# CSE 519 MID PROJECT PROGRESS REPORT

## How much do people sleep?

**Objective:**

Social media analysis sheds considerable light on human behavior, gaining statistical strength from the scale of such interactions. In this project, we will analyze Twitter data to get insight into factors affecting how much sleep different populations in the US receive.

**Introduction:**

Daily rhythms of activity and rest in human life are familiar to all of us, and there is a growing body of evidence indicating that disrupting this pattern has severe consequences for health. Despite its importance, it is poorly understood how daily activity varies geographically and seasonally. Further, it is unclear what role biological clocks and social constraints have in determining geographical variation in activity. Circadian clocks provide an innate pressure to sleep at night and are active during the day, but our internal rhythms can be at odds with the timing of social obligations, such as work and school.

In this project, we try to study how the different time zones in the US affect the sleep cycle of people using Twitter data. Twitter data has a timestamp of when the tweet was created and has location data by which we can determine the place of origin of tweet. We use these two features to categorize the tweets to different time zones and plot the twitter activity graph for these time zones to determine the sleeping pattern.

**Dataset:**

- We are using the hourly dataset provided by Prof. Jason Jones as our main twitter activity data. (We specifically used the data form **11th Nov 2018** for our analysis as the complete dataset wasn't available to us with sufficient time for analysis.)
- To categorize location data to different time zones we used data from below sources https://www.kaggle.com/geonames/geonames-database
- http://www.geonames.org/data-sources.htmlThe sunrise and sunset data are scrapped from https://www.timeanddate.com/

**Data Preprocessing:**

We tried getting data from Twitter API, but we could only get seven days of data which was too less for our analysis. We needed months of data and there were no freely available data online, so we are using Twitter data provided by Prof. Jason Jones as our primary data source.

Twitter data provided by Prof. Jason Jones is in text form which on direct conversion to data frame doesn't yield proper data frame structure, so we created a JSON schema that parses the data from text to a valid JSON and this JSON data is used to create our data frame. Since JSON parsing was taking more time we use **RapidJson** python library which parses JSON more quickly.

The user data is again a nested JSON data, so we extract it and create a new data table for user data. After this, we had two data tables, one with user data the other with the tweet data.
Following features were deprecated from Twitter from **Apr 2018**:

- utc_offset
- time_zone
- lang
- geo_enabled
- following
- follow_request_sent

- has_extended_profile
- notifications
- profile_location
- contributors_enabled
- profile_image_url
- profile_background_color
- profile_background_image_url
- profile_background_image_url_https
- profile_background_tile
- profile_link_color
- profile_sidebar_border_color
- profile_sidebar_fill_color
- profile_text_color
- profile_use_background_image
- is_translator
- is_translation_enabled
- translator_type

So, we couldn't use any of the above features, we dropped all of them from our data table. While proposing this project we thought we could use time_zone, utc_offset, profile_location, coordinates to determine the location from where the tweet was made. But we had to come up with a new way to determine the tweet location.

As we are determining the sleeping patterns for different time zones in the US, we are categorizing data from user-entered location data which is obtained from the location field of the tweet data. User entered location data doesn't give accurate location data as users can enter any text of their choice for the location filed, few users entered location data were "city of stars", "Sugar Trap"," on cloud 5", "I can't be found". We had to remove all data which didn't have the proper location data which reduced our data size considerably.

The hourly data contains tweets from all over the world, so we had to come with a logic where we select maximum possible tweets from the US. We filtered the location data with the map containing the major cities (around 2500) of the US and states (like Texas, Tx, tx, Los Angeles, LA, Los Angeles California). We had a time zone map for each of the cities and states, and we mapped the location data to it to get their respective time zones.

We converted the time from epoch timestamp in milliseconds to readable time for respective time zones and added a new column to which contains the local time for each tweet.
Even after the mapping, we had false location due to the following reasons:

- A location like Washington has ambiguity whether it is Washington, D.C. or Washington state.
- A location like New York can't be determined if it is New York state or New York City.
- The location Birmingham is present in 17 places of US and in 3 places in the UK, so we can't determine location accurately if the user entered just Birmingham. (Complete list can be found here : https://en.wikipedia.org/wiki/Birmingham_(disambiguation) )
    - Birmingham, Alabama
    - Birmingham, Alabama (Amtrak station)
    - Birmingham-Shuttlesworth International Airport
    - Birmingham, Connecticut
    - Birmingham, Kentucky

As it is too difficult to determine the tweet location for the above cases, we have approximated the location data. Where location with just Washington is treated as state and Washington, D.C is treated as a city, similarly New York is treated as state and New York City, NY is treated as city (in this case it doesn't matter which one we choose as both NY state and NYC are in same time zone).

**Approach 1:**

In this approach, we tried fetching the twitter data using the twitter python API. Using the tweepy API we were able to download 7 days of data for any specific twitter id.
The tweepy API response is a nested JSON object. We parsed this nested JSON into a data frame.
As we were not able to get large data using this approach, we moved on to approach 2.

**Approach 2:**

Below are the steps at a high level followed in this approach,

1. Data preprocessing: Read the twitter hourly text file and parse using JSON parser. Save this as a CSV file.
2. Read all the CSV hourly files into a Dataframe object and filter out based on location
3. Use the cleaned data for analysis.

The twitter data in the text file was in the JSON format. As there can be both deleted and tweet-created responses the JSON schema for these will be different. Therefore, to read this JSON data directly into a data frame object will not be proper, as the columns are not parsed properly by pandas.
So, we wrote a JSON schema to parse the JSON data. As json.dumps was taking a lot of time, we improved this using rapidjson. Once the twitter data was parsed using a JSON schema, this data can now be saved as a CSV file, so that it will be easy to load into a data frame using pandas. In the JSON schema, we have used the id, timestamp and location data as required columns because these are the most important columns used for our analysis.

Once all the hour's data is saved as CSV, we can directly load all these files into a single data frame.
In this data frame, we only filter out data that belong to the US timezone. Filtering out US-based records was not as easy as the location data on twitter does not maintain a standard structure.
For ex: A tweet from New York location, can have the location value as NY, US or New York or NYC or NY.
To get only tweets from us locations we have created 2 hashmaps which contains all possible combinations of city names and state names. If the location value of the tweet is present in the hashmap then it implies that the tweet is from a US location.

Timezone information:

Now one more factor to consider is the timezone. The timestamp_ms or created_at columns present in the data are in UTC format.
The twitter data provided was for Nov-11-2018. From May 2018 (https://twittercommunity.com/t/upcoming-changes-to-the-developer-platform/104603) time_zone and utc_offset values are not made available.
Therefore, to infer the timezone information, we used the location column. Based on the location city/state a timezone mapping is created and we added a new timezone column.

Tweet Analysis:
Based on the epoch time present in timestamp_ms and the newly added timezone column we can now calculate the local time at which the tweet was made, by using the python DateTime API.
We decided to do sleep analysis based on

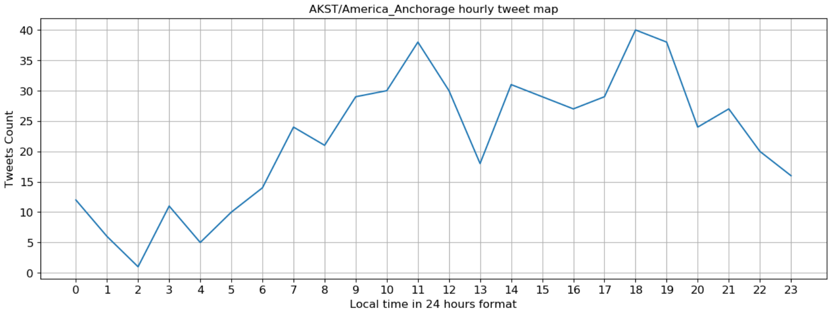
1. Different timezones of US
2. For US cities – LA, Houston, etc.,

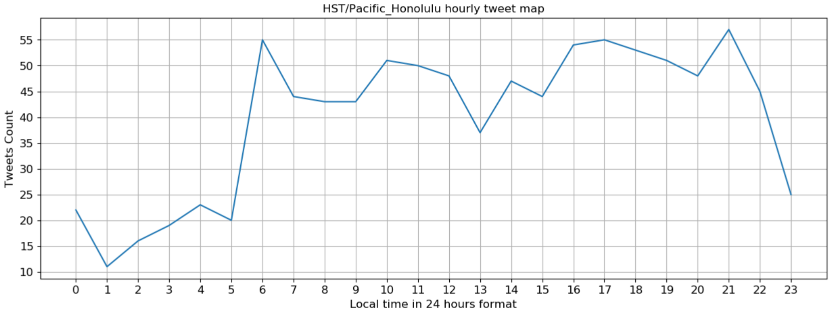
**Sleeping Range:**

To calculate the sleeping range a ranking function was written. This will calculate the ratio of the total number of tweets at any hour with the max number of the tweet (hr) for that day. If this ratio is less than 25% then we can conclude this range as the sleeping range.
Sleeping time range for all cities and different timezones can be calculated using this approach.

## Analysis:

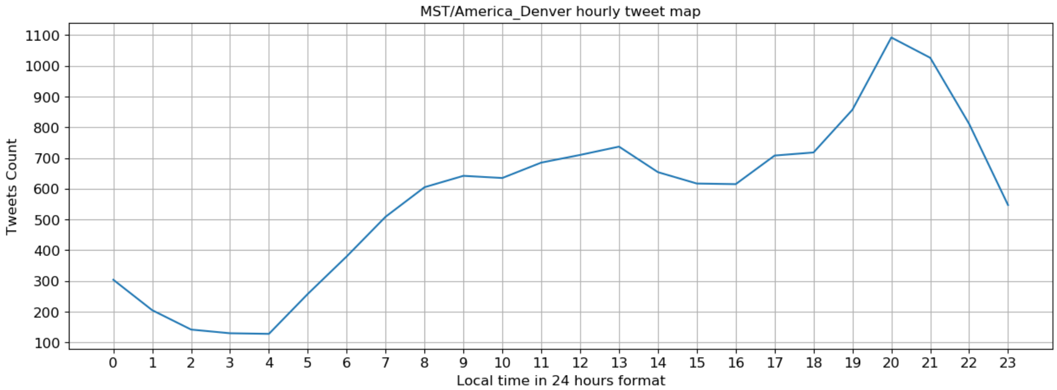
We plotted the hourly tweet map for six time zones in US as below. The time zones AKST, HST have very few tweets due to less population there, as a result the map is spiky but one can guess the sleeping hours by the steep dips in the tweeter activity.

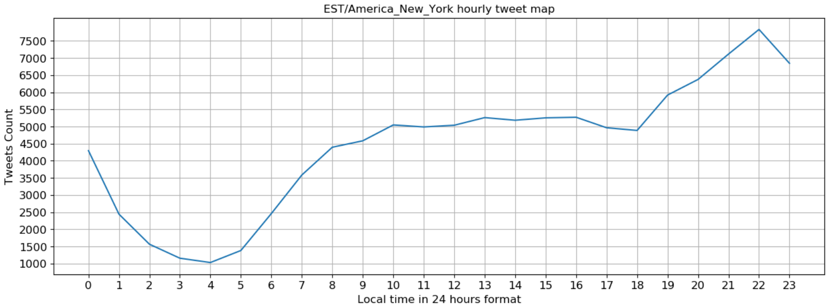




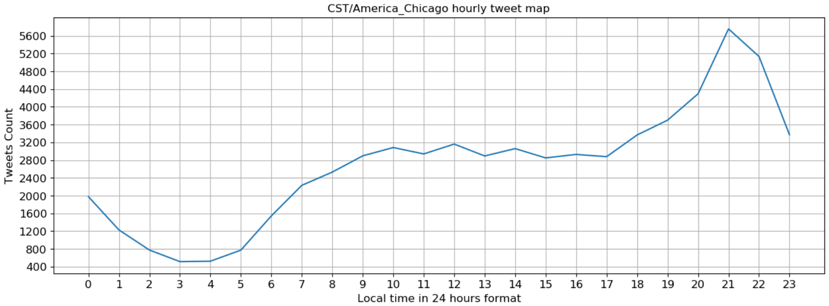
The time zones EST, CST have most twitter activity followed by time zones PST and MST. The curve for EST which has most twitter activity is almost smooth expect for the sudden rise and fall in the hour range 18-23. The smoothness indicates the gradual increase in the twitter activity from morning till evening and the sudden increase indicates that the majority of the population use twitter in late evenings, one reason might be that the vast population belong to the day jobs or are occupied with some activities during day time and they get free time during the evening when they use Twitter. The sudden decrease indicates that vast majority of people use twitter just before their sleep, so when they sleep the twitter activity decreases drastically.

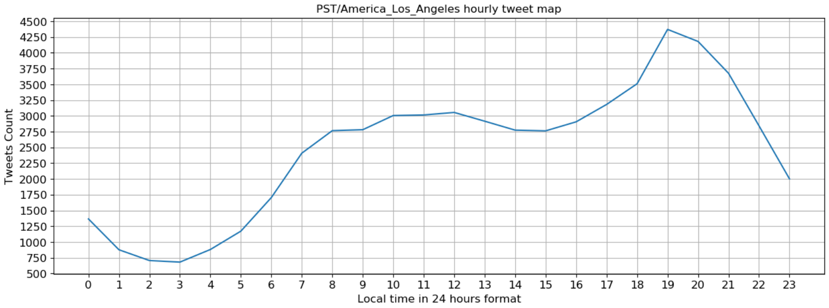
The sleeping hours for MST can be said to be in the time range 12:00am – 6:00am and for EST it will in the range 12:30am – 6:00am. (But this analysis is only from one day of data, we will verify this for the long-term data before predicting the final results.)



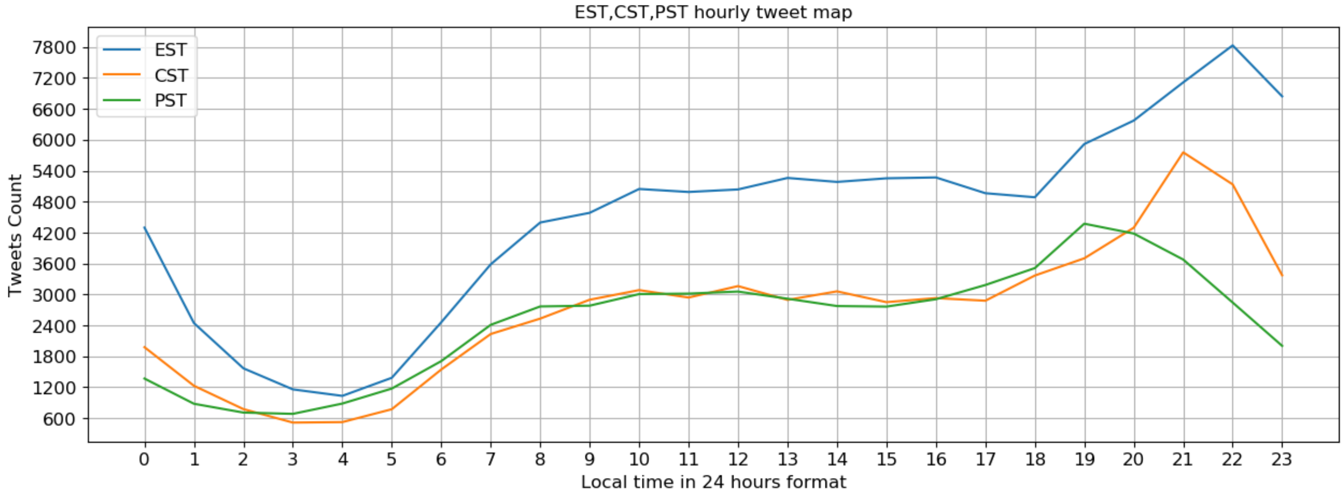


As seen from the graphs people in MST go to sleep around 8:00pm whereas the people in EST go to sleep around 10:00pm.



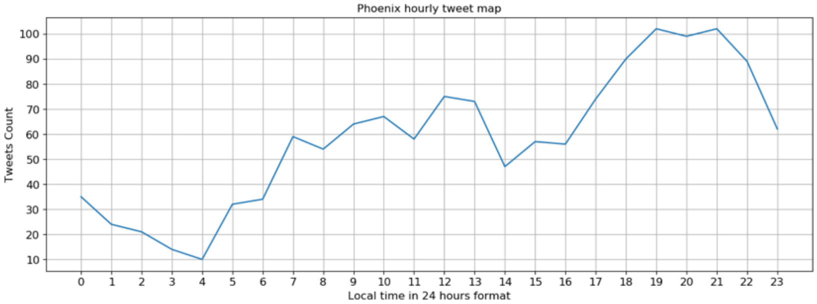


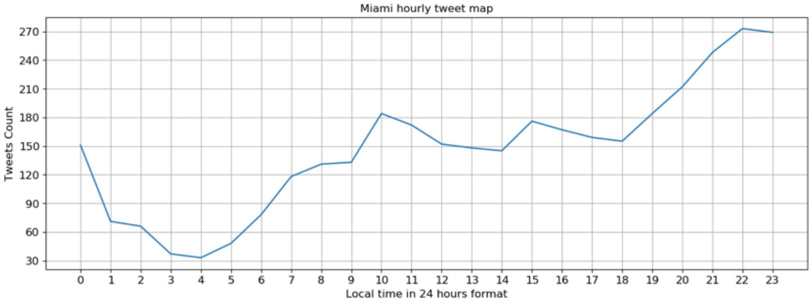
People in CST go to sleep around 9:00pm and people in PST go the sleep around 7:00pm-9:00pm.

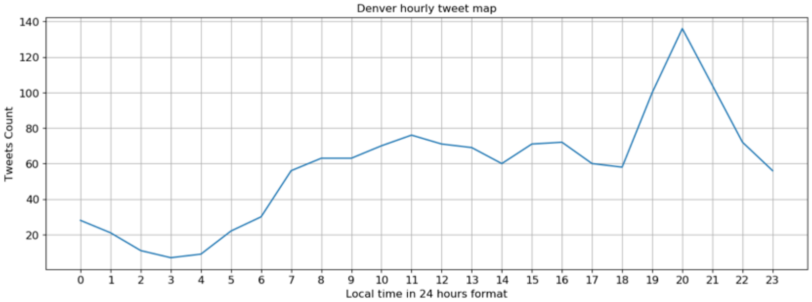


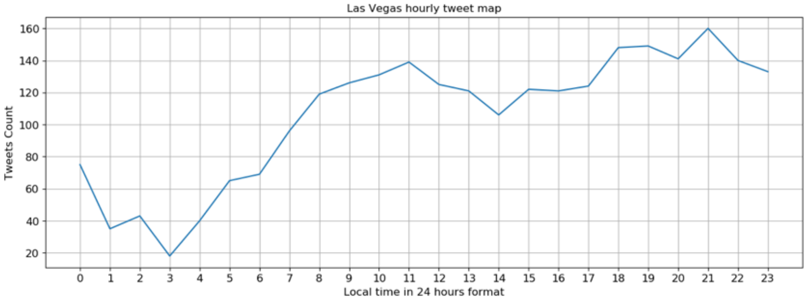
Above is the graph which overlays the EST, CST, PST time zones and all of them follow similar sleep cycles with small variations in the time they go to bed.

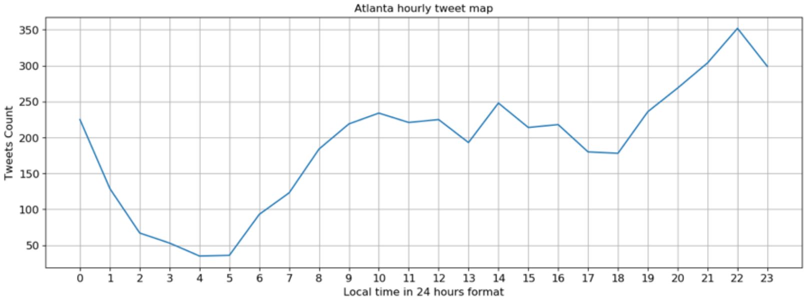
Next, we analysed the sleeping patterns for some major cities in each time zones. Below are the twitter activity graphs for cities Denver, Phoenix, Miami, Las Vegas, Atlanta, Chicago, Los Angeles and Huston.

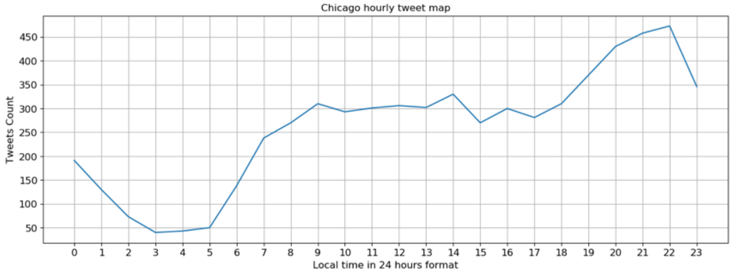


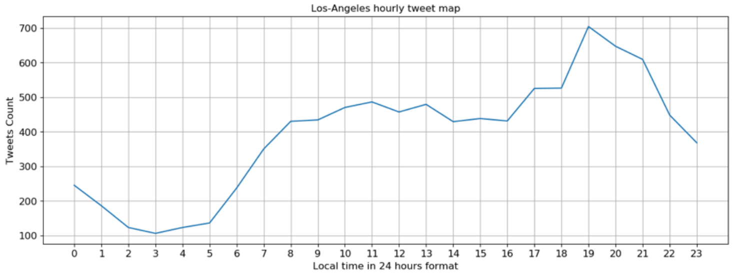


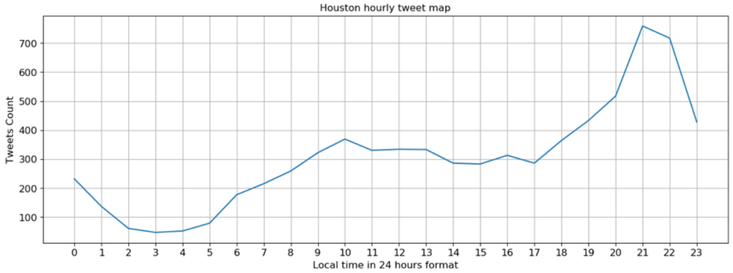


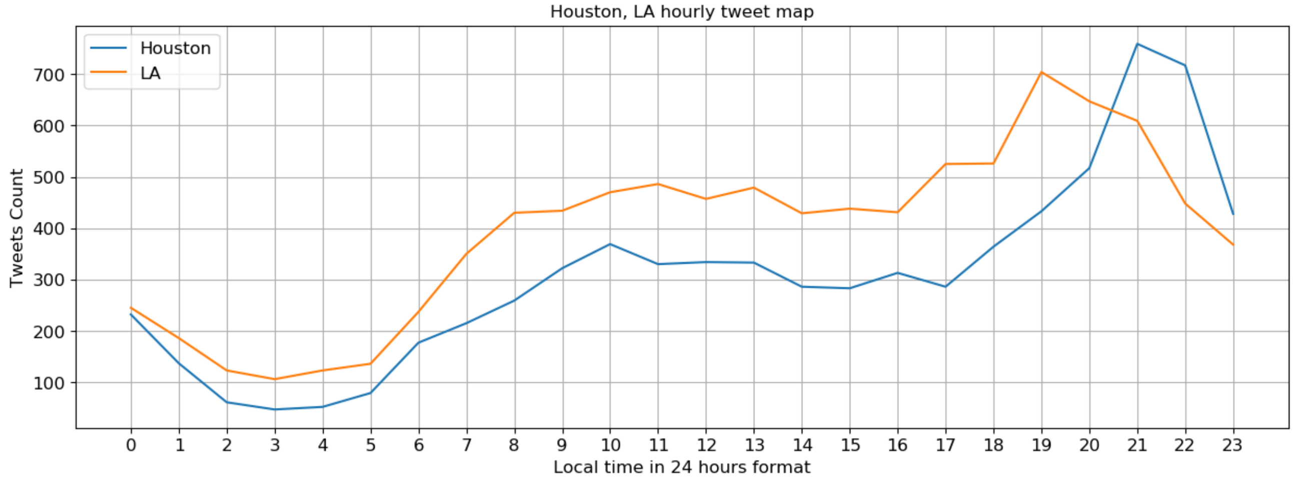












The overlay graph for LA and Houston show the distinctions in them while having similar sleeping time range. The twitter activity reduces for LA around 7:00pm whereas for Houston it reduces around 10:00pm.

**Next Steps:**

- For now, we have done the analysis for one day of data. We plan to extend this to one year of data.
- We have scarped the sunrise and sunset data from timeanddate.com, next we try to get the correlation between the sleeping hours with the sunrise, sunset timings.
- We will find the correlation between the sleeping pattern between the different time zones of the US and the correlation between a city (city with most twitter activity) and its time zone.
- We will find the effects of DST (Daylight saving time) with sleeping patterns.
- We plan to construct a feature engineering model which will predict the sleeping pattern based on the parameters like sunrise, sunset, temperature.