Christina Carty

Earth 361

Dr. Suzan Van Der Lee

5 December 2020

A deeper dive into ocean survival on tidally-locked exoplanets

The question “Is there life outside of earth?” seems as old as time. Attempts to answer the age-old question have evolved significantly throughout the history of mankind, but have yet to find any definitive conclusions. In the present day, one modern attempt to tackle this question is through exoplanet climate modelling—that is, using climate models to estimate the conditions on planets outside of our solar system. An emphasis of exoplanet climate modelling is understanding the potential habitability (or lack thereof) of these far-off bodies. These efforts are not only interesting and relevant as standalone modelling experiments, but also help guide the exploratory endeavors of actual space probes being sent out to collected samples and data from exoplanets. Since probes can only take samples of specific patches of a given planet, models regarding the planets atmosphere, climate, and behavior can help provide context for those sampled patches that might otherwise only be a fraction of the whole story. Additionally, modelling can help inform *where* those patches should be taken in order to most accurately understand the conditions of said planet. Although exoplanet modelling is a relatively new field, strides are being made in modelling and understanding the climates of planets light years away

In 2019, PhD Candidate Howard Chen released a paper that used a General Climate Model with a fully coupled atmosphere to investigate the climate of Inner-Habitable zone, tidally-locked, M-dwarf exoplanets (Chen et. al, 2019). This paper was one of the first to run a fully coupled atmosphere with a 3D climate model, and thus allowed Chen to obtain atmospheric data with an accuracy and range that other efforts hadn’t quite yet achieved. One consequence of this advanced model was the ability to simulate Hydrogen mixing ratios throughout the entire atmosphere of these modeled exoplanets.

With fully simulated H mixing ratios, Chen was able to calculate the survival of one earth ocean per billion years on these simulated planets. The hydrogen escape rate of a given atmosphere indicates how readily hydrogen escapes from the thermosphere. The hydrogen that escapes must come from somewhere, so the idea is that hydrogen escape can be extrapolated to estimate water survival. This is then extrapolated to estimate ocean survival. The model estimates exoplanets with earth mass, radius and terrain, and so the size of an earth ocean (2.8 x 10^8 H atoms cm^{-2} (Wolf, 2015)) is used to estimate ocean survival.

While ocean survival was not the focus of the Chen et al analysis, ocean loss is a hugely consequential component of planet habitability given that water is one of the primary indicators of habitability. Not only is ocean loss an important factor, but the specific timescale of said ocean loss can be equally telling. For example, on earth we know that in 1 billion years, the sun will be bright enough to increase hydrogen escape rates such that Earth loses all water (Leconte et. al, 2013). However, given a timescale of 1 billion years (in the context of the last ~4.5 billion years of earths existence), we still consider Earth to be very habitable in terms of ocean survival. For other planets whose age we can only estimate (Burgasser et al 2017, Gonzalez et al 2019) this timescale becomes even more important. A greater uncertainty regarding planetary means we should hope for a greater timescale of ocean survival.

Exoplanets often orbit stars that are different temperatures than our sun, and varying temperatures of a host star is strongly correlated with H escape rate and therefore ocean survival. Additionally, with respect to Chen's analysis, tidally-locked planets have only one side that faces its star (the dayside) while the other is permanently oriented away from the star (the nightside). This behavior can cause drastically different temperatures on the dayside/nightside, and thus further complicates the question of ocean survival and hydrogen escape rates.

Past models did not have values for Hydrogen mixing ratios at the appropriate parts of the thermosphere, and therefore could only estimate hydrogen escape rate by doubling their H2O (vapor) mixing ratios (Kopparapu 2017). When true H mixing ratio values are used, Chen et. al found a significant difference in ocean survival timescale than studies who could only approximate Hydrogen mixing ratios.

With my program, I hope to recreate the ocean survival calculations done by Chen et. al and compare them with calculations on the same data using approximated hydrogen ratios to further understand how much of a difference using accurate H mixing ratios makes on ocean survival. Additionally, I hope to introduce a dayside/nightside component to escape rate calculations as Chen et. al only calculated escape rates using global averages. Finally, I would like my program to display these results visually, as well as save them for future use by the program user. Ocean survival timescales are often on the order of billions of years, and a magnitude of such degree may not show significant differences across different experiments. However, such small differences may indeed be consequential in assessing the habitability of a planet.

Methods

This section details the methods used in the program FinalProgram.py, including the supplemental program HydrogenEscape.py. Before any calculations begin, at the start of the program the user must manually input the names of each file. This allowed the program to be used for any file the user wants to analyze. Additionally, the program asks the user to name the run that is about to happen under a variable, **title**. The program then uses **title** to name variables, export images and organize outputs so that the outputs are unique to the run and can be used for multiple runs with minimal confusion.

I. Hydrogen Escape Functions

The equations necessary for my calculations are rather simple and the code used to define each function is in a supplementary program, HydrogenEscape.py, that will be attached to this report. The equations for the hydrogen escape itself was taken directly from Chen 2014;

$$r = \frac{(6.5 \times 10^7 * {T_{thermo}}^{0.7}) * Q_H}{H}$$

Where Q_H is the hydrogen mixing ratio (and goes as $2*Q_{H2O}$ for approximated values), and H is the atmospheric scale height. The equation for H is

$$H = \frac{k * T}{m * g}$$

Where K is Boltzmann's constant, T is temperature, m is mean molecular mass of the earth (used earths value of 29.8) and g is gravity.

Ocean loss was simply the number of H atoms in an earth ocean divided by the escape rate.

II. Data Manipulation and calculations

One of the main complications from this project instead arises from dealing with the multidimensional data outputted by 3D climate models. The program requires three datasets: atmospheric temperatures, hydrogen mixing ratios, and H2O mixing ratios. Each of these variables are in turn attached to a latitude, longitude, and level coordinate. To work with thermospheric values, the datasets need to be set at a specific level (which correlated to vertical altitude). To work with dayside/nightside values, the datasets likewise need to be set at specific longitudinal values. A flowchart of the programs' approach to obtaining these subsets is shown in Figure 1 below.

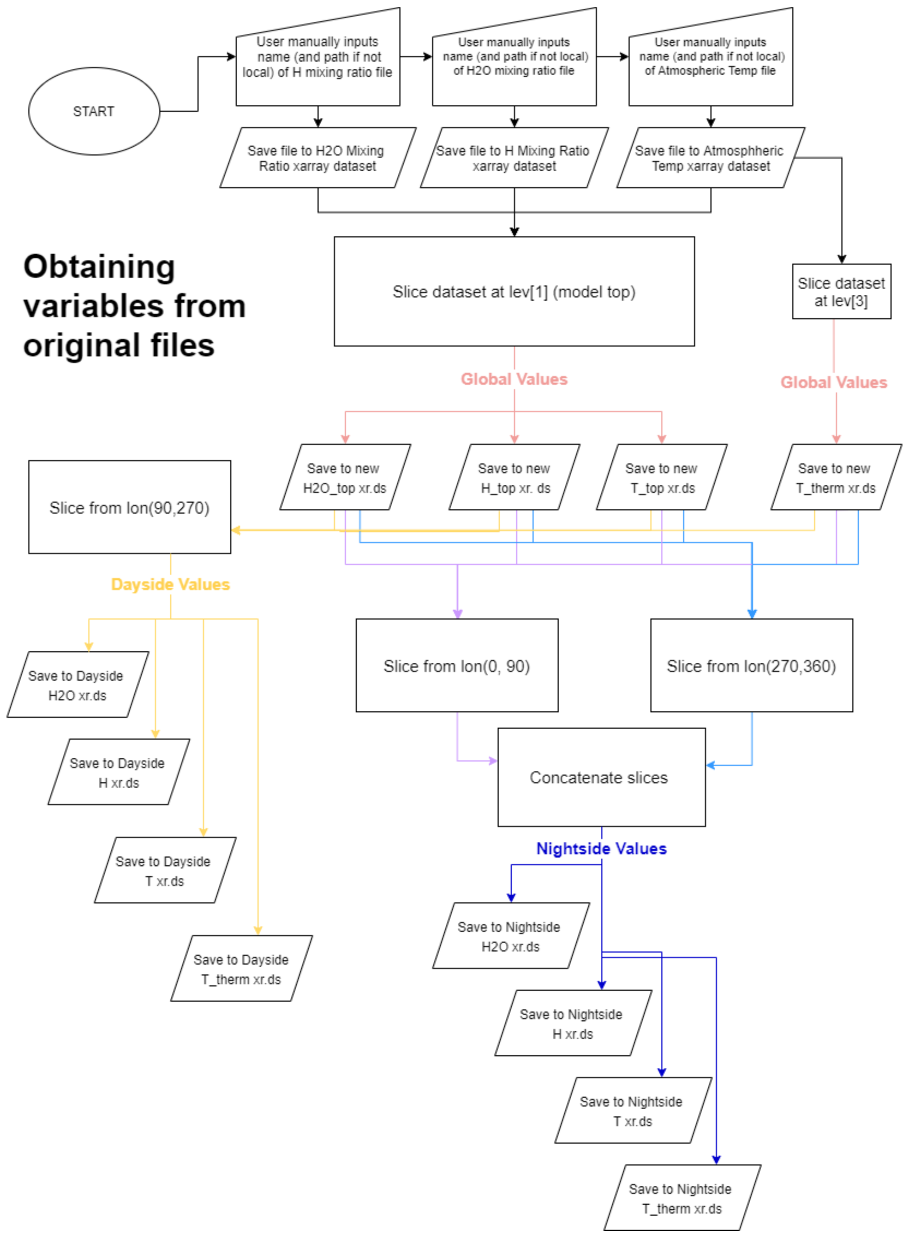

START
User manually inputs name (and path if not local) of H mixing ratio file
User manually inputs name (and path if not local) of H2O mixing ratio file
User manually inputs name (and path if not local) of Atmospheric Temp file
Save file to H2O Mixing Ratio xarray dataset
Save file to H Mixing Ratio xarray dataset
Save file to Atmosphheric Temp xarray dataset
Obtaining variables from original files
Slice dataset at lev[1] (model top)
Slice dataset at lev[3]
Global Values
Global Values
Save to new H2O_top xr.ds
Save to new H_top xr. ds
Save to new T_top xr.ds
Save to new T_therm xr.ds
Slice from lon(90,270)
Dayside Values
Save to Dayside H2O xr.ds
Save to Dayside H xr.ds
Save to Dayside T xr.ds
Save to Dayside T_therm xr.ds
Slice from lon(0, 90)
Slice from lon(270,360)
Concatenate slices
Nightside Values
Save to Nightside H2O xr.ds
Save to Nightside H xr.ds
Save to Nightside T xr.ds
Save to Nightside T_therm xr.ds

Figure 1 shows a flowchart summary of the programs approach to obtaining necessary variables from source files

One challenge posed by this section of the code was organizing the nightside datasets. The substellar point for this model was placed at 180° Longitude, making the dayside terminators 90° and 270° longitude. Subsequently, this split the nightside data in half at (0,90) and (270,360). I originally attempted to move forward with two nightside datasets and combine them by taking the mean of both sets and then taking the mean of the resulting two floats. However, this method was messy and took up unnecessary space—especially given that a *much* easier solution was to simply concatenate the two nightside datasets into one data.

Once I had global, dayside, and nightside values for all the relevant variables, the next step was to turn the datasets into floats that could be plugged into my equations. Specifically, I wanted the average values of each variable across the latitude and longitude. This step was fairly easy as xarray's built in *DataSet.mean()* function allowed me to automatically take the mean over an entire dataset.

Another challenge arose here as the *DataSet.mean()* function outputs floats that are still embedded within an xarray Dataset. So taking the mean of the global Temperature dataset, for example, returned a 0 dimensional dataset containing the variable T and its mean as a float. Xarray loses much of its value when working with data that is not multidimensional and becomes pretty much useless when working with single float values, so leaving the values like this was not ideal. Unfortunately, xarray likes to retain its formatting as much as possible and obtaining a float from the mean function required some workaround. A flow chart displaying this workaround is shown in Figure 2

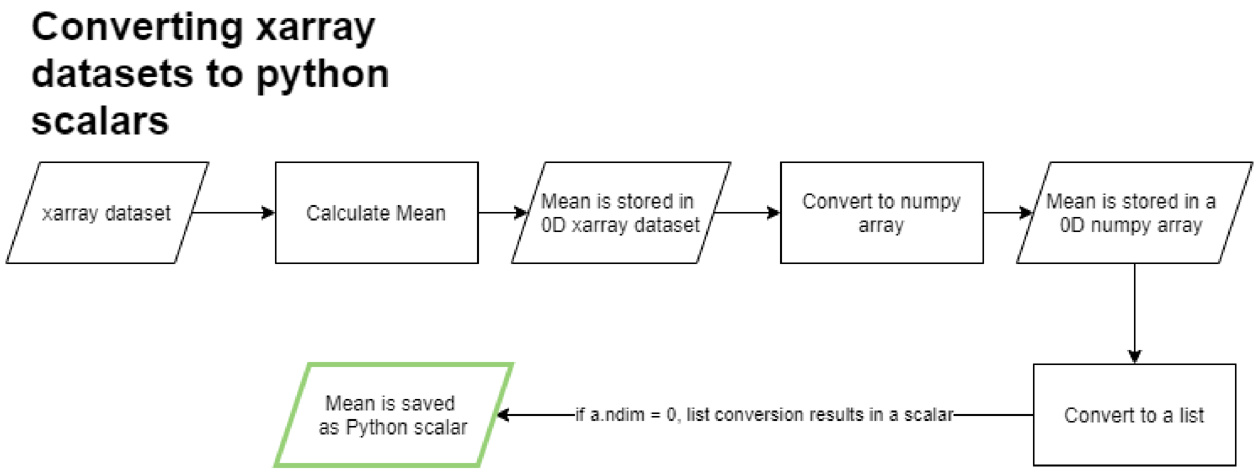

Converting xarray datasets to python scalars
xarray dataset
Calculate Mean
Mean is stored in 0D xarray dataset
Convert to numpy array
Mean is stored in a 0D numpy array
Convert to a list
if a.ndim = 0, list conversion results in a scalar
Mean is saved as Python scalar

Figure 2 shows the programs approach to obtaining python scalars from the xarray dataset format

Once again, doing this workaround one step at a time resulted in many needless variables. Nesting the operations made the code much less messy. An example of this nested code is shown below;

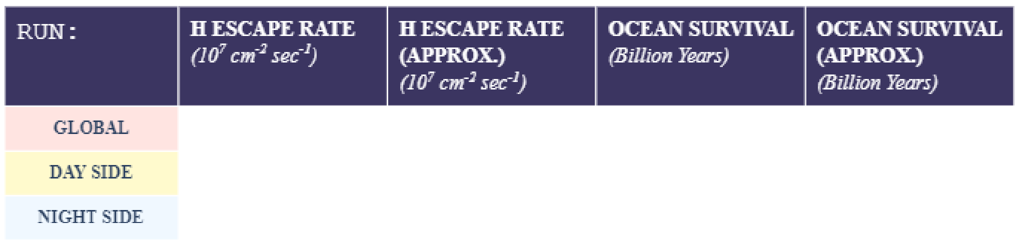
```
$ Q = h_top.mean().H.values.tolist()
```

Once I had my scalar means, it was simply a matter of plugging them into my predefined functions and my calculations were complete.

III. Data visualization

I chose to visualize the outputs of the program in a table. While there were possible ways to visualize the data graphically or geographically/cartographically, I thought that a table was the most straightforward way to compare and analyze calculation outputs.

The table was created using a plotly figure, with the variable label in the horizontal headers and the "parameter" (i.e., Global, Dayside or Nightside) as vertical headers. The cell in the top left corner shows the specific run the table represents using the manually inputted "title" variable. An example of an empty table can be seen below.



RUN :	H ESCAPE RATE *(10^7 cm^{-2} sec^{-1})*	H ESCAPE RATE (APPROX.) *(10^7 cm^{-2} sec^{-1})*	OCEAN SURVIVAL *(Billion Years)*	OCEAN SURVIVAL (APPROX.) *(Billion Years)*
GLOBAL				
DAY SIDE				
NIGHT SIDE				

Figure 3 is an empty version of the table used in the program to organize outputs

I then give the user the option to export the table to a local file using another input command. If yes, the table is exported as "Table(**title**).png" where title is the previously inputted run name.

One challenge of this section was formatting everything to look the way I wanted. For example, I was unhappy with that my units were being visualized as "10^7 cm^-2 sec^-1" vs "10^7 cm^{-2} sec^{-1}", because I was unsure how to format superscripts within a python string. To fix this, I realized that in plotly prints can be written with html code to properly format them in standard output. This not only allowed me to adjust the format of the units, but bold and italics to further stylize the table.

Another challenge faced in stylizing the table was that the calculation outputs were always floats. Given that these numbers were often quite large, putting the floats as is into the table was messy and confusing. To amend this, I imported the module “decimal” which allows you to convert floats into scientific notation. I then used these converted floats as values in the table, which made the values much easier to read and comprehend.

The third most prevalent challenge in this section was exporting the image. I noticed that when the table was exported, the columns would compress and text would get cut off. I realized this was because the standard export settings were set to 700x500 pixels, which was too narrow to fit my table. I found that by importing the module plotly.io, I could adjust the default export settings to a wider ratio that would not compress my png when exported.

IV. Data storage

Research projects such as the that conducted by Chen et. al often run models for multiple simulations (Chen et. al ran 14 in total), each with their own unique mixing ratios and temperatures (and consequently hydrogen escape rates and ocean survival timescales). I designed my program with the understanding that a user may want to use the program for multiple simulations in on session and keep the outputs for each run to perform further analyses. I originally had the calculation outputs saving to a generic dictionary, but unless the user manually saved the dictionary to a new name, it would simply be overwritten with the subsequent run of the program.

To solve this, I had the data storage be part of an if else statement where if the dictionary already existed (i.e., the program found the name listed in the local list of variable names), it would simply be appended with the values from this run. If it could not find the dictionary, the dictionary would be created from scratch. To avoid doubles and overwriting, the keys for each run were created using the **title** input from the top of the program, so each key in the dictionary is labeled by its run title. A flow chart detailing this process is seen below.

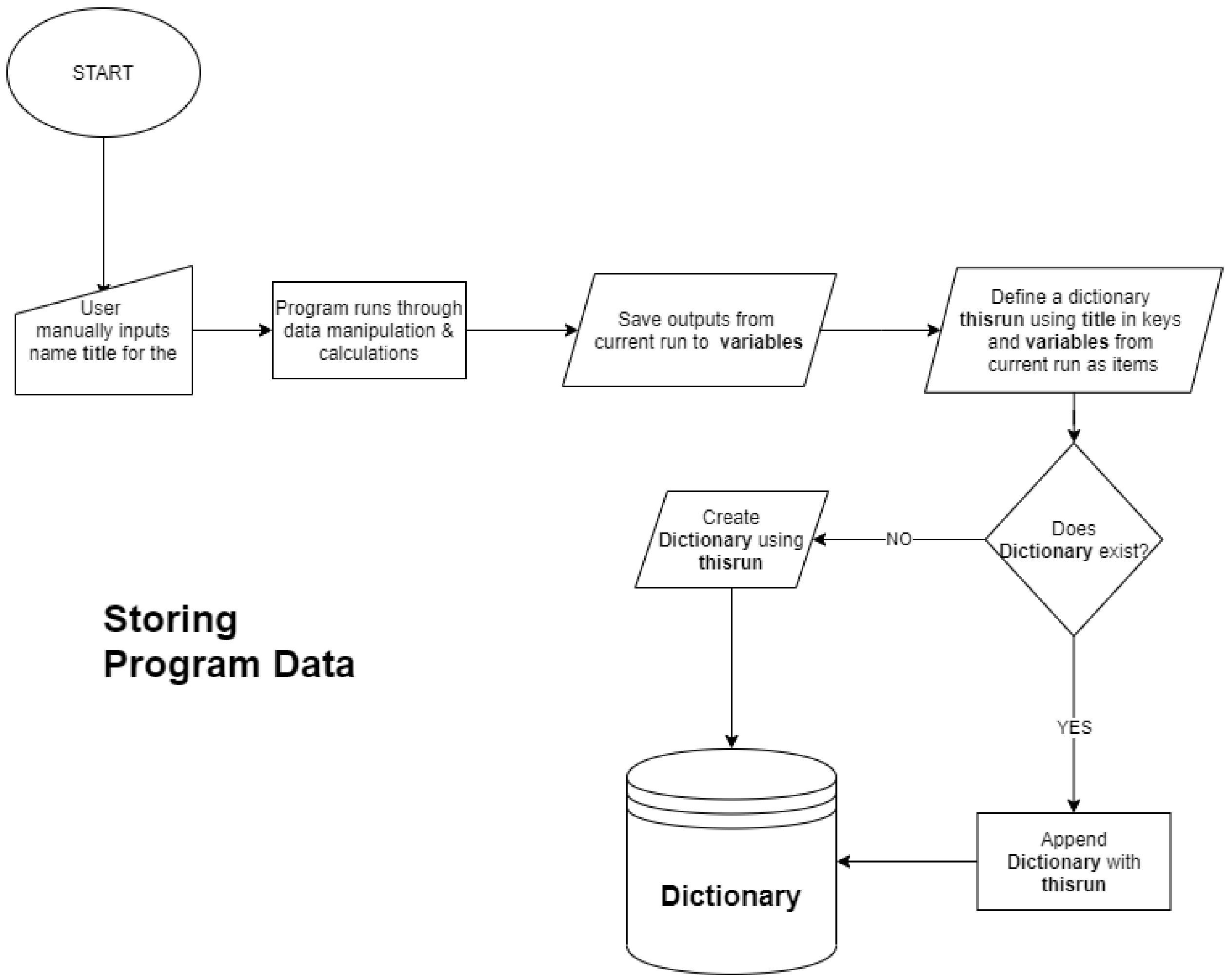

START
User manually inputs name title for the
Program runs through data manipulation & calculations
Save outputs from current run to variables
Define a dictionary thisrun using title in keys and variables from current run as items
Does Dictionary exist?
NO
Create Dictionary using thisrun
YES
Append Dictionary with thisrun
Dictionary
Storing Program Data

Figure 4 shows the programs approach to storing data in running dictionaries

I also created a Variables dictionary, containing the long names of all variables in case a user was unfamiliar or confused by any nomenclature used.

Results

I used the data from two of Chen's runs for this program. The first, '10F26T', represents a planet around a host start with a temperature of 2600 degK. The second, '19F40T' represents a planet orbiting a host start with a temperature of 4000 degK. Full information regarding both datasets can be seen in the appendix.

The resultant tables from both runs, respectively, are shown below.

RUN : 10F26T	H ESCAPE RATE *(10^7 cm^{-2} sec^{-1})*	H ESCAPE RATE (APPROX.) *(10^7 cm^{-2} sec^{-1})*	OCEAN SURVIVAL *(Billion Years)*	OCEAN SURVIVAL (APPROX.) *(Billion Years)*
GLOBAL	8.25E-03	1.11E-02	7.67E+07	5.68E+07
DAY SIDE	7.14E-03	1.10E-02	4.43E+07	2.88E+07
NIGHT SIDE	9.47E-03	1.13E-02	3.34E+07	2.80E+07

Table 1 shows results of the data from 10F26T, or a planet orbiting a host star of 2600 deg Kelvin

RUN : 19F40T	H ESCAPE RATE *(10^7 cm^{-2} sec^{-1})*	H ESCAPE RATE (APPROX.) *(10^7 cm^{-2} sec^{-1})*	OCEAN SURVIVAL *(Billion Years)*	OCEAN SURVIVAL (APPROX.) *(Billion Years)*
GLOBAL	1.73E-01	2.81E+01	3.66E+06	2.26E+04
DAY SIDE	2.56E-01	2.80E+01	1.24E+06	1.13E+04
NIGHT SIDE	7.91E-02	2.81E+01	4.00E+06	1.13E+04

Table 2 shows results of the data from 19F40T, or a planet orbiting a host star of 4000 deg Kelvin

The code did have some failures. Primarily, I did not realize that, although defined within the module, variables and dictionaries will not be saved outside of the module after the program has run. As such, my if loops appending dictionaries were useless since the dictionaries couldn't be accessed after the program was saved. A way to fix this in the future could be to use the shelf function. I also added a quick fix of simply printing the dictionary within the program so at least the user can still see the values even if they cannot access them later. However this 'quick fix' was *very* messy and would not be an acceptable solution if I were to share this program more widely.

Additionally, the user has to manually reload the module to run it again for a new set of data. This is inconvenient and costly in terms of computing power. In the future, a solution would be to set the program within a while loop that would run the program continuously until the user manually exits. A quick solution with the current program was just to print directions for the user to run the program again.

Of note is also the fact that the global values calculated here do not match those calculated by Chen et. Al. The figure below shows the results from the referenced paper.

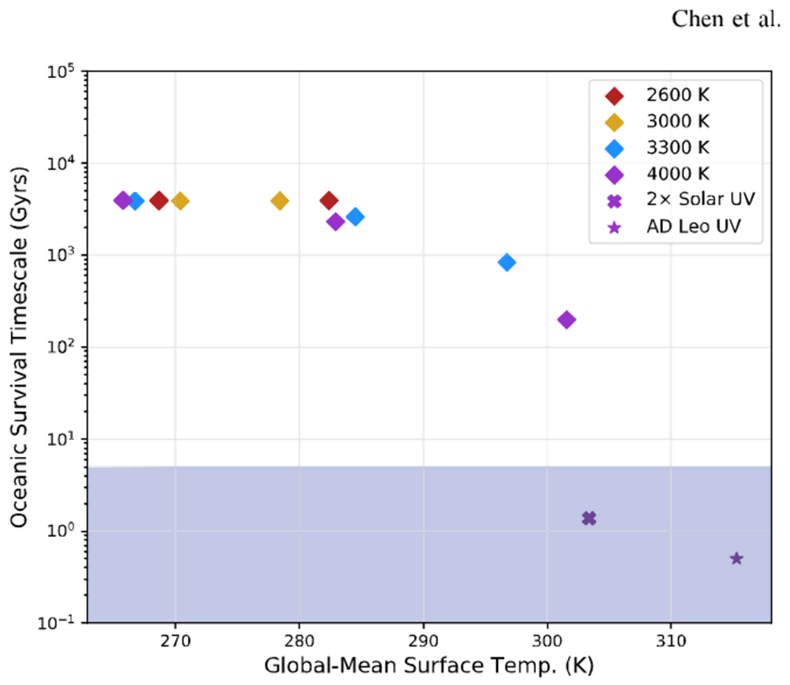


*Figure 5, curtsey of Chen et al. Original caption**: Ocean survival timescale as a function of stellar Teff (2600, 3000, 3300, 4000 K), UV activity (2× solar and AD Leo), and their corresponding global-mean Ts. Our results suggest that only simulations around active M dwarfs enter the classical moist greenhouse regime as defined by Kasting et al. (1993). Blue shading indicates timescales less than the age of the Earth.***

Differences on the order of 2-4 billion years are notable and will need to be further explored to be explained. However, on such large scales, even small differences in initial values used could add up, so the answers could simply be chalked up to rounding errors.

Discussion & Conclusion

As expected, differences in values are generally not drastic (both for approximated/non approximated mixing ratios and day/night side differences). However, some interesting information can be gleaned from the runs. In general, the data from 19F40T appears to be more sensitive than that from 10F26T. This is understandable, as 19F40T's host star is hotter by 1400 degK, and thus creates more volatile conditions.

Some other caveats to these results is that the model run by Howard uses a slab ocean, (i.e., an ocean of fixed depth that does not have any traditional ocean movement or dynamics) so the role of hydrodynamics is ignored here. Additionally, this method of calculating escape rate and ocean survival uses the assumption of Jeans diffusion limited escape means which is only one of many mechanisms involved in atmospheric gas escape (Hunten, 1973). All of this to say that while the feat of making such precise estimates for planetary bodies so far off is impressive, results should certainly be taken with a grain of salt.

I. Approximated vs. Non-Approximated values

Firstly, we see that for approximated values there are no differences in day/nightside escape rates or ocean survival. This is because at such high altitudes, the difference between H2O mixing ratios on the day and nightside are very small. The large scale of the escape rate (raised to the power of 7) and the survival timescale (on the order of 10s of billions of years) tends to erase such small differences, so this result is expected.

From this, the necessity of using true simulated hydrogen mixing ratios is reinforced. As mentioned, organizations such as NASA will often use exoplanet modeling research to direct the target sampling efforts of space probes and exploration missions. Models that do not use true H mixing ratios will fail to account for the day/nightside differences of mixing ratios and ocean survival scales, which would fail to accurately inform exploration efforts of such probes.

Regardless of day/night side differences, we also find a noticeable difference between approximated and non-approximated global values. In terms of ocean survival, the approximated values indicated a timescale that is shorter than that calculated by non-approximated values, on the order of approx. 1 billion years for 10F26T and 1.0×10^2 billion years for 19F40T. From this we can hypothesize that approximating H escape rates with H2O data will result in an overly conservative estimate of ocean survival. This is reinforced by understandings of Hydrogen behavior in the upper atmosphere, and how various processes can cause anomalies that divert hydrogen mixing ratio values from those of H2O mixing ratio values (Sonneman et. al, 2003).

II. Dayside vs. Nightside Values

Interestingly enough, for 10F26T the escape rate on the nightside is faster than that of the dayside, whereas the escape rate for 19F40T is faster on the dayside than the nightside. My original hypothesis was that daysides would always have faster escape rates, as the side facing the sun is likely to have warmer temperatures. However, upon looking at the average temperatures for both experiments, 10F26T has a higher average model top temperature on the nightside than on the dayside. I am not entirely sure why this might be, as Chen et. al show higher surface temperatures on the dayside for this run (which I assume would translate to higher model top temperatures as well). This could also be an instance of user error, and would require more investigation on my end.

These results for 10F26T show that the dayside will hold on to its ocean for a billion years longer than the nightside. For 19F40T, we conversely see that the nightside will hold on to its ocean for two billion years longer than the dayside. These results are meaningful. The true age of tidally locked M-dwarf planets is often hard to estimate. If models show that a day or nightside is more likely to retain water for longer, exploration efforts could be able to direct their efforts accordingly to ensure the highest chance of finding water.

III. Conclusion and Next Steps

In conclusion, the program ran well for the most part, and gave results that could have interesting implications for the field of exoplanet climate modelling. The program found that for a planet orbiting a host star of 2600 degree kelvin, the dayside hydrogen escape rate could potentially be *slower* than the nightside and thus result in the dayside holding its ocean more readily. The program also found that using H_2O mixing ratios to estimate hydrogen escape not only underestimates global ocean survival, but also severely simplifies the day/night side dynamics of hydrogen escape.

Next steps with the program would certainly be to clean up the output issues, mainly finding a way to save dictionaries or lists outside of the program and rerunning automatically without forcing the user to reimport the module. Outside of the bugs, I would also be interested in finding more ways to express these results, primarily through plots and maps that could better delineate the day and nightside dynamics. In this vein I also think the program could benefit by displaying a bit more of the process to standard output. For example the anomaly of the 10F26T dayside/nightside dynamic would have been better understood if the program allowed the user to see the data used for the calculations on a map or in a table. As shown, these interim steps are important pieces of the puzzle in understanding the results.

In terms of research in this field, research efforts looking to better understand these types of dynamics should be focused around running models with dynamic oceans. Not being able to account for the processes caused by dynamical oceans leaves out a huge piece of the picture when analyzing the existence and behavior of water on exoplanets. Additionally, future research into ocean survival and escape rate should look to include more systems of Hydrogen thermospheric escape such as those detailed in Hunten, 1973.

APPENDIX

Full summary of experiments done by Chen et. al. Experiments used in this paper are highlighted. Original caption included.

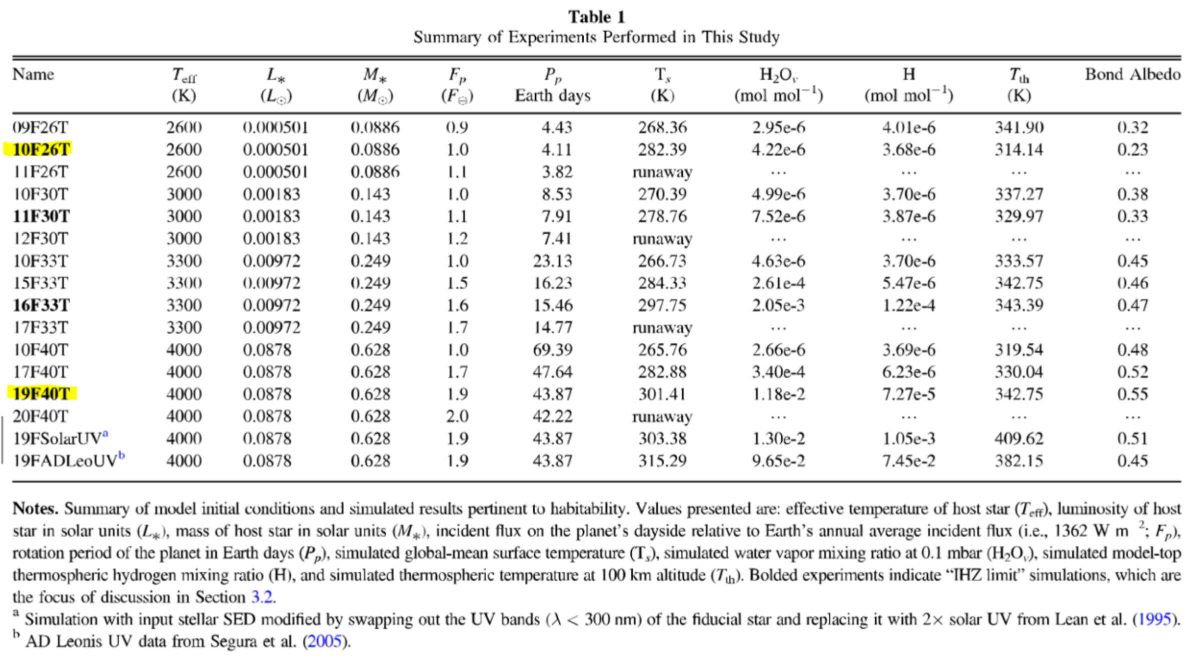


Table 1
Summary of Experiments Performed in This Study

Name	$T_{\rm eff}$ (K)	L_* ($L_\odot$)	M_* ($M_\odot$)	F_p ($F_\oplus$)	P_p Earth days	T_s (K)	H_2O_v (mol mol^{-1})	H (mol mol^{-1})	$T_{\rm th}$ (K)	Bond Albedo
09F26T	2600	0.000501	0.0886	0.9	4.43	268.36	2.95e-6	4.01e-6	341.90	0.32
10F26T	2600	0.000501	0.0886	1.0	4.11	282.39	4.22e-6	3.68e-6	314.14	0.23
11F26T	2600	0.000501	0.0886	1.1	3.82	runaway	…	…	…	…
10F30T	3000	0.00183	0.143	1.0	8.53	270.39	4.99e-6	3.70e-6	337.27	0.38
11F30T	3000	0.00183	0.143	1.1	7.91	278.76	7.52e-6	3.87e-6	329.97	0.33
12F30T	3000	0.00183	0.143	1.2	7.41	runaway	…	…	…	…
10F33T	3300	0.00972	0.249	1.0	23.13	266.73	4.63e-6	3.70e-6	333.57	0.45
15F33T	3300	0.00972	0.249	1.5	16.23	284.33	2.61e-4	5.47e-6	342.75	0.46
16F33T	3300	0.00972	0.249	1.6	15.46	297.75	2.05e-3	1.22e-4	343.39	0.47
17F33T	3300	0.00972	0.249	1.7	14.77	runaway	…	…	…	…
10F40T	4000	0.0878	0.628	1.0	69.39	265.76	2.66e-6	3.69e-6	319.54	0.48
17F40T	4000	0.0878	0.628	1.7	47.64	282.88	3.40e-4	6.23e-6	330.04	0.52
19F40T	4000	0.0878	0.628	1.9	43.87	301.41	1.18e-2	7.27e-5	342.75	0.55
20F40T	4000	0.0878	0.628	2.0	42.22	runaway	…	…	…	…
19FSolarUV[a]	4000	0.0878	0.628	1.9	43.87	303.38	1.30e-2	1.05e-3	409.62	0.51
19FADLeoUV[b]	4000	0.0878	0.628	1.9	43.87	315.29	9.65e-2	7.45e-2	382.15	0.45

Notes. Summary of model initial conditions and simulated results pertinent to habitability. Values presented are: effective temperature of host star ($T_{\rm eff}$), luminosity of host star in solar units (L_*), mass of host star in solar units (M_*), incident flux on the planet's dayside relative to Earth's annual average incident flux (i.e., 1362 W m^{-2}; F_p), rotation period of the planet in Earth days (P_p), simulated global-mean surface temperature (T_s), simulated water vapor mixing ratio at 0.1 mbar (H_2O_v), simulated model-top thermospheric hydrogen mixing ratio (H), and simulated thermospheric temperature at 100 km altitude ($T_{\rm th}$). Bolded experiments indicate "IHZ limit" simulations, which are the focus of discussion in Section 3.2.

[a] Simulation with input stellar SED modified by swapping out the UV bands ($\lambda < 300$ nm) of the fiducial star and replacing it with 2× solar UV from Lean et al. (1995).

[b] AD Leonis UV data from Segura et al. (2005).

REFERENCES

Burgasser, Adam J., and Eric E. Mamajek. “On the Age of the TRAPPIST-1 System.” *The Astrophysical Journal*, vol. 845, no. 2, 2017, p. 110., doi:10.3847/1538-4357/aa7fea.

Chen, Howard, et al. “Habitability and Spectroscopic Observability of Warm M-Dwarf Exoplanets Evaluated with a 3D Chemistry-Climate Model.” *The Astrophysical Journal*, vol. 886, no. 1, 2019, p. 16., doi:10.3847/1538-4357/ab4f7e.

Gonzales, Eileen C., et al. “A Reanalysis of the Fundamental Parameters and Age of TRAPPIST-1.” *The Astrophysical Journal*, vol. 886, no. 2, 2019, p. 131., doi:10.3847/1538-4357/ab48fc.

Hunten, Donald M. “The Escape of Light Gases from Planetary Atmospheres.” *Journal of the Atmospheric Sciences*, vol. 30, no. 8, 1973, pp. 1481–1494., doi:10.1175/1520-0469(1973)030<1481:teolgf>2.0.co;2.

Hunten, Donald M. “Thermal and Nonthermal Escape Mechanisms for Terrestrial Bodies.” *Planetary and Space Science*, vol. 30, no. 8, 1982, pp. 773–783., doi:10.1016/0032-0633(82)90110-6.

Kopparapu, Ravi Kumar, et al. “Habitable Moist Atmospheres on Terrestrial Planets near the Inner Edge of the Habitable Zone around M Dwarfs.” *The Astrophysical Journal*, vol. 845, no. 1, 2017, p. 5., doi:10.3847/1538-4357/aa7cf9.

Wolf, E. T., and O. B. Toon. “The Evolution of Habitable Climates under the Brightening Sun.” *Journal of Geophysical Research: Atmospheres*, vol. 120, no. 12, 2015, pp. 5775–5794., doi:10.1002/2015jd023302.

ADDITIONAL SOURCES

List of named colors¶. (n.d.). Retrieved December 06, 2020, from https://matplotlib.org/3.1.0/gallery/color/named_colors.html

N-D labeled arrays and datasets in Python¶. (n.d.). Retrieved December 06, 2020, from http://xarray.pydata.org/en/stable/

Python Dicitonaries. (n.d.). Retrieved December 06, 2020, from https://www.w3schools.com/python/python_dictionaries.asp

Setting the Font, Title, Legend Entries, and Axis Titles. (n.d.). Retrieved December 06, 2020, from https://plotly.com/python/figure-labels/

Static Image Export. (n.d.). Retrieved December 06, 2020, from https://plotly.com/python/static-image-export/

Tables. (n.d.). Retrieved December 06, 2020, from https://plotly.com/python/table/

ACKNOWLEDGEMENTS

I would like to primarily acknowledge Howard Chen for graciously allowing me to use his data and providing insight into his methods and approaches. I would also like to thank Dr. Suzan Van Der lee and Igor Eufrasio for their guidance on this project and on my development as a programmer in Python in general. I would like to acknowledge the help and support of Regan Stelander in figuring out how to make flowcharts, which was for some reason much more complex than it probably needed to be. I additionally want to acknowledge the guidance of Matthew Liu in coding etiquette and providing advice on formatting my final code.