ME396P FINAL PROJECT REPORT

CURVE FITTING GUI WITH BOUNDARY CONDITIONS

Team G03 – Pynapples
(Anastasia B Timoshenko, Jahnavi Valivi Reddy, John C Tanir)
University of Texas at Austin

OBJECTIVES:

The objective of our project is to design a Curve-fitting GUI Tool capable of handling multiple curve-fitting requirements and to fit the data with boundary conditions. The tool must be able to:

- Take user inputs for fit parameters, such as:
- Type of fit between polynomial or custom equation.
    - Order of the polynomial equation.
    - Custom equation as a text input.
    - Type of boundary condition to be identified (min/max, jump, or none).
    - Dataset to be fitted to text input or excel file input.
- Fit the given data to the user specified equations/polynomial types and generate multiple curves by detecting the specified boundary conditions.
- Output a figure with the fitted data along with the resultant equations and r^2 value.
    - Along with visualizing confidence bounds on the plot if the user specifies the requirement.

APPROACH:

A. PACKAGES

The following packages were used to complete the project:

- PyQt5
- Numpy (polyfit, etc.), Pandas, Random
- Matplotlib.pyplot and Matplotlib.backends.backend_qt5agg
- Scipy.optimize (curve_fit) and Scipy.stats
- Sklearn.metrics (r2_score)

B. ALGORITHM

Four major algorithms had to be implemented to complete the objectives of this project. They include:

- Detecting type of boundary condition (BC) and BC location.
    - We aimed to identify two types of boundary conditions:
        - Maximum/minimum point on a curve.
            - The max and min points of noisy data aren't trustworthy as BC's because they can incorrectly identify the location of the BC.
            - Utilized moving average to "smooth" noisy input data.
            - Max/min point of the "smooth" data was used as the BC.
        - Discontinuous point on the curve.
            - Significant gaps in the data were identified with the moving average method.

        - The "threshold value" (TV), a magnitude for a significant enough gap between neighboring points, determines the discontinuous BC.
        - TV is user-specified or auto-generated (from standard deviation of errors).
- Curve Fitting for different types of curves (polynomials of different degrees/custom equation).
    - Polynomial fit.
        - Utilized np.polyfit(x, y, deg).
        - User selects degree and type of BC – fits based on the boundary condition.
    - Custom equation fit.
        - Utilized Scipy.optimise.curve_fit(func, x, y)
        - Converted the custom equation into a usable function with x and *nums (tuple of coeff. values) as the only inputs. See the algorithm for the conversion of text input into an executable function below.
        - Performs curve fitting using the provided equation and x and y data points.
- Custom equation: Conversion of text input into a usable function.
    - To be able to perform curve fitting using the curve_fit function in the Scipy.optimize package the equation of interest must be present in the code as an executable function. Therefore, the user input of the desired equation must be converted to a function.
    - Two important functions were utilized to make the conversion:
        - Utilized exec() to execute a miniature string based function that was able to convert a string into a variable name and set the function input coefficient as the value of the variable.
        - Utilized the eval() function to evaluate the equation in string form once all of the coefficients were defined using the above method.
- Evaluation of fit: This was done by performing error calculations using Scipy.stats and Sklearn.metrics (r2_score)
    - The coefficient of determination, r^2 was calculated to check how closely the data matches the fitted polynomial, with it being more accurate if it's closer to 1.
    - These errors included the Sum of Squares due to the error, and the Standard Error of the Estimate, Residuals, Total Sum of Squares and Sum of Squares of the Regression.
    - In addition, statistical analysis was performed to calculate the 95% confidence and prediction intervals.
    - Confidence intervals provide an interval estimate with a measure of reliability about the mean and Prediction intervals provide reliability about future observations.

RESULTS:

The code successfully identifies varying trends and gaps in the data based on the set boundary conditions (with a few limitations explained in the later sections). A few examples are given below depicting its functionality. Fig.1 shows a curve fit to the 2^{nd} order custom equation entered by the user, disregarding the value '1' in the order box. With the BC set to 2, it detected a gap in the data using the standard deviation of errors by default (since the threshold value text box is empty).

Whereas, in fig.2, the TV is set to 1.8 and thus the first gap, which is less than 1.8 is ignored and plotted as a single curve. Fig.3 shows two separate curves for the two varying trends as the Trend BC (BC =1) was selected, and fig.4 shows the 95% confidence and prediction bounds of the 4th order curve that was fit to the manually entered data points.

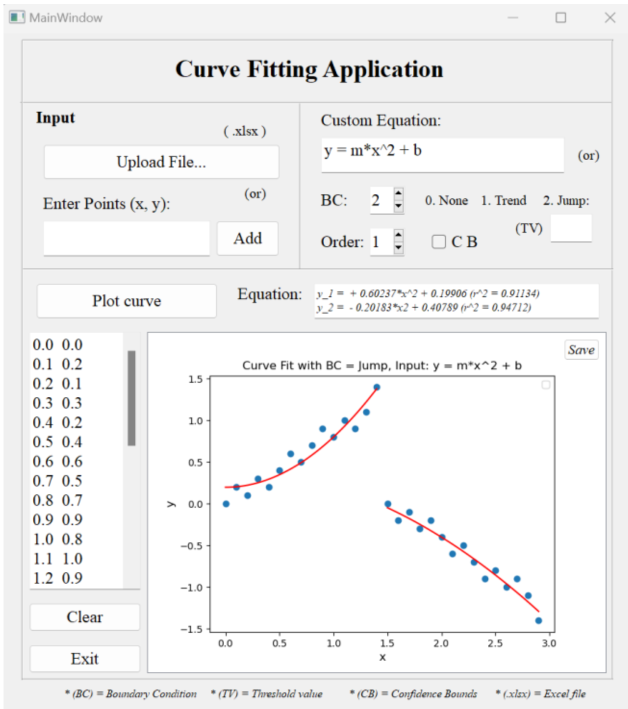

Fig 1. 2nd order custom equation (BC=2)

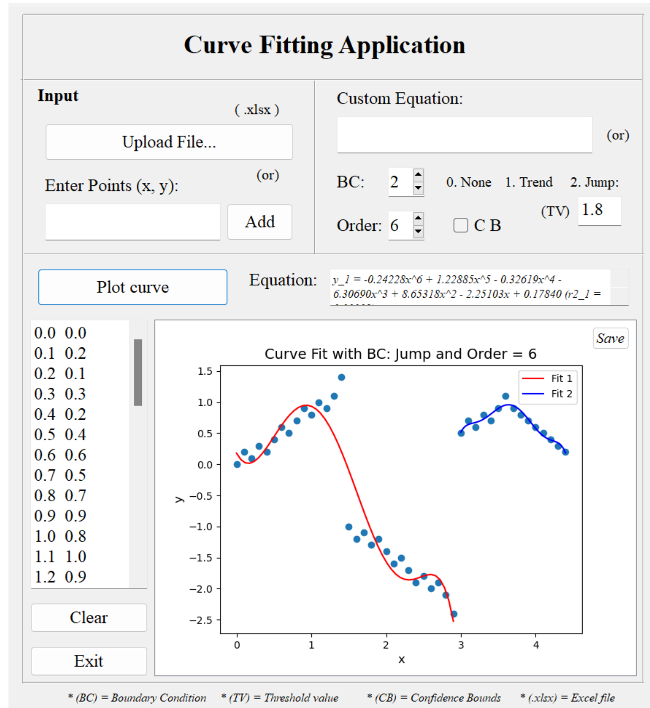

Fig 2. Jump BC with threshold value of order 6

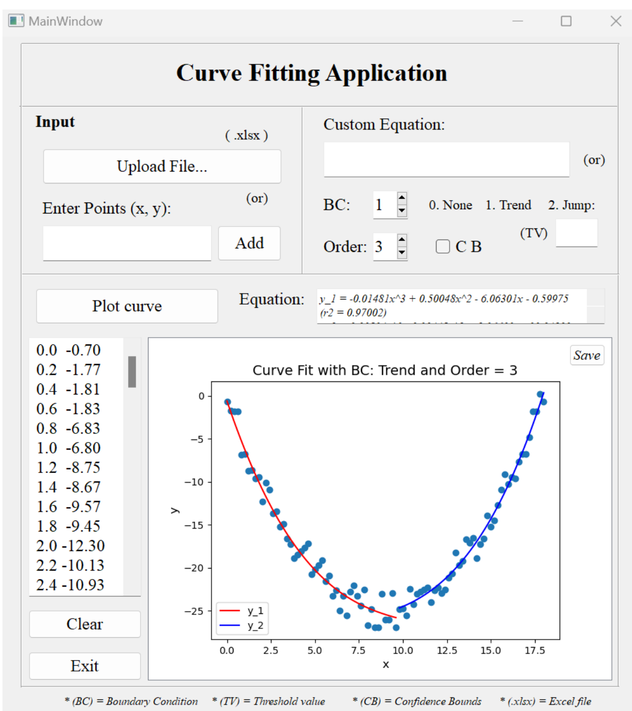

Fig 3. Trend BC of order 3

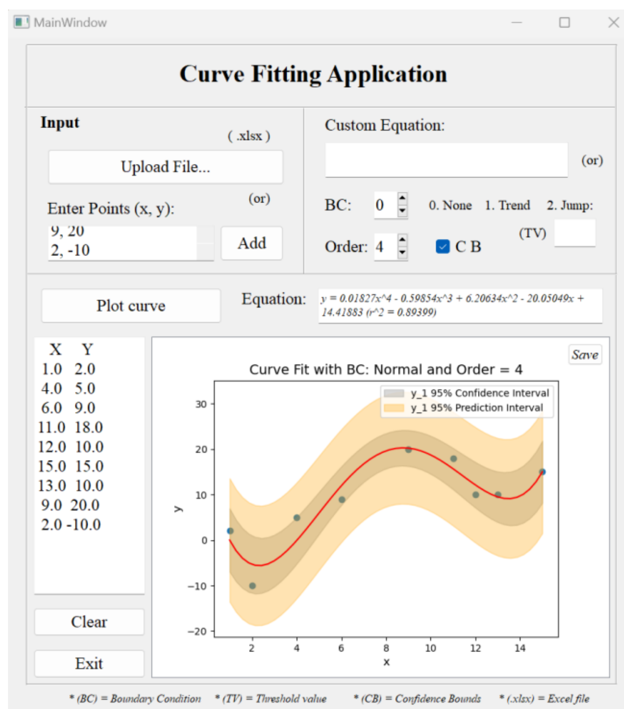

Fig 4. Confidence Bounds & Manual point entry

DEMO LINKS:

- Github - https://github.com/Jaanv99/Team-Pynapples.git

  Read the 'How to' section of the report or the README.md on github before using the GUI

HOW TO:

1. INSTALL PYQT5 & QT DESIGNER

- **Anaconda on Windows**
  PyQt is already integrated with Spyder/Anaconda. If you're running the code with spyder on windows, installation is not necessary. For other operating systems, PyQt must be installed before running the code which might involve restarting the system as well.

- **Installation**
  The simplest way to install PyQt5 on Linux/windows or macOS is to use Python's pip packaging tool, just as for other packages. For Python3 installations this is usually called pip3.

```
pip3 install pyqt5
```

  After installing, you should be able to run python and import PyQt5. Note that if you want to access Qt Designer or Qt Creator you will need to download this from https://qt.io/download [the Qt downloads site]. *You do not need Qt Designer to run our demo*. It's only necessary when creating new GUI designs.

2. USE THE GUI

- **Download demo**
  From the Github repository link above, download the code 'finalprojectCT.py' and the GUI design file 'CFA.ui' into the same folder on your PC. Try curve fits for any random data sets or use the available excel files 'Book 1, Book 2…' by downloading them to the same folder.

- **Available options**
  - Input points: Two methods, either through a .xlsx file upload with the x and y points in the first 2 columns, or manually entering points in the text box in x, y format.
  - Order of fit: Choose from 1-10. It is set by default to order '1' (linear). Check to make sure these boxes are *always* populated with the desired values. When a custom equation is given, the input order box value is neglected.
  - Type of Boundary Condition: Choose from None, Trend and Jump as per requirement. Select '0' (None) for a simple fit without considering jumps or trends. Select '1' (Trend) to detect a change in the trend of the data (increasing to decreasing or decreasing to increasing) and select '2' (Jump) to detect gaps in the data based on the threshold value.
  - Threshold value for BC Jump: TV can be manually input to a value based on desired fit type or in to *neglect unintended gaps* in the curve. The code will then only detect gaps which are more than the input threshold value. If the TV field is left empty, jumps are detected based on the standard deviation of errors in the data by default (only if BC = 2).
  - Custom equation text input: Manually enter the equation type (linear or higher order) in the text box. For example, a 3rd order equation would be 'y = a*x^3 + b*x + d'.
  - Save Plot: Clicking the 'Save' button saves the plot to the code working directory location.
  - Display Confidence Bounds (CB): 95% confidence intervals and prediction intervals are plotted to the curve based on the fit type. This can be enabled by checking the 'C B' check box.

- Fit equation: The curve fit equations are displayed in the 'equation' box above the plot along with their r^2 value (coefficient of determination). *When $r^2 = 1$, no confidence bounds* are plotted as it's a perfect fit.
- Clear: Erases all the inputs and displays of the GUI to start over.

- **Choosing the right fit**
  Judge the final plot based on your requirements and through the following:
  - Coefficient of determination: R^2 value (the closer it is to 1, the better the fit)
  - Order of fit (higher order curves fit better but their equations are computationally complex for future predictions)
  - Boundary Conditions (Not all curves work for all kinds of data, if the plot is empty or not in the desired manner, change the BC and order values to get the best one)

  Play around with all the options to find the required fits and choose what's best for you.

LIMITATIONS:

1. Data Type:
   - For the Trend BC: Only one Increasing - decreasing /(or) Decreasing - increasing trend can be identified with BC = 1. For example, for a sine wave with multiple ups and downs, multiple curves aren't generated. The code is *limited to a simple 'U' or 'inverted U*' type data sets.
   - For the Jump BC: The code is constructed to identify *only up to 2 gaps/breaks* in the data. If the data set has more than 2 large jumps, only the first two curves would fit correctly and the last one would include the rest of the data in a single curve.
     - Future improvements could certainly include making the code more flexible to 'n' number of gaps and trends, but it was out of scope for our current project.
   - Certain combinations of input *might not generate plots* depending upon fit types and compatibility. The user must use their best judgment and change/choose the boundary condition and order values accordingly.
2. Custom equation text input:
   - The custom equation can be input in virtually any form if the characters used are python recognized, with one exception: the ^ symbol has been built into the program instead of **.
   - Spacing does not matter for the most part, however, a space must be used between the = and the rest of the expression due to how the code was written.
     - This can be changed by editing the equation function to use '=' instead of '= 'for the split and by utilizing strip() on the text after splitting.
   - The custom text input equation code is currently unable to take an input that has an 'e' in the equation. However, since the numpy package is utilized in the program code, it can recognize np.e as e. Because of this, np.e should be utilized in the text input instead.
     - To simplify this, an additional line of code can be added to the custom equation text input code:
       - Within the text input string, replace e with np.e.
       - A similar piece of code is already currently implemented to allow the user to type in the ^ symbol rather than using **.

REFERENCES:

- https://learnpython.com/blog/average-in-matplotlib/
- https://learnpython.com/blog/filter-rows-select-in-pandas/
- https://stackoverflow.com/questions/74364841/how-to-find-peaks-in-a-noisy-signal-or-estimate-its-number
- https://www.geeksforgeeks.org/how-to-calculate-moving-average-in-a-pandas-dataframe/
- https://stackoverflow.com/questions/13728392/moving-average-or-running-mean
- https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.rolling.html
- https://pandas.pydata.org/docs/getting_started/intro_tutorials/03_subset_data.html
- https://stackoverflow.com/questions/29949757/creating-pandas-dataframe-between-two-numpy-arrays-then-draw-scatter-plot
- https://www.geeksforgeeks.org/get-the-index-of-maximum-value-in-dataframe-column/
- https://stackoverflow.com/questions/16060899/alphabet-range-in-python
- https://stackoverflow.com/questions/9685946/math-operations-from-string
- https://www.geeksforgeeks.org/convert-string-into-variable-name-in-python/
- https://doc.qt.io/qtforpython-6/tutorials/datavisualize/index.html
- https://stackoverflow.com/questions/30964669/qt-designer-qgraphicsview-load-image
- https://www.appsloveworld.com/coding/python3x/6/how-to-get-confidence-intervals-from-curve-fit?expand_article=1