Ideas for Sucrose Prediction Problem and Research Questions

Survey on Sucrose Prediction Using Spectral Indices:

* [Generalizable Prediction](#_iuih0nwew2ow)[[1]](#footnote-0) [--- Mixed Model (with Fixed and Random Effects)](#_iuih0nwew2ow)
* [Selecting Variables](#_aybeiyt71mo0)[[2]](#footnote-1) [--- Bayesian Causal Inference for Spatial Data](#_aybeiyt71mo0)
* [Platform for Analyzing Spectral Indices at scale ---](#_bvyvj4ma7jfv) [Wherobots](https://wherobots.com/use-cases/)
* [White Papers/ Blogs from Startups ---](#_dtb1sxk7ef93)
* [Research Questions ---](#_jmvuexfpwfw8)
* [Modeling - V1](#_83gs3h5pvq86)

# Sucrose Prediction Techniques (Generalizable over geography):

| **Descriptive Stats/ Techique** | **Metric Calculated** | **Operational Use Case** |
| --- | --- | --- |
| NDVI based mathematical model for Crop Stage Detection  Observe progression in key moments of crop growth -- germination, vegetative, flowering, seed fill, maturity | Growing Degree Days:  Based on heat units, gives:  -- Current Stage of Crop  -- Progression of Crop | Metrics help gauge whether  -- harvest is on time,  -- crop is progressing healthily towards maturity |
| Descriptive Statistics of various spectral indices -  CG - Derived from Normalized Difference Vegetation Index (NDVI)  CNU - Utilizing Normalized Difference RedEdge Index (NDRE)  CWS - Based on Land Surface Water Index (LSWI) | Canopy Greenness (CG)  Canopy Nitrogen Uptake (CNU)  Canopy Water Stress (CWS) | CG -- Vegetation Health  CNU -- Measure of Crop Vigor  CWS -- Total Liquid Water Available in both Vegetation and Supporting Soil |
| Sucrose Estimation in Sugarcane  Commercial Cane Sugar (CCS) |  | Measure quality of sugar |

**Overview of Method -**

* Static + semi-empirical relationships between Ground Biomass and Vegetation Index
* Vegetation Index = Calculated from combinations of different bands
* Crop Field Polygon → Extract farm of interest → Obtain Spectrum Information (Spectrum = Light Intensity Distribution as a function of wavelength)
* Traditional method of Yield Calculation →

Field Images Extraction →

NDVI Calculation for each image →

Modeling and index calculation →

Yield Prediction Using Regression Analysis

* Model:
* Inputs:
  + NDVI (peak of biomass growth) ---- RapidEye Images
  + NDVI (peak of sucrose growth) ---- Rapid Eye Images
  + NDVI (elongation and ripening) --- time series LandSat images (3 months)

Modeling Setup

Here,

* =
* =
* =
* U0 = a random effect which is the site number.
* p = NDWI in the period of peak sucrose accumulation.

Robustness checks for weather

Rainfall and humidity plays an important role in sugarcane sucrose accumulation. How?

**Rainfall ↑** → **Body Growth Steps ↑** → **Potential for Sucrose Accumulation ↑**

**Humidity ↓ → Ripening Steps ↑ → Potential for Sucrose Accumulation ↑**

Modeling Choice -- Introduce soil moisture in the model → Capture Using NDWI

→ Mixed Method Model Using Both Fixed Effects and Random Effects

For random effects specification, divide the 21 sampling points to two groups using two different sites which have different environments for sugarcane growth.

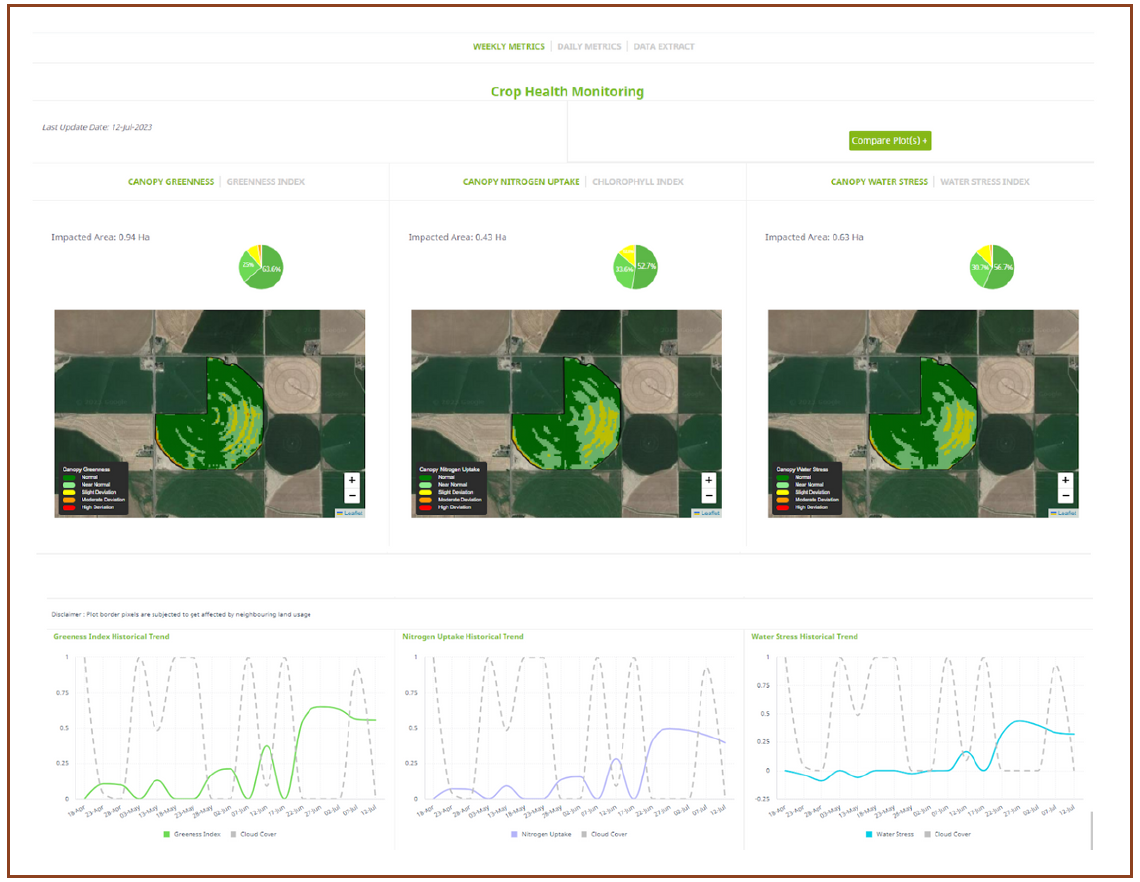
**Prediction Results**

| Method | Model (NDWI) | Model | USDA |
| --- | --- | --- | --- |
| **Accuracy** | 90.4% | 81.0% | 57.3% |
| **MAE** | 0.24 | 0.33 | 0.76 |
| **SD** | 0.27 | 0.33 | 0.76 |

Random Effects -- Varying across sites.

Fixed Effects -- Constant across individual sites, can change with time.

Crop Health Summary Statistics --



*Source - Crop Health Metrics: A Comprehensive Guide to Monitoring Crop Vitality - Whitepaper, CropIn Plot-Level Intelligence*

# Selecting Variables[[3]](#footnote-2) --- Bayesian Causal Inference for Spatial Data

* Biggest challenge in spatial causal inference→
  + Complex Correlation Structures
  + Interference between treatment at one location and outcome at others
* Paper discusses
  + Methods that exploit spatial structure to account for unmeasured confounding variables
  + Causal Analysis in presence of spatial interference including several common assumptions to reduce the complexity of interference patterns under consideration
  + Method extended to spatio-temporal analysis
  + Code to implement many of the methods using the popular Bayesian software OpenBUGS is open-sourced.

# Platforms for Analyzing Spectral Indices:

* [Wherebots](https://wherobots.com/use-cases/)
* Write about Use Cases, and GeoParquet Workshop Highlights

Uses:

* + develop scalable spatial data analytics applications on data, processed in your own data stack, and deployed anywhere.

Innovation in the Data Format:

* [GeoParquet and Its Ecosystem](https://geoparquet.org/): Started as a repository to survey ecosystem for geospatial cloud data warehouses
* Parquet is a format that supports interoperability
* [Key Stakeholders maintaining the repository](https://azure.microsoft.com/en-us/blog/geospatial-imagery-unlocks-new-cloud-computing-scenarios-on-azure/)
* Platforms:
  + Apache Sedona™ is a cluster computing system for processing large-scale spatial data. Sedona extends existing cluster computing systems, such as Apache Spark, Apache Flink, and Snowflake, with a set of out-of-the-box distributed Spatial Datasets and Spatial SQL that efficiently load, process, and analyze large-scale spatial data across machines.
  + Sedona runs 2X - 10X faster than other Spark-based geospatial data systems on computation-intensive query workloads.

# White Papers/ Blogs from Startups ---

* [CropIn Blog](https://www.cropin.com/blogs):Techniques are not open source

# Research Questions ---

* Given a set of farm parcels’ spatially tracked metrics over time (spectral indices), and another few farms’ ground truth, how to build a robust model that predicts sucrose content?

# **Modeling - V1 ---**

Sources ----

---- *Advanced and Specialized Statistics with Stata*

<https://www.linkedin.com/learning/advanced-and-specialized-statistics-with-stata>

--- *Machine Learning and Foundations: Prediction, Causal Inference and Modeling*

<https://www.linkedin.com/learning/machine-learning-and-ai-foundations-causal-inference-and-modeling/>

Problem Statement

* Explore variables Yield and it’s transitions
* Estimate a Pooled OLS, Random Effects and Fixed Effects regression model
* Regress Yield on NDWI, NDVI, and rainfall/humidity
* Test which of these models should be used: Hausman Test

AIC, BIC, **MAE**, Adjusted R^2

Hypothesis:

---- Control variables --- Sunlight, Moisture, Vigor

---- Treatment: Moisture, Vigor (Humidity and Rainfall)

[Colab - Code Book](https://colab.research.google.com/drive/1sSR6stZDJMtT2SUMnNfrq_thVEiV3ygO?usp=sharing):

Steps to run the code --

1. Use the spreadsheet “Sugarcane farm data.xlsx”
2. Variables or column names currently --- “Master Farm ID”, “Date”, “NDVI”, “NDWI”, “CIG”, “Yield”, “Sucrose Content”
3. Run the “Pooled OLS” code snippet below
4. Add columns “rainfall”, “humidity” for each farm ID.
5. Run the “Fixed Effects Model” snippet below and “Fixed plus Random Effects Model”
6. Run the “Hausman Test” code snippet below to compare the three models
7. Run the “Prediction Accuracy” code snippet below to measure the preduction accuracy of the three models.

Description of Models:

The terms "OLS" (Ordinary Least Squares) and "Pooled OLS" refer to regression analysis methods, but they are used in slightly different contexts, primarily distinguished by the structure of the data they are applied to.

### **Ordinary Least Squares (OLS)**

OLS is a type of linear regression that estimates the relationship between one or more independent variables and a dependent variable by minimizing the sum of the squares of the differences between the observed and predicted values. OLS is used across various data settings, from simple cross-sectional data analyses to more complex time series and panel data analyses. The key assumption of OLS is that the relationship between the variables is linear, and it aims to find the linear equation that best predicts the dependent variable from the independent variables.

### **Pooled Ordinary Least Squares (Pooled OLS)**

Pooled OLS is a specific application of OLS in the context of panel data, where you have observations over time for the same cross-sectional units (e.g., individuals, companies, countries). Panel data, therefore, have two dimensions: a cross-section dimension and a time series dimension. Pooled OLS treats the panel data as one large cross-section, ignoring the panel structure. It assumes that the slope coefficients are constant across time and entities, pooling all observations together to estimate a common intercept and slope coefficients for the independent variables.

### **Key Differences ----**

* **Data Structure:**
  + OLS can be applied to simple cross-sectional data or time series data.
  + Pooled OLS is specifically used for panel data, combining both cross-sectional and time series data.
* **Assumptions:**
  + OLS assumes a linear relationship between variables in a single cross-section or time series.
  + Pooled OLS assumes that this linear relationship is constant across both time and entities in panel data, ignoring any within-entity autocorrelation or heterogeneity across entities.
* **Application:**
  + OLS is more general and can be applied to various types of data structures.
  + Pooled OLS is particularly used when the analyst believes that the cross-sectional units in the panel data are homogeneous, meaning they have the same intercept and slope, and there is no need to account for any entity-specific or time-specific effects.
* **Interpretation:**
  + OLS results are interpreted as the average effect of changes in independent variables on the dependent variable for the given data.
  + Pooled OLS results are interpreted similarly but under the assumption that these effects are constant across all entities and time periods in the panel data.

### **Conclusion**

While OLS is a broader term for a regression technique, Pooled OLS is a specific application of OLS for panel data under certain assumptions. The choice between using OLS or Pooled OLS depends on the structure of the data and the research questions being addressed. If there's evidence of significant variation over time or across entities, more sophisticated panel data techniques such as fixed effects or random effects models may be more appropriate than Pooled OLS.

# **Pooled OLS**

import pandas as pd

import numpy as np

from statsmodels.api import OLS, add\_constant

# Load the dataset

df = pd.read\_excel("/path/to/your/dataset.xlsx", header=1, names=['Index', 'Farm ID', 'Date', 'NDVI', 'NDWI', 'CIG', 'Yield', 'Sucrose Content'])

# Convert 'Date' to datetime format and ensure other relevant columns are in numeric format

df['Date'] = pd.to\_datetime(df['Date'])

df['NDVI'] = pd.to\_numeric(df['NDVI'])

df['NDWI'] = pd.to\_numeric(df['NDWI'])

df['CIG'] = pd.to\_numeric(df['CIG'])

df['Sucrose Content'] = pd.to\_numeric(df['Sucrose Content'])

# Prepare the data for regression

X = df[['Date', 'NDVI', 'NDWI', 'CIG']]

X = X.copy() # To avoid SettingWithCopyWarning

# Convert 'Date' to a numeric value (e.g., days since the earliest date)

X['Date'] = (X['Date'] - X['Date'].min()) / np.timedelta64(1, 'D')

# Add constant for OLS regression

X = add\_constant(X)

y = df['Sucrose Content']

# Perform the regression

model = OLS(y, X).fit()

# Print the summary of the regression

print(model.summary())

# **Fixed Effects Model**

A fixed effects model accounts for entity-specific heterogeneity by allowing for a unique intercept for each entity (e.g., Farm ID in your dataset). This approach effectively controls for all unobserved, time-invariant differences across entities. Here, we cannot directly include time-invariant variables (like Farm ID) as regressors, but we control for them through entity-specific intercepts.

import statsmodels.api as sm

import pandas as pd

# Assuming 'df' is your DataFrame and it now includes 'rainfall' and 'humidity'

# Convert 'Date' to a numeric value (e.g., days since the earliest date)

df['Date'] = (df['Date'] - df['Date'].min()) / np.timedelta64(1, 'D')

# Create dummy variables for each Farm ID to include as fixed effects

fe = pd.get\_dummies(df['Farm ID'], drop\_first=True)

# Prepare the model data

X = df[['Date', 'NDVI', 'NDWI', 'CIG', 'rainfall', 'humidity']]

X = pd.concat([X, fe], axis=1) # Add fixed effects

X = sm.add\_constant(X) # Add constant

y = df['Sucrose Content']

# Fit the fixed effects model

model\_fe = sm.OLS(y, X).fit()

print(model\_fe.summary())

# **Fixed Plus Random Effects Model (Mixed Effects Model)**

A mixed effects model includes both fixed effects (as in the fixed effects model) and random effects, where random effects allow for random variation across entities (e.g., different slopes for different farms). This model is particularly useful when you believe that some of the variation in your dependent variable is attributable to differences that vary across entities but are not directly observed.

from statsmodels.regression.mixed\_linear\_model import MixedLM

# Prepare the data

X = df[['Date', 'NDVI', 'NDWI', 'CIG', 'rainfall', 'humidity']]

X = sm.add\_constant(X) # Add constant

y = df['Sucrose Content']

groups = df['Farm ID'] # Define groups for random effects

# Fit the mixed effects model

model\_mixed = MixedLM(y, X, groups=groups).fit()

print(model\_mixed.summary())

In both models, we added "rainfall" and "humidity" to the list of independent variables. The fixed effects model (model\_fe) controls for unobserved, entity-specific heterogeneity by including a dummy variable for each Farm ID, effectively giving each farm its intercept. The mixed effects model (model\_mixed) goes further by allowing both fixed effects (as global coefficients for your predictors) and random effects (allowing for random variation in intercepts or slopes across farms, defined by groups).

Please note that these examples assume that your DataFrame df now includes the columns for "rainfall" and "humidity". You'll need to adjust the code to match the exact structure of your data and ensure that all necessary libraries are installed. Mixed models, in particular, can be complex to interpret and require careful consideration of the model structure and assumptions.

**Hausman Test to calculate the best of the three models:**

The Hausman test is a statistical test that is commonly used to decide between a fixed effects model and a random effects model in panel data analysis. The test evaluates whether the unique errors (ui) are correlated with the regressors, the null hypothesis being that the preferred model is random effects versus the alternative fixed effects.

However, applying the Hausman test directly to compare "Pooled OLS", "Fixed Effects", and "Fixed plus Random Effects" models is not straightforward, as the test is specifically designed to compare fixed and random effects models. To approach this systematically, you could first compare Fixed Effects vs. Random Effects, and then separately assess whether Pooled OLS or the preferred model from the first test is more appropriate based on theoretical considerations and model diagnostics rather than using the Hausman test for the latter comparison.

Here's how to perform the Hausman test in Python using statsmodels for comparing Fixed Effects and Fixed plus Random Effects models. Note that performing the Hausman test requires the estimation of both models first. We'll assume you have already fitted these models (model\_fe for Fixed Effects and model\_mixed for Mixed Effects) as described in previous responses.

### **Step 1: Fit Fixed Effects and Random Effects Models**

For demonstration purposes, let's assume you have already fitted these models:

* Fixed Effects Model: model\_fe
* Mixed Effects Model: model\_mixed

### **Step 2: Perform the Hausman Test**

The Hausman test is not directly implemented in statsmodels for these models, but you can perform it manually by comparing the coefficients and variance matrices of the fixed effects and mixed (random effects) models. Here's a way to manually calculate the Hausman statistic:

import numpy as np

import scipy.stats

def hausman\_test(fe\_estimates, re\_estimates, fe\_var, re\_var):

"""

Compute the Hausman test for fixed effects vs random effects models.

Parameters:

- fe\_estimates: Coefficients from the fixed effects model

- re\_estimates: Coefficients from the random effects model

- fe\_var: Covariance matrix of the fixed effects model estimates

- re\_var: Covariance matrix of the random effects model estimates

Returns:

- Hausman statistic and p-value

"""

# Calculate the difference in coefficients

diff = fe\_estimates - re\_estimates

# Calculate the variance difference

var\_diff = fe\_var - re\_var

# Hausman statistic

hausman\_stat = np.dot(np.dot(diff.T, np.linalg.inv(var\_diff)), diff)

# Degrees of freedom

df = len(fe\_estimates)

# Compute p-value

p\_value = 1 - scipy.stats.chi2.cdf(hausman\_stat, df)

return hausman\_stat, p\_value

# Extract the estimates and their covariance matrices

fe\_estimates = model\_fe.params

re\_estimates = model\_mixed.params

fe\_var = model\_fe.cov\_params()

# For mixed model, extract the covariance matrix for fixed effects only

re\_var = model\_mixed.cov\_params().loc[fe\_estimates.index, fe\_estimates.index]

# Perform the Hausman test

hausman\_stat, p\_value = hausman\_test(fe\_estimates, re\_estimates, fe\_var, re\_var)

print(f"Hausman Test Statistic: {hausman\_stat}")

print(f"P-value: {p\_value}")

This function calculates the Hausman statistic and its p-value. A significant p-value (typically <0.05) suggests rejecting the null hypothesis in favor of the fixed effects model, indicating that the unique errors are correlated with the regressors and that the fixed effects model is more appropriate.

Remember, this test is specifically for comparing Fixed Effects with Random Effects models. The choice between Pooled OLS and these models would typically depend on whether you believe your data has significant cross-sectional or time-series variation that needs to be accounted for with entity-specific intercepts (fixed effects) or if you can assume homogeneity across entities (Pooled OLS).

**Measuring the Prediction Accuracy of the Model:**

To evaluate the prediction accuracy of the Pooled OLS, Fixed Effects, and Mixed Effects models, we will use the Root Mean Squared Error (RMSE) as the metric for this example, given its common application in assessing the accuracy of continuous variable predictions in regression models. Note that to directly compare the Fixed Effects and Mixed Effects models' predictions with the actual values, we need to adjust our approach slightly for each model, particularly because the Fixed Effects model involves within-entity variations.

Step 1: Calculate Predictions First, we need to generate predictions from each of the models. For the Pooled OLS and Fixed Effects models, this process is straightforward. For the Mixed Effects model, predictions are also direct but remember to account for both fixed and random effects.

Assuming you have already fitted models named model\_pooled, model\_fe, and model\_mixed (note: the actual fitting of these models is not shown here), the following code snippets show how to calculate predictions:

**# Pooled OLS predictions**

**predictions\_pooled = model\_pooled.predict(X) # Assuming X is your predictors matrix**

**# Fixed Effects predictions - we use the same predictors but must handle fixed effects**

**# Note: This may involve creating dummy variables for each entity and subtracting entity-specific intercepts if they were included**

**predictions\_fe = model\_fe.predict(X\_fe) # Assuming X\_fe is adjusted for fixed effects**

**# Mixed Effects predictions**

**# Here, we can directly use the .predict() method as it accounts for both fixed and random effects**

**predictions\_mixed = model\_mixed.predict(X\_mixed) # Assuming X\_mixed includes the necessary predictors**

Step 2: Calculate RMSE The RMSE can be calculated by comparing the predictions from each model to the actual observed values. Assuming y is your vector of actual observed values:

1. Dynamics of Modeling for Sugarcane Sucrose Estimation Using Time Series Satellite Imagery - Yu Zhao, Diego Della Justina, Yoriko Kazama, Jansle Vieira Rocha, Paulo Sergio Graziano, Rubens Augusto Camargo Lamparelli [↑](#footnote-ref-0)
2. A Review of Spatial Causal Inference Methods for Environmental and Epidemiological Applications - Brian J. Reich, Shu Yang, Yawen Guan, Andrew B. Giffin, Matthew J. Miller [↑](#footnote-ref-1)
3. A Review of Spatial Causal Inference Methods for Environmental and Epidemiological Applications - Brian J. Reich, Shu Yang, Yawen Guan, Andrew B. Giffin, Matthew J. Miller [↑](#footnote-ref-2)