Estimate Missing Climate Data

The EMCD tool is a Windows-based software application designed for estimating missing climate data. It facilitates the estimation process by modeling data from its own historical data or data from other stations. Known for its user-friendly interface, the tool is particularly well-suited for researchers and analysts working with climate data, providing an accessible solution for handling missing data in the field.

In this tool, we have two distinct sections. The first section is dedicated to estimating missing data using information from other stations, while the second section focuses on estimating missing data based on the historical records of the current station.

A- Estimate Using Other stations

For this method, stations must possess historical data for both dependent and independent stations to build a model. Additionally, independent stations need to have records corresponding to the missing data for the method to be effective. In this section, the tool presents four models for estimating missing data as follows:

1- Multiple Linear Regression

Utilizing Multiple Linear Regression (MLR) to impute missing data in time series by incorporating information from other stations is a widely used technique. This method entails constructing regression models using data from nearby or spatially connected stations to forecast missing values at a designated station. The formula is as follows:

$$Y_{dep} = \alpha_0 + \sum_{i=1}^{n} \alpha_i \times X_i$$

Where, Y_{dep} represents the data of the dependent station, X_i is the data of independent stations, n is the number of independent stations, α_o is the constant value, and α_i is the coefficient of station i.

2- Inverse distance weighting

The missing data for the dependent station (target station) are calculated by incorporating values observed in independent stations (neighboring stations), where weights are assigned based on the inverse distance between the target station and its neighboring stations. The formula is as follows:

$$Y_{dep} = \frac{\sum_{i=1}^{n} d_i \times X_i}{\sum_{i=1}^{n} d_i}$$

Where, Y_{dep} represents the data of the dependent station, X_i is the data of independent stations, n is the number of independent stations, and d_i is the Euclidean distance between station i and dependent(target) station as follows:

$$d_i = \sqrt{\sum_{i=1}^{n} \left(Y_{dep} - X_i\right)^2}$$

3- **Correlation coefficient weighting**

In this method, the concept of distance is replaced by Pearson's correlation coefficients. The estimating of the missing value is expressed as:

$$Y_{dep} = \frac{\sum_{i=1}^{n} r_i \times X_i}{\sum_{i=1}^{n} r}$$

Where r_i is the Pearson's correlation coefficient between station i and dependent(target) station.

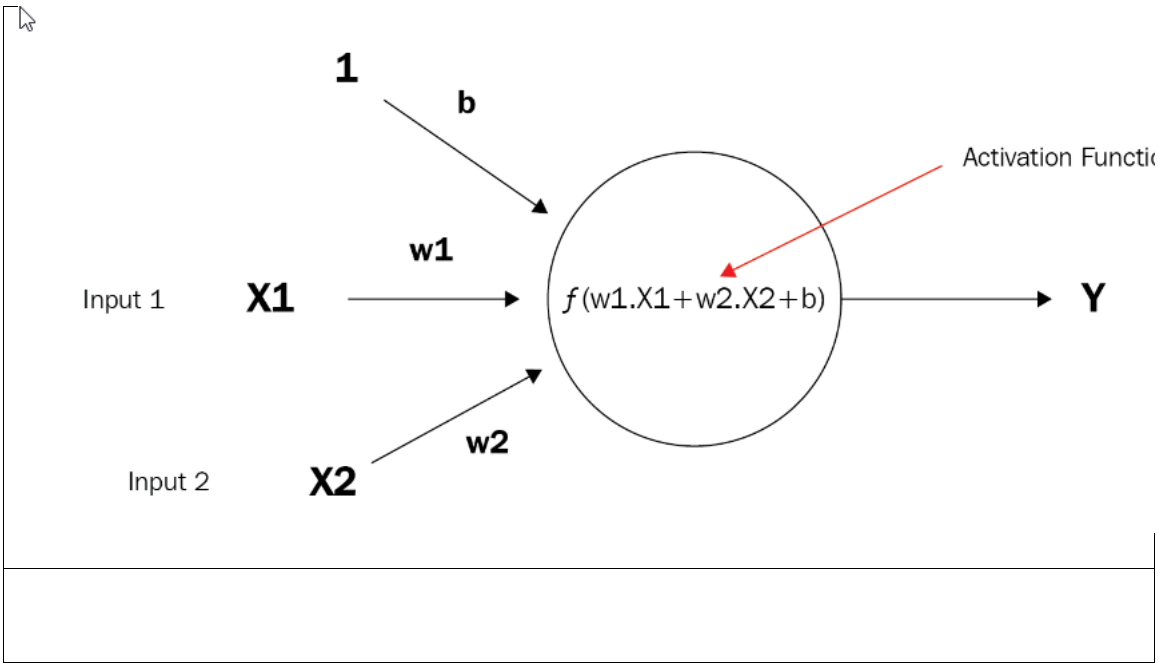
4- Artificial Neural Networks

In the context of estimating missing climate data, an Artificial Neural Network is a computational model inspired by the structure and

functioning of the human brain. It is a machine learning technique that can be used for various tasks, including imputing missing values in climate datasets.

The artificial neural network (ANN) section offers various configuration options that require your specification. Regarding the network setup, you must input the number of neurons in the first layer along with the desired activation function. It's important to note that the output layer has a single neuron.

In artificial neural networks (ANNs), an activation function is a mathematical operation applied to the input of a neuron, determining the neuron's output. It introduces non-linearities to the network, allowing it to learn complex patterns and relationships in data. The activation functions include the Bipolar Sigmoid Function, Identity Function, Linear Function, Rectified Linear Function, Sigmoid Function, Threshold Function, Bernoulli Function, Gaussian Function. These functions help in modeling and capturing the non-linear nature of real-world data, enabling ANNs to perform tasks like classification and regression effectively.



Bipolar Sigmoid Function

The bipolar sigmoid activation function, also known as the hyperbolic tangent (tanh) function, is a type of activation function commonly used in artificial neural networks. It is an extension of the standard sigmoid function and is defined by the formula:

$$f(x) = \frac{2}{1 + \exp(-\alpha x)} - 1$$

http://accord-framework.net/docs/html/T_Accord_Neuro_BipolarSigmoidFunction.htm

Identity Function

The identity activation function, denoted by f(x) = x, operates by directly transmitting the output of neuronal summation to subsequent neurons without any alteration.

http://accord-framework.net/docs/html/T_Accord_Neuro_IdentityFunction.htm

Linear Function

This item employs a linear activation function constrained within the interval (a, b), defined by the piecewise formula:

$$f(x) = \begin{cases} \alpha x, & b < \alpha x < a \\ a, & x = a \\ b, & x = b \end{cases}$$

Where a = -1 and b = +1

http://accord-framework.net/docs/html/T_Accord_Neuro_LinearFunction.htm

Rectified Linear Function

This function, non-differentiable at zero, implements a rectified linear activation function, as defined by the piecewise formula below:

$$f(x) = \begin{cases} x, & x > 0 \\ 0, & x \leq 0 \end{cases}$$

http://accord-framework.net/docs/html/T_Accord_Neuro_RectifiedLinearFunction.htm

Sigmoid Function

The sigmoid activation function, also known as the logistic function, is a common activation function used in artificial neural networks (ANNs). It is defined by the following mathematical formula:

$$f(x) = \frac{1}{1 + e^{-\alpha x}}$$

http://accord-framework.net/docs/html/T_Accord_Neuro_SigmoidFunction.htm

Threshold Function

The threshold activation function, also known as the step function, is a simple activation function used in artificial neural networks (ANNs). It is a binary activation function that outputs either 0 or 1 based on a specified threshold. The mathematical representation of the threshold activation function is as follows:

$$f(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

http://accord-framework.net/docs/html/T_Accord_Neuro_ThresholdFunction.htm

Bernoulli Function

The Bernoulli activation function serves as a foundation for constructing Stochastic Neurons, facilitating the development of Deep Belief Networks and Restricted Boltzmann Machines. Its application is well-suited for scenarios where the inputs of a problem are discrete, taking on values of either 0 or 1. In cases where the inputs are continuous, the Gaussian function may be a more appropriate choice.

Functioning as a stochastic activation function, the Bernoulli function has the capability to generate values following a statistical probability distribution. Specifically, the Bernoulli function adheres to a Bernoulli distribution, with its mean determined by the output of the associated sigmoidal function within this class.

http://accord-framework.net/docs/html/T_Accord_Neuro_ActivationFunctions_GaussianFunction.htm

Gaussian Function

The Gaussian activation function is employed to generate Stochastic Neurons, forming the basis for constructing Deep Belief Networks and Restricted Boltzmann Machines. Unlike the Bernoulli function, the Gaussian function proves useful in modeling continuous inputs within Deep Belief Networks. In cases where the input variables of the problem exhibit a discrete nature, opting for a Bernoulli function would be more suitable.

Modeled after a Gaussian (Normal) probability distribution, this activation function presupposes that output variables have undergone normalization to achieve zero mean and unit variance.

http://accord-framework.net/docs/html/T_Accord_Neuro_ActivationFunctions_GaussianFunction.htm

In the realm of Artificial Neural Networks (ANNs), learning algorithms are techniques employed to fine-tune the parameters of the network throughout the training process. The primary objective of training an ANN is

to empower it to generate precise predictions based on input data. Learning algorithms play a pivotal role in this process, iteratively adjusting the weights and biases of the network in response to the training data. Various methods can be utilized for this purpose, including Back Propagation Learning, Delta Rule Learning, Levenberg Marquardt Learning, Parallel Resilient Backpropagation Learning, Perceptron Learning, and Resilient Backpropagation Learning.

An epoch in the context of neural network training refers to the process of exposing the entire training dataset to the network for a single cycle with N times repeating. Please input the desired value for N.

Autocorrelation Function (ACF)

The Autocorrelation Function (ACF) is a statistical tool widely employed in time series analysis to investigate the correlation between a variable and its previous values. It quantifies the relationship between the lagged version of a variable and its original version within a time series. The formula is expressed as:

$$r_k = \frac{\sum_{t=k+1}^{T}[(y_t - \bar{y}) \times (y_{t-k} - \bar{y})]}{\sum_{t=1}^{T}(y_t - \bar{y})^2}$$

$$Bounds = \pm \frac{Z}{\sqrt{T-k}}$$

Where T represents the number of samples, k is the lag number, and Z is the inverse cumulative distribution (CD) of the standard normal distribution, determined by the Confidence Level.

Values that fall outside the expected bounds exhibit a notable correlation with the station. It is advisable to choose lag values that demonstrate the strongest correlation for optimal results.

B- Estimate Using Historical

These techniques can address the absence of climate data without relying on data from other stations. If you only have data from a single station, you can employ methods that leverage the historical data of that specific station for imputation.

1- Last Observation Carried Forward (LOCF)

The Last Observation Carried Forward (LOCF) method is a simple imputation technique commonly used in time series or longitudinal data analysis to fill in missing values. The basic idea behind LOCF is to carry forward the last observed value to replace any missing values that occur subsequently.

Mathematically, the LOCF method can be expressed as:

$$Y_t = Y_{t-1}$$

Y_t is the missing value at time t.

Y_{t-1} is the last observed value before time t.

In words, the value for a missing data point (Y_t) is replaced with the value of the last observed data point (Y_{t-1}). This method assumes that the most recent observed value is a reasonable estimate for the missing value until a new observation is available.

While LOCF is straightforward and easy to implement, it has limitations. It may not be suitable for all types of data, especially if the assumption that the most recent observation is a good estimate for the missing value is not reasonable.

2- Next Observation Carried Backward (NOCB)

The Next Observation Carried Backward (NOCB) method is an imputation technique used in time series or longitudinal data analysis to fill in missing values. It is the counterpart to the Last Observation Carried Forward (LOCF) method. While LOCF carries the last observed value forward to replace missing values, NOCB carries the next observed value backward to fill in missing values.

Mathematically, the NOCB method can be expressed as:

$$Y_t = Y_{t+1}$$

Y_t is the missing value at time t.

Y_{t+1} is the next observed value after time t.

3- Interpolation

Interpolation is a method used to estimate values that are missing in a dataset by predicting them based on the values of other data points. It is particularly useful for time series or sequential data where there is a presumed order or continuity between data points. There are several interpolation methods, and the choice depends on the characteristics of the data and the assumptions made about its behavior.

- Linear Interpolation

  Linear interpolation assumes a linear relationship between consecutive data points. It connects the two neighboring data points with a straight line and estimates the missing value based on the position along that line.

  The formula for linear interpolation between points (x_1, y_1) and (x_2, y_2) is given by:

$$Y_t = y_1 + \frac{(x_t - x_1) \times (y_2 - y_1)}{(x_2 - x_1)}$$

- Akima Cubic Spline Interpolation

The Akima cubic spline interpolation is based on the Akima method for constructing a piecewise cubic Hermite interpolating polynomial. The Akima spline is designed to provide a smooth interpolation even in the presence of irregularly spaced data points. The formula for the Akima cubic spline interpolation between two adjacent data points (x_i, y_i) and (x_{i+1}, y_{i+1}) is as follows:

$$P_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3$$

$$a_i = y_i$$

$$b_i = s_i$$

$$c_i = \frac{3m_i - 2s_i - s_{i+1}}{x_{i+1} - x_i}$$

$$d_i = \frac{s_i + s_{i+1} - 2m_i}{(x_{i+1} - x_i)^2}$$

$$m_i = \frac{y_{i+1} - y_i}{x_{i+1} - x_i}$$

$$s_i = \frac{m_{i-1} + m_i}{2}$$

The key idea is that the slope mi is calculated using a weighted average of neighboring slopes, providing smoothness in the interpolated function.

The MathNet.Numerics library takes care of the details of fitting these cubic polynomials between adjacent data points, constructing a piecewise cubic spline that passes through each data point. The InterpolateAkima method then evaluates this spline at a given point to obtain the interpolated value.

4- Substitute Statistics

The "Substitute Statistics" method is a straightforward approach that involves replacing missing values in a dataset with one of the following statistics: mean, median, minimum, maximum, or quantiles at specific percentiles such as 25, 75, 90, and 95.