x, y):
    return 8.134471514136+np.exp(np.sqrt(y+np.sqrt(np.exp(x)))/(x-1))

x = np.linspace(7.5, 13, 100)
y = np.linspace(-6, 3, 100)

X_t, Y_t = np.meshgrid(x, y)
Z_t = f(X_t, Y_t)

ax.plot_surface(X_t, Y_t, Z_t, cmap='Reds', antialiased=False)
plt.show()
```