Time Series Analysis Program

Spencer Lutz and John Gunasar

May 2020

Introduction

For the past few months, John and I have been working on a program to analyze and forecast time series of multiple dimensions. In other words, the program takes in the values of many independent variables (e.g. time, temperature, etc.) and a dependent variable (e.g. stock price, wind speed, etc.) A series of matrix operations are performed on this data set, and the result is a function that approximates the value of the dependent variable for different values of each independent variable. This function can be used to more efficiently store existing data, or to predict values of the dependent variable.

One example of this in action is stock analysis. We gave the program a series of days and the stock volume on each day as the independent variables, and we gave it the price for that day as the dependent variable. We were able to generate a function that would accurately predict stock prices a few days into the future given the date and volume.

Equations

These hypothesis functions can take many forms, but for our project we decided to look at summations of cosines and polynomials. Our equations take the following form:

- **For Cosines:**

$$\sum_{i=1}^{p}\sum_{j=1}^{n_i}\theta_{i,j}cos(jx)$$

- **For Polynomials:**

$$\sum_{i=1}^{p}\sum_{j=1}^{n_i}\theta_{i,j}x^j$$

where p is the total number of input variables, n_i is the number of parameters generated for the ith input variable, $\theta_{i,j}$ is the parameter generated for the jth cosine of the ith input variable, and x is the value of the independent variable for which we want to estimate the dependent.

Reasoning

We chose to use cosines due to the fact that they can easily capture the seasonality of data; for example, if the price of a certain stock peaks every Friday, a cosine with a period of 5 days would represent this well. Polynomials, on the other hand, are useful for capturing the trend of the data; for example, an increasing stock price could have a positive coefficient on the x^1 term.

Method

Once we have given the program our input matrices and our output matrix, which consist of the independent variable data and corresponding dependent variable data, we generate a design matrix that will be fed into the algorithm. This design matrix is what will be used to generate the parameters for each value of i and j. In order to do this for multiple input variables, we must find the design matrix for each independent variable and append them together, adding a column of 1s on the left to account for a y intercept.

We generate the design matrix X_i for independent variable i as follows:

$$X_i = \begin{bmatrix} cos(x_{i,1}) & cos(2x_{i,1}) & \dots & cos(nx_{i,1}) \\ cos(x_{i,2}) & cos(2x_{i,2}) & \dots & cos(nx_{i,2}) \\ \vdots & \vdots & \ddots & \vdots \\ cos(x_{i,n}) & cos(2x_{i,n}) & \dots & cos(nx_{i,n}) \end{bmatrix}$$

This matrix gives us the values for each variable that will be multiplied by the generated coefficient. We generate X_i for all i and append them together, left to right. We then add the aforementioned column of 1s. This final matrix is the design matrix for the entire system, denoted X. In order to estimate these coefficients, we apply ordinary least squares as follows:

$$\theta = (X^T X)^{-1} X^T y$$

Where θ is the parameter vector and y is a matrix of the values of the dependent variable. Plugging these parameters into the equation shown in the introduction will produce an accurate representation of the data.

Results

Finally, we can use the data to make predictions on new data. When doing so, the independent variables will fall into one of two categories: known data and unknown data. Known data is information that we know to be true about the system; for example, if we are predicting stock price at a later date, the date for which we are predicting is a known variable. Unknown data is information that is not available to us; for example, the volume of a stock at a future date is not something we know.

In order to solve for these unknowns, we generate a univariate hypothesis function for each unknown variable to predict it for future values of known variables; for example, we can generate a univariate hypothesis function for stock volume, with the date as the independent variable. We then plug in the desired values of this known variable into the hypothesis functions of each unknown

variable to get estimates. These estimates can then be plugged into the multivariate hypothesis function to generate predictions.

After extensive testing on stock data, we have found that the most accurate and efficient predictor is the cosine model. This is likely because of the difficulty of computation inherent to the polynomial model for large degrees. In other words, it is much easier for a computer to sum the values of 100 cosine functions than it is to calculate x^{100}, x^{99}, x^{98}, and so on. Future testing could implement a combination between the two models, summing multiple cosines and only 1 or 2 polynomial degrees. This would theoretically be able to capture both seasonality and trend of data more accurately.