Interior Structure Modeling of Sub-Neptune Exoplanets: From TOI-270d to Population-Level Inference

Thesis submitted in partial fulfillment of the requirement for

Honors in Physics

Biruk Nardos Abebe

Adviser, Leslie Hebb

April 8, 2025

Contents

Acknowledgments

First and foremost, I thank God for the strength and clarity to complete this work. I am deeply grateful to my adviser at Hobart and William Smith Colleges (HWS), Dr. Leslie Hebb, for her constant encouragement, thoughtful feedback, and generous support throughout this project. I also sincerely thank Dr. Matthew C. Nixon, my mentor at the University of Maryland, for introducing me to interior structure modeling and for guiding me through every step of this research.

This work was supported in part by the HWS Summer Research Program, RECONS Fellow Grant, and the GRAD-MAP (Graduate Resources Advancing Diversity with Maryland Astronomy and Physics) program at the University of Maryland, where I began developing this project.

Finally, I acknowledge the use of the free version of Grammarly, a writing assistance tool which provided real-time grammar, spelling, punctuation, word-choice, sentence structure, and style suggestions throughout the text.

1 Introduction

1.1 Discovery of Exoplanets

1.1.1 Historical Perspective

The study of exoplanets, planets orbiting stars beyond our solar system, was once confined in the realm of speculation. For centuries, astronomers debated whether planetary systems like our own were commonplace or exceptional, but lacked the observational tools to resolve the question. That changed in 1995, when Mayor and Queloz (1995) announced the discovery of 51 Pegasi b, a Jupiter-mass planet orbiting a sun-like star just 50 light-years away. Detected via the radial velocity method, this hot Jupiter changed previous assumptions about extrasolar planets by showing that planetary systems could differ dramatically from our own.

In the decades since, exoplanetary science has evolved from single-point discovery to thousands of exoplanets. At the time of this thesis, more than 5,000 exoplanets that have been discovered, with potentially thousands of other "candidates" in line (NASA Science, 2017). Perhaps the most surprising finding has been the ubiquity of planets between the size of Earth and Neptune, so-called "Sub-Neptunes", which are now known to be the most common type of planets in the galaxy, despite having no solar system analogues (Fulton et al., 2017).

This observational revolution has prompted a shift in how interior structure models are developed. Early interpretations of exoplanets' interior structure often relied on simple two-layer models or empirical mass-radius relationships (Seager and et al., 2007; Van Eylen et al., 2021). However, as measurements become more precise and atmospheric characterization entered the picture, it became clear that more sophisticated models were needed, ones that account for realistic thermal gradients, phase transitions, and compositional degeneracies (i.e., cases where different internal structures can yield the same mass and radius).

Today, with the launch of the James Webb Space Telescope (JWST), we can detect different compositions that make up a specific exoplanet. From these results, atmospheric mean molecular weight, in particular, can be a key link between spectral retrievals and bulk composition inference. Spectral retrieval is the process of fitting atmospheric models to observed spectra; the distribution of light intensity across wavelengths infers the chemical and physical properties of the atmosphere (Carroll and Ostlie, 2007).

This provides a bridge between observational and interior structure modeling efforts. The framework used in this thesis represents this new era of interior modeling.

1.1.2 Observational Techniques

The interpretation of an exoplanet's internal structure begins with a small set of measurable physical properties: mass (M_p), radius (R_p), equilibrium temperature ($T_{\rm eq}$), and, when available, atmospheric composition via spectroscopy. These observables are not direct products of a single method, but rather arise from the combination of several detection and characterization techniques. In this section, we focus on the methods most relevant to this work—those that yield the parameters used as inputs to our interior structure models.

Radial Velocity (RV): The radial velocity method detects the periodic Doppler shifts in a star's spectral lines induced by the gravitational pull of an orbiting planet. As the star moves toward and away from us, the resulting redshift and blueshift allow measurement of the stellar velocity semi-amplitude K, which can be related to the planet's minimum mass ($M_p \sin i$), the orbital period (P), and eccentricity (e) through Keplerian dynamics (Mayor and Queloz, 1995; Wright and Gaudi, 2013). Though RV only provides a lower limit on M_p without knowledge of inclination (i), it becomes especially powerful when combined with transits, which may provide i from geometry.

RV sensitivity increases with planet mass and proximity to the star, making it well-suited to detecting massive, short-period planets (e.g., hot Jupiters). However, improvements in spectrograph precision—down to $\sim$1 m/s—have enabled the detection of small planets around low-mass stars, including Earth-mass planets like Proxima Centauri b (Fischer et al., 2016; Anglada-Escudé et al., 2016). Since mass is a critical input to structure models, RV is foundational to any interior inference effort.

Transit Photometry: The transit method detects planets by observing the periodic dimming of a star as a planet passes in front of it. The depth of this flux decrement yields the ratio of planet to star radius (Winn, 2010):

$$\delta = \left(\frac{R_p}{R_*}\right)^2.$$

When combined with accurate stellar parameters, the planetary radius R_p can be extracted. Transit surveys like *Kepler* and *TESS* have dramatically expanded the known planet population, particularly for close-in, sub-Neptune-sized planets (Borucki et al., 2010; Ricker et al., 2015).

Because the probability of transit is inversely proportional to the orbital separation, transit detections are biased toward close-in planets. However, these are also the planets most amenable to atmospheric characterization and thus form the core of our sample. Radius is one of two key quantities (alongside mass) used to constrain interior composition, and thus forms a cornerstone of our modeling framework.

Transmission and Emission Spectroscopy: For transiting planets, follow-up spectroscopic observations can reveal the composition and thermal structure of the atmosphere. During a transit, starlight filters through the planetary limb, allowing for the detection of molecular absorption features—a technique known as transmission spectroscopy. Similarly, during secondary eclipse, emission spectra can be obtained by measuring the planet's thermal contribution.

These spectra provide constraints on the presence of molecules (e.g., H_2O, CO_2, CH_4), atmospheric metallicity, and the mean molecular weight (MMW) (Kempton and Knutson, 2024). The MMW, in particular, is vital to this work: it acts as a proxy for the envelope composition, enabling us to constrain the bulk composition of our target exoplanets. Additionally, observations from *HST* and *JWST* play a central role in bridging atmospheric data with interior structure models.

Equilibrium Temperature ($T_{\rm eq}$): Though not directly measured, $T_{\rm eq}$ is calculated from the stellar flux incident on the planet, using known stellar parameters and the orbital distance inferred from transits or RV. This quantity sets the outer boundary temperature in our models and influences the atmospheric scale height and thermal profile. We assume a Bond albedo (typically 0 unless otherwise stated) and full redistribution to estimate $T_{\rm eq}$ for each planet in our sample.a

Together, these techniques yield the physical parameters—M_p, R_p, $T_{\rm eq}$, and MMW—used as inputs to the `SMILE` structure solver. Understanding how each of these quantities is derived, and the limitations and biases of the methods that produce them, is essential for interpreting the internal structures of sub-Neptune planets.

Summary of Observable Parameters

Interior models rely on the following observables:

- **Mass (M)** — from Radial Velocity (RV)
- **Radius (R)** — from Transit Photometry
- **Equilibrium Temperature ($T_{\rm eq}$)** — from stellar flux and orbital separation
- **Mean Molecular Weight (MMW)** — from Atmospheric Spectroscopy

These form the key inputs to the `SMILE` structure solver.

1.2 Exoplanet Demographics

1.2.1 Diversity of Exoplanets

The discovery of exoplanets over the past three decades has revealed a staggering diversity in planetary systems. The first confirmed exoplanet orbiting a Sun-like star—51 Pegasi b—was discovered in 1995 via the radial velocity method (Mayor and Queloz, 1995). This hot Jupiter, orbiting its star every 4.2 days, was unlike anything in the Solar System, immediately highlighting the surprising variety of planetary system architectures. Since then, thousands of exoplanets have been detected, with radii ranging from smaller than Earth to more than twice that of Jupiter, and with orbital periods spanning from a few hours to several years. The known exoplanet population now includes massive gas giants, rocky planets, and volatile-rich sub-Neptunes. Many of these planets have no clear Solar System analogue—especially those in the intermediate radius regime between Earth and Neptune, where sub-Neptunes dominate the population.

This diversity is apparent in both the radius–period and mass–radius parameter

spaces (Figures 1 and 2). These distributions demonstrate that exoplanets occupy a wide range of physical and orbital regimes, although their detectability is strongly shaped by observational biases. Detection techniques such as transits and radial velocities are most sensitive to planets with large sizes or masses and short orbital periods, leading to an overrepresentation of close-in, gas-rich planets in current exoplanet catalogs. As a result, occurrence rates derived from these samples must be interpreted carefully (Youdin, 2011; Cassan et al., 2012).

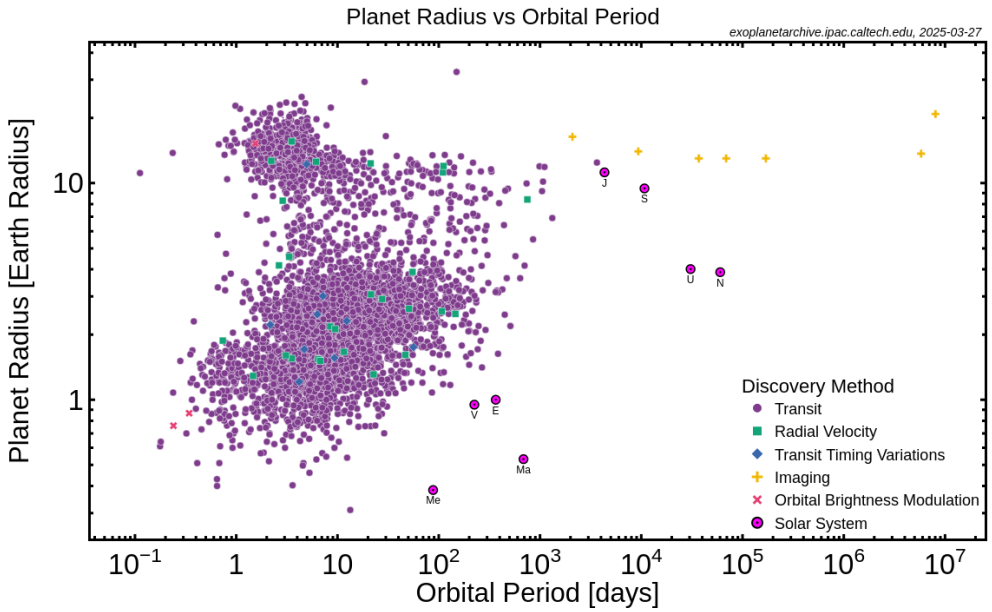

Figure 1: Radius–period distribution of known exoplanets, color-coded by detection method. NASA Exoplanet Science Institute (2025).

Despite these biases, it is still possible to categorize exoplanets into several broad, phenomenologically motivated classes:

Gas giants: These planets have masses and radii comparable to Jupiter and Saturn. Many are located on short-period orbits and are therefore subject to intense stellar irradiation, earning the designation "hot Jupiters." Others occupy wider orbits and are cooler, more akin to the Solar System's giant planets. Some massive objects exceed 13 Jupiter masses and blur the boundary with brown dwarfs.

Ice giants: Analogous in size and mass to Uranus and Neptune, these planets tend to have significant volatile content but lower H/He fractions than gas giants. While

most detected ice giants reside on short orbits ($P \lesssim 100$ days), future missions like *Roman* are expected to expand our ability to detect longer-period analogues (Spergel et al., 2015).

This diversity is further illustrated by the mass–radius distribution (Figure 2), which reveals a wide spread in bulk density and highlights the compositional continuum from rocky to gas-rich planets.

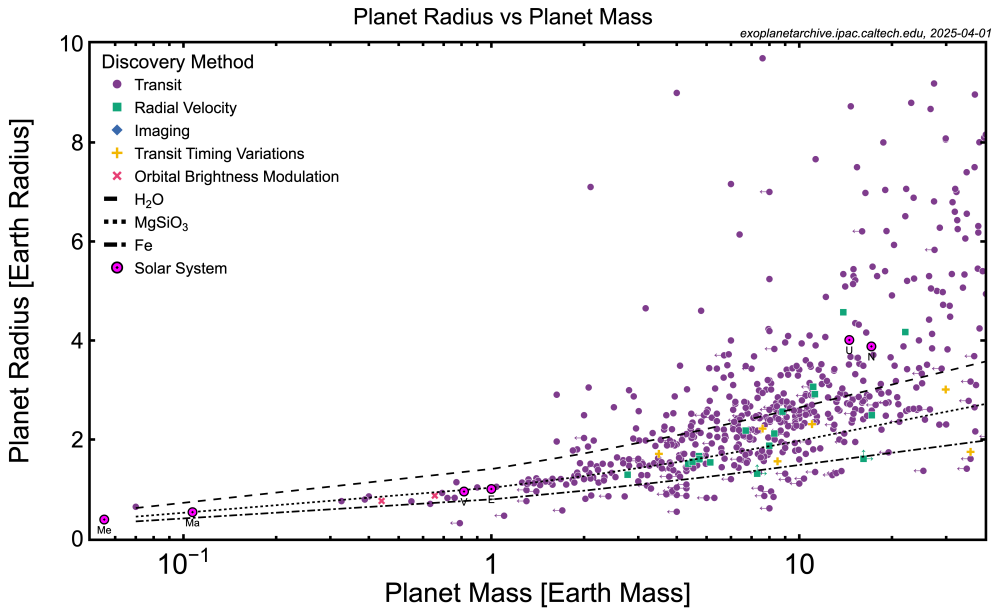

Figure 2: Mass–radius distribution of confirmed exoplanets up to $40M_{\oplus}$, with composition curves for pure iron, silicates ($MgSiO_3$), and water. The broad scatter reflects the diversity in bulk composition among similarly sized planets. NASA Exoplanet Science Institute (2025).

Sub-Neptunes: This class includes planets with radii between roughly 1 and 4 $R_{\oplus}$, and is the dominant population in the size range probed by *Kepler* and *TESS*. These planets exhibit a wide range of compositions—from rocky cores with thin hydrogen atmospheres to volatile-rich water worlds. They are often divided into *super-Earths* (1–1.5 $R_{\oplus}$) and *mini-Neptunes* (2–4 $R_{\oplus}$), depending on their likely bulk structure, as inferred from their radius and density. (Mulders, 2018; Rogers and Owen, 2021). Sub-Neptunes are particularly abundant around M dwarfs and are the focus of this study.

Rocky planets: Planets with radii below $\sim$1.5 $R_\oplus$ and high bulk densities are inferred to be primarily rocky. These include true Earth analogues, though their detection is challenging due to their small size and low signal-to-noise ratios in both RV and transit data. Interestingly, the population exhibits a deficit of planets near 1.5–2 $R_\oplus$, known as the *radius valley*, which may reflect divergent evolutionary pathways driven by atmospheric loss (see figure 4, Rogers and Owen (2021)). Whether rocky exoplanets resemble the terrestrial planets of the Solar System in structure and composition remains an open question.

Beyond these size-based groupings, exoplanets also orbit an extraordinary range of stellar hosts—from cool M dwarfs to hot, massive O- and A-type stars. Host star properties can influence planetary formation, retention of volatiles, and the likelihood of atmospheric loss. For instance, metal-rich stars have been linked to an increased likelihood of hosting giant planets (Quirrenbach et al., 2011; Fischer and Valenti, 2005), and high-energy radiation from young or active stars can strip away lightweight atmospheres—particularly for low-mass planets in close-in orbits

Overall, the combination of these observational biases, stellar dependencies, and compositional diversity motivates the development of flexible, physics-based interior models capable of disentangling this complexity—one of the primary aims of this thesis. We will focus on characterizing the interiors of sub-Neptunes, since their ubiquity and lack of solar system analogues makes them particularly interesting targets for further study.

1.2.2 Bulk Composition of sub-Neptunes

For planets with well-measured masses and radii, it is possible to constrain their bulk densities and thereby gain insight into their interior compositions. These quantities serve as critical inputs to planetary interior structure models like `SMILE` (see Section 2.2), which use them to infer plausible combinations of core, mantle, water, and gas envelope components. A first-order interpretation involves com-

paring observed properties to theoretical mass–radius relations for planets of pure composition—iron, silicate rock ($MgSiO_3$), water, or hydrogen–helium. While such models are admittedly idealized, they offer valuable baselines for interpreting the diversity of planetary interiors.

This diversity is further illustrated in Figure 3, which shows the mass–radius distribution for sub-Neptune to Neptune-sized planets orbiting M dwarfs (Rogers et al., 2023). Overlaid are theoretical curves for Earth-like, water-rich, and H/He-enveloped planets, highlighting the wide range of plausible compositions among small planets. The light orange band denotes the range of sizes consistent with thin H/He atmospheres atop rocky cores, while some low-density planets lie above even these curves—suggesting significant water content. However, degeneracies remain: planets with intermediate densities can often be fit by either water-rich interiors or rocky compositions enveloped in thin hydrogen atmospheres (Mousis et al., 2020; Turbet et al., 2020; Aguichine et al., 2021). The equilibrium temperature also plays a key role, influencing atmospheric retention and escape.

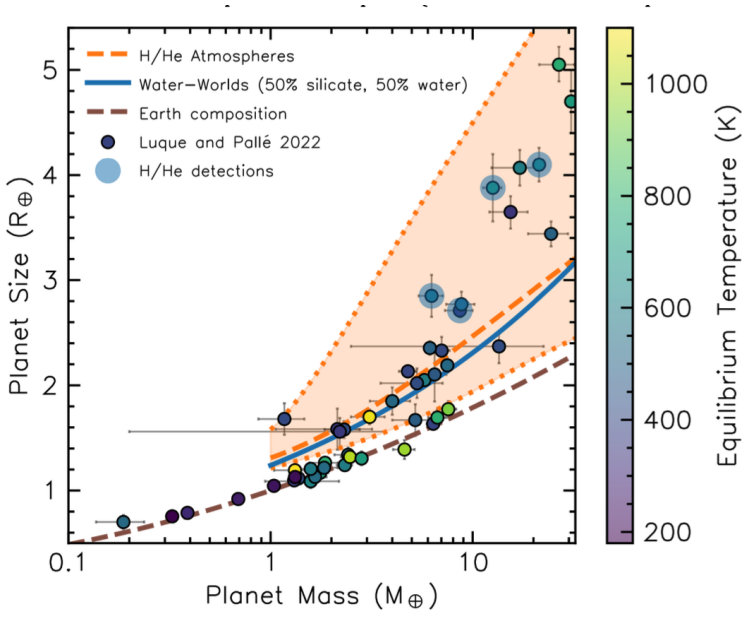

Figure 3: Mass–radius diagram of planets orbiting M dwarfs, overlaid with composition models for rocky, water-rich, and H/He-enveloped interiors. Orange shading marks the regime of low-density planets consistent with H/He atmospheres. Adapted from Rogers et al. (2023).

The interpretation of sub-Neptune compositions is particularly complex. These planets exhibit a wide range of densities and interior makeups, suggesting that their cores, water layers, and atmospheres can vary greatly from one planet to another. Figure 4 illustrates one of the most striking population-level features: the so-called *radius valley*, a dearth of planets between 1.5 and 2 $R_\oplus$ that separates rocky super-Earths from larger mini-Neptunes (Fulton et al., 2017). This feature is thought to arise from evolutionary processes such as photoevaporation (Owen and Wu, 2013) or core-powered mass loss (Gupta and Schlichting, 2019), which preferentially strip atmospheres from low-mass, close-in planets—leaving behind smaller, denser cores. The valley therefore encodes valuable information about atmospheric retention and the thermal history of exoplanets.

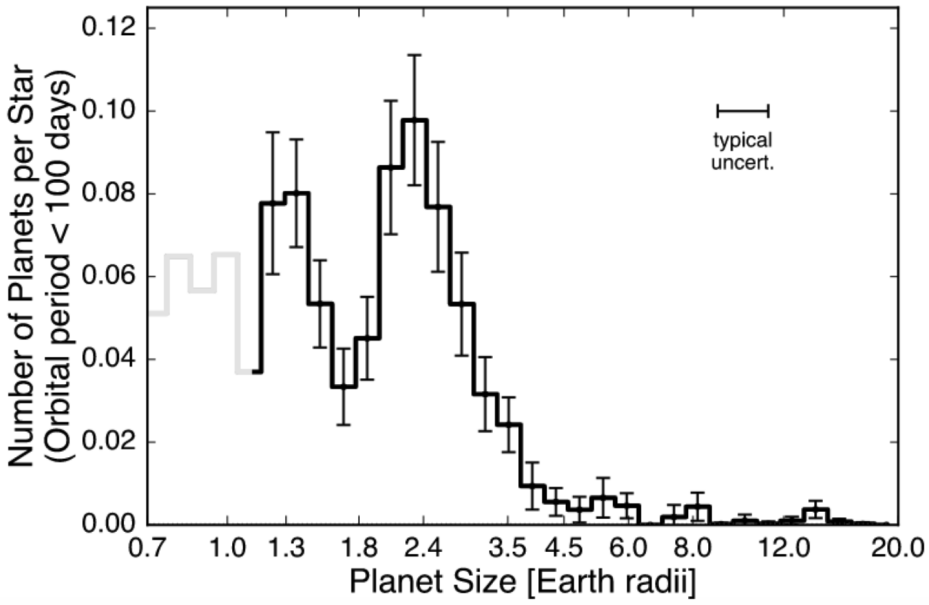

Figure 4: Completeness-corrected histogram of planet radii for short-period planets, showing a bimodal distribution with peaks near 1.3 $R_\oplus$ and 2.4 $R_\oplus$. The deficit near 1.7 $R_\oplus$ is known as the radius valley. Adapted from Fulton et al. (2017).

While mass and radius provide first-order constraints on planetary compositions, equilibrium temperature and atmospheric metallicity can influence a planet's interior structure and what is observable from spectral retrievals. Benneke et al. (2024) proposed a temperature-dependent classification scheme for sub-Neptunes, distinguishing three distinct interior regimes: Hycean worlds (hydrogen-rich planets

with potential subsurface oceans), Stratified mini Neptunes (planets with hydrogen-rich low metallicity atmosphere above denser volatile layer), and miscible-envelope sub-Neptunes (where volatiles remain well mixed with hydrogen throughout the envelope, Benneke et al., 2024). Volatiles, in this context, refer to molecules like water (H_2O), methane (CH_4), and carbon dioxide (CO_2) that can exist in gas or liquid form and easily respond to changes in temperature and pressure.

These regimes differ in whether water and other volatiles condense out of the upper atmosphere or remain well-mixed with hydrogen, which in turn affects the atmospheric composition accessible to transmission spectroscopy, as shown in Figure 5. In colder, stratified (layered) interiors, heavy volatiles may be hidden beneath the observable atmosphere, making the planet appear more hydrogen-rich than it truly is. In contrast, warmer planets with fully mixed envelopes allow volatiles to remain suspended throughout the atmosphere, enabling a more accurate retrieval of the bulk envelope composition. In the population-level study presented in this study, we model the planets assuming a mixed envelope scenario.

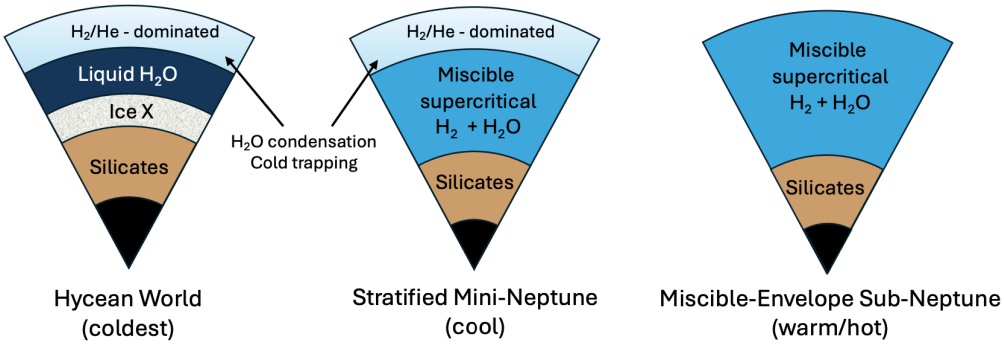

Figure 5: Temperature-dependent interior structure of sub-Neptunes driven by the phase changes of H_2O. For example, TOI-270d's high atmospheric metal mass fraction indicates that high-molecular-weight volatiles (H_2O, CH_4, CO, CO_2) are well-mixed with the H_2/He in a warm miscible envelope (right scenario). Adapted from Benneke et al. (2024).

Nonetheless, compositional inference remains highly degenerate. Many observed planets are consistent with a range of interiors, depending on assumptions about temperature, composition, and formation history. For example, the same bulk

density can correspond to either a rocky planet with a thin H/He envelope or a water-rich world with little to no gas (Lozovsky et al., 2018). Atmospheric metallicity, temperature, and host star activity all further modulate observable properties and must be accounted for in any robust interior model (Thorngren et al., 2016, 2019). The presence of volatile-rich "water worlds" is especially intriguing, particularly around M dwarfs, where ice-rich material can be accreted even in close-in orbits due to the proximity of the snow line (Kimura and Ikoma, 2022). Indeed, some studies suggest that low-mass stars may preferentially host planets with water-dominated envelopes (Luque and Pallé, 2022).

As more precise mass and radius measurements become available, alongside spectroscopic observations with JWST to measure atmospheric composition, the ability to distinguish between rocky, water-rich, and gas-enveloped planets will improve. In the meantime, models like `SMILE` (more details on section 2.2) are essential tools for interpreting this structural diversity. They allow us to move beyond bulk density alone and extract more meaningful inferences about the internal structure of planets like TOI-270d (see Section 3) within the broader context of the exoplanet population.

1.2.3 Habitability

The broader search for life remains one of the core motivations behind exoplanet science. A planet is often considered potentially habitable if it resides within the "habitable zone" of its host star—the region where stellar insolation allows for liquid water to exist on the surface (Dressing and Charbonneau, 2015). However, this orbital definition alone is not sufficient. A planet's atmosphere, interior composition, and the activity of its host star all strongly influence its surface and subsurface conditions (Meadows and Barnes, 2018). Particularly around M dwarfs, flare activity and atmospheric erosion pose challenges to habitability, even for planets within this nominal zone.

In recent years, the concept of habitability has expanded to include exotic configurations such as Hycean worlds—sub-Neptunes with hydrogen-rich atmospheres overlying high-pressure liquid water layers (Madhusudhan et al., 2021). While these planets may not have solid surfaces, they could still offer stable, temperate environments shielded from harmful radiation. The James Webb Space Telescope (JWST), already operational, is actively characterizing the atmospheres of small exoplanets, including several Hycean candidates. Meanwhile, the upcoming Extremely Large Telescope (ELT), expected to begin science operations around 2028, will push the frontier even further by enabling high-resolution ground-based spectroscopy (Observatory, 2024). Together, these observatories provide our best opportunity yet to detect atmospheric biosignatures and probe the limits of planetary habitability beyond the Solar System.

1.3 Motivation for This Study

Sub-Neptune exoplanets, planets with radii between Earth and Neptune, represent the most common class of exoplanets in our galaxy. Yet, their composition and formation histories remain poorly understood. Unlike the terrestrial or gas giant planets of our solar system, sub-Neptunes have uncertain interior structure properties. This presents a major modeling challenge: multiple compositions can produce the same mass and radius, a problem known as compositional degeneracy.

Although thousands of sub-Neptunes have been discovered, most existing models struggle to resolve this degeneracy. Many rely on oversimplified assumptions, such as isothermal interiors or stratified, unmixed volatile layers, and often neglect constraints from atmospheric observations. As a result, key questions remain unanswered: What is the true compositional diversity of sub-Neptunes? Are they water-rich, hydrogen-rich, a combination of sub-populations, or something else? How do these objects form and evolve?

This study addresses these challenges by developing interior structure models

that incorporate temperature-dependent equations of state, isothermal–adiabatic thermal profiles, and mixed H/He–H_2O envelopes. Importantly, this work connects interior structure models to observable quantities such as atmospheric mean molecular weight (MMW), enabling direct comparison with JWST spectral retrievals.

The analysis begins with TOI-270d, a well-characterized sub-Neptune with new JWST constraints, and expands to a broader population of planets selected for upcoming or ongoing atmospheric observations. Through this combined case-study and population-level approach, the goal is to demonstrate that detailed interior structure modeling—when done carefully—can yield meaningful insights into planetary composition, even in the absence of detailed atmospheric spectra. As high-precision observations become more common, frameworks like this will be essential for interpreting the growing diversity of small exoplanets.

2 Modeling Framework and Physical Foundations

2.1 Initial Steps to Interior Structure Modeling

Understanding a planet's internal structure requires linking physical laws with material properties. At the heart of this connection is the equation of state (EOS), which describes how material's density responds to change in pressure and temperature:

$$\rho = \rho(P, T) \tag{1}$$

Each planetary material-such as iron, silicates, water, or hydrogen-helium, has its own EOS, which determines how compressible it is under planetary conditions. The EOS is essential for evaluating the planet's internal density structure and,

ultimately, its radius.

Once the EOS provided a way to get the local density, the stage was set to numerically integrate the structure equations that shape a planet's interior layer by layer.

The first equation I used to integrate was the mass continuity equation, which ensures that mass is correctly distributed throughout the planet's volume. It describes how the radius changes with increasing enclosed mass:

$$\frac{dR}{dM} = \frac{1}{4\pi R^2 \rho} \tag{2}$$

At each step of the integration, the local density ρ is required. This value is obtained using the equation of state (EOS), which defines density as a function of pressure and temperature. Once ρ is known, the mass continuity equation provides the next value of the radius.

The second key equation is hydrostatic equilibrium, which ensures that the inward force of gravity is balanced by the pressure gradient at each layer:

$$\frac{dP}{dM} = -\frac{GM}{4\pi R^4} \tag{3}$$

This equation is evaluated using the current radius (from mass continuity) and the mass enclosed up to that layer. Together, these two differential equations describe the planet's interior structure when linked by the provided EOS.

The equation of state (EOS) describes how materials compress under pressure and temperature—directly determining the planet's internal density structure and, ultimately, its radius. During this initial step, my work didn't consider the temperature part from the EOS. For rocky interiors (iron and silicate), I adopted isothermal EOS models from Seager and et al. (2007), as thermal effects in these layers have minimal influence on the mass–radius relation (Grasset et al., 2009;

Howe et al., 2014).

To solve this coupled system, I applied Euler's method, which propagates the solution forward using a first-order approximation. At each mass increment ΔM, the radius and pressure are updated according to:

$$R_{i+1} = R_i + \left(\frac{dR}{dM}\right)_i \cdot \Delta M \quad , \quad P_{i+1} = P_i + \left(\frac{dP}{dM}\right)_i \cdot \Delta M \tag{4}$$

Mass increases step-by-step as ΔM is added at each layer:

$$M_{i+1} = M_i + \Delta M$$

Here:

- The subscript i denotes the current step in the numerical integration, corresponding to a shell at a given depth inside the planet.
- M_i is the total mass that has been accumulated up to the current step.
- R_i and P_i are the radius and pressure at step i (the current shell).
- R_{i+1} and P_{i+1} are the updated values at the next shell, after adding a small mass increment ΔM.
- $\left(\frac{dR}{dM}\right)_i$ and $\left(\frac{dP}{dM}\right)_i$ are the derivatives evaluated at step i.

This method allows the structure to be built up incrementally from the surface inward, keeping track of how radius and pressure evolve as more mass is enclosed.

The integration proceeds from a known surface pressure and an initial guess for the planet's surface radius R_p. To find the correct planetary radius, I began by setting a plausible range of surface radii that spans the expected size for Earth-like planets. I then calculated the midpoint of this range and used it as an initial guess

for R_p. This radius guess was passed to the integration routine, which computed the resulting total mass. I compared the integrated mass to the observed mass of the target planet.

If the integrated mass was too large, it meant the radius was underestimated, so I increased the guess. If the integrated mass was too small, the radius was too large, and I decreased it. This process was repeated until the computed mass matched the observed mass to within one part in a million. We call this a bisection method. The program evaluates the radius and pressure profiles as it accumulates mass, and terminates when the total enclosed mass equals the target planet mass.

To validate this method, I first modeled an Earth-mass planet with a pure silicate ($MgSiO_3$) and then a pure iron (Fe) composition. I adopted isothermal EOS models from Seager and et al. (2007) for both $MgSiO_3$ and Fe, interpolated density as a function of pressure, and implemented the structure equations as described above.

I extended the model to planets ranging from 1 to 10 Earth masses to examine how the radius changes with mass. As shown in Figure 6, pure silicate consistently yield larger radii than pure iron planets at a given mass, which shows the lower density of silicate relative to iron. For example, at 5 Earth mass, a silicate planet has a radius of around 1.6, while a pure iron planet is closer to 1.2 Earth radii. My model successfully reproduced the radius of a pure silicate and pure iron Earth to within a few percent of published mass–radius relations (Seager and et al., 2007).

Figure 6 is modeled on an Isothermal temperature profile assumption. An *isothermal profile* assumes a constant temperature throughout the planet's interior. While this is a simplification compared to real planetary interiors, which typically have temperature gradients, this assumption removes temperature as a variable in the EOS and allows the use of a pressure-dependent tabulated EOS. As a result, it provides a computationally simple and physically reasonable baseline for initial validation.

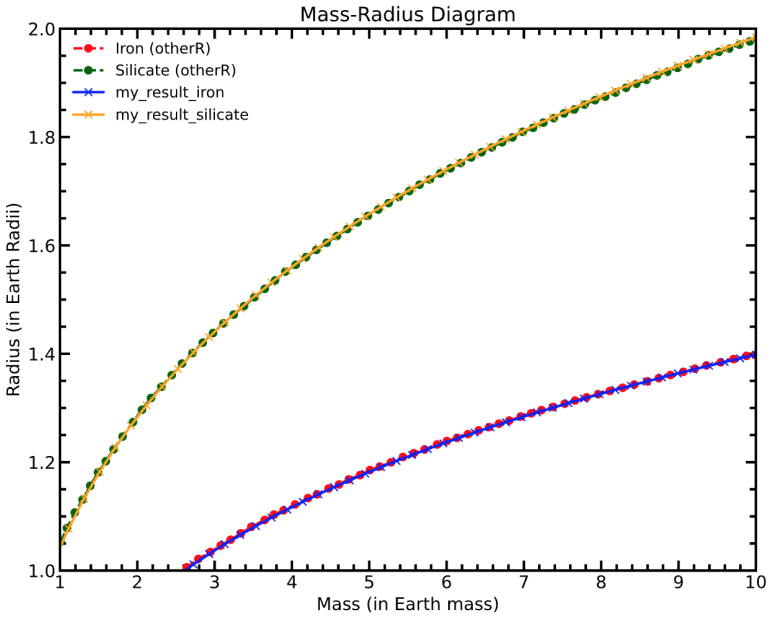

Figure 6: Mass–radius relationship from initial modeling work. The curves represent my calculated results for planets composed of pure iron and pure silicate, compared to known theoretical models.

Although these initial models offered useful insights into how planetary radius responds to mass and composition, they were limited in several key ways. They assumed a single, uniform material throughout the planet, neglected the effects of temperature gradients, and could not account for layered structures or phase transitions. As such, they could not capture the full physical complexity of exoplanets—particularly those with volatile-rich envelopes or differentiated interiors.

To overcome these limitations, I transitioned to a more sophisticated modeling framework: `SMILE`[1]. While it builds on the same physical principles and structure equations as my initial model, `SMILE` expands the scope considerably—it supports multiple compositional layers, temperature-dependent equations of state, and more realistic thermal structures, including isothermal–adiabatic profiles and fully mixed H/He–H_2O envelopes.

An isothermal–adiabatic profile assumes that the outermost part of the atmosphere

[1]Structural Model of Internal Layers of Exoplanets

is at constant temperature (isothermal), anchored to the planet's equilibrium temperature, while deeper layers transition into a convective region where temperature increases with pressure along an adiabatic gradient (Nixon and Madhusudhan, 2021). This two-part structure reflects the physical conditions expected in sub-Neptune atmospheres, where stellar irradiation dominates the upper layers and internal heat transport drives convection below.

This framework forms the foundation for the interior structure modeling presented in this thesis.

2.2 SMILE

While my initial structure models (see Section 2.1) focused on single-layer planets using simplified assumptions, more realistic exoplanet interiors require a flexible framework that incorporates layered compositions, temperature gradients, and temperature-dependent material properties. For this purpose, I utilized the publicly available `SMILE` package (Nixon and Madhusudhan, 2021), developed by Nixon et al., which is designed to simulate the internal structure of exoplanets with arbitrary layerings and thermal profiles.

`SMILE` builds on the same fundamental structure equations introduced earlier—mass continuity and hydrostatic equilibrium—but extends the approach to accommodate complex, multi-material interiors and self-consistent thermal stratification. Instead of solving for radius using a fixed composition and isothermal profile, `SMILE` supports up to four fully differentiated layers:

- An iron (Fe) core,
- A silicate ($MgSiO_3$) mantle,
- A water (H_2O) layer,
- An optional hydrogen–helium (H/He) envelope.

Each layer is defined by a mass fraction, and the sum of the layers must equal the total planet mass.

Model Inputs: Figure 7 summarizes the key inputs required by `SMILE`. These include:

- The total planetary mass (M_p), surface pressure (P_0), and surface temperature (T_0) as boundary conditions,
- Mass fractions for each of the four potential layers,
- A pressure–temperature profile (e.g., isothermal–adiabatic) to describe the thermal structure,
- Equations of state (EOS) for each material, interpolated from tabulated data.

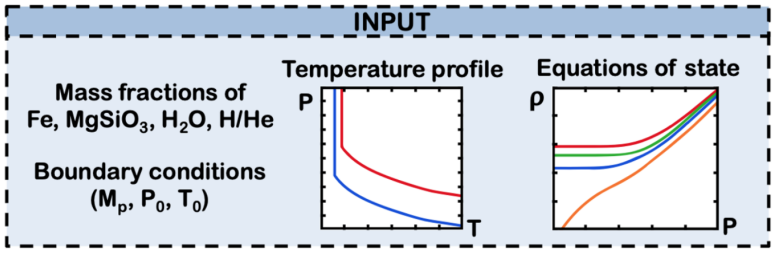

Figure 7: Inputs to the `SMILE` model include layer mass fractions, boundary conditions, a pressure–temperature profile, and material equations of state.

Thermal Structure: Isothermal–Adiabatic Profiles

To accurately model the internal structure of a planet, it is essential to specify how temperature changes with depth. The thermal profile directly influences the equation of state (EOS) and, in turn, the resulting pressure and density profiles.

In this study, we adopt a two-layer temperature structure consisting of an isothermal layer at the top of the atmosphere, transitioning to an adiabatic gradient deeper in the envelope. This *isothermal–adiabatic* profile offers a physically motivated yet computationally tractable approach to representing the thermal structure of sub-Neptune atmospheres (Nixon and Madhusudhan, 2021).

The isothermal region is anchored to the planet's equilibrium temperature $T_{\rm eq}$, defined by stellar and orbital properties:

$$T_{\rm eq} = T_\star \sqrt{\frac{R_\star}{2a}} \cdot (1 - A_B)^{1/4} \tag{5}$$

where $T_\star$ is the effective temperature of the host star, $R_\star$ is the stellar radius, a is the planet's semi-major axis, and A_B is the Bond albedo. The Bond albedo, ranging from 0 (absorbs all energy) to 1 (perfectly reflective), is the fraction of total incident stellar energy that a planet reflects back into space across all wavelengths (Carroll and Ostlie, 2007).

We compute $T_{\rm eq}$ for each planet using Equation 5, assuming zero Bond albedo and full heat redistribution. The resulting equilibrium temperature defines the upper isothermal boundary condition in all `SMILE` simulations.

The transition between the isothermal and adiabatic layers is set by the pressure at the *radiative–convective boundary*, $P_{\rm ad}$, which marks where radiative transport becomes inefficient and convection dominates. Below this boundary, the temperature increases with pressure according to the adiabatic gradient:

$$\left(\frac{dT}{dP}\right)_S = \frac{\alpha T}{\rho c_P} \tag{6}$$

where α is the thermal expansion coefficient, ρ is the local density, and c_P is the specific heat capacity at constant pressure. For mixed H/He and H_2O envelopes, the adiabatic gradient is computed using entropy-weighted mixing (Chabrier et al., 2019; Nixon et al., 2024).

To ensure physical consistency, $P_{\rm ad}$ is selected so that water remains in the vapor or supercritical phase at the base of the isothermal layer. This prevents condensation near the radiative–convective boundary, as shown in Figure 8. The same methods are used for all the planets in this study to calculate the $T_{\rm eq}$ and,$P_{\rm ad}$.

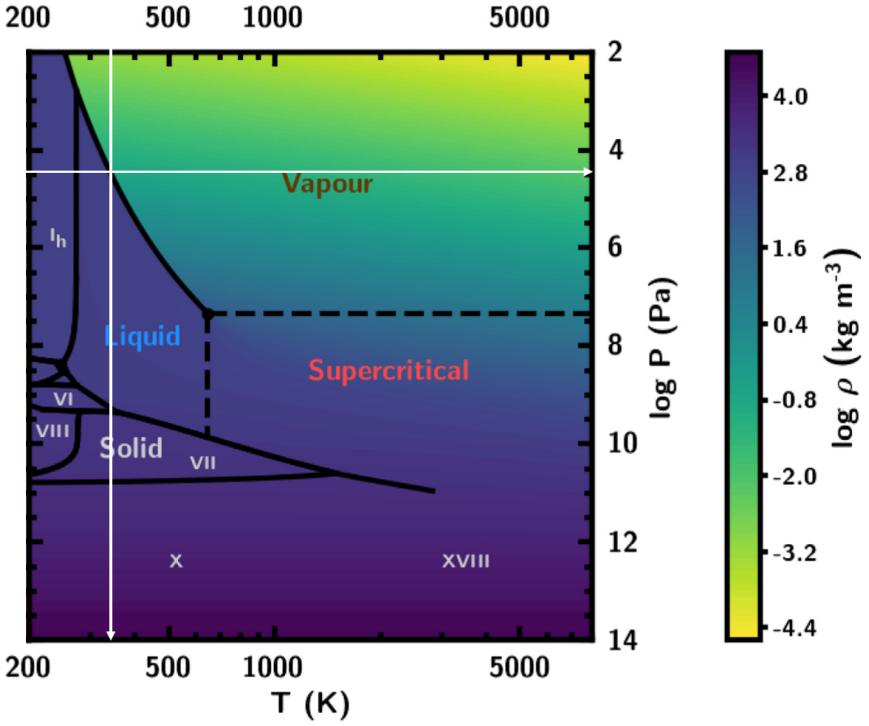

Figure 8: Phase diagram of water adapted from Nixon et al. (2024), showing vapor, liquid, supercritical, and solid regimes. The white lines indicate how the equilibrium temperature constrains the maximum radiative–convective boundary pressure ($P_{\rm ad}$) to prevent condensation for TOI-270d, with $T_{\rm eq}$ = 387 K,$P_{\rm ad}$ = 1.33 barr.

In contrast to the rocky layers, temperature substantially affects how pressure and density evolve in volatile-rich layers. For water, we implement a temperature-dependent EOS from Thomas and Madhusudhan (2016) that captures transitions between vapor, liquid, supercritical, and high-pressure ice phases. For hydrogen–helium, we adopt the EOS from Chabrier et al. (2019), which accounts for thermal and compositional variations relevant to sub-Neptunes.

Figure 9 shows representative EOS curves used in this study, illustrating how density varies with pressure for the major planetary materials modeled in `SMILE`. The hydrogen-helium EOS (red) shows lower densities at low pressures, highlighting its strong contribution to inflating planetary radii even when present in small mass fractions. Water shows a notable density jump corresponding to phase transitions.

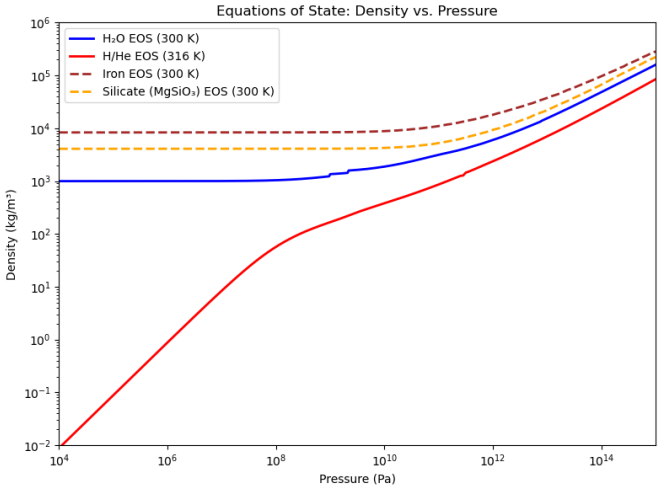

Figure 9: Schematic of representative EOS curves used in this study, showing how density varies with pressure for key planetary materials. Curves are plotted at fixed temperatures around 300 K.

With the compositional and thermal structure specified, `SMILE` proceeds to solve the structure equations numerically, using an iterative approach to determine the planetary radius.

Numerical Integration and Convergence: `SMILE` uses a shooting method with bisection to solve for the planet's radius. Similar to my initial model, an initial guess for surface radius R_p is made. The code then integrates the structure equations inward using a fourth-order Runge–Kutta method, updating pressure, radius, density, and temperature at each step based on the current EOS. The process repeats, adjusting R_p until the integration reaches the center of the planet (i.e., $R(M = 0)$) within a small numerical tolerance.

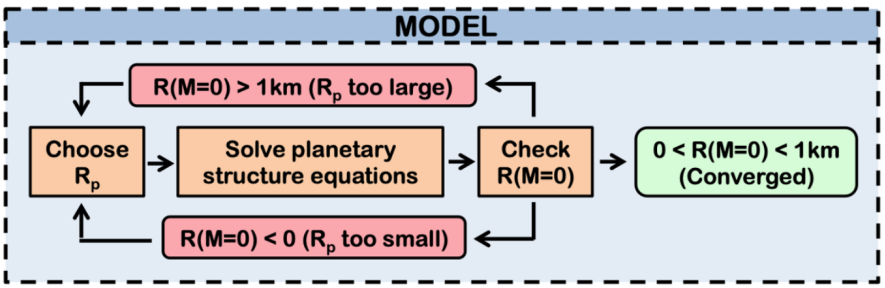

Figure 10: Flowchart of the `SMILE` model's internal loop for solving planetary radius. A bisection algorithm adjusts the radius until the integration reaches the planetary center within a small tolerance.

The use of Runge–Kutta rather than Euler's method allows for greater numerical stability and accuracy, particularly important when modeling stratified planets where density can change steeply across interfaces.

Model Outputs: Once convergence is reached, `SMILE` outputs the final planetary radius R_p along with interior profiles for pressure, temperature, density, and composition. The internal structure is resolved layer-by-layer, allowing direct visualization of the material stratification and the thermal behavior of each component.

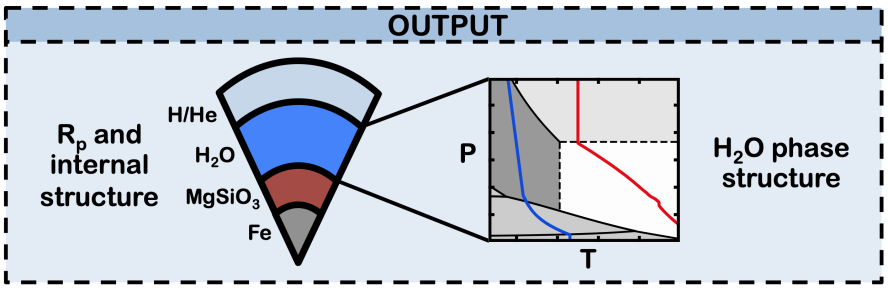

Figure 11: Example `SMILE` output showing the interior layering (Fe, $MgSiO_3$, H_2O, H/He) and corresponding pressure–temperature profile over a water phase diagram. While this shows a fully stratified configuration, the models in this research assume a fully mixed H/He–H_2O envelope, which significantly alters both radius and thermal structure (see also Benneke et al. 2024 and Section 3.4).

Key Advancements Over the Initial Modeling Approach (see Section 2.1):

- `SMILE` supports layered compositions, whereas my model assumed a single uniform material.
- `SMILE` includes temperature-dependent EOS, while my model assumed an isothermal interior.
- `SMILE` incorporates Runge–Kutta integration and bisection convergence, improving numerical accuracy.
- `SMILE` can model fully mixed envelopes and track phase transitions (e.g., water vapor to supercritical), which are not possible in the simplified approach.

Together, these capabilities make SMILE an essential tool for realistic interior modeling, enabling the analysis of planets with complex compositions and observational constraints (e.g., from JWST). It forms the foundation for all structure models used throughout the remainder of this thesis.

2.3 Accelerating SMILE with Multiprocessing

Typically, SMILE has been used to generate a small number of interior models with a fixed composition, often in the context of analyzing a single exoplanet. However, the aim of my research is to explore the internal structure of an entire population of sub-Neptunes. This requires generating thousands of models that span a wide and continuous parameter space, including planetary mass, atmospheric composition, boundary pressures, and temperature profiles. Although each individual SMILE model runs in just a few seconds, evaluating millions of parameter combinations becomes computationally challenging without further optimization.

To address this, I developed a custom multiprocessing framework that parallelizes SMILE evaluations across multiple CPU cores. Instead of computing each model sequentially, the system constructs a full grid of parameter combinations—such as mass, $P_{\rm ad}$, equilibrium temperature, envelope mass fraction, and mean molecular weight (MMW)—and distributes them among all available processing cores. Each core executes SMILE independently, calculating the planetary radius and internal structure for its assigned batch of models. Once the calculations are complete, results are aggregated and saved for downstream analysis.

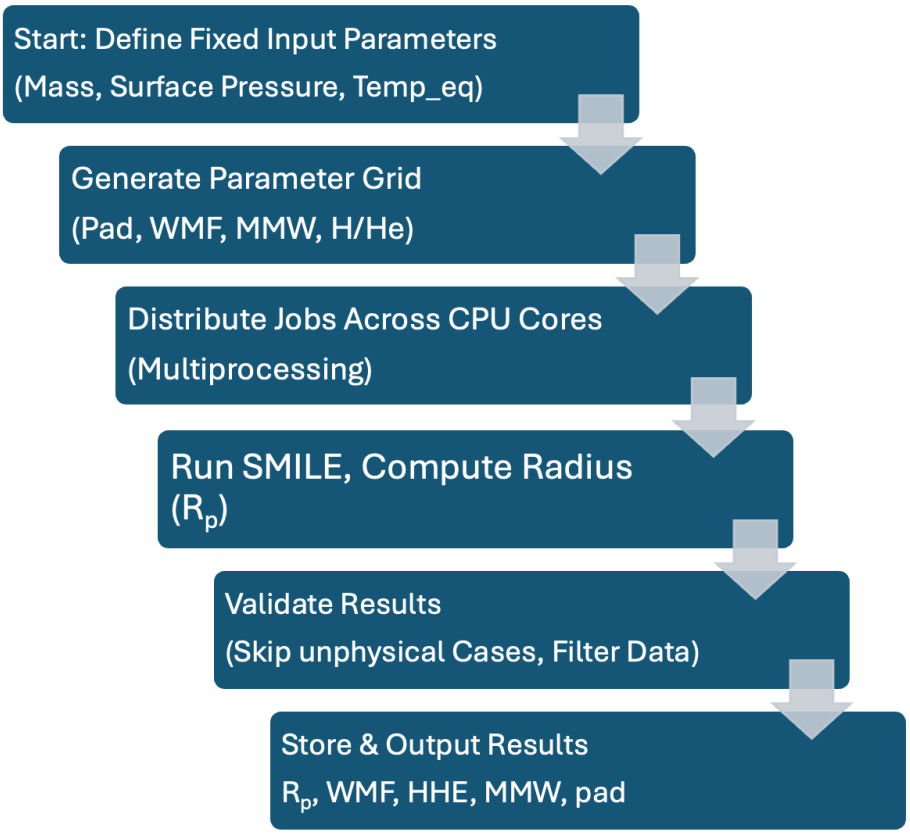

Figure 12: Architecture of the multiprocessing system used for SMILE-based grid modeling for multiple target Exoplanets.

To enhance both physical realism and computational efficiency, the system filters out models deemed unphysical or unstable. In particular, we exclude cases where the hydrogen–helium (H/He) mass fraction exceeds 30%. Such high volatile contents are unlikely to be retained by sub-Neptunes due to a combination of photoevaporative mass loss and limitations set by planet formation theory. Low-mass planets exposed to strong stellar irradiation can lose much of their primordial hydrogen through atmospheric escape, while core accretion models predict that planets in this size regime rarely acquire thick H/He envelopes in the first place (Owen and Wu, 2013; Lee and Chiang, 2015). Additionally, the framework dynamically detects the number of available CPU cores and parallelizes the workload accordingly, ensuring optimal utilization of computational resources across large model grids.

This parallel framework improved computational efficiency by a factor of 5–10, depending on the model grid resolution and hardware. It made high-resolution grid

searches feasible, enabling robust exploration of the mass–radius–composition space. This tool proved especially useful in population-level studies where thousands of planetary models are required to interpret observational trends across exoplanet samples.

3 Analyzing the Internal Structure of TOI-270 d

3.1 Introduction and Scientific Motivation

TOI-270d is a temperate sub-Neptune discovered in 2019 using the transit method by the Transiting Exoplanet Survey Satellite (TESS) (Günther et al., 2019). It orbits a bright M3V-type star with an apparent magnitude of $J = 9.1$ (Mikal-Evans et al., 2023). The planet has a mass of $4.78 \pm 0.43\, M_\oplus$ and a radius of $2.113 \pm 0.065\, R_\oplus$, placing it in the sub-Neptune category (Van Eylen et al., 2021). It completes one orbit every 11.4 days at a distance of 0.0721 AU (Günther et al., 2019).

At the time we began studying TOI-270d, it was considered one of the most promising small exoplanets for atmospheric characterization using transit spectroscopy, due to its relatively low bulk density and temperate equilibrium temperature (Mikal-Evans et al., 2023). These properties suggested that the planet could host a volatile-rich envelope, potentially composed of water vapor or other high–molecular-weight species, making it a strong water-world candidate (Luque and Pallé, 2022; Van Eylen et al., 2021). These traits made TOI-270d an ideal prototype for interior structure modeling.

Initial work exploring the internal structure of TOI-270d simplified its composition by excluding water layers and modeling the planet as a rocky core with a H/He envelope only (Van Eylen et al., 2021). A more recent study by Luque and Pallé (2022) included water as a compositional component, but assumed a purely

isothermal temperature profile throughout the envelope. These simplifications highlighted the need for more physically motivated interior models that incorporate thermal gradients, realistic volatile compositions, and observational constraints such as mean molecular weight (MMW) retrieved from spectra.

TOI-270d was the first planet I analyzed using models generated with the `SMILE` framework. It served as a critical testbed for validating the pipeline, exploring structural degeneracies, and eventually expanding the modeling effort to a larger population of sub-Neptunes.

3.2 First Attempt: Pure H_2O Envelopes

To begin my research exploring sub-Neptune interiors, I explored the hypothesis that TOI-270d could be a pure water world. In this preliminary study, I constructed models assuming a rocky interior (iron core + silicate mantle) overlaid with a pure H_2O envelope. These were the first `SMILE` models I computed.

The key parameters varied in this initial grid were the water mass fraction (WMF) and surface pressure (P_0). The thermal structure was modeled using an adiabatic temperature profile anchored at the equilibrium temperature of the planet.

In these pure water models, the envelope consisted entirely of H_2O vapor, corresponding to a fixed atmospheric mean molecular weight (MMW) of 18.02 g/mol; pure H_2O envelope. Figure 13 and figure 14 illustrate some of the results from this initial analysis.

Note that, to identify physically consistent models, I used a chi-squared-like filtering criterion that compares the modeled mass and radius to the observed values. Models were retained only if they fell within the 1σ observational uncertainties in both mass and radius, satisfying:

$$\chi^2 = \left(\frac{M_{\rm model} - M_{\rm obs}}{\sigma_M}\right)^2 + \left(\frac{R_{\rm model} - R_{\rm obs}}{\sigma_R}\right)^2 \leq 1 \tag{7}$$

where $M_{\rm obs}$ and $R_{\rm obs}$ are the observed mass and radius of TOI-270d, $M_{\rm model}$ and $R_{\rm model}$ are the corresponding model predictions, and σ_M and σ_R are their respective uncertainties. This same filtering criterion was applied consistently across all models, including those with layered and mixed volatile envelopes (see Sections 3.3 and 3.4).

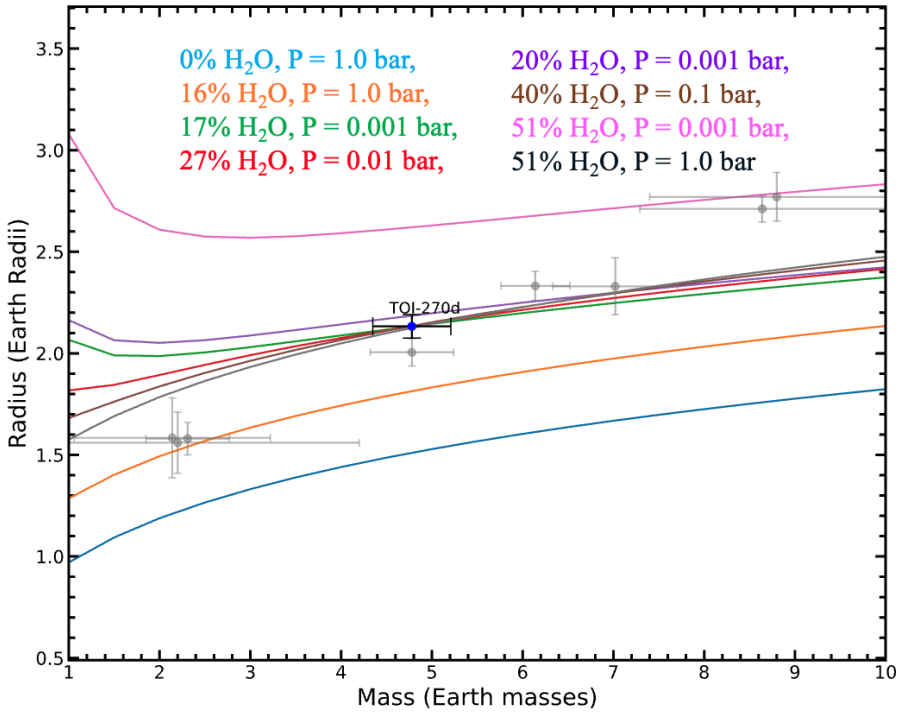

Figure 13: Mass–radius relationships for TOI-270d assuming pure H_2O envelopes with varying water mass fractions and surface pressures. The observed mass and radius of TOI-270d are indicated in blue.

As illustrated in Figure 13, a range of water mass fractions and surface pressures yield models that are consistent with the mass and radius of the planet.

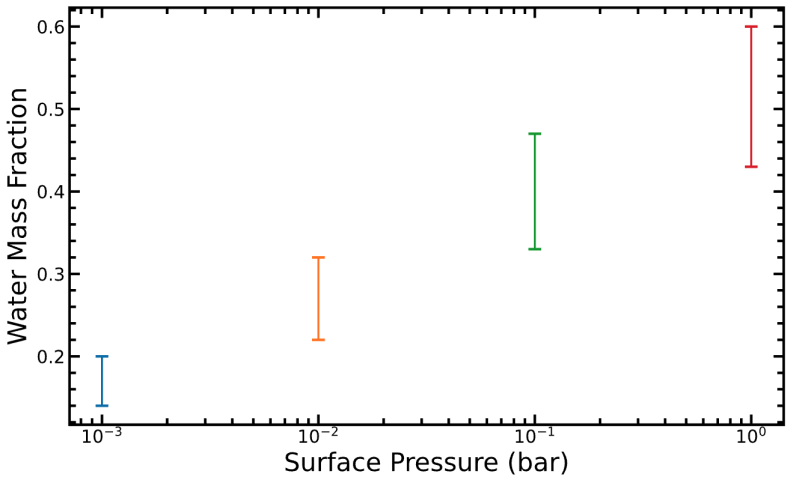

Figure 14: Water mass fraction vs. surface pressure for pure H_2O envelope models of TOI-270d that match its observed radius.

Figure 14 shows the range of envelope water mass fractions and corresponding surface pressures that yield models consistent with TOI-270d's measured mass and radius. As the surface pressure increases, we can see the range of water mass fractions that are consistent with the measured mass and radius of TOI-270d.

In the plots above, the water mass fraction (WMF) refers to the fraction of the *entire planet's* mass made up of water—that is, the bulk WMF. In this pure H_2O envelope scenario, all volatile mass is water, so the envelope WMF is equal to the total WMF. Later sections will distinguish more clearly between these two quantities. Ultimately, we found that a range of WMF values were consistent with the mass and radius of the planet. However, in this preliminary study, we assumed a purely adiabatic temperature profile, rather than setting the radiative-convective boundary at a location appropriate for the planet's equilibrium temperature. Our later work used more realistic temperature profiles, leading to some differences between results.

This first modeling phase served as both a learning experience and a foundational validation of the `SMILE` framework. It also underscored the importance of surface pressure and thermal structure in shaping planetary radii. These early results pointed toward key directions for future modeling—namely, the need to refine the

thermal profile using self-consistent atmospheric models and to incorporate the effects of a hydrogen–helium (H/He) envelope for a more realistic treatment of volatile-rich atmospheres.

3.3 Layered H/He–H_2O Envelopes (Prior to MMW Constraints)

Following this early work, I began incorporating hydrogen and helium into the envelope structure. This marked a shift from the pure H_2O hypothesis to a more realistic two-component envelope model composed of H_2O and H/He. This also introduced the concept of the *total envelope mass fraction*—the fraction of the planet's total mass assigned to the volatile envelope, separate from the solid interior. The envelope was modeled as *layered* rather than mixed, meaning H_2O and H/He occupied distinct regions within the volatile layer.For each model, I independently varied the envelope mass fraction (from 0 to 1) and the envelope's water mass fraction (WMF, also from 0 to 1). Here, envelope WMF refers specifically to the mass fraction of water within the envelope alone, that is the fraction of the envelope's mass composed of H_2O.

The total mass of water and H/He in the planet was calculated as:

$$\text{Total water mass fraction} = \text{envelope fraction} \times \text{WMF}$$

$$\text{Total H/He mass fraction} = \text{envelope fraction} \times (1 - \text{WMF})$$

These models were still constrained only by mass and radius within 1σ of the observed values, with no atmospheric or spectral information incorporated at this stage. The envelope components were not yet treated as miscible, and no mean molecular weight (MMW) constraints were applied.

While useful for initial exploration, these layered-envelope models carried inher-

ent limitations. Recent observations and theoretical work suggest that in warm sub-Neptunes like TOI-270d, hydrogen, helium, and water are likely to exist as a homogeneous, miscible mixture under the planet's high-pressure, high-temperature conditions (Benneke et al., 2024). In such regimes, phase separation is unlikely, and efficient convection promotes compositional mixing throughout the envelope. Moreover, transmission spectra from JWST will provide constraints on the atmospheric mean molecular weight (MMW), enabling a direct link between modeled envelope composition and observables. These insights motivated the transition to fully mixed H/He–H_2O envelopes, guided by MMW-based sampling. This approach not only reflects a more physically realistic interior structure but also enables forward modeling compatible with spectroscopic data.

3.4 Mixed H/He–H_2O Envelopes: Transitioning to MMW-Based Sampling

Recent theoretical and observational advances suggest that many sub-Neptunes possess atmospheres where hydrogen, helium, and water remain well-mixed rather than separated into distinct layers. This configuration is supported by both high-pressure miscibility studies Soubiran and Militzer (2015) and the expectation of strong vertical mixing in volatile-rich envelopes. In particular, steep temperature gradients in sub-Neptune atmospheres-driven by stellar irradiation and internal heat-can trigger deep convection and inhibit the formation of stable, layered structures (Pierrehumbert, 2023). As a result, fully mixed envelopes are now considered a physically realistic configuration for many volatile-rich exoplanets, including recent studies of TOI-270d and GJ 1214b (Benneke et al., 2024; Nixon et al., 2024).

Motivated by this, I transitioned from layered-envelope models to a framework that assumes fully mixed H/He–H_2O envelopes in all `SMILE` simulations. In this setup, the gaseous envelope is treated as a single, well-mixed fluid composed of

hydrogen, helium, and water, under the assumption of fully mixed envelopes.

3.4.1 Exploring Mixed Envelopes with WMF Sampling and MMW Constraints

The atmospheric mean molecular weight (MMW) of TOI-270d's atmosphere was recently determined to be $5.47^{+1.25}_{-1.14}$ g/mol (Benneke et al., 2024). Assuming that the envelope consists primarily of hydrogen, helium, and water, it is possible to map this MMW constraint to an envelope water mass fraction (WMF). At this stage, the envelope was assumed to be fully mixed, but sampling was still conducted in terms of water mass fraction (WMF), rather than directly in MMW space. To bridge the two, I constructed a numerical mapping between WMF and MMW for H/He–H_2O mixtures, as shown in Figure 15.

Assuming a solar hydrogen-to-helium mass ratio of 27.5% He (Asplund et al., 2009), I varied the envelope's WMF from 0 to 1, distributing the remaining mass between H_2 and He, (M_{He} and $M_{\mathrm{H_2}}$, respectively):

$$M_{\mathrm{He}} = (1 - \mathrm{WMF}) \cdot 0.275, \quad M_{\mathrm{H_2}} = (1 - \mathrm{WMF}) \cdot 0.725$$

These mass fractions were then converted to mole counts via $n_i = \frac{M_i}{\mu_i}$, and used to compute the mole fractions $x_i = \frac{n_i}{\sum_j n_j}$. These mole fractions were then used to compute the mean molecular weight:

$$\mu_{\mathrm{atm}} = \sum_i x_i \cdot \mu_i$$

This process yielded a continuous mapping between WMF and MMW, enabling me to determine which WMF values would correspond to atmospheres consistent with TOI-270d's observed MMW range. The results are shown in Figure 15.

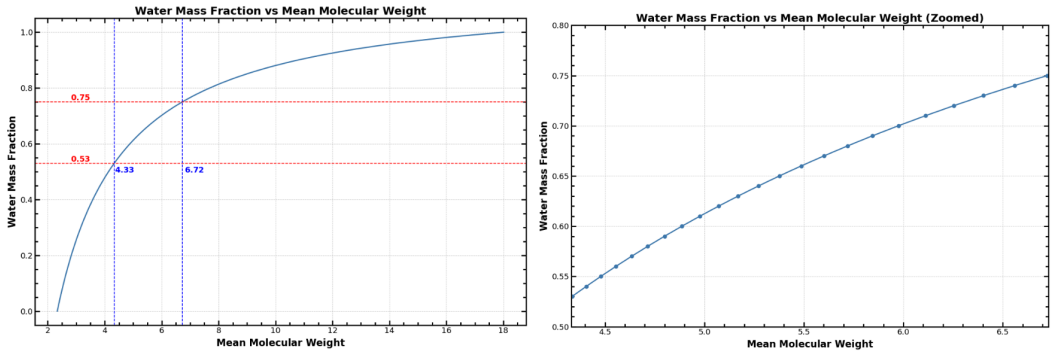

Figure 15: Relationship between atmospheric mean molecular weight (MMW) and water mass fraction (WMF). **Left:** Full curve from MMW = 2 to 18 g/mol, with red dashed lines marking WMF thresholds of 0.53 and 0.75. Vertical blue lines indicate the JWST-retrieved MMW range for TOI-270d (4.33 to 6.72 g/mol). **Right:** Zoomed-in view focusing on the observationally allowed MMW range. This panel reveals that TOI-270d's envelope water mass fraction must lie between approximately 0.53 and 0.75 to remain consistent with the measured MMW.

As shown in the figure 15, only models with envelope WMFs between approximately 0.53 and 0.75 are consistent with TOI-270d's observed atmospheric MMW. The steep slope of the MMW–WMF curve in this range highlights the diagnostic power of MMW as an observable: even small changes in MMW translate to significant differences in envelope composition. Note that this refers specifically to the envelope—not the total planetary water fraction, which must also account for the envelope mass fraction.

Using this mapping, I filtered a precomputed grid of models (originally sampled in WMF space) to retain only those with MMW values within the JWST-retrieved range of 4.33–6.72 g/mol. In this sense, the models were MMW-consistent by construction, even though MMW was not the primary sampling variable.

At this stage, I also explored a wide range of radiative–convective boundary pressures ($P_{\rm ad}$), including 0.1 bar, 1 bar, 10 bar, and 100 bar. However, these values were chosen arbitrarily and not yet constrained by the water phase diagram. As a result, some models permitted condensation in the envelope, potentially inflating the plausibility of high-water-fraction solutions.

Figure 16 shows how these WMF-sampled models populate the mass–radius space, including compositions that match TOI-270d's measured radius across a wide range of $P_{\rm ad}$ assumptions. While the figure demonstrates the flexibility of the WMF-based models, it also underscores the need for tighter thermal constraints.

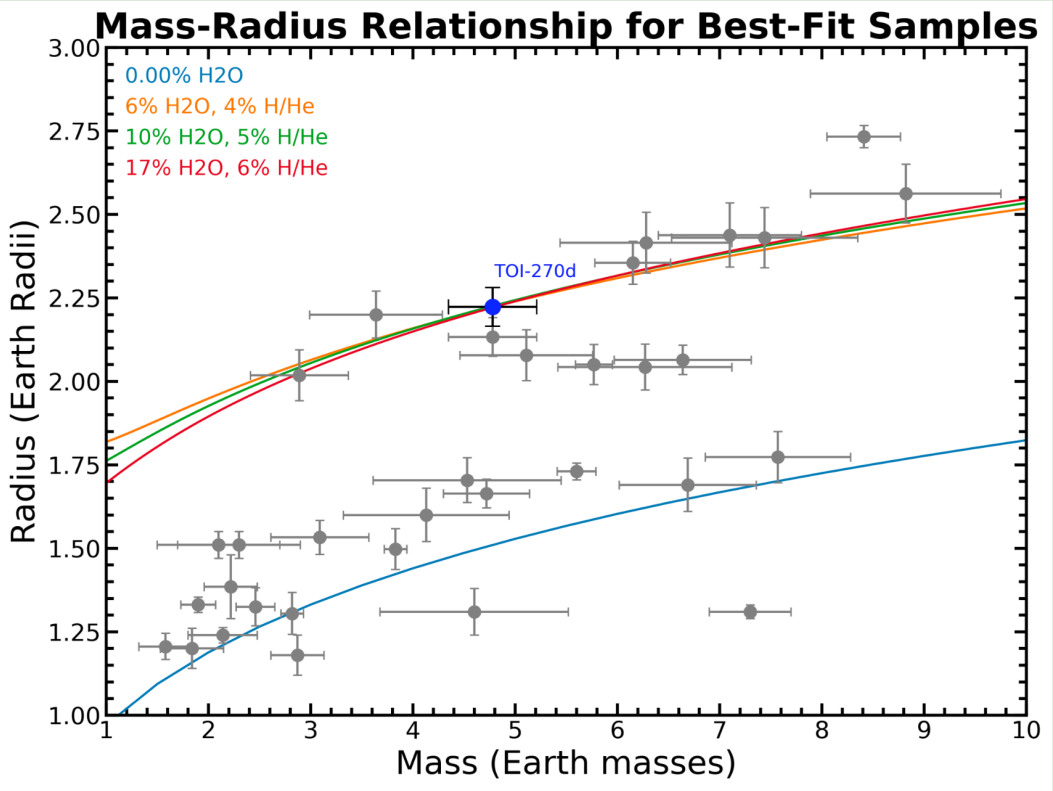

Figure 16: Mass–radius relationships for best-fitting compositions of TOI-270d, constrained to the observed MMW range ($\mu = 5.47^{+1.25}_{-1.14}$ g/mol). These models were generated before switching to MMW-based sampling and do not yet apply a condensation-aware cutoff on $P_{\rm ad}$.

Figure 17 further illustrates how the best-fitting envelope, water, and H/He mass fractions vary across the allowed MMW range. As MMW increases, both the water mass fraction and the overall envelope mass increase steeply, while the H/He fraction remains relatively low. These results reflect the compositional degeneracy present before applying condensation-aware filtering and motivate the need for further refinement in later modeling stages.

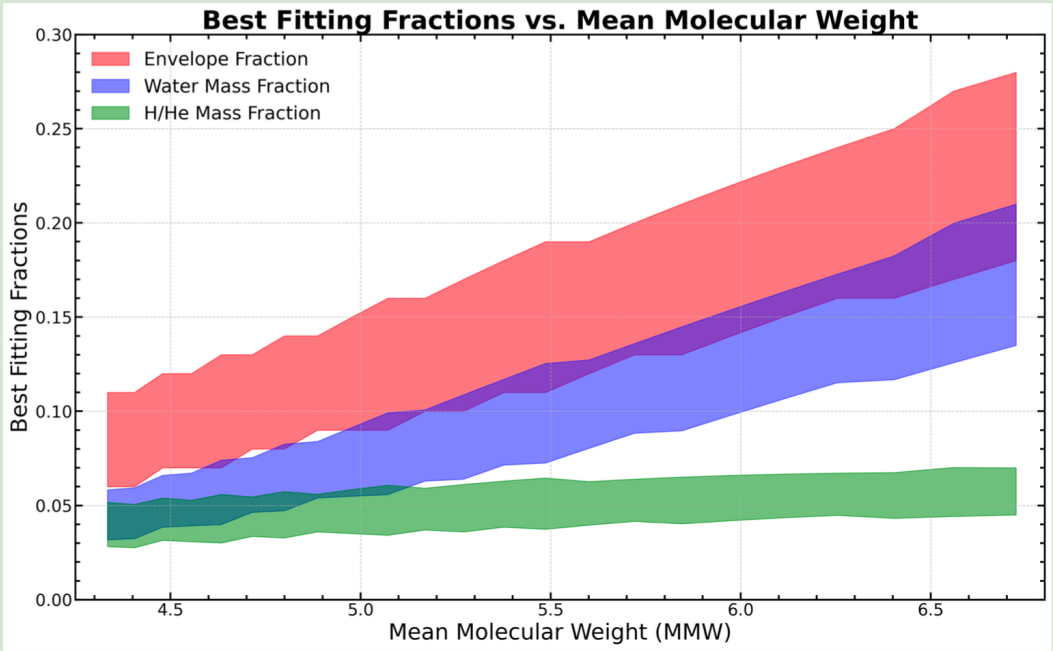

Figure 17: Best-fitting envelope, water, and H/He mass fractions as a function of MMW for TOI-270d. These results were obtained without restricting P_{ad} to values that prevent H_2O condensation, and therefore differ from the final, condensation-aware grid models.

Subsequent modeling phases addressed this shortcoming by imposing physically motivated upper limits on P_{ad}, using the planet's equilibrium temperature and the water phase curve to ensure that all models remained in the vapor or supercritical regime (see Section 2.2). This refinement, informed by studies such as Gupta et al. (2025), eliminated condensation-permitting models and brought the framework into full physical consistency with thermal expectations for close-in sub-Neptunes.

3.4.2 Final MMW-Based Modeling and Constraints on TOI-270d's Composition

Can interior structure modeling alone, without the aid of atmospheric constraints-yield meaningful insights into the bulk composition of a sub-Neptune exoplanet like TOI-270d? This question served as the foundation for my final modeling phase. By systematically exploring a broad grid of compositions, I tested how interior modeling alone could narrow down TOI-270d's possible structure.

To explore the full range of plausible interior compositions for TOI-270d, I generated models across a two-dimensional grid of envelope mass fraction (ranging from 0.01 to 1.0) and atmospheric mean molecular weight (MMW, ranging from 2.35 to 18). This spans compositions from pure H/He to pure water vapor. The grid was computed using a custom multiprocessing framework built on top of `SMILE` (see Section 2.3), enabling efficient evaluation of thousands of composition models.

For each MMW value, the corresponding envelope water mass fraction ($\mathrm{WMF_{env}}$) was computed using the prescription from Nixon et al. (2024):

$$x_{\mathrm{H_2O,env}} = \frac{\mu_{\mathrm{H_2O}}(\mu_{\mathrm{atm}} - \mu_{\mathrm{H/He}})}{\mu_{\mathrm{atm}}(\mu_{\mathrm{H_2O}} - \mu_{\mathrm{H/He}})} \tag{8}$$

where $\mu_{\mathrm{H_2O}} = 18.02$ g/mol, $\mu_{\mathrm{H/He}} = 2.34$ g/mol, and μ_{atm} is the sampled MMW.

The implied H/He and H_2O mass fractions were then scaled by the total envelope mass fraction to compute the total mass fraction of each volatile component for each model. All envelopes were assumed to be fully mixed and followed an isothermal–adiabatic temperature profile.

To identify physically plausible solutions, I evaluated each model using `SMILE` and applied the same chi square filtering criterion described in Equation 7, which requires the model's mass and radius to fall within the 1σ observational uncertainties of TOI-270d. Only models with $\chi^2 \leq 1$ were retained. Additionally, I fixed ($P_{\mathrm{ad}} = 1.32$), the highest value that avoids water condensation at TOI-270d's equilibrium temperature.

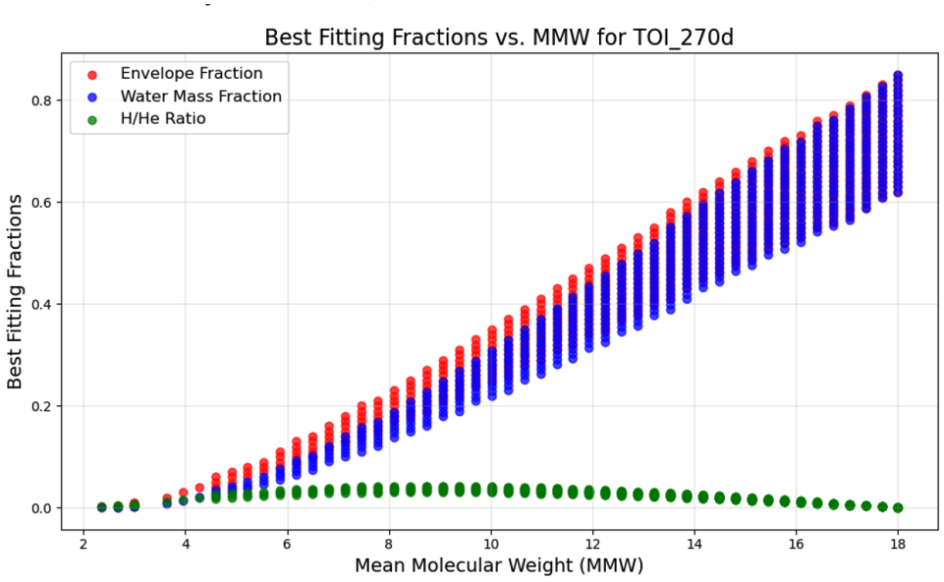

Figure 18: Best-fitting water mass fraction (blue), envelope mass fraction (green), and H/He mass fraction (red) for TOI-270d, plotted as a function of mean molecular weight (MMW). These models were generated using MMW-based sampling and filtered using the criteria listed above, but without applying the 50% WMF cutoff. The H/He Ratio in this plot represents the Hydrogen Helium Mass Fraction

Figure 18 illustrates how the envelope composition evolves across the sampled MMW grid. At low MMW values (near 2.35 g/mol), the envelope is composed almost entirely of hydrogen and helium, with negligible water content. As MMW increases, the required water mass fraction rises steeply to maintain consistency with TOI-270d's observed radius. These results demonstrate the steep degeneracy between H/He and H_2O in the mixed-envelope regime: many combinations can match the observed mass and radius, but only a subset are chemically and physically realistic.

To further constrain the solution space, I applied an upper limit on the planet's total water mass fraction, excluding all models exceeding 50%. Luque and Pallé (2022) This filtering step is motivated by formation and evolution considerations: extremely water-rich planets are likely rare in close-in orbits due to limited icy accretion and long-term atmospheric loss (Rogers and Owen, 2021).

Imposing this cutoff narrows the space of acceptable models, excluding water-

dominated atmospheres and setting an upper limit of $\mu_{\rm atm} \approx 15.4$. Figure 19 shows the distribution of best-fitting H/He mass fractions across the remaining MMW range.

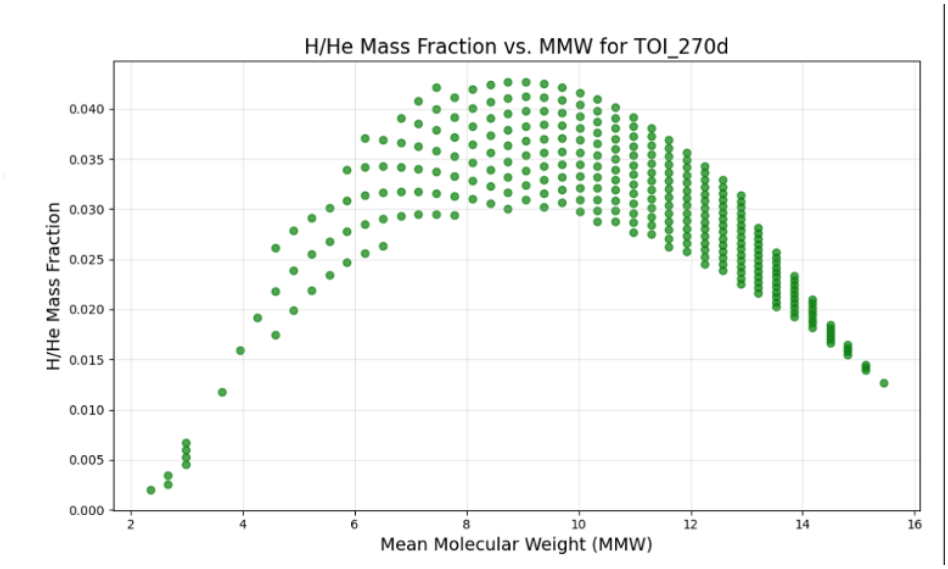

Figure 19: H/He mass fraction vs. mean molecular weight (MMW) for TOI-270d after applying a 50% WMF cutoff. The physically plausible region excludes water-dominated atmospheres (MMW $\gtrsim$ 15.4 g/mol) and reveals a peak in allowable H/He fraction near MMW = 9.1 g/mol.

After filtering, we find that the allowed MMW range is reduced to approximately 2.35–15.4 g/mol. The maximum allowable H/He mass fraction is 4.27%, occurring at MMW = 9.06 g/mol. The minimum viable H/He mass fraction is 0.2%, corresponding to MMW = 2.35 g/mol (a nearly pure H/He atmosphere). Solutions at low MMW (2.35–4.5 g/mol) are H/He-dominated, while those at intermediate MMW (6–12 g/mol) contain mixed envelopes with varying H/He–H_2O ratios. High-MMW models (MMW > 15.4 g/mol) are eliminated due to exceeding the water fraction limit.

The release of JWST spectra for TOI-270d then provided a concrete atmospheric constraint. Benneke et al. (2024) reported a measured mean molecular weight (in g/mol) of:

$$\mu_{\rm atm} = 5.47^{+1.25}_{-1.14}$$

This value allowed me to formally restrict the MMW sampling range to 4.33–6.72 g/mol and rerun the model grid within this window. Since the models had already been filtered for physical viability, the addition of this MMW constraint allowed for a much tighter constraint on TOI-270d's bulk composition. Only models within this MMW range—and consistent with observed mass, radius, and water condensation limits—were retained. We found that the total envelope mass fraction ranges from 8%-17%, with the H_2O fraction ranging from 5%-14%. This led to a narrow constraint on the H/He fraction of 2%-4%.

Even prior to the JWST atmospheric measurement, this detailed interior structure modeling placed an upper limit of MMW $\approx$ 15.4 g/mol, effectively ruling out highly water-dominated envelopes. The observed MMW of 5.47 ± 1.2 g/mol falls well within this predicted range, confirming that the planet lies in the moderate H/He regime. This demonstrates that interior structure modeling alone can yield meaningful composition constraints, even in the absence of atmospheric data.

This result—grounded in detailed interior structure modeling—provided the motivation to extend this approach to a population-level analysis. Even without atmospheric spectra, detailed interior structure modeling can place physically meaningful bounds on the bulk composition of sub-Neptunes. Then, when MMW measurements do become available (as with TOI-270d), they can be layered onto the model grid to further constrain the bulk composition of the target exoplanet.

TOI-270d emerged as a prototypical case study for the `SMILE` framework—guiding the structure modeling approach used across the 18 JWST Cycle 1–3 sub-Neptunes presented in Section 4. Unlike the low envelope mass fractions ($\sim$0.5–1%) proposed in early sub-Neptune formation models (Owen and Wu, 2013), TOI-270d exhibits a significantly more massive volatile envelope, with a constrained H/He fraction of 2–4% and a total water fraction as high as 14%. This finding confirms TOI-270d's status as a mixed-envelope sub-Neptune and provides direct constraints on its bulk water content. The elevated water fraction suggests that TOI-270d must have

formed or evolved in a way that enabled substantial water enrichment—whether through migration from beyond the snow line, reduced atmospheric escape, or water-rich accretion. More broadly, these results point to greater diversity in the atmospheric and interior structures of sub-Neptunes than previously assumed, motivating future studies into the physical mechanisms that govern water enrichment in volatile-rich planets.

4 Exoplanetary Population Study: Interior Structures of Sub-Neptunes

4.1 Method

Building on the detailed modeling of TOI-270d, I expanded the analysis to a broader population of sub-Neptune exoplanets. The goal of this population-level study is to explore the diversity of internal structures across planets with similar sizes but potentially very different compositions and thermal environments. All targets selected for this study are confirmed transiting sub-Neptunes with well-characterized masses and radii and are scheduled for atmospheric observations by JWST during Cycles 1 through 3.

This selection ensures that the targets are not only observationally accessible but also among the most promising candidates for connecting atmospheric data to interior structure. The final sample comprises 27 exoplanets, of which 20 have been fully analyzed with `SMILE` as of this writing. The remaining systems are pending analysis due to either missing parameters or shortage of time.

4.1.1 Sample Overview

The planets in this population span a wide range of equilibrium temperatures, from temperate environments (e.g., LHS-1140b and LP-791-18c) to more irradiated planets such as TOI-824b and WASP-47e. Their radii fall between 1.6 and 3.7 Earth radii, and their masses span from approximately 3 to 20 Earth masses. Bulk densities vary widely, reflecting underlying diversity in envelope composition and thickness (see Table 1 and Figure 20).

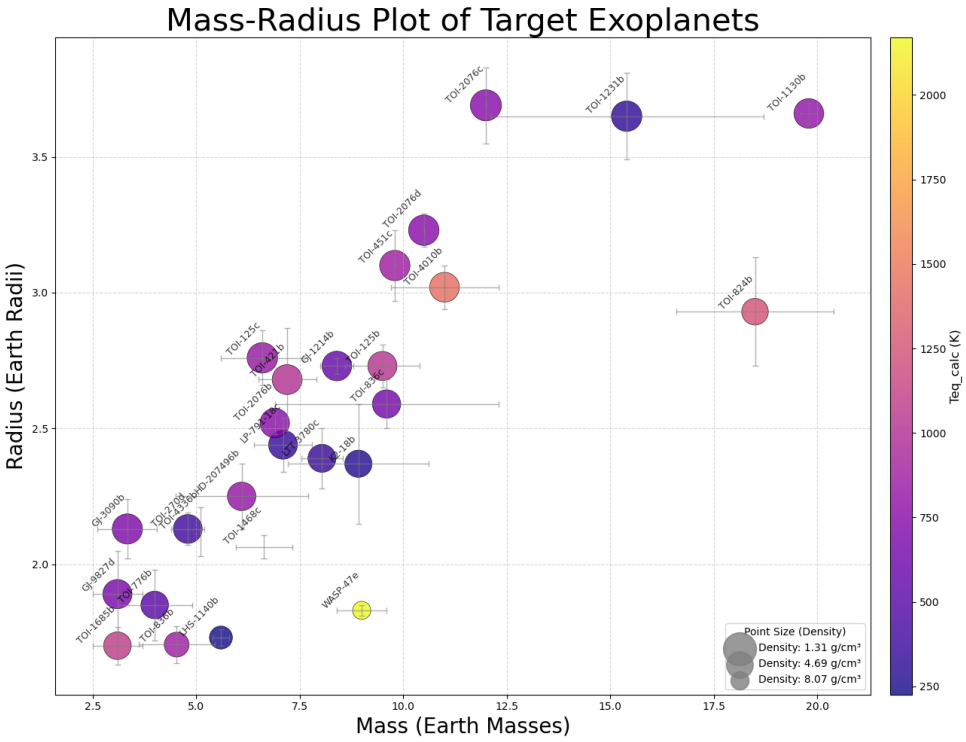

Figure 20: Mass–radius diagram for the 27 JWST Cycle 1–3 sub-Neptune targets. Color corresponds to the calculated equilibrium temperature, and point size scales with bulk density.

Figure 20 illustrates the diverse structural landscape occupied by sub-Neptunes. Some, like LHS-1140b and TOI-1685b, exhibit compact, high-density structures while others, like TOI-270d and TOI-1231b, have lower bulk densities. Notably, several targets occupy similar positions in the mass–radius diagram but differ significantly in equilibrium temperature or density, suggesting possible differences in atmospheric loss, envelope metallicity, or formation history.

Name	Mass ($M_\oplus$)	Radius ($M_\oplus$)	$T_{\rm eq}$ (K)	Cycle
TOI-836 b**	4.5 ± 0.9	1.70 ± 0.07	871 ± 36	1
LHS 1140 b**	5.6 ± 0.2	1.73 ± 0.03	226 ± 4	1
TOI-776 b**	4.0 ± 0.9	1.85 ± 0.13	514 ± 17	1
K2-18 b**	$8.92\ ^{+1.7}_{-1.6}$	2.37 ± 0.22	284 ± 15	1
LP 791-18 c**	7.1 ± 0.7	2.44 ± 0.10	324 ± 2	1
TOI-836 c**	$9.6\ ^{+2.7}_{2.5}$	2.59 ± 0.09	665 ± 27	1
TOI-421 b**	7.2 ± 0.7	$2.68\ ^{+0.19}_{-0.18}$	981 ± 16	1
GJ 1214 b**	8.4 ± 0.4	2.73 ± 0.03	567 ± 8	1
TOI-1685 b**	3.1 ± 0.6	1.70 ± 0.70	1069 ± 16	2
WASP-47 e**	6.8 ± 0.6	1.81 ± 0.03	2208 ± 40	2
GJ 9827 d**	3.0 ± 0.6	$1.89^{+0.16}_{-0.14}$	600 ± 17	2
TOI-1468 c	6.6 ± 0.7	2.06 ± 0.04	$338\ ^{+4}_{-3}$	2
GJ 3090 b	3.3 ± 0.7	2.13 ± 0.11	693 ± 18	2
TOI-270 d**	4.8 ± 0.4	2.13 ± 0.06	387 ± 10	2
LTT 3780 c	8.04 ± 0.5	$2.39\ ^{+0.10}_{-0.11}$	359 ± 10	2
TOI-125 b**	9.5 ± 0.9	2.73 ± 0.08	1037 ± 11	2
TOI-125 c**	6.6 ± 1.0	2.76 ± 0.10	828 ± 9	2
TOI-824 b**	$18.5\ ^{+1.8}_{-1.9}$	$2.93\ ^{+0.20}_{-0.19}$	$1253\ ^{+38}_{-37}$	2
TOI-1130 b**	19.3 ± 1.0	2.56 ± 0.13	632 ± 13	2
TOI-1231 b**	15.4 ± 3.3	$3.65\ ^{+0.16}_{-0.15}$	330 ± 4	2
TOI-4336 b	5.1*	$2.12^{+0.08}_{-0.09}$	308 ± 9	3
HD 207496 b**	6.1 ± 1.6	$2.25^{+0.12}_{-0.10}$	743 ± 26	3
TOI-2076 b	6.9*	2.52 ± 0.04	797 ± 12	3
TOI-4010 b	11.0 ± 1.3	3.02 ± 0.08	$1441\ ^{+14}_{-13}$	3
TOI-451 c	9.8*	3.10 ± 0.13	$875\ ^{+13}_{-11}$	3
TOI-2076 d	10.5*	3.23 ± 0.06	530 ± 8	3
TOI-2076 c	12.0*	3.50 ± 0.04	630 ± 9	3

Table 1: List of JWST Cycle 1–3 sub-Neptune targets selected for interior structure modeling. Columns show planetary name, mass, radius, equilibrium temperature, and JWST observation cycle. A total of 27 planets are listed, of which 18 (highlighted in the main analysis) were fully modeled in this study using the `SMILE` framework. Systems with incomplete or pending analysis are included here for completeness. Planets marked with $\star$* are those fully modeled in this study using the `SMILE` framework. The remaining systems are pending due to incomplete data or time constraints.

4.1.2 Modeling Procedure

Having defined the target sample, I applied a uniform interior modeling approach across all planets. Each planet was modeled with a differentiated interior (iron core, silicate mantle) and a fully mixed H/He–H_2O envelope, following an isothermal–adiabatic temperature profile anchored to the equilibrium temperature ($T_{\rm eq}$). These equilibrium temperatures were computed directly from stellar parameters using the expression in Equation 5. These calculated values were then used to interpolate the radiative–convective boundary pressure ($P_{\rm ad}$), ensuring thermal consistency across all targets.

To prevent water condensation at the top of the atmosphere, we constrained $P_{\rm ad}$ to remain within the vapor or supercritical regime for each planet's equilibrium temperature. This constraint was based on the phase diagram of H_2O (Figure 8), following the methodology of Nixon et al. (2024). For each target, we computed the maximum allowable $P_{\rm ad}$ such that the isothermal layer remained above the vapor–liquid boundary at the corresponding temperature.

For each planet, a grid of models was run, varying the total envelope mass fraction between 0.001 and 1.0, and scanning over mean molecular weights (MMWs) from 2.35 to 18 g/mol. The water mass fraction (WMF) corresponding to each MMW was computed using the relation Equation 5.

To constrain the modeled compositions, I filtered the outputs to include only models that matched the observed mass and radius of each planet within 1σ uncertainties. Additionally, I applied an upper limit of 50% on the total water mass fraction to be consistent with planet formation theory.

4.2 Results and Discussion

4.2.1 Exoplanetary Population Trends

Following the methodology outlined in Section 4, we analyzed the interior structure of 18 sub-Neptune exoplanets using the `SMILE` modeling framework. Each target was selected for having well-characterized mass and radius measurements and scheduled JWST atmospheric observations.

In this section, we present the population-level trends in hydrogen–helium envelope fractions, atmospheric mean molecular weights (MMWs), and composition degeneracies. For clarity, we first describe global correlations across the sample (e.g., between density and maximum H/He content), then highlight several case studies illustrating unique structural outcomes.

TOI-270d appears in both Table 2 and Table 3 for consistency with the rest of the population. Although its modeling is discussed in detail in Section 3, we include its key metrics here to enable direct comparison with other sub-Neptunes in the sample.

4.2.2 Hydrogen–Helium Mass Fractions

One of the main goals of this study was to constrain the maximum hydrogen–helium (H/He) mass fraction that each planet in our sample could retain while remaining consistent with observed mass and radius measurements.

The analysis reveals a wide diversity in atmospheric retention across the sub-Neptune population. The most H/He-rich planet in our sample is TOI-1130b, which supports a maximum hydrogen–helium (H/He) mass fraction of 13.18%, consistent with its large radius ($3.66\,R_\oplus$) and low-to-moderate density ($2.22\,\mathrm{g/cm^3}$). In contrast, high-density planets like TOI-836b and LHS-1140b place some of the tightest constraints on volatile retention, with H/He mass fractions limited to just

Name	Max H/He (%)	MMW @ Max H/He (g/mol)
WASP-47e	< 0.1	—
TOI-836b	0.57	11.61
LHS-1140b	0.62	11.29
TOI-1685b	0.80	11.29
K2-18b	2.20	10.97
TOI-125b	2.50	9.06
TOI-776b	3.02	9.38
GJ 9827d	3.58	9.6959
HD-207496b	3.98	9.38
TOI-270d	4.27	9.06
TOI-824b	5.27	7.4602
TOI-836c	6.38	8.10
LP 791-18 c	6.53	7.46
TOI-421b	6.96	8.74
TOI-125c	8.23	7.7796
GJ 1214b	8.43	7.46
TOI-1231b	10.76	8.10
TOI-1130b	13.18	6.502

Table 2: Maximum hydrogen–helium (H/He) mass fractions and associated mean molecular weights (MMW) before applying the 50% water mass fraction (WMF) constraint. These values represent the most radius-inflating compositions allowed under unconstrained modeling. The table is sorted in order of increasing H/He mass fraction.

Name	**Max H/He (%)**	**MMW @ Max H/He**	**Max MMW (g/mol)**
TOI-1130b	13.18	6.502	7.7796
TOI-125c	8.23	7.7796	10.9735
GJ 1214b	8.43	7.46	9.70
TOI-1231b	10.76	8.10	9.38
TOI-421b	6.96	8.74	14.17
LP 791-18 c	6.53	7.46	13.53
TOI-270d	4.27	9.06	15.44
TOI-836c	6.38	8.10	15.76
TOI-776b	3.02	9.38	18.00
LHS-1140b	0.62	11.29	18.00
K2-18b	2.20	10.97	18.00
HD-207496b	3.98	9.38	18.00
TOI-836b	0.57	11.61	18.00
TOI-125b	2.50	9.06	18.00
TOI-1685b	0.80	11.29	18.00
GJ 9827d	3.58	9.6959	18.00
TOI-824b	5.27	7.4602	18.00
WASP 47e	< 0.1	—	—

Table 3: Same as Table 2, but after applying the 50% WMF constraint. The final column shows the maximum allowable atmospheric MMW consistent with water-rich envelopes (WMF $\leq$ 50%). Planets with Max MMW = 18 g/mol are unconstrained by this filter.

0.57% and 0.62%, respectively.

These constraints are visualized in Figure 21, which shows strong anticorrelation between bulk density and maximum H/He mass fraction (left panel). Low-density planets like TOI-1231b and GJ 1214b accommodate up to 8–11% H/He, while compact, dense planets sharply restrict hydrogen-rich envelopes. The trend reinforces the interpretation that high-density planets have either lost their primordial atmospheres through photoevaporation or never accreted significant H/He during formation (Owen and Wu, 2013).

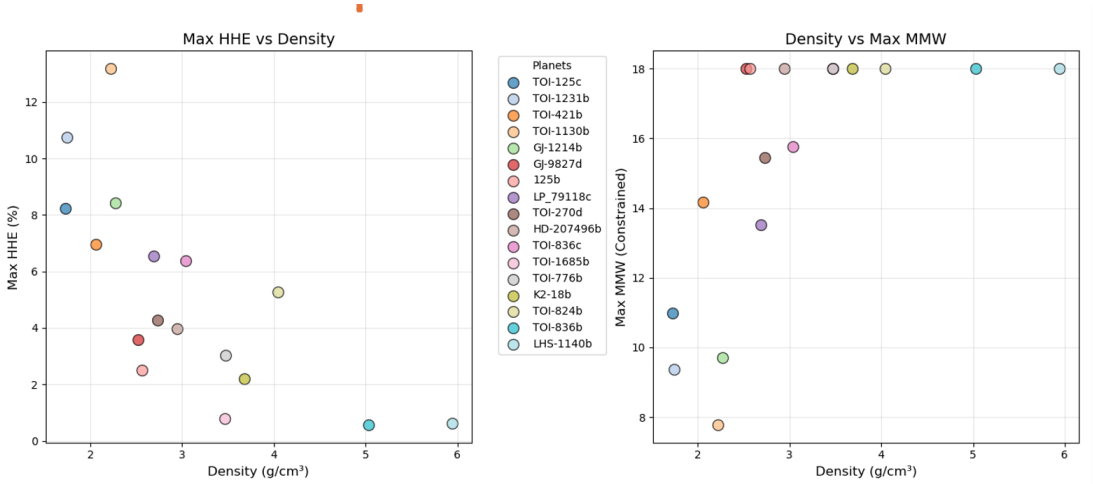

Figure 21: Left: Maximum hydrogen–helium (H/He) mass fraction versus bulk density. Right: Maximum constrained (at 50% wmf) mean molecular weight (MMW) as a function of bulk density. Together, these plots highlight the inverse relationship between hydrogen helium mass fraction and density across the sub-Neptune sample.

The right panel of Figure 21 further illustrates how the maximum constrained MMW increases with planetary density. While low-density planets can reach a compositional ceiling at MMW $\sim$ 9–10 g/mol, denser planets such as TOI-836b, TOI-1685b, and LHS-1140b allow MMW values up to 18 g/mol, even though the corresponding H/He content remains negligible. This combination of low H/He and high MMW offers a useful diagnostic: it implies that, if atmospheres exist at all, they are likely composed of heavier molecules (e.g., water, CO_2, or outgassed species) rather than hydrogen-dominated mixtures.

4.2.3 Maximum Mean Molecular Weight and Composition Degeneracy

In addition to constraining hydrogen–helium (H/He) mass fractions, our modeling framework also reveals how atmospheric mean molecular weight (MMW) shapes—and limits—plausible envelope compositions. For each planet, we computed the maximum MMW allowed before the total water mass fraction (WMF) exceeds 50%, our upper bound for physical viability.

Several planets—such as TOI-776b, TOI-824b, and K2-18b—remain unconstrained by this filter, with their maximum MMW pinned at 18 g/mol, the upper limit of our model grid. This suggests that even fully water-dominated envelopes remain consistent with observed mass and radius for these targets. While degenerate, these planets remain prime candidates for atmospheric characterization, which may rule out extreme compositions.

By contrast, other planets exhibit tighter constraints. When the maximum allowed MMW falls below 18 g/mol (e.g., in the range of 9–15 g/mol), high-MMW, water-dominated solutions are ruled out by the WMF cap. GJ 1214b and TOI-1231b, for instance, permit MMWs up to only 9.7 and 10.8 g/mol, respectively—beyond which water content becomes implausibly high. TOI-270d similarly shows a constrained maximum MMW of 15.4 g/mol, excluding the most water-rich cases.

Figure 21 visualizes this trend: denser planets tend to support higher maximum MMWs, since heavier molecular compositions are needed to reproduce small radii. However, this is not a simple one-to-one relationship. Planets like TOI-421b, with intermediate density, are more constrained than expected—likely due to thermal structure and radiative–convective boundary limits (i.e., P_{ad}) that further restrict the envelope.

These results show that, even in the absence of spectral retrievals, physically motivated MMW–WMF mappings enable us to isolate realistic atmospheric scenarios, rule out extreme compositions, and guide future observations with JWST and

beyond.

4.2.4 Thermal Limits and Water Mass Fractions

Beyond matching observed mass and radius, our interior models were filtered using two physically motivated constraints designed to eliminate unphysical or implausible configurations. These constraints—one thermal, one compositional—play complementary roles in shaping the range of allowable envelope structures.

First, we enforced a temperature-dependent cap on the radiative–convective boundary pressure ($P_{\rm ad}$), requiring that the top of the envelope remain in the vapor or supercritical phase. This constraint was imposed to prevent condensation of water near the upper atmosphere, following the phase data of H_2O. It primarily affects cooler planets, such as LP-791-18c and K2-18b, which can only support modest convective zones before intersecting the vapor–liquid boundary. As a result, the allowed H/He fractions for these planets are more conservative, but grounded in realistic thermal structure assumptions.

Second, we imposed a maximum total water mass fraction (WMF) of 50% to remain consistent with planet formation theory, which disfavors highly water-dominated compositions for close-in sub-Neptunes. This compositional filter proved particularly useful for differentiating plausible envelope scenarios. For instance, GJ 1214b and TOI-1231b allow mean molecular weights (MMWs) up to ∼9–10 g/mol while remaining within the WMF limit. In contrast, denser planets such as TOI-824b and HD-207496b support somewhat heavier envelopes (MMW ∼14 g/mol) without exceeding the cap.

Together, these thermal and compositional constraints help eliminate physically unrealistic models—such as those with condensed water layers or extreme volatile content—thereby refining the space of plausible envelope structures across the sub-Neptune population.

4.2.5 Case Studies and Edge Cases

While global trends reveal broad correlations between planetary properties and interior compositions, several individual systems in the sample stand out for their atypical structural characteristics. These edge cases help illustrate the physical limits of our modeling framework and highlight the diversity of sub-Neptune interiors beyond average trends.

TOI-270d: As detailed earlier (Section 3), TOI-270d lies near the center of the observed sub-Neptune distribution in both radius and temperature. Prior to the release of JWST spectra, interior modeling constrained its maximum MMW to approximately 15.4 g/mol, already ruling out the most water-rich configurations. The observed value of $5.47^{+1.25}_{-1.14}$ g/mol from transmission spectroscopy falls well within the predicted range. TOI-270d serves as a benchmark for validating MMW-based interior models and exemplifies how bulk properties can yield meaningful composition constraints, even in the absence of spectroscopic data.

K2-18b: K2-18b is one of the few planets in the sample whose atmosphere has been observed with JWST, revealing a H_2-dominated atmosphere containing CH_4 and CO_2 (Madhusudhan et al., 2023). However, the low temperature of this planet ($T_{\rm eq} \sim 284\,{\rm K}$) means that the assumption of a mixed envelope may not hold at low pressures, as H_2O may have condensed out of its upper atmosphere, resulting in a stratified structure (e.g., Benneke et al., 2024). It has even been suggested that this planet could host a liquid water ocean beneath its H_2-rich atmosphere (Madhusudhan et al., 2020; Nixon and Madhusudhan, 2021). However, our current observations of this planet can be explained either by a liquid ocean or by a mixed envelope (Shorttle et al., 2024; Wogan et al., 2024). Our work assumes a mixed envelope for the planet, and places an upper limit on the H/He mass fraction of 2.2%. If it is eventually determined that the planet is indeed stratified, follow-up work will be required to revise its bulk composition.

TOI-1130b: TOI-1130b supports one of the highest H/He mass fractions in the sample, with a maximum value of 13.2%. This is consistent with its large radius ($2.56\,R_{\oplus}$) and low-to-moderate density. The planet exemplifies a volatile-rich sub-Neptune in which a thick primordial atmosphere is retained. Its position near the low-density tail of the mass–radius diagram highlights the capacity for substantial hydrogen–helium envelopes in extended sub-Neptunes.

WASP-47e: WASP-47e is a compact sub-Neptune whose high density effectively excludes any significant volatile envelope. Interior models consistently return negligible H/He content, and no well-defined atmospheric mean molecular weight (MMW) can be assigned. These results support a scenario in which the planet has undergone complete atmospheric loss or formed with minimal gas accretion—consistent with expectations for photoevaporated or impact-stripped cores in high-irradiation environments (Rogers and Owen, 2021; Owen and Wu, 2013).

These examples illustrate that the relationship between density, temperature, and interior composition is nontrivial. While high-density planets tend to restrict volatile envelopes, some temperate, low-density planets can still accommodate high-MMW, water-rich solutions. Conversely, there exist targets where the 50% WMF constraint imposes meaningful upper bounds on MMW, excluding steam-dominated atmospheres. These findings reinforce that detailed structure modeling—including realistic equations of state, phase-aware thermal constraints, and envelope composition limits—is essential for distinguishing between degenerate solutions and isolating physically viable planetary interiors.

5 Conclusion

In this thesis, I developed and applied a physically motivated modeling method to understand the interior structure of sub-Neptune exoplanets. Sub-Neptunes are one of the most abundant exoplanets in the exoplanet population, without a

solar system analogue. Using the equations of planetary structure, i.e., the mass continuity equation and hydrostatic equilibrium equation, coupled with equations of state, I developed a simple planetary interior model, which I validated against published results. I subsequently transitioned to a more comprehensive model (`SMILE`, Nixon and Madhusudhan, 2021) which enabled detailed simulations of layered and mixed composition interiors with temperature–dependent equations of state.

The modeling considers different key parameters to make sure that our results are consistent with the measured mass and radius of the exoplanet being analyzed. By incorporating appropriate temperature profiles, taking into account the likely location of the radiative-convective boundary in the atmosphere, I constructed a modeling framework that analyzed thousands of model evaluations to find planetary interior parameters which align with the measured mass and radius of a given exoplanet.

TOI-270d served as the primary test for the modeling framework. I have modeled TOI-270d exploring pure water, layered, and fully mixed H/He/H_2O envelope scenarios, to demonstrate that detailed interior structure modeling alone can place meaningful constraints on the composition, even in the absence of spectroscopic constraints. I found that the maximum mean molecular weight (MMW) of TOI-270d is 15.44 g/mol. A subsequent measurement of the MMW by JWST found $\mu = 5.47^{+1.25}_{-1.14}$ g/mol, within the range suggested by my models. These JWST measurements in turn allowed for more accurate constraints on the interior structure.

Building on this foundation, I extended the analysis to a population of 18 sub-Neptune exoplanets selected for observation during the first 3 cycles of JWST observations. There is a clear relationship between bulk density and volatile content, showing that relatively low density planets can support up to 10-13% hydrogen-helium mass fractions, while high density planets are consistent with minimal

or no primordial gas. By filtering models with a 50% water mass fraction limit, I identified the most physically consistent interiors across the diverse sample of exoplanets. Additionally, we were able to recognize the exoplanets of which MMW is not affected by the 50% constraints as potential exoplanets with high likelihood of envelopes mostly consisting of heavier molecules rather than hydrogen-dominated mixtures.

Together, these results provide a roadmap for integrating interior and atmospheric data to constrain exoplanet compositions. As transmission spectroscopy yields a more precise measurement of the atmospheric MMW, the methods presented here will become very important for interpreting and constraining the bulk composition of exoplanets. Even without direct spectral detections, detailed interior structure modeling can significantly narrow the space of viable planetary compositions. In this way, interior structure modeling becomes not just a tool for interpreting observations, but a framework for guiding them.

References

Aguichine, A., Mousis, O., Deleuil, M., and Marcq, E. (2021). Thermal Evolution and Interiors of Water-rich Sub-Neptunes. *The Astrophysical Journal*, 914(2):84.

Anglada-Escudé, G., Amado, P. J., Barnes, J., Berdiñas, Z. M., Butler, R. P., Coleman, G. A. L., de La Cueva, I., Dreizler, S., Endl, M., Giesers, B., Jeffers, S. V., Jenkins, J. S., Jones, H. R. A., Kiraga, M., Kürster, M., López-González, M. J., Marvin, C. J., Morales, N., Morin, J., Nelson, R. P., Ortiz, J. L., Ofir, A., Paardekooper, S.-J., Reiners, A., Rodríguez, E., Rodríguez-López, C., Sarmiento, L. F., Strachan, J. P., Tsapras, Y., Tuomi, M., and Zechmeister, M. (2016). A terrestrial planet candidate in a temperate orbit around Proxima Centauri. *Nature*, 536(7617):437–440.

Asplund, M., Grevesse, N., Sauval, A., and Scott, P. (2009). The Chemical Composition of the Sun. *Annual Review of Astronomy and Astrophysics*, 47:481–522.

Benneke, B., Vats, A., Taylor, J., Lim, O., Kammerer, J., Claytor, Z. H., Roy, P.-A., Boley, A. C., Barstow, J., Tsiaras, A., Changeat, Q., Irwin, P., Radica, M., Angerhausen, D., Spake, J., Beatty, T. G., Fetherolf, T., Hinz, J., Lewis, N., Clampin, M., Pelletier, S., Hu, R., Seager, S., and Lafrenière, D. (2024). JWST Reveals CH_4, CO_2, and H_2O in a Metal-rich Miscible Atmosphere on a Two-Earth-Radius Exoplanet. *arXiv preprint arXiv:2403.03325*. Submitted to ApJ.

Borucki, W. J., Koch, D., Basri, G., et al. (2010). Kepler Planet-Detection Mission: Introduction and First Results. *Science*, 327:977–980.

Carroll, B. W. and Ostlie, D. A. (2007). *An Introduction to Modern Astrophysics*. Pearson Addison-Wesley, San Francisco, 2nd edition.

Cassan, A., Kubas, D., Beaulieu, J.-P., and et al. (2012). One or more bound planets per Milky Way star from microlensing observations. *Nature*, 481:167–169.

Chabrier, G., Mazevet, S., and Soubiran, F. (2019). New Hydrogen-Helium Equation of State for Giant Planet Interiors. *The Astrophysical Journal*.

Dressing, C. D. and Charbonneau, D. (2015). The Occurrence of Potentially Habitable Planets Orbiting M Dwarfs Estimated from the Full Kepler Dataset and an Empirical Measurement of the Detection Sensitivity. *The Astrophysical Journal*, 807(1):45.

Fischer, D. A., Anglada-Escudé, G., Arriagada, P., et al. (2016). State of the Field: Extreme Precision Radial Velocities. *Publications of the Astronomical Society of the Pacific*, 128(964):066001.

Fischer, D. A. and Valenti, J. (2005). The Planet-Metallicity Correlation. *The Astrophysical Journal*, 622(2):1102–1117.

Fulton, B. J., Petigura, E. A., Howard, A. W., and et al. (2017). The California-Kepler Survey. III. A Gap in the Radius Distribution of Small Planets. *The Astronomical Journal*, 154(3):109.

Grasset, O., Schneider, J., and Sotin, C. (2009). A Study of the Accuracy of Mass-Radius Relationships for Silicate-rich and Ice-rich Planets up to 100 Earth Masses. *The Astrophysical Journal*.

Gupta, A. and Schlichting, H. E. (2019). Sculpting the Valley in the Radius Distribution of Small Exoplanets as a by-product of Planet Formation: The Core-powered Mass-loss Mechanism. *Monthly Notices of the Royal Astronomical Society*, 487(1):24–33.

Gupta, A., Stixrude, L., and Schlichting, H. E. (2025). The Miscibility of Hydrogen and Water in Planetary Atmospheres and Interiors. *The Astrophysical Journal Letters*, 982(2):L35.

Günther, M. N. et al. (2019). A super-Earth and two sub-Neptunes transiting the nearby and quiet M dwarf TOI-270. *Nature Astronomy*, 3:1099–1108.

Howe, A. R., Burrows, A., and Verne, W. (2014). Mass-radius Relations and Core-envelope Decompositions of Super-Earths and Sub-Neptunes. *The Astrophysical Journal*.

Kempton, E. M.-R. and Knutson, H. A. (2024). Transiting Exoplanet Atmospheres in the Era of JWST. *arXiv preprint arXiv:2404.15430*. Accepted in Reviews in Mineralogy and Geochemistry (RiMG).

Kimura, T. and Ikoma, M. (2022). Retention of Water-rich Atmospheres by Sub-Earth-sized Planets Orbiting M Dwarfs. *Nature Astronomy*, 6:1296–1301.

Lee, E. J. and Chiang, E. (2015). To Cool is to Accrete: Analytic Scalings for Nebular Accretion of Planetary Atmospheres. *The Astrophysical Journal*, 811(1):41.

Lozovsky, M., Helled, R., Dorn, C., and Venturini, J. (2018). Interior Structure of Water-rich Super-Earths: Implications for GJ 1214b and HD 97658b. *The Astrophysical Journal*, 866(1):49.

Luque, R. and Pallé, E. (2022). Density, not radius, separates rocky and water-rich small planets orbiting M dwarf stars. *Science*, 377(6611):1211–1214.

Madhusudhan, N., Nixon, M. C., Welbanks, L., Piette, A. A. A., and Booth, R. A. (2020). The Interior and Atmosphere of the Habitable-zone Exoplanet K2-18b. *The Astrophysical Journal Letters*, 891(1):L7.

Madhusudhan, N., Piette, A. A. A., and Constantino, T. (2021). Habitability and Biosignatures of Hycean Worlds. *The Astrophysical Journal*, 918(1):L5.

Madhusudhan, N., Sarkar, S., Constantinou, S., Holmberg, M., Piette, A. A. A., and Moses, J. I. (2023). Carbon-bearing Molecules in a Possible Hycean Atmosphere. *The Astrophysical Journal Letters*, 956(1):L13.

Mayor, M. and Queloz, D. (1995). A Jupiter-mass companion to a solar-type star. *Nature*, 378:355–359.

Meadows, V. S. and Barnes, R. K. (2018). Factors Affecting Exoplanet Habitability. In Deeg, H. J. and Belmonte, J. A., editors, *Handbook of Exoplanets*, page 57. Springer.

Mikal-Evans, T., Madhusudhan, N., Dittmann, J., Günther, M. N., Welbanks, L., Van Eylen, V., Crossfield, I. J. M., Daylan, T., and Kreidberg, L. (2023). Hubble Space Telescope Transmission Spectroscopy for the Temperate Sub-Neptune TOI-270 d: A Possible Hydrogen-rich Atmosphere Containing Water Vapor. *The Astronomical Journal*, 165(3):84.

Mousis, O., Aguichine, A., Helled, R., Lunine, J. I., and Madhusudhan, N. (2020). The Role of Water in the Evolution of Super-Earths. *The Astrophysical Journal*.

Mulders, G. D. (2018). Exoplanet Demographics from Transit and Radial Velocity Surveys. In Deeg, H. J. and Belmonte, J. A., editors, *Handbook of Exoplanets*, page 153. Springer.

NASA Exoplanet Science Institute (2025). NASA Exoplanet Archive. Accessed on April 1, 2025.

NASA Science (2017). Accessed: 2025-04-06.

Nixon, M. C. and Madhusudhan, N. (2021). How deep is the ocean? Exploring the phase structure of water-rich sub-Neptunes. *Monthly Notices of the Royal Astronomical Society*, 505(3):3414–3432.

Nixon, M. C., Piette, A. A. A., Kempton, E. M.-R., Gao, P., Bean, J. L., Steinrueck, M. E., Mahajan, A. S., Eastman, J. D., Zhang, M., and Rogers, L. A. (2024). New Insights into the Internal Structure of GJ 1214b Informed by JWST. *The Astrophysical Journal Letters*, 970:L28.

Observatory, E. S. (2024). The ELT - Extremely Large Telescope. Accessed: 2025-04-01.

Owen, J. E. and Wu, Y. (2013). Kepler Planets: A Tale of Evaporation. *The Astrophysical Journal*, 775(2):105.

Pierrehumbert, R. T. (2023). A runaway greenhouse explanation for sub-Neptune radius inflation. *Nature Astronomy*, 7:1002–1010.

Quirrenbach, A., Reffert, S., and Bergmann, C. (2011). Giant Planets Around Giant Stars. In Schuh, S., Drechsel, H., and Heber, U., editors, *AIP Conference Proceedings*, volume 1331, pages 102–109. American Institute of Physics.

Ricker, G. R., Winn, J. N., Vanderspek, R., Latham, D. W., Bakos, G. Á., Bean, J. L., Berta-Thompson, Z. K., Brown, T. M., Buchhave, L., Butler, N. R., Butler, R. P., Chaplin, W. J., Charbonneau, D., Christensen-Dalsgaard, J., Clampin, M., Deming, D., Doty, J., De Lee, N., Dressing, C., Dunham, E. W., Endl, M., Fressin, F., Ge, J., Henning, T., Holman, M. J., Howard, A. W., Ida, S., Jenkins, J. M., Jernigan, G., Johnson, J. A., Kaltenegger, L., Kawai, N., Kjeldsen, H., Laughlin, G., Levine, A. M., Lin, D., Lissauer, J. J., MacQueen, P., Marcy, G., McCullough, P. R., Morton, T. D., Narita, N., Paegert, M., Palle, E., Pepe, F., Pepper, J., Quirrenbach, A., Rinehart, S. A., Sasselov, D., Sato, B., Seager, S., Sozzetti, A., Stassun, K. G., Sullivan, P., Szentgyorgyi, A., Torres, G., Udry, S., and Villasenor, J. (2015). Transiting Exoplanet Survey Satellite (TESS). *Journal of Astronomical Telescopes, Instruments, and Systems*, 1:014003.

Rogers, J. G. and Owen, J. E. (2021). Unveiling the planet population at birth. *Monthly Notices of the Royal Astronomical Society*, 503(1):1526–1542.

Rogers, J. G., Schlichting, H. E., and Owen, J. E. (2023). Conclusive evidence for a population of water-worlds around M-dwarfs remains elusive. *arXiv preprint arXiv:2301.04321*. Draft version April 4, 2023.

Seager, S. and et al. (2007). Mass-Radius Relationships for Solid Exoplanets. *The Astrophysical Journal*.

Shorttle, O., Jordan, S., Nicholls, H., Lichtenberg, T., and Bower, D. J. (2024). Distinguishing Oceans of Water from Magma on Mini-Neptune K2-18b. *The Astrophysical Journal Letters*, 962(1):L8.

Soubiran, F. and Militzer, B. (2015). Miscibility calculations for water and hydrogen in giant planets. *The Astrophysical Journal*, 806(2):228.

Spergel, D., Gehrels, N., Baltay, C., Bennett, D., and et al. (2015). Wide-Field InfrarRed Survey Telescope-Astrophysics Focused Telescope Assets WFIRST-AFTA Final Report. arXiv:1503.03757.

Thomas, S. and Madhusudhan, N. (2016). Thermal Effects on the Mass-Radius Relation of Water Worlds. *Astronomy & Astrophysics*.

Thorngren, D. P., Fortney, J. J., Murray-Clay, R. A., and Lopez, E. D. (2016). The Mass–Metallicity Relation for Giant Planets. *The Astrophysical Journal*, 831(1):64.

Thorngren, D. P., Marley, M. S., and Fortney, J. J. (2019). Exploring Planetary Internal Structure with Mass and Radius. *Research Notes of the AAS*, 3(11):128.

Turbet, M., Boukrouche, M., Gillmann, C., Bolmont, E., and Forget, F. (2020). The Water Cycle on TRAPPIST-1 Planets. *Space Science Reviews*.

Van Eylen, V., Astudillo-Defru, N., Bonfils, X., Livingston, J., Hirano, T., Luque, R., Lam, K. W. F., Justesen, A. B., Winn, J. N., Gandolfi, D., Nowak, G., Pallé, E., Albrecht, S., Dai, F., Campos Estrada, B., Owen, J. E., Foreman-Mackey, D., Fridlund, M., Korth, J., Mathur, S., Forveille, T., Mikal-Evans, T., Osborne, H. L. M., Ho, C. S. K., Almenara, J. M., Artigau, E., Barragán, O., Barros, S. C. C., Bouchy, F., Cabrera, J., Caldwell, D. A., Charbonneau, D., Chaturvedi, P., Cochran, W. D., Csizmadia, S., Damasso, M., Delfosse, X., De Medeiros, J. R., Díaz, R. F., Doyon, R., Esposito, M., Fűrész, G., Figueira, P., Georgiev, I., Goffo, E., Grziwa, S., Guenther, E., Hatzes, P. A., Jenkins, J. M., Kabath, P., Knudstrup, E., Latham, D. W., Lavie, B., Lovis, C., Mennickent, R. E.,

Mullally, S. E., Murgas, F., Narita, N., Pepe, A. F., Persson, C. M., Redfield, S., Ricker, G. R., Santos, N. C., Seager, S., Serrano, L. M., Smith, A. M. S., Suárez Mascareño, A., Subjak, J., Twicken, J. D., Udry, S., Vanderspek, R., and Zapatero Osorio, M. R. (2021). Masses and compositions of three small planets orbiting the nearby M dwarf L231-32 (TOI-270) and the M dwarf radius valley. *arXiv e-prints*.

Winn, J. N. (2010). Exoplanet Transits and Occultations. In Seager, S., editor, *Exoplanets*, pages 55–77. University of Arizona Press.

Wogan, N. F., Batalha, N. E., Zahnle, K. J., Krissansen-Totton, J., Tsai, S.-M., and Hu, R. (2024). JWST Observations of K2-18b Can Be Explained by a Gas-rich Mini-Neptune with No Habitable Surface. *The Astrophysical Journal Letters*, 963(1):L7.

Wright, J. T. and Gaudi, B. S. (2013). Exoplanet Detection Methods. In Oswalt, T. D., French, L. M., and Kalas, P., editors, *Planets, Stars and Stellar Systems. Volume 3: Solar and Stellar Planetary Systems*, page 489. Springer.

Youdin, A. N. (2011). Exoplanet demographics: planet formation and migration. *The Astrophysical Journal*, 742(1):38.

Appendix: Code Listings

The full code for the SMILE package can be found at:

https://github.com/mcnixon/smile

`grid_search.py` – Multiprocessing Framework

```
1
2  import sys
3  import os
4  import logging
5  import pandas as pd
6  import numpy as np
7  import time
8  from datetime import datetime
9  import multiprocessing as mp
10 from scipy.interpolate import interp1d
11 import smile
12
13 # Path to SMILE Package
14 sys.path.insert(0, "/Users/biruknardos/a_UMD_Research/smile")
15
16 # Setup logging to see progress on the Jupyter notebook
17 logging.basicConfig(level=logging.INFO, format='%(message)s',
       handlers=[
18      logging.FileHandler("Parallel_grid.log"),
19      logging.StreamHandler(sys.stdout)
20 ])
21
22 # Function that gives insight about what pad values to use
23 """
24 - The function reads in the target file from target path that will
       be provided
25 - The user provides the path, the codes checks if it is csv or
      excel and reads in the data
```

```
- From the data it get the temprature equlibrium (teq_val) of the planet that the user is interested in
- Taking the phase diagram of h20 it will math the teq to the pad to get the max pad that could like in the
- phase line without becoming liquid. Then it will tell the user the advised pad max value
-
"""

# Load liquid-vapor data from the txt file

def load_phase_data(file_path):
    """
        Load the phase boundary data from the liquid_vapour_bd.txt file

        Args:
        - file_path (str): Path to the liquid_vapour_bd.txt file.

        Return:
        - DataFrame with temperature and pressure boundaries
    """

    phase_data = pd.read_csv(file_path, delim_whitespace=True, header=None, names=["Pressure_Pa", "Temprature_K"])
    return phase_data

# Function to calculate max Pad (pressure) given an equilibrium temprature
# Automatically load phase data when the module is imported

PHASE_DATA = load_phase_data('/Users/biruknardos/a_UMD_Research/General/liquid_vapour_bd.txt')


```

```
54 def find_max_pad(Teq_val, extrapolate=False):
55     """
56     Find the maximum allowable Pad (pressure) for a given
           equilibrium temperature (Teq).
57
58     Args:
59     - Teq_val (float): The equilibrium temperature of the planet (
           in K).
60     - extrapolate (bool): If True, allows extrapolation for Teq
           values outside data range.
61
62     Returns:
63     - max_pad (float): The maximum pressure (Pad) in Pa before
           transitioning to vapor.
64     """
65
66     # Set up interpolation function
67     interp_fun = interp1d(
68         PHASE_DATA["Temprature_K"],
69         PHASE_DATA["Pressure_Pa"],
70         fill_value="extrapolate" if extrapolate else None,
71         bounds_error=not extrapolate
72     )
73
74     # Check if the Teq_val is within the data range
75     min_temp, max_temp = PHASE_DATA["Temprature_K"].min(),
           PHASE_DATA["Temprature_K"].max()
76
77     if Teq_val < min_temp or Teq_val > max_temp:
78         if not extrapolate:
79             print(f"Error: Teq value {Teq_val} K is outside the
                   interpolation range ({min_temp} K - {max_temp} K).
                    Set extrapolate=True to see an extrapolated
                   result.")
80             return None
```

```
        else:
            print(f"Warning: Teq value {Teq_val} K is outside the
                data range. Result will be extrapolated.")

    # Calculate max pad using the interpolated function
    max_pad = interp_fun(Teq_val)

    max_pad_bar = max_pad / 1e5
    print(f"The maximum Pad value for Teq = {Teq_val} K is {
        max_pad:.3f} Pa or {max_pad_bar:.4f} bar.")

    if Teq_val < min_temp or Teq_val > max_temp:
        print(f"Note: This value is extrapolated. The maximum
            reliable Teq range is {min_temp} K to {max_temp} K.")

    return max_pad  # Return Max pad in Pa

# Function to calculate WMF from MMW (Nixon et al., 2024)
def derive_wmf(mmw, h2_mw=2.016, he_mw=4.003, h2o_mw=18.015):

    """
    Derives the Water mass fraction (WMF) from the given MMW using
        formula from Nixon et al., 2024
    """

    he_maf = 0.275      # 27.5% of H/He is helium
    h2_maf = 1 - he_maf # The rest is hydrogen

    # The average molecular weight of the H/He mixture
    total_hhe = h2_maf/h2_mw + he_maf/he_mw

    h2_mof = (h2_maf/h2_mw) / total_hhe
    he_mof = (he_maf/he_mw) / total_hhe

    # Hydrogen/Helium
```

```
    hhe_mw  = h2_mof*h2_mw + he_mof*he_mw

    # Using the equation from Nixon et al., 2024
    wmf = (h2o_mw * (mmw-hhe_mw))/ (mmw * (h2o_mw - hhe_mw))

    return wmf

def calc_mmw(h2_maf, he_maf, h2o_maf):
    h2_mw = 2.016
    he_mw = 4.003
    h2o_mw = 18.015

    # Convert mass fractions to mole fractions
    total_moles = h2_maf/h2_mw + he_maf/he_mw + h2o_maf/h2o_mw

    if total_moles == 0 or np.isnan(total_moles):
        return np.nan  # Return NaN to indicate an invalid case

    h2_mof = (h2_maf/h2_mw) / total_moles
    he_mof = (he_maf/he_mw) / total_moles
    h2o_mof = (h2o_maf/h2o_mw) / total_moles

    # Calculate MMW using mole fractions
    mmw = h2_mof * h2_mw + he_mof * he_mw + h2o_mof * h2o_mw

    return mmw

def validate_mmw(x_env_w, x_env_g):
    """
    Validate teh MMW by recalculating it from x_env_w (Water mass
       fraction) and
    x_env_g (H/He mass fraction)
    """

    # Split the H/He into He and H2
```

```
146     he_maf = 0.275 * x_env_g
147     h2_maf = x_env_g - he_maf
148     h2o_maf = x_env_w
149 
150     # Compute MMW
151     mmw = calc_mmw(h2_maf, he_maf, h2o_maf)
152     return mmw
153 
154 def error_calc(file_path, mass_val, mass_unc, rad_val, rad_unc,
        pname, save_files=True):
155     """
156     This function calculates chi-squared error for a given dataset
            based on pre-calculated MMW.
157 
158     Parameters:
159     - df: DataFrame containing the full grid of models (from the
            grid search), including the MMW
160     - mass_val: observed mass
161     - mass_unc: uncertainty in mass
162     - rad_val: observed radius
163     - rad_unc: uncertainty in radius
164     - pname: name of the planet or specific dataset, used for
            filenames
165     - save_files: If True, saves CSV files for both full data and
            filtered data (default: True)
166 
167     Returns:
168     - filtered_df: DataFrame filtered by error <= 1 sigma.
169     """
170 
171     df = pd.read_csv(file_path)
172 
173     # Calculate error metric (chi-squared)
174     df['mass_diff'] = df['Mass'] - mass_val
175     df['radius_diff'] = df['Radius'] - rad_val
```

```
176     df['error'] = ((df['mass_diff'] / mass_unc)**2 + (df['
            radius_diff'] / rad_unc)**2)
177 
178     if save_files:
179         # Save all data (with error)
180         all_data_file = f"all_data_error_{pname}_{datetime.now().
                strftime('%Y%m%d_%H%M%S')}.csv"
181         df.to_csv(all_data_file, index=False)
182         print(f"All data with error saved to: {all_data_file}")
183 
184     # Filter the results where error <= 1
185     filtered_df = df[df['error'] <= 1].copy()
186     filtered_df = filtered_df.sort_values('error').reset_index(
            drop=True)
187 
188     if save_files:
189         # Save the filtered results (with error) to a CSV
190         filtered_file = f"filtered_with_error_{pname}_{datetime.
                now().strftime('%Y%m%d_%H%M%S')}.csv"
191         filtered_df.to_csv(filtered_file, index=False)
192         print(f"Filtered data saved to: {filtered_file}")
193 
194     print(f"Number of rows with error <=1: {len(filtered_df)}")
195 
196     return filtered_df
197 
198 
199 def single_run(m, pad, mmw, env_fraction, temp):
200 
201     # Skip calculation if env_fraction is too small or zero
202     """
203     - The envelope fraction (env_fraction) represents the fraction
            of the planet's mass that is made up of the gaseous
            envelope,
204     which includes water (H2O), hydrogen (H2), and helium (He).
```

```
    - If the env_fraction is 0, both the water mass fraction (
       x_env_w) and the H/He mass fraction (x_env_g) must also be
        0.
    This is because the envelope doesn't exist, so these
       components cannot contribute to the planet's mass.
    - Instead of calculating the radius using the SMILE package,
       the code skips the radius calculation and sets radius_full
        to 0 directly.
    """
    if env_fraction == 0:
        x_env_w = 0
        x_env_g = 0
        radius_full = 0  # Avoid calling smile.get_radius if
           envelope fractions are zero
        logging.info(f"Skipping radius calculation for mass={m},
           pad={pad}, env_frac={env_fraction:.2f}, Radius={
           radius_full}")
    else:
        # Calculate x_env_w (Water mass fraction) from the MMW
        x_env_w = derive_wmf(mmw) * env_fraction  # Water mass
           fraction in the whole planet
        x_env_g = env_fraction - x_env_w  # H/He mass fraction in
           the whole planet

        # Skip calculation if x_env_g > 0.3
        if x_env_g > 0.3:
            logging.info(f"Skipping calculation for H/He Mass
               fraction={x_env_g:.2f}, which is above 30%")
            return None

        # Validate the calculated MMW
        if x_env_w + x_env_g == 0:
            mmw_calculated = np.nan
        else:
            mmw_calculated = validate_mmw(x_env_w, x_env_g)
```

```
229
230         logging.info(f"MMW Validation: Original MMW={mmw:.3f},
                Calculated MMW={mmw_calculated:.3f}")
231
232         # Only calculate radius if there's a non-zero envelope and
                H/He ratio is within the limit
233
234         if x_env_w + x_env_g > 0 and x_env_g <0.31:
235             # Calculate radius using smile
236             radius_full = smile.get_radius(mass=m, P0=1e3, T0=temp
                    , Pad=pad, x_g=x_env_g, x_w=x_env_w, mixed=True)
237
238             if isinstance(radius_full, list) or radius_full is
                    None:
239                 radius_full = 0
240         else:
241             radius_full = 0
242
243         logging.info(f"Calculated for mass={m}, pad={pad}, MMW={
                mmw:.3f}, WMF={x_env_w}, H/He={x_env_g}, env_frac={
                env_fraction:.2f}, Radius={radius_full}")
244
245     row_data = {
246         'Mass': m,
247         'Radius': radius_full,
248         'Pad': pad,
249         'WMF': x_env_w,
250         'H/He': x_env_g,
251         'Env_Fraction': env_fraction,
252         'MMW': mmw
253     }
254
255     return row_data
256
257
```

```
258 # Use parallel Processing to run the grid model
259 def parallel_run(mass_val, mass_unc, rad_val, rad_unc, teq_val,
    pad_list, env_fractions, mmw_list=None, pname="PlanetName"):
260     data = []
261 
262     masses = np.linspace(mass_val - mass_unc, mass_val + mass_unc,
        10)
263 
264     # If no mmw_list is provided, default to MMW range from 2 to
        18
265     if mmw_list is None:
266         mmw_list = np.linspace(2.35,18,50)  # Default range of MMW
            values
267 
268     # Generate arguments (without executing)
269     args = [
270         (m, pad, mmw, env_fraction)
271         for m in masses
272         for pad in pad_list
273         for mmw in mmw_list
274         for env_fraction in env_fractions
275     ]
276 
277     # Count valid combinations by skipping unwanted cases
278     valid_combinations = 0
279     for m, pad, mmw, env_fraction in args:
280         if env_fraction == 0:
281             continue  # Skip if env_fraction is 0
282         x_env_w = derive_wmf(mmw) * env_fraction
283         x_env_g = env_fraction - x_env_w
284         if x_env_g > 0.31:
285             continue  # Skip if H/He fraction > 0.3
286         valid_combinations += 1
287 
288     # Log valid combinations and calculate time estimate
```

```
289     logging.info(f"Total combinations generated: {len(args)}")
290     logging.info(f"Valid combinations after skipping: {
            valid_combinations}")
291
292     # Set base timing
293     base_combinations = 10000
294     base_time_in_hours = 0.4167  # 25 minutes in hours
295
296     # Estimate time based on valid combinations
297     num_cores = mp.cpu_count()
298     cores_to_use = num_cores - 1
299     estimated_time = (valid_combinations / base_combinations) *
            (10 / cores_to_use) * base_time_in_hours
300
301     logging.info(f"SMILE :) Starting parallel computation with {
            valid_combinations} combinations, Using {cores_to_use} CPU
             cores, Estimated total time for computation: {
            estimated_time:.2f} hours")
302
303     # Start parallel computation
304     start_time = time.time()
305     with mp.Pool(cores_to_use) as pool:
306         results = pool.starmap(single_run, [(m, pad, mmw,
                env_fraction, teq_val) for m, pad, mmw, env_fraction
                in args if env_fraction != 0 and (env_fraction -
                derive_wmf(mmw) * env_fraction) <= 0.3])
307
308     # Processing and saving results
309     for result in results:
310         if result is not None:
311             data.append(result)
312
313     result_file = f'{pname}_Model_grid_full_{datetime.now().
            strftime("%Y%m%d_%H%M%S")}.csv'
314     df_full = pd.DataFrame(data)
```

```
315     df_full.to_csv(result_file, index=False)
316
317     # Log runtime
318     total_runtime = time.time() - start_time
319     logging.info(f"Total runtime: {total_runtime / 60:.2f} minutes
            ({total_runtime / 3600:.2f} hours)")
320
321     return result_file
322
323
324 def specific_run(m, pad, x_env_w, x_env_g, temp):
325
326     # Ensure the envelope fractions are valid
327     env_frac = x_env_w + x_env_g
328     if env_frac > 1:
329         logging.warning(f"Warning: Envelope fraction exceeds 1.
                Total env_frac = {env_frac}")
330
331     # Calculate radius using SMILE with provided x_env_w and
            x_env_g
332     radius_full = smile.get_radius(mass=m, P0=1e3, T0=temp, Pad=
            pad, x_g=x_env_g, x_w=x_env_w, mixed=True)
333
334     if isinstance(radius_full, list) or radius_full is None:
335         radius_full = 0
336
337     logging.info(f"Calculated for mass={m}, pad={pad}, x_env_w={
            x_env_w:.3f}, x_env_g={x_env_g:.3f}, Radius={radius_full}"
            )
338
339     row_data = {
340         'Mass': m,
341         'Radius': radius_full,
342         'Pad': pad,
343         'WMF': x_env_w,
```

```
        'H/He': x_env_g,
        'Env_Fraction': env_frac  # Total fraction for reference
    }

    return row_data

def specific_parallel_run(mass_list, rad_val, rad_unc, teq_val,
    pad_list, xw_list, xg_list, pname, save_files=True):

    data = []

    # Ensure that the input lists have the same length
    if not (len(xw_list) == len(xg_list) == len(pad_list) == len(
        mass_list)):
        raise ValueError("All input lists must have the same
            length!")

    # Create the list of arguments for each combination
    args = [
        (mass, pad, x_env_w, x_env_g, teq_val)
        for mass, pad, x_env_w, x_env_g in zip(mass_list, pad_list
            , xw_list, xg_list)
    ]

    logging.info(f"SMILE :) Starting specific grid computation
        with {len(args)} combinations...")

    # Use multiprocessing to compute the results
    with mp.Pool(mp.cpu_count() - 1) as pool:
        results = pool.starmap(specific_run, args)

    logging.info("Specific grid computation completed, processing
        results...")

    for result in results:
```

```
373             if result is not None:
374                 data.append(result)
375
376         # Save the results with a timestamp in the filename
377         result_file = f'{pname}_Specific_grid_{datetime.now().strftime
                ("%Y%m%d_%H%M%S")}.csv'
378
379         df = pd.DataFrame(data)
380
381         # Check for save_files flag before saving
382         if save_files:
383             result_file = f'{pname}_Specific_grid_{datetime.now().
                    strftime("%Y%m%d_%H%M%S")}.csv'
384             df.to_csv(result_file, index=False)
385             logging.info(f"Results saved to {result_file}")
386             return result_file
387         else:
388             return df  # Return the DataFrame directly if not saving
389
390 def run_grid_search(planet_name, file_path, pad_list,
        env_fractions, mmw_list=None,teq_val=None):
391     # Log the start time
392     start_time = datetime.now()
393     logging.info(f"Grid search started at: {start_time.strftime('%
            Y-%m-%d %H:%M:%S')}")
394
395     # Track start time in seconds for calculating total runtime
396     time_start = time.time()
397
398     # Get the file extension
399     _, file_extension = os.path.splitext(file_path)
400
401     # Read the file based on its extension
402     if file_extension == '.xlsx' or file_extension == '.xls':
403         planets_df = pd.read_excel(file_path)
```

```
404     elif file_extension == '.csv':
405         planets_df = pd.read_csv(file_path)
406     else:
407         raise ValueError("Unsupported file format, Provide Excel
                or CSV file")
408 
409     planet_data = planets_df[planets_df['Name'] == planet_name]
410 
411     planet_data = planet_data.iloc[0]
412 
413     mass_val = planet_data['Mass_value']
414     mass_unc = planet_data['Mass_unc']
415     rad_val = planet_data['Radius_value']
416     rad_unc = planet_data['Radius_unc']
417 
418     # I added this because it helps me in testing for TOI-270d,
        but providing it is not mandatory
419 
420     if teq_val is None:
421         teq_val = planet_data['Teq_calc']
422 
423     result_file = parallel_run(mass_val, mass_unc, rad_val,
        rad_unc, teq_val, pad_list, env_fractions, mmw_list,
        planet_name)
424 
425     # Log the end time
426     end_time = datetime.now()
427     logging.info(f"Grid search finished at: {end_time.strftime('%Y
        -%m-%d %H:%M:%S')}")
428 
429     # Calculate and log total runtime
430     total_runtime = time.time() - time_start
431     logging.info(f"Total runtime: {total_runtime:.2f} seconds ({
        total_runtime/60:.2f} minutes)")
432 
```

```
    return result_file


# Main execution block

if __name__ == "__main__":
    logging.info("This script is intended to be imported into
        Jupyter Notebook or other script. It doesn't calculate
        error")
```

`general.py` – Analysis and Utilities

```
# from astropy import units as u
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy.ndimage import gaussian_filter1d


# Function calculating Equilibrium Temperature
def Calculate_Teq(file_path, planet_name, albedo=0, f=1):
    """
    Calculate equilibrium temperature with given file_path and
        planet name
    """

    data = pd.read_csv(file_path)
    planet_index  = data[data['Name'] == planet_name].index[0]

    # Extract the values
    T_star = data.loc[planet_index, 'T_star_(K)']
    T_star_unc = data.loc[planet_index, 'T_star_unc']

```

```
    R_star = data.loc[planet_index, 'R_star_(solar_radii)'] *
        6.957e+8  # Convert solar radii to m
    R_star_unc = data.loc[planet_index, 'R_star_unc'] * 6.957e+8
        # Convert solar radii to m

    a = data.loc[planet_index, 'sm_axis_(au)'] * 1.496e+11
    Teq_nasa = data.loc[planet_index, 'Teq(nasa_arch)']

    # Calculating R_star / a with geometric factor
    Rd = R_star / (2 * a)

    # Calculate Teq with heat redistribution factor
    Teq = T_star * np.sqrt(Rd) * ((1-albedo)*f) ** 0.25

    # Calculating Uncertainty
    unc_rd = np.sqrt((R_star_unc / R_star)**2) * Rd
    unc_tp = np.sqrt((T_star_unc / T_star)**2 + (0.25*unc_rd/Rd)
        **2)
    Tp_unc = Teq * unc_tp

    # Calculate the difference from NASA Archive value
    Teq_diff = Teq - Teq_nasa

    # Print Results
    print(f"\nCalculated Teq for {planet_name}: {Teq:.2f} K")
    print(f"Uncertainty in Teq: +/-{Tp_unc:.2f} K")
    print(f"NASA Archive Teq: {Teq_nasa} K")
    print(f"Difference: {Teq_diff:.2f} K")

    # Update the DataFrame with the Calculated Values
    data.at[planet_index, 'Teq_calc'] = round(Teq, 4)
    data.at[planet_index, 'Teq_calc_unc'] = round(Tp_unc, 4)
    data.at[planet_index, 'Teq_diff_nasa'] = round(Teq_diff, 4)

    # Save file
```

```
    data.to_csv(file_path, index=False)
    print(f"Updated file saved at {file_path}")

    return Teq, Tp_unc, Teq_diff


def filter_pad(file_path: str, pad_value: float, pname: str):
    """
    Filters the data by a specified Pad value and saves the result
        to a new CSV file.

    Parameters:
    - file_path: str, the path to the CSV file with all Pad values
       .
    - pad_value: float, the Pad value in bars to filter by.
    - pname: str, a specific name or identifier to include in the
       output file name.
    """

    # Convert pad_value from bars to Pascals
    pad_value_pa = pad_value * 1e5

    # Read the input file
    df = pd.read_csv(file_path)

    # Filter by the specified Pad value
    df_filtered = df[df['Pad'] == pad_value_pa]

    # Check if there are any rows with the specified Pad value
    if df_filtered.empty:
        print(f"No data available for Pad value: {pad_value} bar")
    else:
        # Create the output file name
        output_file_path = f"filtered_with_error_pad{pad_value}_{
           pname}.csv"
```

```
84
85          # Save the filtered data to the output file
86          df_filtered.to_csv(output_file_path, index=False)
87          print(f"Filtered data for Pad = {pad_value} bar saved to
                '{output_file_path}'")
88
89  # Function to plot best fitting vs MMW
90  def analyze_results(file_path: str, pname: str, save_plot: bool =
        True, ylim: tuple = (0, 0.15)):
91      """
92      - The function takes a csv file path that contains the grid
            models with calculated error and is
93        filtered by one sigma error.
94      - Plots Best fitting vs MMW (Scatter plot)
95      - Plots Best fitting H/He vs MMW (Scatter plot)
96
97      Parameters:
98      - file_path: Path to the CSV file with the filtered data
99      - pname: Name of the planet to include in the plot title and
            saved filenames
100     - save_plot: If True, saves the plots as PNG files (default:
            True)
101     - ylim: Tuple to set y-axis limits for the "Best Fitting H/He
            vs MMW" plot (default: (0, 0.15))
102     """
103
104     df_filtered = pd.read_csv(file_path)
105
106     # Best fitting vs MMW (Scatter) plot
107     plt.figure(figsize=(10, 6))
108     plt.scatter(df_filtered['MMW'], df_filtered['Env_Fraction'],
            color='red', alpha=0.7, label='Envelope Fraction')
109     plt.scatter(df_filtered['MMW'], df_filtered['H/He'], color='
            green', alpha=0.7, label='H/He Mass Fraction')
```

```
    plt.scatter(df_filtered['MMW'], df_filtered['WMF'], color='
        blue', alpha=0.7, label='Water Mass Fraction')

    plt.xlabel('Mean Molecular Weight (MMW)', fontsize=14)
    plt.ylabel('Best Fitting Fractions', fontsize=14)
    plt.title(f'Best Fitting Fractions vs Mean Molecular Weight
        for {pname}', fontsize=16)
    plt.grid(True)
    plt.xticks(fontsize=12)
    plt.yticks(fontsize=12)

    plt.tight_layout()
    if save_plot:
        plt.savefig(f'best_fitting_vs_mmw_{pname}.png')
    plt.show()

    # Best Fitting H/He vs MMW (Scatter) plot
    plt.figure(figsize=(10, 6))
    plt.scatter(df_filtered['MMW'], df_filtered['H/He'], color='
        green', alpha=0.7, label='H/He Mass fraction')
    plt.xlabel('Mean Molecular Weight (MMW)', fontsize=14)
    plt.ylabel('Best Fitting Fractions', fontsize=14)
    plt.title(f'Best Fitting H/He vs Mean Molecular Weight for {
        pname}', fontsize=16)
    plt.grid(True)

    # Apply user-defined ylim or the default
    plt.ylim(ylim)

    plt.tight_layout()
    if save_plot:
        plt.savefig(f'best_fitting_hhe_vs_mmw_{pname}.png')
    plt.show()
```

```python

def fill_between(file_path: str, pname: str, save_plot: bool =
    False, pad_value: float = None):
    """
    Creates a shaded plot of best fitting mass fractions vs. Mean
        Molecular Weight (MMW) for a specific Pad value.

    Parameters:
    - file_path: str, the path to the CSV file with data.
    - pname: str, the name of the planet to include in the plot
        title and save filenames.
    - save_plot: bool, if True, saves the plot as a PNG file.
    - pad_value: float, optional, the specific Pad value in bars
        to filter by.
    """

    # Read the filtered file
    df_filtered = pd.read_csv(file_path)

    # Check unique Pad values in the data
    unique_pads = df_filtered['Pad'].unique() / 1e5  # Convert
        from Pascals to bars
    unique_pads = sorted(set(unique_pads))

    # Determine the Pad value to use
    if pad_value is None:
        if len(unique_pads) == 1:
            pad_value = unique_pads[0]
            print(f"Using the only available Pad value: {pad_value
                } bar")
        else:
            print(f"Multiple Pad values found: {unique_pads}")
            print("Please specify a pad_value from the list above.
                ")
            return
```

```
    else:
        if pad_value not in unique_pads:
            print(f"Specified Pad value {pad_value} bar is not in
                the data. Available values are: {unique_pads}")
            return

    # Filter data by the specified Pad value in Pascals
    pad_value_pa = pad_value * 1e5
    df_filtered_pad = df_filtered[df_filtered['Pad'] ==
        pad_value_pa]

    # Calculate min/max for plotting shaded regions
    min_max_df = df_filtered_pad.groupby('MMW').agg(
        min_ef=('Env_Fraction', 'min'),
        max_ef=('Env_Fraction', 'max'),
        min_wmf=('WMF', 'min'),
        max_wmf=('WMF', 'max'),
        min_hhe=('H/He', 'min'),
        max_hhe=('H/He', 'max')
    ).reset_index()

    plt.figure(figsize=(10, 6))

    sigma = 0.7

    min_ef_smooth = gaussian_filter1d(min_max_df['min_ef'], sigma=
        sigma)
    max_ef_smooth = gaussian_filter1d(min_max_df['max_ef'], sigma=
        sigma)

    min_wmf_smooth = gaussian_filter1d(min_max_df['min_wmf'],
        sigma=sigma)
    max_wmf_smooth = gaussian_filter1d(min_max_df['max_wmf'],
        sigma=sigma)

```

```
    min_hhe_smooth = gaussian_filter1d(min_max_df['min_hhe'],
        sigma=sigma)
    max_hhe_smooth = gaussian_filter1d(min_max_df['max_hhe'],
        sigma=sigma)

    # Create shaded areas between min and max values
    plt.fill_between(min_max_df['MMW'], min_ef_smooth,
        max_ef_smooth, color='red', alpha=0.5, label='Envelope
        Fraction')
    plt.fill_between(min_max_df['MMW'], min_wmf_smooth,
        max_wmf_smooth, color='blue', alpha=0.5, label='Water Mass
         Fraction')
    plt.fill_between(min_max_df['MMW'], min_hhe_smooth,
        max_hhe_smooth, color='green', alpha=0.5, label='H/He Mass
         Fraction')

    plt.xlabel('Mean Molecular Weight (MMW)', fontsize=14)
    plt.ylabel('Best Fitting Mass Fraction', fontsize=14)
    plt.title(f'Best Fitting Mass Fraction vs Mean Molecular
        Weight for {pname} (Pad = {pad_value} bar)', fontsize=16)

    plt.grid(True)
    plt.legend(fontsize=12, loc='upper left')
    plt.xticks(fontsize=12)
    plt.yticks(fontsize=12)
    plt.yscale('log')
    plt.tight_layout()

    if save_plot:
        plt.savefig(f'filled_between_plot_{pname}_pad_{pad_value}
            bar.png')
    plt.show()
```

```
222 def clean_display(file_path: str, output_file: str = None,
        mmw_range: tuple = None, save_file: bool = False,
        mmw_threshold: float = 10):
223     # Read the CSV file
224     df = pd.read_csv(file_path)
225 
226     # Apply MMW range filtering if provided
227     if mmw_range:
228         df = df[(df['MMW'] >= mmw_range[0]) & (df['MMW'] <=
                mmw_range[1])]
229         print(f"Filtering MMW between {mmw_range[0]} and {
                mmw_range[1]}")
230 
231     # Remove duplicates
232     df.drop_duplicates(subset=['MMW', 'Env_Fraction', 'H/He', 'WMF
            '], inplace=True)
233 
234     # Save only if save_file is True and output_file is specified
235     if save_file:
236         if output_file is None:
237             raise ValueError("Please provide an output file path
                    when save_file is True.")
238         df.to_csv(output_file, index=False)
239         print(f"Updated data saved to '{output_file}'")
240 
241     # Check and display duplicates
242     num_dup = df.duplicated(subset=['MMW', 'Env_Fraction', 'H/He',
            'WMF']).sum()
243     print(f"The file has {num_dup} duplicates." if num_dup else "
            The file has no duplicates.")
244 
245     # Display unique Pad values in bars
246     unique_pads = sorted(df['Pad'].unique())
247     unique_pads_bar = [f"{pad / 1e5} bar" for pad in unique_pads]
```

```
248     print("Available unique Pad values:", ", ".join(
            unique_pads_bar))
249
250     # Display min and max for columns of interest
251     min_max_vals = {
252         'WMF': (df['WMF'].min(), df['WMF'].max()),
253         'H/He': (df['H/He'].min(), df['H/He'].max()),
254         'Env_Fraction': (df['Env_Fraction'].min(), df['
                Env_Fraction'].max())
255     }
256     for name, (min_val, max_val) in min_max_vals.items():
257         print(f"{name} - Min: {min_val:.4f}, Max: {max_val:.4f}")
258
259     print("\n")
260     # Display Pad values associated with min and max H/He
261     hhe_max, hhe_min = df['H/He'].max(), df['H/He'].min()
262
263     # Get the rows for max and min H/He
264     max_row = df.loc[df['H/He'] == hhe_max]
265     min_row = df.loc[df['H/He'] == hhe_min]
266
267     # Extract Pad and MMW values for max and min H/He
268     pad_max = max_row['Pad'].values[0] if not max_row.empty else
            None
269     pad_min = min_row['Pad'].values[0] if not min_row.empty else
            None
270     mmw_max = max_row['MMW'].values[0] if not max_row.empty else
            None
271     mmw_min = min_row['MMW'].values[0] if not min_row.empty else
            None
272
273     # Print the results for max and min H/He
274     print(f"Pad value for max H/He ({hhe_max:.4f}): {pad_max:.4f}
            Pa or {pad_max / 1e5 if pad_max else None} bar, MMW: {
            mmw_max}")
```

```
275     print(f"Pad value for min H/He ({hhe_min:.4f}): {pad_min:.4f}
            Pa or {pad_min / 1e5 if pad_min else None} bar, MMW: {
            mmw_min}")
276
277
278     # Find the row with minimum MMW
279     mmw_min_row = df.loc[df['MMW'].idxmin()]  # Row with the
            absolute minimum MMW
280     min_mmw_value = mmw_min_row['MMW']        # Minimum MMW
281     hhe_for_min_mmw = mmw_min_row['H/He']     # Corresponding H/He
282     pad_for_min_mmw = mmw_min_row['Pad']      # Corresponding Pad
283     print(f"Absolute Min MMW: {min_mmw_value:.4f}, H/He: {
            hhe_for_min_mmw:.4f}")
284     print(f"Pad corresponding to Min MMW: {pad_for_min_mmw:.4f} Pa
             or {pad_for_min_mmw / 1e5:.2f} bar")
285
286     # Find the smallest H/He with a reasonable MMW
287     filtered_df = df[df['MMW'] <= mmw_threshold]  # Filter rows
            with MMW <= threshold
288     if not filtered_df.empty:
289         min_hhe_row = filtered_df.loc[filtered_df['H/He'].idxmin()
                ]  # Row with smallest H/He
290         smallest_hhe = min_hhe_row['H/He']
                                       # Smallest H/He
291         corresponding_mmw = min_hhe_row['MMW']
                                  # Corresponding MMW for smallest
                H/He
292         print(f"Smallest H/He within threshold (MMW <= {
                mmw_threshold}): {smallest_hhe:.4f}, Corresponding MMW
                : {corresponding_mmw:.4f}")
293
294         # Compare the two rows
295         if min_mmw_value == corresponding_mmw:
296             print("The minimum MMW also gives the smallest H/He
                    within the threshold.")
```

```
297         else:
298             print("The minimum MMW does NOT give the smallest H/He
                    within the threshold.")
299     else:
300         print(f"No rows found with MMW <= {mmw_threshold}.")
301
302
303 # def clean_display(file_path: str, output_file: str = None,
    mmw_range: tuple = None, save_file: bool = False):
304 #     # Read the CSV file
305 #     df = pd.read_csv(file_path)
306
307 #     # Apply MMW range filtering if provided
308 #     if mmw_range:
309 #         df = df[(df['MMW'] >= mmw_range[0]) & (df['MMW'] <=
    mmw_range[1])]
310 #         print(f"Filtering MMW between {mmw_range[0]} and {
    mmw_range[1]}")
311
312 #     # Remove duplicates
313 #     df.drop_duplicates(subset=['MMW', 'Env_Fraction', 'H/He', '
    WMF'], inplace=True)
314
315 #     # Save only if save_file is True and output_file is
    specified
316 #     if save_file:
317 #         if output_file is None:
318 #             raise ValueError("Please provide an output file path
     when save_file is True.")
319 #         df.to_csv(output_file, index=False)
320 #         print(f"Updated data saved to '{output_file}'")
321
322 #     # Check and display duplicates
323 #     num_dup = df.duplicated(subset=['MMW', 'Env_Fraction', 'H/He
    ', 'WMF']).sum()
```

```
#     print(f"The file has {num_dup} duplicates." if num_dup else "The file has no duplicates.")

#     # Display unique Pad values in bars
#     unique_pads = sorted(df['Pad'].unique())
#     unique_pads_bar = [f"{pad / 1e5} bar" for pad in unique_pads]
#     print("Available unique Pad values:", ", ".join(unique_pads_bar))

#     # Display min and max for columns of interest
#     min_max_vals = {
#         'WMF': (df['WMF'].min(), df['WMF'].max()),
#         'H/He': (df['H/He'].min(), df['H/He'].max()),
#         'Env_Fraction': (df['Env_Fraction'].min(), df['Env_Fraction'].max())
#     }
#     for name, (min_val, max_val) in min_max_vals.items():
#         print(f"{name} - Min: {min_val:.4f}, Max: {max_val:.4f}")

    # # Display Pad values associated with min and max H/He
    # hhe_max, hhe_min = df['H/He'].max(), df['H/He'].min()

    # # Get the rows for max and min H/He
    # max_row = df.loc[df['H/He'] == hhe_max]
    # min_row = df.loc[df['H/He'] == hhe_min]

    # # Extract Pad and MMW values for max and min H/He
    # pad_max = max_row['Pad'].values[0] if not max_row.empty else None
    # pad_min = min_row['Pad'].values[0] if not min_row.empty else None
    # mmw_max = max_row['MMW'].values[0] if not max_row.empty else None
```

```
351     # mmw_min = min_row['MMW'].values[0] if not min_row.empty else
            None
352
353     # # Print the results for max and min H/He
354     # print(f"Pad value for max H/He ({hhe_max:.4f}): {pad_max:.4f
            } Pa or {pad_max / 1e5 if pad_max else None} bar, MMW: {
            mmw_max}")
355     # print(f"Pad value for min H/He ({hhe_min:.4f}): {pad_min:.4f
            } Pa or {pad_min / 1e5 if pad_min else None} bar, MMW: {
            mmw_min}")
356
357 #     # Find the row with minimum MMW
358 #     mmw_min_row = df.loc[df['MMW'].idxmin()]
359 #     hhe_for_min_mmw = mmw_min_row['H/He']
360 #     pad_for_min_mmw = mmw_min_row['Pad']
361 #     min_mmw_value = mmw_min_row['MMW']  # The minimum MMW itself
362
363 #     print(f"Min MMW: {min_mmw_value:.4f}")
364 #     print(f"H/He corresponding to Min MMW: {hhe_for_min_mmw:.4f
    }")
365 #     print(f"Pad corresponding to Min MMW: {pad_for_min_mmw:.4f}
    Pa or {pad_for_min_mmw / 1e5:.2f} bar")
366
367 #     # Find the smallest H/He with a reasonable MMW
368 #     sorted_hhe = df.sort_values(by='H/He', ascending=True)
369 #     reasonable_mmw_threshold = 10
370 #     filtered_hhe_mmw = sorted_hhe[sorted_hhe['MMW'] <=
    reasonable_mmw_threshold]
371 #     smallest_hhe_row = filtered_hhe_mmw.iloc[0]  # Get the first
     row after filtering
372 #     smallest_hhe = smallest_hhe_row['H/He']
373 #     corresponding_mmw = smallest_hhe_row['MMW']
374
375 #     # Print the results for smallest H/He and its corresponding
    MMW
```

```
376 #     print(f"Smallest H/He with reasonable MMW: {smallest_hhe:.4f}")
377 #     print(f"Corresponding MMW for this H/He: {corresponding_mmw:.4f}")
378
379 def pad_plot(file_path, pname: str, pad_value=None, save_plot=False):
380     """
381     Plots best fitting fractions vs. Mean Molecular Weight (MMW) for a given Pad value (in bars) from the filtered dataset,
382     and displays a summary of min and max values for H/He, WMF, and Env_Fraction for the specific Pad value.
383
384     Parameters:
385     - file_path: str, the path to the filtered CSV file.
386     - pname: str, the name of the planet to include in the plot title and save filenames.
387     - pad_value: float, optional, the specific Pad value in bars to filter and plot.
388     - save_plot: bool, if True, saves the plots as PNG files (default: False).
389     """
390
391     # Read the filtered file
392     df_filtered = pd.read_csv(file_path)
393
394     # Check unique Pad values in the data
395     unique_pads = df_filtered['Pad'].unique() / 1e5  # Convert from Pascals to bars
396     unique_pads = sorted(set(unique_pads))
397
398     # Determine the Pad value to use
399     if pad_value is None:
400         if len(unique_pads) == 1:
401             pad_value = unique_pads[0]
```

```
            print(f"Using the only available Pad value: {pad_value} bar")
        else:
            print(f"Multiple Pad values found: {unique_pads}")
            print("Please specify a pad_value from the list above.")
            return
    else:
        if pad_value not in unique_pads:
            print(f"Specified Pad value {pad_value} bar is not in the data. Available values are: {unique_pads}")
            return

    # Filter data by the specified Pad value in Pascals
    pad_value_pa = pad_value * 1e5
    df_filtered_pad = df_filtered[df_filtered['Pad'] == pad_value_pa]

    # Calculate summary statistics for H/He, WMF, and Env_Fraction
    min_max_summary = {
        'WMF': (df_filtered_pad['WMF'].min(), df_filtered_pad['WMF'].max()),
        'H/He': (df_filtered_pad['H/He'].min(), df_filtered_pad['H/He'].max()),
        'Env_Fraction': (df_filtered_pad['Env_Fraction'].min(), df_filtered_pad['Env_Fraction'].max())
    }

    # Find MMW values for min and max H/He
    hhe_min, hhe_max = min_max_summary['H/He']
    mmw_min_hhe = df_filtered_pad.loc[df_filtered_pad['H/He'] == hhe_min, 'MMW'].values[0]
    mmw_max_hhe = df_filtered_pad.loc[df_filtered_pad['H/He'] == hhe_max, 'MMW'].values[0]

```

```
    # Display summary information with formatted output
    print(f"Summary for {pname} at Pad = {pad_value:.1f} bar:")
    for name, (min_val, max_val) in min_max_summary.items():
        print(f"  {name} - Min: {min_val:.2f}, Max: {max_val:.2f}")
    print(f"  MMW corresponding to min H/He ({hhe_min:.2e}): {mmw_min_hhe:.2f}")
    print(f"  MMW corresponding to max H/He ({hhe_max:.2f}): {mmw_max_hhe:.2f}")

    # Plot 1: Best Fitting Fractions vs MMW
    plt.figure(figsize=(10, 6))
    plt.scatter(df_filtered_pad['MMW'], df_filtered_pad['Env_Fraction'], color='red', alpha=0.7, label='Envelope Fraction')
    plt.scatter(df_filtered_pad['MMW'], df_filtered_pad['WMF'], color='blue', alpha=0.7, label='Water Mass Fraction')
    plt.scatter(df_filtered_pad['MMW'], df_filtered_pad['H/He'], color='green', alpha=0.7, label='H/He Ratio')

    plt.xlabel('Mean Molecular Weight (MMW)', fontsize=14)
    plt.ylabel('Best Fitting Fractions', fontsize=14)
    plt.title(f'Best Fitting Fractions vs. Mean Molecular Weight for {pname} (Pad = {pad_value} bar)', fontsize=16)
    plt.grid(True)
    plt.legend(fontsize=12)
    plt.tight_layout()
    if save_plot:
        plt.savefig(f'best_fitting_fractions_{pname}_pad_{pad_value}bar.png')
    plt.show()

    # Plot 2: H/He vs MMW
    plt.figure(figsize=(10, 6))
```

```
453     plt.scatter(df_filtered_pad['MMW'], df_filtered_pad['H/He'],
            color='green', alpha=0.7, label='H/He Ratio')
454
455     plt.xlabel('Mean Molecular Weight (MMW)', fontsize=14)
456     plt.ylabel('H/He Mass Fraction', fontsize=14)
457     plt.title(f'H/He Mass Fraction vs. Mean Molecular Weight for {
            pname} (Pad = {pad_value} bar)', fontsize=16)
458     plt.grid(True)
459     plt.legend(fontsize=12)
460     plt.tight_layout()
461     if save_plot:
462         plt.savefig(f'hhe_mass_fraction_{pname}_pad_{pad_value}bar
                .png')
463     plt.show()
464
465 def wmf_filter(df, pname: str, wmf_cutoff: float, save_plot: bool
        = False):
466     """
467     Filters data by WMF cutoff and performs additional analysis
            and plotting.
468
469     Parameters:
470     - df: DataFrame, the dataset to analyze.
471     - pname: str, the name of the planet to include in plot titles
            and filenames.
472     - wmf_cutoff: float, maximum allowed WMF value for filtering.
473     - save_plot: bool, if True, saves the generated plots.
474     """
475     # Filter rows where WMF exceeds the cutoff
476     df_filtered = df[df['WMF'] <= wmf_cutoff]
477     print(f"Applying WMF cutoff: Excluding rows with WMF > {
            wmf_cutoff}")
478
479     # Plot 1: Best Fitting Fractions vs MMW
480     plt.figure(figsize=(10, 6))
```

```
    plt.scatter(df_filtered['MMW'], df_filtered['Env_Fraction'],
        color='red', alpha=0.7, label='Envelope Fraction')
    plt.scatter(df_filtered['MMW'], df_filtered['WMF'], color='
        blue', alpha=0.7, label='Water Mass Fraction')
    plt.scatter(df_filtered['MMW'], df_filtered['H/He'], color='
        green', alpha=0.7, label='H/He Ratio')
    plt.xlabel('Mean Molecular Weight (MMW)', fontsize=14)
    plt.ylabel('Best Fitting Fractions', fontsize=14)
    plt.title(f'Best Fitting Fractions vs. MMW for {pname}',
        fontsize=16)
    plt.legend(fontsize=12)
    plt.grid(alpha=0.3)
    plt.tight_layout()
    if save_plot:
        plt.savefig(f'best_fitting_fractions_wmf_cutoff_{pname}.
            png')
    plt.show()

    # Plot 2: H/He vs MMW (No dotted lines)
    plt.figure(figsize=(10, 6))
    plt.scatter(df_filtered['MMW'], df_filtered['H/He'], color='
        green', alpha=0.7, label='H/He Ratio')
    plt.xlabel('Mean Molecular Weight (MMW)', fontsize=14)
    plt.ylabel('H/He Mass Fraction', fontsize=14)
    plt.title(f'H/He Mass Fraction vs. MMW for {pname}', fontsize
        =16)
    plt.grid(alpha=0.3)
    plt.tight_layout()
    if save_plot:
        plt.savefig(f'hhe_vs_mmw_wmf_cutoff_{pname}.png')
    plt.show()

    # Additional Summary Information
    # Maximum H/He and corresponding MMW
    hhe_max = df_filtered['H/He'].max()
```

```
    mmw_max_hhe = df_filtered.loc[df_filtered['H/He'] == hhe_max,
        'MMW'].values[0]
    print(f"Max H/He ({hhe_max:.4f}) corresponds to MMW: {
        mmw_max_hhe:.4f}")

    # Maximum MMW
    max_mmw = df_filtered['MMW'].max()
    print(f"Max MMW after cutoff: {max_mmw:.4f}")

    # Minimum MMW and corresponding H/He
    min_mmw_row = df_filtered.loc[df_filtered['MMW'].idxmin()]
    min_mmw = min_mmw_row['MMW']
    hhe_for_min_mmw = min_mmw_row['H/He']
    print(f"Min MMW ({min_mmw:.4f}) corresponds to H/He: {
        hhe_for_min_mmw:.4f}")

    # Smallest H/He with a "reasonable" MMW
    sorted_hhe = df_filtered.sort_values(by='H/He', ascending=True
        )
    reasonable_mmw_threshold = 10
    filtered_hhe_mmw = sorted_hhe[sorted_hhe['MMW'] <=
        reasonable_mmw_threshold]

    if not filtered_hhe_mmw.empty:
         smallest_hhe_row = filtered_hhe_mmw.iloc[0]
         smallest_hhe = smallest_hhe_row['H/He']
         corresponding_mmw = smallest_hhe_row['MMW']
         print(f"Smallest H/He (MMW <= {reasonable_mmw_threshold}):
             {smallest_hhe:.4f}")
         print(f"Corresponding MMW for this H/He: {
            corresponding_mmw:.4f}")

         # Compare Min MMW and Smallest H/He
         if min_mmw == corresponding_mmw:
```

```
536             print("The absolute minimum MMW gives the smallest H/
                    He.")
537         else:
538             print("The absolute minimum MMW does NOT give the
                    smallest H/He.")
539     else:
540         print(f"No rows found with MMW <= {
                reasonable_mmw_threshold} for smallest H/He.")
541
542 def best_display(file_path, pname: str, output_file=None,
    mmw_range=None, pad_value=None, wmf_cutoff=None, save_file=
    False, save_plot=False):
543     """
544     Merges the functionalities of clean_display and pad_plot
            functions with an optional WMF cutoff.
545
546     Displays filtered data, summaries, and plots for a specific
            pressure value (Pad), Mean Molecular Weight (MMW) range,
547     and an optional cutoff for Water Mass Fraction (WMF).
548
549     Parameters:
550     - file_path: str, path to the filtered CSV file.
551     - pname: str, the name of the planet to include in the plot
            titles and save filenames.
552     - output_file: str, optional, path to save the filtered data
            if save_file is True.
553     - mmw_range: tuple, optional, range of MMW values to filter
            the data.
554     - pad_value: float, optional, the specific Pad value in bars
            to filter and plot.
555     - wmf_cutoff: float, optional, maximum allowed WMF value to
            filter the data.
556     - save_file: bool, if True, saves the filtered data (default:
            False).
```

```
557     - save_plot: bool, if True, saves the generated plots (default
            : False).
558     """
559     # Read the dataset
560     df = pd.read_csv(file_path)
561 
562     # Filter by MMW range if specified
563     if mmw_range:
564         df = df[(df['MMW'] >= mmw_range[0]) & (df['MMW'] <=
                mmw_range[1])]
565         print(f"Filtering MMW between {mmw_range[0]} and {
                mmw_range[1]}")
566 
567     # Remove duplicates
568     df.drop_duplicates(subset=['MMW', 'Env_Fraction', 'H/He', 'WMF
            '], inplace=True)
569 
570     # Save the filtered file if required
571     if save_file:
572         if output_file is None:
573             raise ValueError("Please provide an output file path
                    when save_file is True.")
574         df.to_csv(output_file, index=False)
575         print(f"Updated data saved to '{output_file}'")
576 
577     # Check for duplicates
578     num_duplicates = df.duplicated(subset=['MMW', 'Env_Fraction',
            'H/He', 'WMF']).sum()
579     print(f"The file has {num_duplicates} duplicates." if
            num_duplicates else "The file has no duplicates.")
580 
581     # Display unique Pad values
582     unique_pads = sorted(df['Pad'].unique() / 1e5)  # Convert from
            Pascals to bars
```

```
583     print("Available unique Pad values:", ", ".join([f"{pad} bar"
            for pad in unique_pads]))
584
585     # Calculate min and max for columns of interest
586     min_max_vals = {
587         'WMF': (df['WMF'].min(), df['WMF'].max()),
588         'Env_Fraction': (df['Env_Fraction'].min(), df['
                Env_Fraction'].max()),
589     }
590     for name, (min_val, max_val) in min_max_vals.items():
591         print(f"{name} - Min: {min_val:.4f}, Max: {max_val:.4f}")
592
593     # Find Pad for max and min H/He
594     hhe_min = df['H/He'].min()
595     hhe_max = df['H/He'].max()
596     mmw_min_hhe = df.loc[df['H/He'] == hhe_min, 'MMW'].values[0]
597     mmw_max_hhe = df.loc[df['H/He'] == hhe_max, 'MMW'].values[0]
598
599     print(f"Min H/He ({hhe_min:.7f}) corresponds to MMW: {
            mmw_min_hhe:.4f}")
600     print(f"Max H/He ({hhe_max:.4f}) corresponds to MMW: {
            mmw_max_hhe:.4f}")
601
602     # Find row for min MMW
603     min_mmw_row = df.loc[df['MMW'].idxmin()]
604     min_mmw = min_mmw_row['MMW']
605     hhe_for_min_mmw = min_mmw_row['H/He']
606     print(f"Absolute Min MMW: {min_mmw:.4f}, H/He corresponding to
            Min MMW: {hhe_for_min_mmw:.4f}")
607
608     # Find the smallest H/He value and its corresponding "
            reasonable" MMW
609     sorted_hhe = df.sort_values(by='H/He', ascending=True)
610     reasonable_mmw_threshold = 10
```

```
    filtered_hhe_mmw = sorted_hhe[sorted_hhe['MMW'] <=
        reasonable_mmw_threshold]
    if not filtered_hhe_mmw.empty:
        smallest_hhe_row = filtered_hhe_mmw.iloc[0]
        smallest_hhe = smallest_hhe_row['H/He']
        corresponding_mmw = smallest_hhe_row['MMW']
        print(f"Smallest H/He with reasonable MMW: {smallest_hhe
            :.4f}")
        print(f"Corresponding MMW for this H/He: {
            corresponding_mmw:.4f}")
        if min_mmw == corresponding_mmw:
            print("The absolute minimum MMW gives the smallest H/
                He.")
        else:
            print("The absolute minimum MMW does NOT give the
                smallest H/He.")
    else:
        print("No rows found with MMW <= 10 for smallest H/He.")

    # Plot 1
    plt.figure(figsize=(10, 6))
    plt.scatter(df['MMW'], df['Env_Fraction'], color='red', alpha
        =0.7, label='Envelope Fraction')
    plt.scatter(df['MMW'], df['WMF'], color='blue', alpha=0.7,
        label='Water Mass Fraction')
    plt.scatter(df['MMW'], df['H/He'], color='green', alpha=0.7,
        label='H/He Ratio')
    plt.xlabel('Mean Molecular Weight (MMW)', fontsize=14)
    plt.ylabel('Best Fitting Fractions', fontsize=14)
    plt.title(f'Best Fitting Fractions vs. MMW for {pname}',
        fontsize=16)
    plt.legend(fontsize=12)
    plt.grid(alpha=0.3)
    plt.tight_layout()
    if save_plot:
```

```
637             plt.savefig(f'best_fitting_fractions_{pname}.png')
638         plt.show()
639 
640         # Plot 2: H/He vs MMW with vertical and horizontal lines
641         plt.figure(figsize=(10, 6))
642         plt.scatter(df['MMW'], df['H/He'], color='green', alpha=0.7,
                label='H/He Ratio')
643         plt.axvline(x=corresponding_mmw, color='darkblue', linestyle='
                --', label=f'MMW for Smallest H/He = {corresponding_mmw:.4
                f}')
644         plt.axhline(y=smallest_hhe, color='darkred', linestyle='--',
                label=f'Smallest H/He = {smallest_hhe:.4f}')
645         plt.axvline(x=min_mmw, color='red', linestyle='--', label=f'
                Min MMW = {min_mmw:.4f}')
646         plt.axhline(y=hhe_for_min_mmw, color='blue', linestyle='--',
                label=f'H/He for Min MMW = {hhe_for_min_mmw:.4f}')
647         plt.axvline(x=mmw_min_hhe, color='purple', linestyle='--',
                label=f'MMW for Min H/He = {mmw_min_hhe:.4f}')
648         plt.axhline(y=hhe_max, color='orange', linestyle='--', label=f
                'Max H/He = {hhe_max:.4f}')
649         plt.axhline(y=hhe_min, color='cyan', linestyle='--', label=f'
                Min H/He = {hhe_min:.4f}')
650         plt.xlabel('Mean Molecular Weight (MMW)', fontsize=14)
651         plt.ylabel('H/He Mass Fraction', fontsize=14)
652         plt.title(f'H/He Mass Fraction vs. MMW for {pname}', fontsize
                =16)
653         plt.grid(alpha=0.3)
654         plt.tight_layout()
655         if save_plot:
656             plt.savefig(f'hhe_vs_mmw_{pname}.png')
657         plt.show()
658 
659         # Additional Analysis with WMF Cutoff (if provided)
660         if wmf_cutoff is not None:
661             wmf_filter(df, pname, wmf_cutoff, save_plot)
```

run_target.py – Remote Execution

```
1
2  # run_target.py
3  import sys
4  import os
5  import numpy as np
6  import grid_search  # grid_search.py should import smile
7
8  def run_grid_search_for_target(target_name, mass_val, mass_unc,
       rad_val, rad_unc, teq_val, pad_values):
9      # Convert pad_values from bars to Pascals
10     pad_list = np.array(pad_values) * 1e5
11     env_fractions = np.arange(0, 1, 0.01)
12     mmw_list = np.linspace(2.35, 18, 50)
13
14     # Define the target path relative to the current file's
           directory
15     current_dir = os.path.dirname(__file__)
16     target_path = os.path.join(current_dir, 'Exoplanet_Target_List
           .csv')
17
18     result_file = grid_search.run_grid_search(
19         file_path=target_path,
20         planet_name=target_name,
21         pad_list=pad_list,
22         env_fractions=env_fractions,
23         mmw_list=mmw_list
24     )
25     print(f"Grid search completed for {target_name}. Results saved
           in {result_file}")
26
27 if __name__ == "__main__":
28     # Get command-line arguments
29     target_name = sys.argv[1]
30     mass_val = float(sys.argv[2])
```

```
    mass_unc = float(sys.argv[3])
    rad_val = float(sys.argv[4])
    rad_unc = float(sys.argv[5])
    teq_val = float(sys.argv[6])
    pad_values = list(map(float, sys.argv[7:]))  # Capture
        remaining args as pad values

    # Run the grid search with the specified parameters
    run_grid_search_for_target(target_name, mass_val, mass_unc,
        rad_val, rad_unc, teq_val, pad_values)
```

TOI-270d Analysis Code

```

import sys
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from IPython.display import clear_output, Image, display


# Import Path to Import
sys.path.insert(0, "/Users/biruknardos/a_UMD_Research/smile")
sys.path.append('/Users/biruknardos/a_UMD_Research/General')

import smile
import grid_search
import general
import testing
# Initial Setup
target_path = '/Users/biruknardos/a_UMD_Research/General/
    Exoplanet_target_list.csv'
```

```
21
22 df = pd.read_csv(target_path)
23 print(df.iloc[4])
24 mass_val = 4.8 # Observed mass for TOI-1231b
25 mass_unc = 0.4 # Uncertainty in observed mass
26 rad_val = 2.13 # Observed radius for TOI-270d
27 rad_unc = 0.06 # Uncertainty in observed radius
28
29 grid_search.find_max_pad(387.0975)
30 # Run Grid
31 pad_list = np.array([0.001, 0.1])
32 env_fractions = np.arange(0,1,0.01)
33 mmw_list = np.linspace(2.35,18,100)
34
35 # grid_search.run_grid_search(
36
37 #     file_path = target_path,
38 #     planet_name = 'TOI-1231b',
39 #     pad_list = pad_list,
40 #     env_fractions = env_fractions,
41 #     mmw_list = mmw_list,
42 # )
43
44
45 result1 = 'TOI-270d_Model_grid_full_20250107_092647.csv'
46 grid_search.error_calc(result1, mass_val, mass_unc, rad_val,
       rad_unc, 'TOI_270d', save_files=False)
47 filtered1 = 'filtered_with_error_TOI_270d_20250107_095721.csv'
48
49
50 general.clean_display(filtered1)
51 # general.filter_pad(filtered1, pad_value= 1.6, pname='TOI_270dNEW
       ')
52 filtered2 = 'filtered_with_error_pad1.6_TOI_270dNEW.csv'
```

```
53 general.best_display(file_path=filtered2, wmf_cutoff = 0.5, pname=
       'TOI_270d')
54 general.analyze_results(filtered1, pname='TOI_170d', ylim=(0,0.06)
       , save_plot=False)
55 general.clean_display(filtered1)
56 general.pad_plot(filtered2, pname='TOI_270d')
57 general.fill_between(filtered2, pname='TOI_270d')
58 general.clean_display(filtered2)
```