Project Update

2019-03-19

Calculating thermal time indices and merging UAV data.

Name: Byron Evers

Semester: Spring 2019

Project area: Agronomy

Table of contents

Objective

1. Write a python function to calculate three thermal time indices for all UAV collection dates
   - growing degree days (GDD)
   - physiological days (Pdays)
   - biometeorological time (BMT)
2. Merge all of the UAV reflectance data, plot level phenotypic data and the calculated thermal values into one .csv file.

Motivation

- Poland labs current UAV pipeline includes stitching photos and extracting plot level reflectance data through Agisoft software.
- Data recived from this process is in either an Excel or csv file.
- The data set includes reflectance values for 5 indiviual bands (R,G,B,RE and NIR) and 3 vegitative indices (NDVI, NDRE and GNDVI).
- Thermal time indices are important for data analysis between years
- Comparing thermal time indices maybe usefull in plots with diverse germplam

Equations

Growing Degree Days (GDD):

$$GDD = \sum_{Planting}^{Harvest}(\frac{Tmax+Tmin}{2})-Tbase$$

Physological Days (Pdays):

$$Pdays = \frac{1}{24}(5P(T_1)+8P(T_2)+8P(T_3)+3P(T_4))$$
Where
$T_1=Tmin$
$T_2=\frac{(2Tmin)+Tmax}{3}$

$T_3=\frac{Tmin+(2Tmax)}{3}$
$T_4=Tmax$

And P is
$P=0$ When $T <=Tmin$
$P=k(1-\frac{(T-Topt)2}{(Topt-Tmin)2})$ when $Tmin <= T <=Topt$
$P=k(1-\frac{(T-Topt)2}{(Tmax-Topt)2})$ when $Topt <= T <=Tmax$
$P=0$ when $T >=Tmax$

Biometeorological Time

$$BMT =\sum_{Planting}^{Harvest}{[a_1(L-a_0) + a_2(L-a_0)^2]}$$
$${[b_1(L-b_0) + b_2(L-b_0)^2]}$$
$$$$

Where
L= daily photoperiod
a_0= base daylength
b_0= base temperature
$a_1, a_2, b_1, b_2, d_1, d_2$ are response coefficents

Progress

- [X] Downloaded data from KSU Mesonet as a csv
- [X] Defined and imported needed modules
- [X] Imported data as a pands dataframe
- [X] Edited the dataframe
- Define user inputs needed for the cacluations and provide and place to enter
  - [X] Planting Date
  - [X] Harvest Date
  - [X] tbase
  - [X] topt
  - [X] tmax
- Define function for:
  - [x] Pdays
  - [ ] Pdays

    - [ ] BMT
- Visulize GDD with matplotlib. Include biophysical thermal time predictors
    - [x] Tillering
    - [x] Flowering
    - [x] Grain Fill

Road Blocks

- Pdays how to incorperate two conditions
    - np.where?
- Source data and how to incorperate photoperiod
- Importing UAV data from Excel particularly multiple tabs
- Formating UAV data from a table format to a database formate

Examples

Mesonet Input

sketch_image

Define GDD Function

```
#Define function
def GDD(df):
    """
    Clacluates an individual and cumluative value for GDD for ea

    Input: Pandas dataframe. Minimum required columns include:
        T_max= daily maximum temprerature
        T_min= daily minimum temperatrue
        Date = date of collection

    Output: Pandas dataframe. In addition the columns in the dat
        GDD= The growing degree day value for that individual da
        cum_GDD = The cumlative growing degree day value for all
    Byron Evers

    2019-03-06
    """
    df.Tmin = df.Tmin.astype(float)
    df.Tmax = df.Tmax.astype(float)
    df.Date =  pd.to_datetime(df.Date,format='%Y-%m-%d')
    df = df.drop(df[df.Date < plantDate].index)
    df = df.drop(df[df.Date > harvestDate].index)
    df['Tbase']=tbase
    df['tavg'] =((df.Tmax+df.Tmin)/2)
    values = np.where(df.tavg < tbase, df.Tbase, df.tavg).astype
    df['GDD']=(values)-df.Tbase
    df['cum_GDD'] = df.GDD.cumsum()
    return df
```

Dataframe Output

sketch_image

Graph

sketch_image

```
'''#size plot
plt.figure(figsize=(12,14))



#plot 2017 Data
plt.subplot(2,1,2)
plt.plot(df17.Date,df17.cum_GDD)
plt.ylabel('GDD', fontsize =24)
plt.xticks(rotation=60)

#plot 2018 Data
plt.subplot(2,1,2)
plt.plot(df18.Date,df18.cum_GDD, 'k')
plt.ylabel('GDD', fontsize =24)
plt.xticks(rotation=60)

#plot stage prediction lines
plt.plot(df17.Date, df17.Tillering, '--y')
plt.plot(df17.Date, df17.Flower, '--k')
plt.plot(df17.Date, df17.GrainFill, '--r')
plt.plot(df18.Date, df18.Tillering, '--y')
plt.plot(df18.Date, df18.Flower, '--k')
plt.plot(df18.Date, df18.GrainFill, '--r')



# edit plot
plt.title('GDD Comparison by Year', size=24)

plt.legend(['17-GDD','18-GDD','Tillering Est.','Flowering Est.',
plt.show()'''
```

```
"#size plot\nplt.figure(figsize=(12,14))\n\n\n#plot 2017 Data\np
```