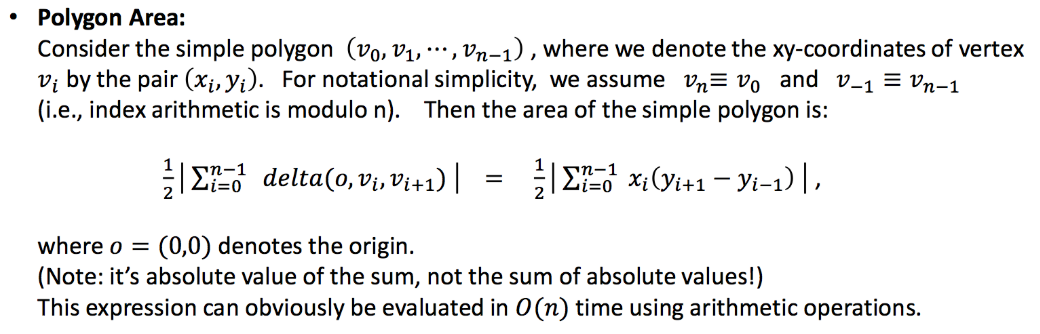
**1. Introduction**

Given the GPS records, time, and weather information, the program calculates the distance traveled, path sinuosity, woodland preference index, convex hull points, convex hull area, and the time slots of the activities for each cow.

1. **Logic**

* All the GPS data are read from the data file, and are stored and **sorted by cowid, date, time**.
* The program is coded to calculate the distance traveled, path sinuosity, woodland preference index. The logic of this part is hard to follow. Time is needed to understand it.
* The code of the convex hull is calculate following [Andrew's monotone chain convex hull algorithm](https://en.wikibooks.org/wiki/Algorithm_Implementation/Geometry/Convex_hull/Monotone_chain). The function accept the GPS locations of one cow, **the vertices of the convex hull are returned**. The complexity for calculation on **two-dimensional space** is **O(nlog(n))**.
* **The returned points are used to calculate the area of convex hull** by using following formula

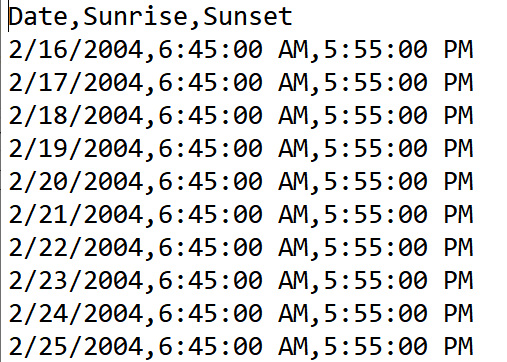


* Given two velocity parameters, one for walking **v\_walking**, one for running **v\_running**. The velocity of a cow is computed by two consecutive GPS records. If the cow's velocity is **[0,v\_walking]**, we assume **the cow was resting**. Similarly, if the cow's velocity is **(v\_walking, v\_running]**, **the cow is assumed that it was walking**. If the speed of the cow is **greater that v\_running**, we think **the cow is running**.
* The function **movementParitionForCow** counts the percentage of each cow that was resting, walking and running during the whole day, **the pre-day, the daytime, and the post-day** according given time date.
* The function **time\_slot** **identifies the activity time slots of the each cow in each day**.

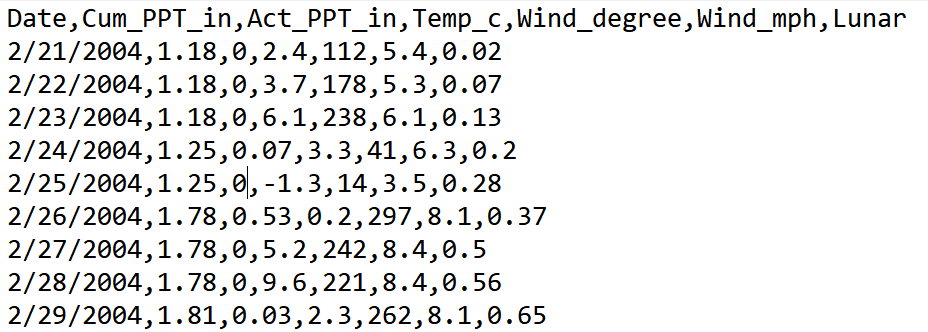
1. **Put the files into the folder**
2. Create a new folder in your Desktop or where you desire to put the GRAZEACT program.
3. Put the jar file “GRAZEACT.jar” into the folder.
4. Put files into the folder, which **default names are DataWithPosition.csv, Weather.csv and Time.csv respectively.**

The **format** of the file is showed as following:

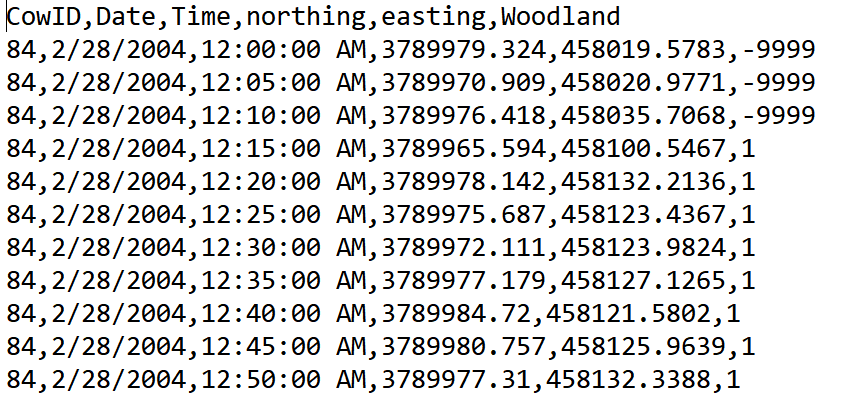
1. Convert your **time data(Time.csv)** into the CSV format and use the comma as the delimiter. In total, the CSV file should have 3 columns of basic information. They are: **Date, Sunrise, Sunset.** The format of Date is **yyyy-MM-dd**. The format of Sunrise and Sunset is **HH:mm:dd AM/PM**.



1. Convert your **weather data(Weather.csv)** into the CSV format and use the comma as the delimiter. In total, the CSV file should have 7 columns of basic information. They are: **Date, Cum\_PPT\_in, Act\_PPT\_in, Temp\_c, Wind\_degree, Wind\_mph, Lunar.** The format of Date is **MM/dd/yyyy**. The Cum\_PPT\_in, Act\_PPT\_in, Temp\_c, Wind\_degree, Wind\_mph, Lunar are **float numbers**.



1. Convert your **GPS data (DataWithPosition.csv)** into the CSV format and use the comma as the delimiter. In total, the CSV file should have 6 columns of basic information. They are: **CowID, Date, Time, northing, easting, Woodland.** The CowID is the unique id of each cow. It’s could be an integer or a string of text. The format of Date is **MM/dd/yyyy**. The format of time is **HH:mm:dd AM/PM**. The northing and easting are **float numbers** that represents the location of the cow at the time. The **Woodland** are **integer numbers** that indicates a cow is visiting a woodland or not in current time**.**



1. **Run The Code**

You could run the code from the Command line on Windows/Linux/MacOS. The results will put in the same folder as the input files and the java program

Text, letter

Description automatically generated

**The parameters:**

* Input the **position file name** (Default: DataWithPosition.csv)
* Input the **weather file name** (Default: Weather.csv)
* Input the **time file Name** (Default: Time.csv)
* Input the **upper bound of the speed (m/min) of the rest movement** (Default 5 m/min)
* Input the **upper bound of the speed (m/min) of the grazing movement** (Default 15 m/min)
* Input the **threshold of the speed (m/min)** to determine whether one GPS record is active or not (Default: 4 m/min)
* Input **the number of the lag** (number of GPS records need to be considered in previous and following, default is 5)

1. **Results of the code**

* **time\_slots.csv** (the time slots of the activities for each cow in each day, which are sorted by CowID, then by date)

Table

Description automatically generated

* **movement\_partition.csv(**calculates cows movement partition, represented as percentage, according to the user inputted upper bound of the rest movement speed and the grazing movement speed. And all data are partitioned into three time periods (pre-day, day-time and post-day), which are sorted by Cow\_id, then by date**)**

Text

Description automatically generated

* **MCP\_Results.csv** (Calculates convex hull vertices of cow movements on each day, pre-day, day-time and post-day, which are sorted by Cow\_id, then by day time partition, Date)

Text

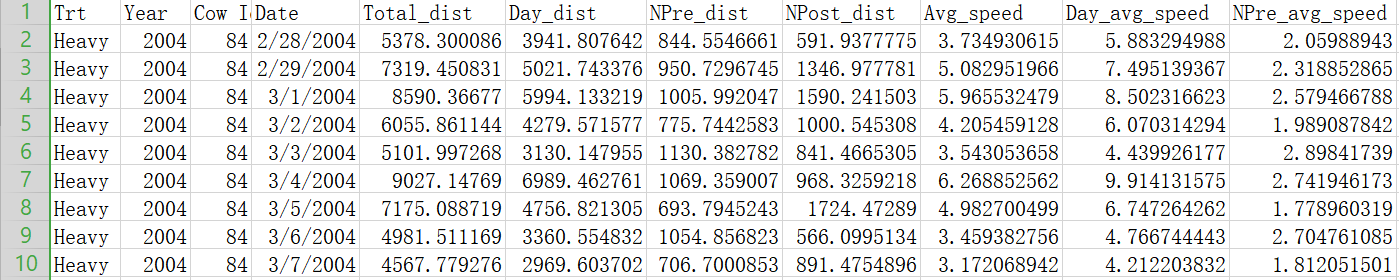
Description automatically generated

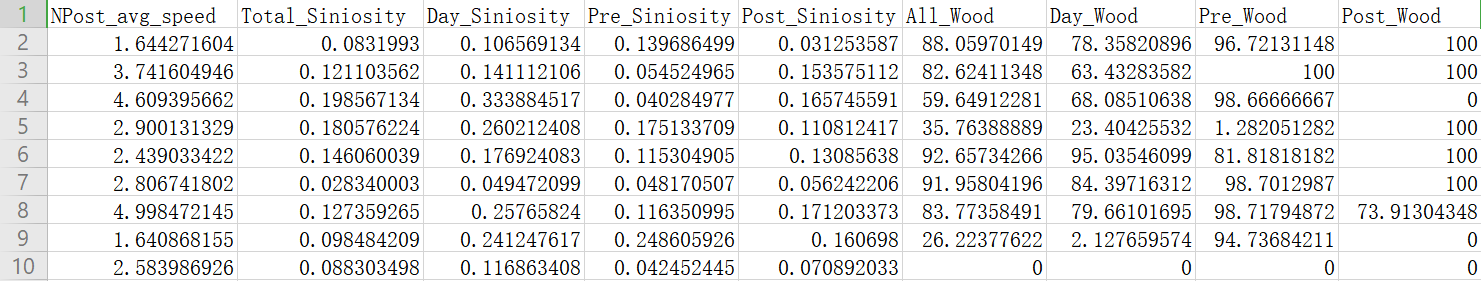
* **MCP\_Area\_Results.csv**(Calculates convex hull area of cow movements on each day, pre-day, day-time and post-day, which are sorted by Cow\_id, then by Date)

Text

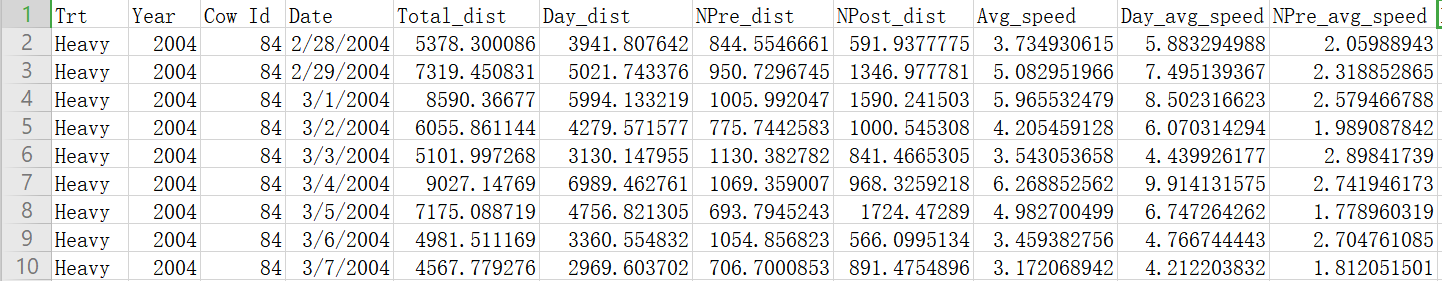
Description automatically generated

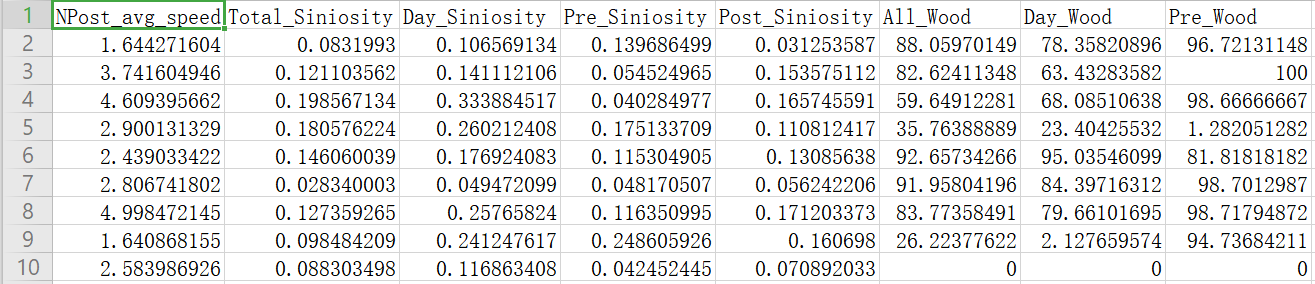
* **DistanceandSiniosity.csv (**calculates the distance traveled, path sinuosity, average speed, and woodland preference index (the percentage of a cow that visited woodland among whole GPS records) of a cow in the whole day, pre-day, day-time and post-day, which are sorted by cowID, then by date. And all data are partitioned into three time periods**)**

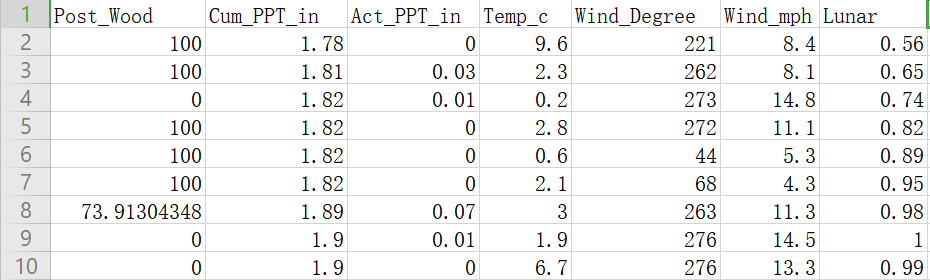




* **CompleteProcessedData.csv (**Besides the results in the **DistanceandSiniosity.csv,** the weather data are extracted from weather.csv and appended to the end of the file.)







Design logic:

The program needs to calculate (1) distance traveled, (2) path sinuosity, (3) woodland preference index, (4) convex hull points, (5) convex hull area, and (6) the time slots of the activities for each cow.

1. The distance is calculated this way:
   1. The inputs are (pseudocode input)
   2. The calculation steps (pseudocode to explain the logic)
   3. Related Java source files:
      1. xxx. Java (functions xx, xx)
      2. xxx2.Java (functions xx, xx)
      3. xxx3.hava (all the functions)
2. The path sinuosity