# 1 Basics of Climate Data Arrays, Statistics, and Visualization

People talk about climate data frequently, read or imagine climate data, and yet rarely play with or use climate data, because they often think it takes a computer expert to do that. However, that is changing. With today's technology, anyone can use a computer to play with climate data, such as a sequence of temperature values of a weather station at different observed times or a matrix of data for a station for temperature, air pressure, precipitation; wind speed and wind direction at different times; and an array of temperature data on a 5-degree latitude–longitude grid for the entire world for different months. The first is a vector. The second is a variable-time matrix, and the third a space-time 3-dimensional array. When considering temperature variation in time at different air pressure levels and different water depths, we need to add one more dimension: altitude. The temperature data for the ocean and atmosphere for the Earth is a 4-dimensional array, with 3D space and 1D time. This chapter attempts to provide basic statistical and computing methods to describe and visualize some simple climate datasets. As the book progresses, more complex statistics and data visualization will be introduced.

We use both R and Python computer codes in this book for computing and visualization. Our method description is stated in R. A Python code following each R code is included in a box with a light yellow background. You can also learn the two computer languages and their applications to climate data from the book *Climate Mathematics: Theory and Applications* (Shen and Somerville 2019) and its website `www.climatemathematics.org`.

The climate data used in this book are included in the `data.zip` file downloadable from our book website `www.climatestatistics.org`. You can also obtain the updated data from the original data providers, such as `www.esrl.noaa.gov` and `www.ncei.noaa.gov`.

After studying this chapter, a reader should be able to analyze simple climate datasets, compute data statistics, and plot the data in various ways.

## 1.1 Global Temperature Anomalies from 1880 to 2018: Data Visualization and Statistical Indices

In a list of popular climate datasets, the global average annual mean surface air temperature anomalies might be on top. Here, *anomalies* means the temperature departures from the normal temperature or the *climatology*. Climatology is usually defined as the mean of temperature data in a given period of time, such as from 1971 to 2020. Thus, temperature anomaly data are the differences of the temperature data minus the climatology.

This section will use the global average annual mean surface air temperature anomaly dataset as an example to describe some basic statistical and computing methods.

### 1.1.1 The NOAAGlobalTemp Dataset

The 1880–2018 global average annual mean surface air temperature (SAT) anomaly data are shown as follows:

```
  [1] -0.37 -0.32 -0.32 -0.40 -0.46 -0.47 -0.45 -0.50 -0.40 -0.35 -0.57
 [12] -0.49 -0.55 -0.56 -0.53 -0.47 -0.34 -0.36 -0.50 -0.37 -0.31 -0.39
 [23] -0.49 -0.58 -0.66 -0.53 -0.46 -0.62 -0.69 -0.68 -0.63 -0.68 -0.58
 [34] -0.57 -0.39 -0.32 -0.54 -0.56 -0.45 -0.45 -0.46 -0.39 -0.47 -0.46
 [45] -0.50 -0.39 -0.31 -0.40 -0.42 -0.54 -0.34 -0.32 -0.36 -0.49 -0.35
 [56] -0.38 -0.36 -0.26 -0.27 -0.26 -0.15 -0.05 -0.09 -0.09  0.04 -0.08
 [67] -0.25 -0.30 -0.30 -0.30 -0.41 -0.26 -0.22 -0.15 -0.36 -0.38 -0.44
 [78] -0.19 -0.13 -0.18 -0.22 -0.16 -0.15 -0.13 -0.39 -0.32 -0.27 -0.26
 [89] -0.27 -0.15 -0.21 -0.32 -0.22 -0.08 -0.32 -0.24 -0.32 -0.05 -0.13
[100] -0.02  0.02  0.06 -0.06  0.10 -0.10 -0.11 -0.01  0.13  0.13  0.05
[111]  0.19  0.16  0.01  0.03  0.09  0.21  0.08  0.27  0.39  0.20  0.18
[122]  0.30  0.35  0.36  0.33  0.41  0.37  0.36  0.30  0.39  0.45  0.33
[133]  0.38  0.42  0.50  0.66  0.70  0.60  0.54
```

The data are part of the dataset named as the NOAA Merged Land Ocean Global Surface Temperature Analysis (NOAAGlobalTemp) V4. The dataset was generated by the NOAA National Centers for Environmental Information (NCEI) (Smith et al. 2008; Vose et al. 2012). Here, NOAA stands for the United States National Oceanic and Atmospheric Administration.

The anomalies are with respect to the 1971–2000 climatology, i.e., 1971–2000 mean. An anomaly of a weather station datum is defined by the datum minus the station climatology. The first anomaly datum, `-0.37` °C, indexed by `[1]` in the above data table, corresponds to 1880 and the last to 2018, a total of 139 years. The last row is indexed from `[133]` to `[139]`.

One might be interested in various kinds of statistical characteristics of the data, such as mean, variance, standard deviation, skewness, kurtosis, median, 5th percentile, 95th percentile, and other quantiles. Is the data's probabilistic distribution approximately normal? What does the box plot look like? Are there any outliers? What is a good graphic representation of the data, i.e., popularly known as a climate figure?

When considering global climate changes, why do scientists often use anomalies, instead of the full values directly from the observed thermometer readings? This is because the observational estimates of the global average annual mean surface temperature are less accurate than the similar estimates for the changes from year to year. There is a concept of characteristic spatial correlation length scale for a climate variable, such as surface temperature. The length scale is often computed from anomalies.

The use of anomalies is also a way of reducing or eliminating individual station biases. A simple example of such biases is that due to station location, which is usually fixed in a long period of time. It is easy to understand, for instance, that a station located in

the valley of a mountainous region might report surface temperatures that are higher than the true average surface temperature for the entire region and cannot be used to describe the behavior of climate change in the region. However, the anomalies at the valley station may synchronously reflect the characteristics of the anomalies for the region. Many online materials give justifications and examples on the use of climate data anomalies, e.g., NOAA NCEI (2021).

The global average annual mean temperature anomalies quoted here are also important for analyzing climate simulations. When we average over a large scale, many random errors cancel out. When we investigate the response of such large scale perturbations as the variations of Sun brightness or atmospheric carbon dioxide, these averaged data can help validate and improve climate models. See the examples in the book by Hennemuth et al. (2013) that includes many statistical analyses of both observed and model data.

## 1.1.2 Visualize the Data of Global Average Annual Mean Temperature

Many different ways have been employed to visualize the global average annual mean temperature anomalies. The following three are popular ones appearing in scientific and news publications: (a) a simple point-line graph, (b) a curve of staircase steps, and (c) a color bar chart, as shown in Figures 1.1–1.3. This subsection shows how to generate these figures by R and Python computer programming languages. The Python codes are in yellow boxes. To download and run the codes, visit www.climatestatistics.org.

### 1.1.2.1 Plot a Point-Line Graph of Time Series Data

Figure 1.1 is a simple line graph that connects all the data points, denoted by the small circles, together to form a curve showing the historical record of global temperature

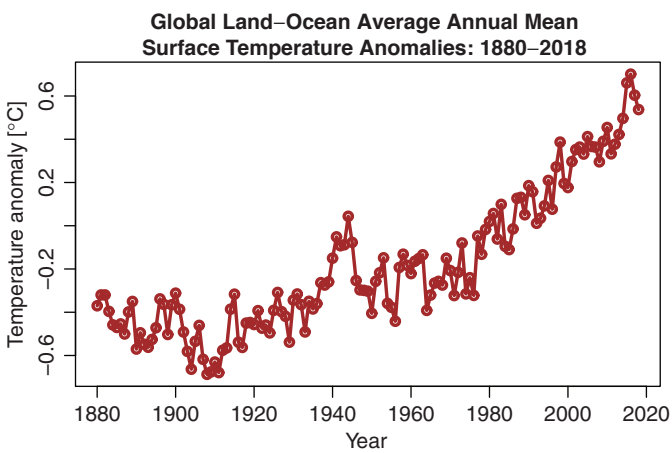


**Figure 1.1** Point-line graph of the 1880–2018 global average annual mean temperature anomalies with respect to the 1971–2000 climatology, based on the NOAAGlobalTemp V4 data.

anomalies. It is plotted from the NOAAGlobalTemp V4 data quoted in Section 1.1.1. The figure can be generated by the following computer code.

```
#R plot Fig. 1.1: A simple line graph of data
# go to your working directory
setwd("/Users/sshen/climstats")
# read the data file from the folder named "data"
NOAAtemp = read.table(
  "data/aravg.ann.land_ocean.90S.90N.v4.0.1.201907.txt",
  header=FALSE) #Read from the data folder
dim(NOAAtemp)# check the data matrix dimension
#[1] 140   6  #140 years from 1880 to 2019
#2019 will be excluded since data only up to July 2019
#col1 is year, col2 is anomalies, col3-6 are data errors
par(mar=c(3.5,3.5,2.5,1), mgp=c(2,0.8,0))
plot(NOAAtemp[1:139,1], NOAAtemp[1:139,2],
     type ="l", col="brown", lwd=3, cex.lab=1.2, cex.axis=1.2,
     main ="Global␣Land-Ocean␣Average␣Annual␣Mean
␣␣␣␣␣Surface␣Temperature␣Anomalies:␣1880-2018",
     xlab="Year",
     ylab=expression(
       paste("Temperature␣Anomaly␣[", degree,"C]")))
```

```
#Python plot Fig. 1.1:  A simple line graph of data
# Go to your working directory
os.chdir("/Users/sshen/climstats")
# read the data file from the folder named "data"
NOAAtemp = read_table(\
"data/aravg.ann.land_ocean.90S.90N.v4.0.1.201907.txt",
    header = None, delimiter = "\s+")
# check the data matrix dimension
print("The␣dimension␣of␣our␣data␣table␣is:", NOAAtemp.shape)
x = np.array(NOAAtemp.loc[0:138, 0])
y = np.array(NOAAtemp.loc[0:138, 1])
plt.plot(x, y, 'brown', linewidth = 3);
plt.title("Global␣Land-Ocean␣Average␣Annual␣Mean␣\n␣\
Surface␣Temperature␣Anomaly:␣1880␣-␣2018", pad = 15)
plt.xlabel("Year", size = 25, labelpad = 20)
plt.ylabel("Temperature␣Anomaly␣[$\degree$C]",
           size = 25, labelpad = 20)
plt.show() # display on screen
```

### 1.1.2.2 Plot a Staircase Chart

The staircase chart Figure 1.2 shows both data and the year-to-year annual changes of temperature. One can clearly see that some years have a larger change from the previous year, while other years have a smaller change. This information can be used for further climate analysis.

Denote the global average annual mean temperature by $T(t)$ where $t$ stands for time in year, the 1971–2000 climatology by $C$, and the anomalies by $A(t)$. We have

$$T(t) = C + A(t). \tag{1.1}$$

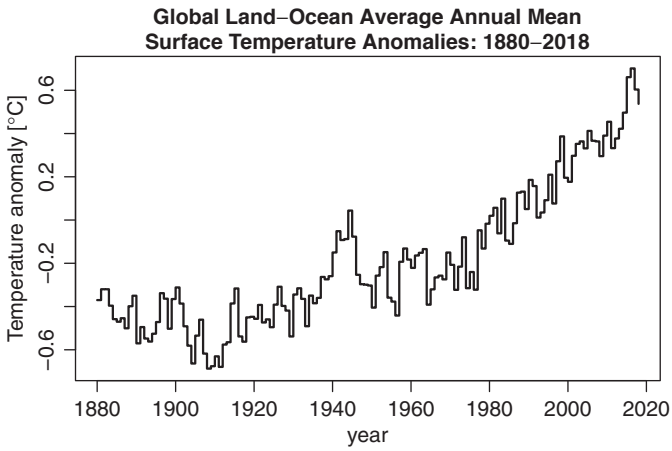


**Figure 1.2** Staircase chart of the 1880–2018 global average annual mean temperature anomalies.

The temperature change from its previous year is

$$\Delta T = T(t) - T(t-1) = A(t) - A(t-1) = \Delta A. \tag{1.2}$$

This implies that an anomaly change is equal to its corresponding temperature change. We do not need to evaluate the 1971–2000 climatology $C$ of the global average annual mean, when studying changes, as discussed earlier.

Figure 1.2 can be generated by the following computer code.

```
#R plot Fig. 1.2: Staircase chart of data
plot(NOAAtemp[1:139,1], NOAAtemp[1:139,2],
     type="s", #staircase curve for data
     col="black", lwd=2,
     main="Global␣Land-Ocean␣Average␣Annual␣Mean
␣␣␣␣␣Surface␣Temperature␣Anomalies:␣1880-2018",
     cex.lab=1.2,cex.axis=1.2,
     xlab="year",
     ylab=expression(paste(
       "Temperature␣Anomaly␣[", degree,"C]"))
)
```

The corresponding Python code is in the following box.

```
#Python plot Fig. 1.2: A staircase chart of data
# keyword arguments
kwargs = {'drawstyle' : 'steps'}
plt.plot(x, y, 'black', linewidth = 3, **kwargs);
plt.title("Global␣Land-Ocean␣Average␣Annual␣Mean␣\n␣␣\
Surface␣Temperature␣Anomaly:␣1880␣-␣2018", pad = 15)
plt.xlabel("Year", size = 25, labelpad = 20)
plt.ylabel("Temperature␣Anomaly␣[$\degree$C]",
           size = 25, labelpad = 20)
plt.show()
```

### 1.1.2.3 Plot a Bar Chart with Colors

Figure 1.3 shows the anomaly size for each year by a color bar: red for positive anomalies and blue for negative anomalies. This bar chart style has an advantage when visualizing climate extremes, since these extremes may leave a distinct impression on viewers. The black curve is a 5-point moving average of the annual anomaly data, computed for each year by the mean of that year, together with two years before and two years after. The moving average smooths the high-frequency fluctuations to reveal the long-term trends of temperature variations.

```
#R plot Fig. 1.3: A color bar chart of data
x  <-  NOAAtemp[,1]
y  <-  NOAAtemp[,2]
z  <-  rep(-99, length(x))
# compute 5-point moving average
for (i in 3:length(x)-2) z[i] =
  mean(c(y[i-2],y[i-1],y[i],y[i+1],y[i+2]))
n1 <- which(y>=0); x1 <- x[n1]; y1 <- y[n1]
n2 <- which(y<0); x2 <- x[n2]; y2 <- y[n2]
x3 <- x[2:length(x)-2]
y3 <- z[2:length(x)-2]
plot(x1, y1, type="h", #bars for data
     xlim = c(1880,2016), lwd=3,
     tck = 0.02,  #tck>0 makes ticks inside the plot
     ylim = c(-0.7,0.7), xlab="Year", col="red",
     ylab = expression(paste(
       "Temperature␣Anomaly␣[", degree,"C]")),
     main ="Global␣Land-Ocean␣Average␣Annual␣Mean
␣␣␣␣␣Surface␣Temperature␣Anomalies:␣1880-2018",
     cex.lab = 1.2, cex.axis = 1.2)
lines(x2, y2, type="h",
      lwd = 3, tck = -0.02, col = "blue")
lines(x3, y3, lwd = 2)
```

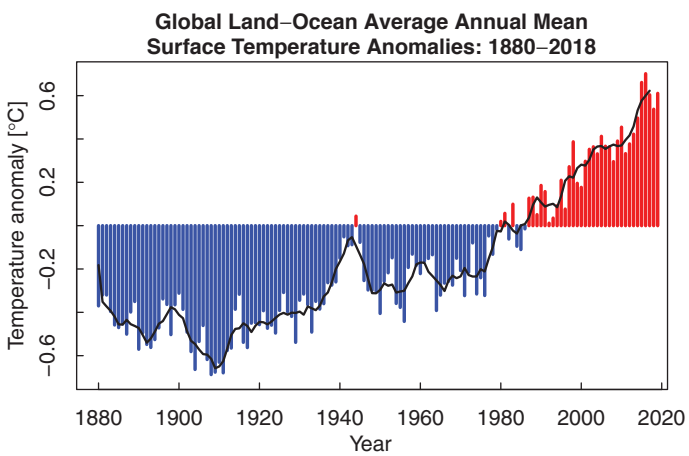


**Figure 1.3** Color bar chart of the 1880–2018 global average annual mean temperature anomalies.

```
#Python plot Fig. 1.3: A color bar chart of data
# define an array z with y number of ones
z = np.ones(y.size)
z[0] = -99
z[1] = -99
# computer 5-point moving average
for i in range(2, z.size - 2):
    z[i] = np.mean(y[i-2:i+3])
z[z.size - 2] = -99
z[z.size - 1] = -99
# define variables based on range
y1 = [y[i] for i in range(y.size) if y[i] >= 0]
x1 = [x[i] for i in range(y.size) if y[i] >= 0]
y2 = [y[i] for i in range(y.size) if y[i] < 0]
x2 = [x[i] for i in range(y.size) if y[i] < 0]
y3 = z[2:x.size-2]
x3 = x[2:x.size-2]

# plot the moving average
plt.plot(x3, y3, 'black', linewidth = 3)
plt.title("Global␣Land-Ocean␣Average␣Annual␣Mean␣\n␣\
Surface␣Temperature␣Anomaly:␣1880␣-␣2018", pad = 20)

# create bar chart
plt.bar(x1, y1, color = 'red');
plt.bar(x2, y2, color = 'blue');
plt.xlabel("Year", size = 25, labelpad = 20)
plt.ylabel("Temperature␣Anomaly␣[$\degree$C]",
           size = 25, labelpad = 20)
plt.show()
```

### 1.1.3 Statistical Indices

The commonly used basic statistical indices include mean, variance, standard deviation, skewness, kurtosis, and quantiles. We first use R to calculate these indices for the global average annual mean temperature anomalies. Then we describe their mathematical formulas and interpret the numerical results.

```
#R code for computing statistical indices
setwd("/Users/sshen/climstats")
NOAAtemp = read.table(
  "data/aravg.ann.land_ocean.90S.90N.v4.0.1.201907.txt",
  header=FALSE)
temp2018=NOAAtemp[1:139,2] #use the temp data up to 2018
head(temp2018) #show the first six values
#[1] -0.370221 -0.319993 -0.320088 -0.396044 -0.458355 -0.470374
mean(temp2018) #mean
#[1] -0.1858632
sd(temp2018) #standard deviation
#[1] 0.324757
var(temp2018) #variance
```

```
#[1] 0.1054671
library(e1071)
#This R library is needed to compute the following parameters
#install.packages("e1071") #if it is not in your computer
skewness(temp2018)
#[1] 0.7742704
kurtosis(temp2018)
#[1] -0.2619131
median(temp2018)
#[1] -0.274434
quantile(temp2018,probs= c(0.05, 0.25, 0.75, 0.95))
#       5%        25%        75%        95%
#  -0.5764861 -0.4119770  0.0155245  0.4132383
```

We use $x = \{x_1, x_2, \ldots, x_n\}$ to denote the sampling data for a time series. The statistical indices computed by this R code are based on the following mathematical formulas for mean, variance, standard deviation, skewness, and kurtosis:

$$\text{Mean: } \mu(x) = \frac{1}{n}\sum_{k=1}^{n} x_k, \tag{1.3}$$

$$\text{Variance by unbiased estimate: } \sigma^2(x) = \frac{1}{n-1}\sum_{k=1}^{n} (x_k - \mu(x))^2, \tag{1.4}$$

$$\text{Standard deviation: } \sigma(x) = (\sigma^2(x))^{1/2}, \tag{1.5}$$

$$\text{Skewness: } \gamma_3(x) = \frac{1}{n}\sum_{k=1}^{n} \left(\frac{x_k - \mu(x)}{\sigma}\right)^3, \tag{1.6}$$

$$\text{Kurtosis: } \gamma_4(x) = \frac{1}{n}\sum_{k=1}^{n} \left(\frac{x_k - \mu(x)}{\sigma}\right)^4 - 3. \tag{1.7}$$

A *quantile* cuts a sorted sequence of data. For example, the 25th quantile, also called 25th percentile or 25% quantile, is the value at which 25% of the sorted data is smaller than this value and 75% is larger than this value. The 50th percentile is also known as the median.

```
#Python code for computing statistical indices
temp2018 = np.array(NOAAtemp.loc[0:138, 1]) # data string
# arithmetic average of the data
print("The␣Mean␣is␣%f.\n" % stats.mean(temp2018))
# standard deviation of the data
print("The␣Standard␣Deviation␣is␣%f.\n" %
      stats.stdev(temp2018))
# variance of the data
print("The␣Variance␣is␣%f.\n" % stats.variance(temp2018))
# skewness of the data
print("The␣Skewness␣is␣%f.\n" % skewness(temp2018))
# kurtosis of the data
print("The␣Kurtosis␣is␣%f.\n" % kurtosis(temp2018))
# median of the data
print("The␣Median␣is␣%f.\n" % stats.median(temp2018))
```

```
# percentiles
print("The␣5th,␣25th,␣75th␣and␣95th␣percentiles␣are:")
probs = [5, 25, 75, 95]
print([round(np.percentile(temp2018, p),5) for p in probs])
print()
```

The meaning of these indices may be explained as follows. The mean is the simple average of samples. The variance of climate data reflects the strength of variations of a climate system and has units of the square of the data entries, such as $[^\circ\text{C}]^2$. You may have noticed the denominator $n-1$ instead of $n$, which is for an estimate of unbiased sample variance. The standard deviation describes the spread of the sample entries and has the same units as the data. A large standard deviation means that the samples have a broad spread.

Skewness is a dimensionless quantity. It measures the asymmetry of sample data. Zero skewness means a symmetric distribution about the sample mean. For example, the skewness of a normal distribution is zero. Negative skewness denotes a skew to the left, meaning the existence of a long tail on the left side of the distribution. Positive skewness implies a long tail on the right side.

The words "kurtosis" and "kurtic" are Greek in origin and indicate peakedness. Kurtosis is also dimensionless and indicates the degree of peakedness of a probability distribution. The kurtosis of a normal distribution[1] is zero when 3 is subtracted as in Eq. (1.7). Positive kurtosis means a high peak at the mean, thus the distribution shape is slim and tall. This is referred to as leptokurtic. "Lepto" is Greek in origin and means thin or fine. Negative kurtosis indicates a low peak at the mean, thus the distribution shape is fat and short, referred to as platykurtic. "Platy" is also Greek in origin and means flat or broad.

For the 139 years of the NOAAGlobalTemp global average annual mean temperature anomalies, mean is $-0.1959^\circ$C, which means that the 1880–2018 average is lower than the average during the climatology period 1971–2000. During the climatology period, the temperature anomaly average is approximately zero, and can be computed by the R command `mean(temp2018[92:121])`.

The variance of the data in the 139 years from 1880 to 2018 is 0.1055 $[^\circ\text{C}]^2$, and the corresponding standard derivation is 0.3248 [°C]. The skewness is 0.7743, meaning skewed with a long tail on the right, thus with more extreme high temperatures than extreme low temperatures, as shown by Figure 1.4. The kurtosis is $-0.2619$, meaning the distribution is flatter than a normal distribution, as shown in the histogram Figure 1.4.

The median is $-0.2744^\circ$C and is a number characterizing a set of samples such that 50% of the sample values are less than the median, and another 50% are greater than the median. To find the median, sort the samples from the smallest to the largest. The median is then the sample value in the middle. If the number of the samples is even, then the median is equal to the mean of the two middle sample values.

[1] Chapter 2 will have a detailed description of the normal distribution and other probabilistic distributions.

Quantiles are defined in a way similar to median by sorting. For example, 25-percentile (also called the 25th percentile) −0.4120°C is a value such that 25% of sample data are less than this value. By definition, 60-percentile is thus larger than 50-percentile. Here, percentile is a description of quantile relative to 100. Obviously, 100-percentile is the largest datum, and 0-percentile is the smallest one. Often, a box plot is used to show the typical quantiles (see Fig. 1.5).

The 50-percentile (or 50th percentile) −0.2744°C is equal to the median. If the distribution is symmetric, then the median is equal to the mean. Otherwise, these two quantities are not equal. If the skew is to the right, then the mean is on the right of the median: the mean is greater than the median. If the skew is to the left, then the mean is on the left of the median: the mean is less than the median. Our 139 years of temperature data are right skewed and have mean equal to −0.1959°C, greater than their median equal to −0.2969°C.

## 1.2 Commonly Used Climate Statistical Plots

We will use the 139 years of NOAAGlobalTemp temperature data and R to illustrate some commonly used statistical figures: histogram, boxplot, Q-Q plot, and linear regression trend line.

### 1.2.1 Histogram of a Set of Data

A histogram of the NOAAGlobalTemp global average annual mean temperature anomalies data from 1880 to 2018 is shown in Figure 1.4, which can be generated by the following computer code.

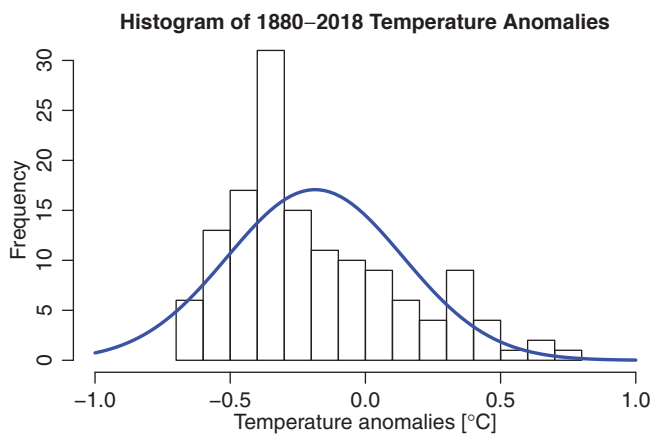


**Figure 1.4** Histogram and its normal distribution fit of the global average annual mean temperature anomalies from 1880 to 2018. Each small interval in the horizontal coordinate is called a bin. The frequency in the vertical coordinate is the number of temperature anomalies in a given bin. For example, the frequency for the bin [−0.5, −0.4]°C is 17.

```
#R plot Fig. 1.4: Histogram and its fit
par(mar=c(3.5,3.5,2.5,1), mgp=c(2,0.8,0))
h <- hist(NOAAtemp[1:139, 2],
          main="Histogram␣of␣1880-2018␣Temperature␣Anomalies",
          xlab=expression(paste(
                  "Temperature␣anomalies␣[", degree, "C]")),
          xlim=c(-1,1), ylim=c(0,30),
          breaks=10, cex.lab=1.2, cex.axis=1.2)
xfit <- seq(-1, 1, length=100)
areat <- sum((h$counts)*diff(h$breaks[1:2]))#Normalization area
yfit <- areat*dnorm(xfit,
                    mean=mean(NOAAtemp[1:139,2]),
                    sd=sd(NOAAtemp[1:139,2]))
#Plot the normal fit on the histogram
lines(xfit, yfit, col="blue", lwd=3)
```

```
#Python plot Fig. 1.4: Histogram and its fit
mu, std = scistats.norm.fit(y)
# create an evenly spaced sequence of 100 points in [-1,1]
points = np.linspace(-1, 1, 100)
plt.hist(y, bins = 20, range=(-1,1), color ='white',
         edgecolor = 'k', density = True);
plt.plot(points, scistats.norm.pdf(points, mu, std),
         color = 'b', linewidth = 3)
plt.title(r"Histogram␣of␣1880␣-␣2018␣Temperature␣Anomalies",
          pad = 20)
plt.xlabel("Temperature␣Anomalies␣[$\degree$C]",
           size = 25, labelpad = 20)
plt.ylabel("Frequency", size = 25, labelpad = 20)
plt.show()
```

The shape of the histogram agrees with the characteristics predicted by the statistical indices in the previous subsection:

(i) The distribution is asymmetric and skewed to the right with skewness equal to 0.7743.

(ii) The distribution is platykurtic with a kurtosis equal to 0.2619, i.e., it is flatter than the standard normal distribution indicated by the blue curve.

## 1.2.2 Box Plot

Figure 1.5 is the box plot of the 1880-2018 NOAAGlobalTemp global average annual mean temperature anomaly data, and can be made from the following R command.

```
#R plot Fig. 1.5: Box plot
boxplot(NOAAtemp[1:139, 2], ylim = c(-0.8, 0.8),
        ylab=expression(paste(
          "Temperature␣anomalies␣[", degree, "C]")),
        width=NULL, cex.lab=1.2, cex.axis=1.2)
```

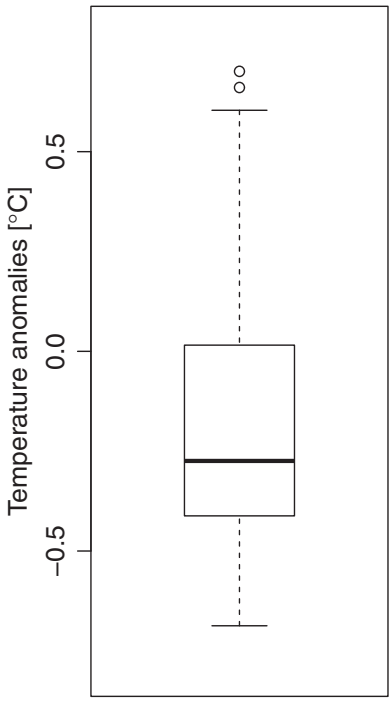


**Figure 1.5** Box plot of the global average annual mean temperature anomalies from 1880 to 2018.

The rectangular box's mid line indicates the median, which is 0.2744°C. The rectangular box's lower boundary is the first quartile, i.e., 25th percentile 0.4120°C. The box's upper boundary is the third quartile, i.e., the 75th percentile 0.0155°C. The box's height is the third quartile minus the first quartile, and is called the interquartile range (IQR). The upper "whisker" is the third quartile plus 1.5 IQR. The lower whisker is supposed to be at the first quartile minus 1.5 IQR. However, this whisker would then be lower than the lower extreme. In this case, the lower whisker takes the value of the lower extreme, which is −0.6872°C. The points outside the two whiskers are considered outliers. Our dataset has two outliers, which are 0.6607 and 0.7011°C, and are denoted by two small circles in the box plot. The two outliers occurred in 2015 and 2016, respectively.

```
#Python plot Fig. 1.5: Box plot
medianprops = dict(linestyle='-', linewidth=2.5, color='k')
whiskerprops = dict(linestyle='--',linewidth=2.0, color='k')
plt.boxplot(y, medianprops = medianprops,
            whiskerprops = whiskerprops);
plt.title("Boxplot␣of␣1880␣-␣2018␣Temperature␣Anomalies",
          pad = 20)
plt.ylabel("Temperature␣Anomalies␣[$\degree$C]",
           size = 25, labelpad = 20)
y_ticks = np.linspace(-0.5,0.5,3)
plt.yticks(y_ticks)
plt.show()
```

## 1.2.3 Q-Q Plot

Figure 1.6 shows quantile-quantile (Q-Q) plots, also denoted by q-q plots, qq-plots, or QQ-plots.

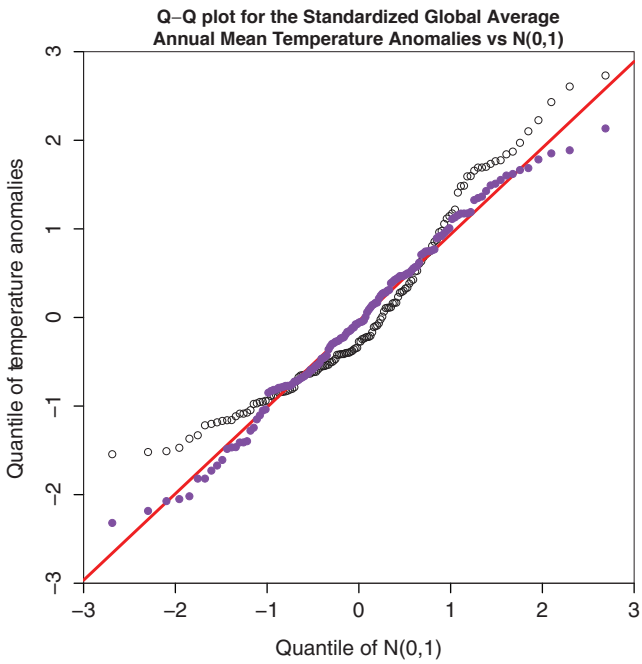


**Figure 1.6** Black empty-circle points are the Q-Q plot of the standardized global average annual mean temperature anomalies versus standard normal distribution. The purple points are the Q-Q plot for the data simulated by `rnorm(139)`. The red is the distribution reference line of $N(0,1)$.

The function of a Q-Q plot is to compare the distribution of a given set of data with a specific reference distribution, such as a standard normal distribution with zero mean and standard deviation equal to one, denoted by $N(0,1)$. A Q-Q plot lines up the percentiles of data on the vertical axis and the same number of percentiles of the specific reference distribution on the horizontal axis. The pairs of the quantiles $(x_i, y_i), i = 1, 2, \ldots, n$ determine the points on the Q-Q plot. Here, $x_i$ and $y_i$ correspond to the same cumulative percentage or probability $p_i$ for both $x$ and $y$ variables, where $p_i$ monotonically increases from approximately 0 to 1 as $i$ goes from 1 to $n$. A red Q-Q reference line is plotted as if the vertical axis values are also the quantiles of the given specific distribution. Thus, the Q-Q reference line should be diagonal.

The black empty circles in Figure 1.6 compare the quantiles of the standardized global average annual mean temperature anomalies marked on the vertical axis with those of the standard normal distribution marked on the horizontal axis. The standardized anomalies are equal to anomalies divided by the sample standard deviation. The purple dots shows a Q-Q plot of a set of 139 random numbers simulated by the standard normal distribution. As expected, the simulated points are located close to the red diagonal line, which is the distribution reference line of $N(0,1)$. On the other hand, the temperature Q-Q plot shows a considerable degree of scattering of the points away from the reference line. We may intuitively conclude that the global average annual temperature anomalies from 1880 to 2018 are not exactly distributed according to a normal (also known as Gaussian) distribution.

However, we may also conclude that the distribution of these temperatures is not very far away from the normal distribution either, because the points on the Q-Q plot are not very far away from the distribution reference line, and also because even the simulated $N(0,1)$ points are noticeably off the reference line for the extremes.

Figure 1.6 can be generated by the following computer code.

```
#R plot Fig. 1.6: Q-Q plot for the standardized
# global average annual mean temperature anomalies
temp2018 <- NOAAtemp[1:139,2]
tstand <- (temp2018 - mean(temp2018))/sd(temp2018)
set.seed(101)
qn <- rnorm(139) #simulate 139 points by N(0,1)
qns <- sort(qn) # sort the points
qq2 <- qqnorm(qns,col="blue", lwd = 2)

setEPS() #Automatically saves the eps file
postscript("fig0106.eps", height=7, width=7)
par(mar = c(4.5,5,2.5,1), xaxs = "i", yaxs = "i")
qt = qqnorm(tstand,
  main = "Q-Q␣plot␣for␣the␣Standardized␣Global␣Average
␣␣Annual␣Mean␣Temperature␣Anomalies␣vs␣N(0,1)",
   ylab="Quantile␣of␣Temperature␣Anomalies",
   xlab="Quantile␣of␣N(0,1)",
  xlim=c(-3,3), ylim = c(-3,3),
          cex.lab = 1.3, cex.axis = 1.3)
qqline(tstand, col = "red", lwd=3)
points(qq2$x, qq2$y, pch = 19,
       col ="purple")
dev.off()
```

In the R code, we first standardize (also called normalize) the global average annual mean temperature data by subtracting the data mean and dividing by the data's standard deviation. Then, we use these 139 years of standardized global average annual mean temperature anomalies to generate a Q-Q plot, which is shown in Figure 1.6.

```
#Python plot Fig. 1.6: Q-Q plot for the standardized
# global average annual mean temperature anomalies
NOAAtemp = read_table(
     "data/aravg.ann.land_ocean.90S.90N.v4.0.1.201907.txt",
                    header = None, delimiter = "\s+")
x = np.array(NOAAtemp.loc[0:138, 0])
y = np.array(NOAAtemp.loc[0:138, 1])
line = np.linspace(-3, 3, y.size)
tstand = np.sort((y - np.mean(y))/np.std(y))
# simulate 139 points following N(0,1)
qn = np.random.normal(size = y.size)
qns = np.sort(qn) # sort the points
qq2 = sm.qqplot(qns)
fig = plt.figure(figsize=(12,12)) # set up figure
sm.qqplot(tstand, color = "k", linewidth = 1)
# plot diagonal line
plt.plot(line, line, 'r-', linewidth = 3)
```

```
# Q-Q plot of standard normal simulations
plt.plot(line, qns, 'mo')
plt.tick_params(length=6, width=2, labelsize=20)
plt.title("Q-Q plot for the Standardized Global \n \
Temperature Anomalies vs N(0,1)", pad = 20)
plt.xlabel("Quantile of N(0,1)", size = 25, labelpad = 20)
plt.ylabel("Quantile of Temperature Anomalies", size = 25)
plt.show()
```

## 1.2.4 Plot a Linear Trend Line

Climate data analysis often involves plotting a linear trend line for time series data, such as the linear trend for the global average annual mean surface temperature anomalies, shown in Figure 1.7. The R code for plotting a linear trend line of data sequence y and time sequence t is `abline(lm(y ~ t))`.

Figure 1.7 can be generated by the following computer code.

```
#R plot Fig. 1.7: Data line graph with a linear trend line
par(mar=c(3.5,3.5,2.5,1), mgp=c(2,0.8,0))
plot(NOAAtemp[1:139,1], NOAAtemp[1:139,2],
     type="l", col="brown", lwd=3,
     main=" Global Land-Ocean Average Annual Mean
Surface Temperature Anomaly: 1880-2018",
     cex.lab=1.2,cex.axis=1.2,
     xlab="Year",
     ylab=expression(paste(
       "Temperature Anomaly [", degree,"C]"))
)
abline(lm(NOAAtemp[1:139,2] ~ NOAAtemp[1:139,1]),
       lwd=3, col="blue")
```

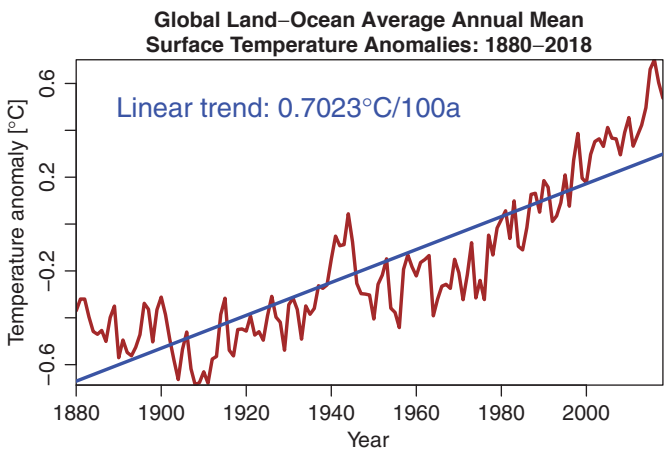


**Figure 1.7** Linear trend line with the 1880–2018 global average annual mean surface temperature based on the NOAAGlobalTemp V4.0 dataset.

```
lm(NOAAtemp[1:139,2] ~ NOAAtemp[1:139,1])
#       (Intercept)  NOAAtemp[1:139, 1]
#-13.872921             0.007023
#Trend 0.7023 degC/100a
text(1930, 0.5,
 expression(paste("Linear trend: 0.7023",
                  degree,"C/100a")),
     cex = 1.5, col="blue")
```

```
#Python plot Fig. 1.7: Data line graph with a trend line
trend = np.array(np.polyfit(x, y, 1))
abline = trend[1] + x*trend[0]
plt.plot(x, y, 'k-',
         color = 'tab:brown', linewidth = 3);
plt.plot(x, abline, 'k-', color = 'b', linewidth = 3);
plt.title("Global Land-Ocean Average Annual Mean \n \
Surface Temperature Anomaly: 1880 - 2018", pad = 20);
plt.text(1880, 0.5, r"Linear trend: 0.7023$\degree$C/100a",
         color= 'b', size = 28)
plt.xlabel("Year", size = 25, labelpad = 20)
plt.ylabel("Temperature Anomaly [$\degree$C]",
           size = 25, labelpad = 20)
```

## 1.3 Read netCDF Data File and Plot Spatial Data Maps

### 1.3.1 Read netCDF Data

Climate data are at spatiotemporal points, such as at the grid points on the Earth's surface and at a sequence of time. NetCDF (Network Common Data Form) is a popular file format for modern climate data with spatial locations and temporal records. The gridded NOAAGlobalTemp data have a netCDF version, and can be downloaded from
`www.esrl.noaa.gov/psd/data/gridded/data.noaaglobaltemp.html`

The data are written into a 3D array, with 2D latitude–longitude for space, and 1D for time. R and Python can read and plot the netCDF data. We use the NOAAGlobalTemp as an example to illustrate the netCDF data reading and plotting. Figure 1.8 displays a temperature anomaly map for the entire globe for December 2015.

Figure 1.8 can be generated by the following computer code.

```
#R read the netCDF data: NOAAGlobalTemp
setwd("/Users/sshen/climstats")
#install.packages("ncdf4")
library(ncdf4)
nc = ncdf4::nc_open("data/air.mon.anom.nc")
nc # describes details of the dataset
Lat <- ncvar_get(nc, "lat")
```

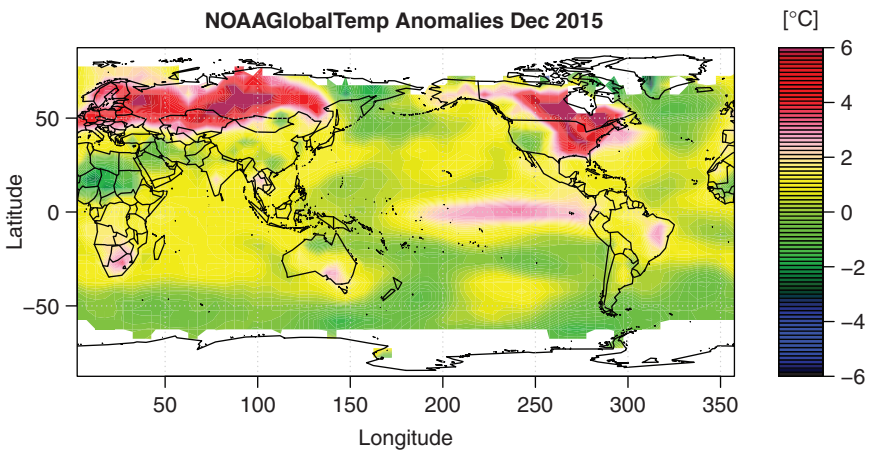


**Figure 1.8** The surface temperature anomalies of December 2015 with respect to the 1971–2000 climatology (data source: The NOAAGlobalTemp V4.0 gridded monthly data).

```
Lat # latitude data
#[1] -87.5 -82.5 -77.5 -72.5 -67.5 -62.5
Lon <- ncvar_get(nc, "lon")
Lon # longitude data
#[1]   2.5   7.5  12.5  17.5  22.5  27.5
Time <- ncvar_get(nc, "time")
head(Time) # time data in Julian days
#[1] 29219 29250 29279 29310 29340 29371
library(chron) # convert Julian date to calendar date
nc$dim$time$units # .nc base time for conversion
#[1] "days since 1800-1-1 00:00:0.0"
month.day.year(29219,c(month = 1, day = 1, year = 1800))
#1880-01-01 # the beginning time of the dataset
tail(Time)
#[1] 79988 80019 80047 80078 80108 80139
month.day.year(80139,c(month = 1, day = 1, year = 1800))
#2019-06-01 # the end time of the dataset

# extract anomaly data in (lon, lat, time) coordinates
NOAAgridT <- ncvar_get(nc, "air")
dim(NOAAgridT) # dimensions of the data array
#[1]  72   36 1674 #5-by-5, 1674 months from Jan 1880-Jun 2019
```

```
#Python read the netCDF data: NOAAGlobalTemp
import netCDF4 as nc # import netCDF data reading package
# go to the working directory
os.chdir('/Users/sshen/climstats')
# read a .nc file from the folder named "data"
nc = nc.Dataset('data/air.mon.anom.nc')
# get the detailed description of the dataset
print(nc)
```

```
# extract latitude, longitude, time and temperature
lon = nc.variables['lon'][:]
lat = nc.variables['lat'][:]
time = nc.variables['time'][:]
air = nc.variables['air']
# covert Julian date to calendar date
from netCDF4 import num2date
from datetime import datetime, date, timedelta
from matplotlib.dates import date2num, num2date
units = nc.variables['time'].units
print(units)
dtime = num2date(time)
```

### 1.3.2 Plot a Spatial Map of Temperature

The NOAAGlobalTemp anomaly data on a 5-degree latitude–longitude grid for a given time can be represented by a color map. Figure 1.8 shows the temperature anomaly map for December 2015, an El Niño month. The eastern tropical Pacific has positive anomalies, which is a typical El Niño signal. That particular month also exhibited a very large anomaly across Europe, and the eastern United States and Canada. The white areas over the high latitude regions lack data.

The figure can be generated by the following computer code.

```
#R plot Fig. 1.8: Dec 2015 surface temp anomalies map
library(maps)# requires maps package
mapmat=NOAAgridT[,,1632]
# Column of NOAAgridT 1632 corresponds to Dec 2015
mapmat=pmax(pmin(mapmat,6),-6) # put values in [-6, 6]
int=seq(-6, 6, length.out=81)
rgb.palette=colorRampPalette(c('black','blue',
   'darkgreen', 'green', 'yellow','pink','red','maroon'),
                              interpolate='spline')
par(mar=c(3.5, 4, 2.5, 1), mgp=c(2.3, 0.8, 0))
filled.contour(Lon, Lat, mapmat,
               color.palette=rgb.palette, levels=int,
  plot.title=title(main="NOAAGlobalTemp␣Anomalies␣Dec␣2015",
           xlab="Latitude", ylab="Longitude", cex.lab=1.2),
  plot.axes={axis(1, cex.axis=1.2, las=1);
     axis(2, cex.axis=1.2, las=2); map('world2', add=TRUE); grid()},
  key.title=title(main=expression(paste("[", degree, "C]"))),
  key.axes={axis(4, cex.axis=1.2)})
```

```
#Python plot Fig. 1.8: Dec 2015 surface temp anomalies map
dpi = 100
fig = plt.figure(figsize = (1100/dpi, 1100/dpi), dpi = dpi)
ax  = fig.add_axes([0.1, 0.1, 0.8, 0.9])
```

```
# create map
dmap = Basemap(projection = 'cyl', llcrnrlat = min(lat),
               urcrnrlat = max(lat), resolution = 'c',
               llcrnrlon = min(lon), urcrnrlon = max(lon))
# draw coastlines, state and country boundaries, edge of map
dmap.drawcoastlines()
dmap.drawstates()
dmap.drawcountries()
# convert latitude/longitude values to plot x/y values
x, y = dmap(*np.meshgrid(lon, lat))
# draw filled contours
cnplot = dmap.contourf(x, y, mapmat, clev, cmap=myColMap)
# tick marks
ax.set_xticks([0, 50, 100, 150, 200, 250, 300, 350])
ax.set_yticks([-50,0,50])
ax.tick_params(length=6, width=2, labelsize=20)
# add colorbar
# pad: distance between map and colorbar
cbar = dmap.colorbar(cnplot, pad = "4%",drawedges=True,
           shrink=0.55, ticks = [-6,-4,-2,0,2,4,6])
# add colorbar title
cbar.ax.set_title('[$\degree$C]', size= 17, pad = 10)
cbar.ax.tick_params(labelsize = 15)
# add plot title
plt.title('NOAAGlobalTemp␣Anomalies␣Dec␣2015',
          size = 25, fontweight = "bold", pad = 15)
# label x and y
plt.xlabel('Longitude', size = 25, labelpad = 20)
plt.ylabel('Latitude', size = 25, labelpad = 10)
# display on screen
plt.show()
```

### 1.3.3 Panoply Plot of a Spatial Map of Temperature

You can also use the Panoply software package to plot the map (see Fig. 1.9). This is a very powerful data visualization tool developed by NASA specifically for displaying netCDF files. The software package is free and can be downloaded from `www.giss.nasa.gov/tools/panoply`.

To make a Panoply plot, open Panoply and choose `Open` from the `File` menu. `Open` dropdown, which will allow you to go to the right directory to find the netCDF file you wish to plot. In our case, the file is `air.mon.anom.nc`. Choose the climate parameter `air`. Click on `Create Plot`. A map will show up. Then you have many choices to modify the map, ranging from the data `Array`, `Scale`, and `Map`, etc., and finally produce the figure. You can then tune the figure by choosing different graphics parameters underneath the figure, such as `Array(s)` to choose which month to plot, `Map` to choose the map projection types and the map layout options, and `Labels` to type proper captions and labels.

To learn more about Panoply, please use an online Panoply tutorial, such as the NASA Panoply help page `www.giss.nasa.gov/tools/panoply/help/`.

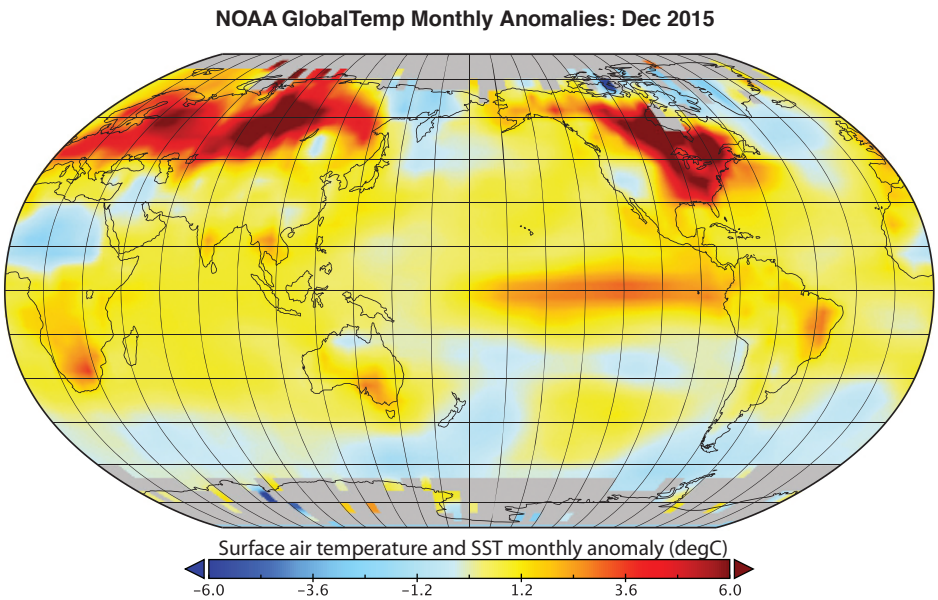


**Figure 1.9** A Panoply plot of Robinson projection map for the surface temperature anomalies of December 2015.

Compared with the R or Python map Figure 1.8, the visualization effect of the Panoply map seems more appealing. However, R and Python have the advantage of flexibility and can deal with all kinds of data. For example, the Plotly graphing library in R `https://plotly.com/r/` and in Python `https://plotly.com/python/` can even make interactive and 3D graphics. You may find some high-quality figures from the Intergovernmental Panel on Climate Change (IPCC) report (2021) `www.ipcc.ch` and reproduce them using the computing tools described here.

## 1.4 1D-space-1D-time Data and Hovmöller Diagram

A very useful climate data visualization technique is the Hovmöller diagram. It displays how a climate variable varies with respect to time along a given line section of latitude, longitude, or altitude. It usually has the abscissa for time and ordinate for the line section. A Hovmöller diagram can conveniently show time evolution of a spatial pattern, such as wave motion from south to north or from west to east.

Figure 1.10 is a Hovmöller diagram for the sea surface temperature (SST) anomalies at a longitude equal to 240°, i.e., 120° W, in a latitude interval [30° S, 30° N], with time range from January 1989 to December 2018. When the red strips become strong from the south to north, a strong El Niño occurs, such as those in the 1997–1998 and 2015–2016 winters.

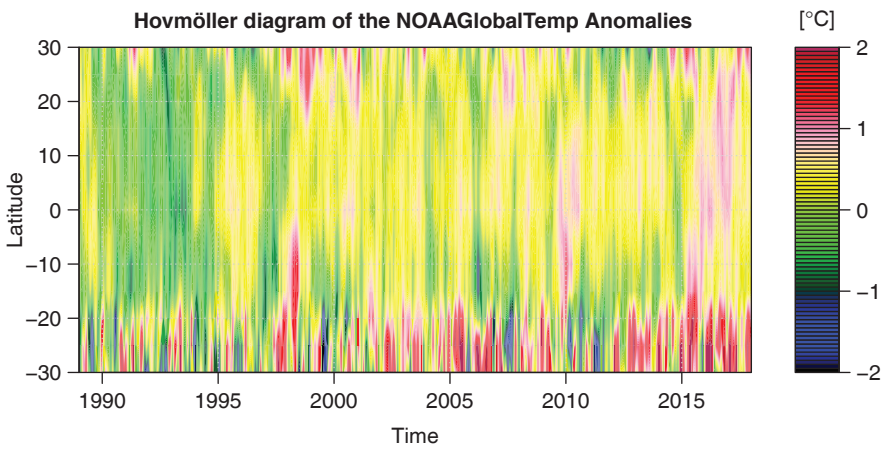


**Figure 1.10** Hovmöller diagram for the gridded NOAAGlobalTemp monthly anomalies at longitude 120° W and a latitude interval [30° S, 30° N].

The Hovmöller diagram Figure 1.10 may be plotted by the following computer code.

```
#R plot Fig. 1.10: Hovmoller diagram
library(maps)
mapmat=NOAAgridT[30,12:24,1309:1668]
#Longitude= 240 deg, Lat =[-30 30] deg
#Time=Jan 1989-Dec 2018: 30 years
mapmat=pmax(pmin(mapmat,2),-2) # put values in [-2,2]
par(mar=c(4,5,3,0))
int=seq(-2,2,length.out=81)
rgb.palette=colorRampPalette(c('black','blue',
 'darkgreen','green', 'yellow','pink','red','maroon'),
                              interpolate='spline')
par(mar=c(3.5,3.5,2.5,1), mgp=c(2.4, 0.8, 0))
x = seq(1989, 2018, len=360)
y = seq(-30, 30, by=5)
filled.contour(x, y, t(mapmat),
       color.palette=rgb.palette, levels=int,
       plot.title=title(main=
    "Hovmoller␣diagram␣of␣the␣NOAAGlobalTemp␣Anomalies",
                xlab="Time",ylab="Latitude", cex.lab=1.2),
      plot.axes={axis(1, cex.axis=1.2);
                 axis(2, cex.axis=1.2);
                 map('world2', add=TRUE);grid()},
        key.title=title(main =
                expression(paste("[", degree, "C]"))),
        key.axes={axis(4, cex.axis=1.2)})
```

```
#Python plot Fig. 1.10: Hovmoller diagram
mapmat2 = NOAAgridT[1308:1668,11:23,29]
```

```
# define values between -2 and 2
mapmat2 = np.array([[j if j < 2 else 2 for j in i]
                    for i in mapmat2])
mapmat2 = np.array([[j if j > -2 else -2 for j in i]
                    for i in mapmat2])
# find dimensions
mapmat2 = np.transpose(mapmat2)
print(mapmat2.shape)
lat3 = np.linspace(-30,30, 12)
print(lat3.shape)
time = np.linspace(1989, 2018, 360)
print(time.shape)

# plot functions
myColMap = LinearSegmentedColormap.from_list(
    name='my_list',
    colors=['black','blue','darkgreen','green','lime',
            'yellow', 'pink','red','maroon'],  N=100)
clev2 = np.linspace(mapmat2.min(), mapmat2.max(), 501)
contf = plt.contourf(time, lat3, mapmat2,
                     clev2, cmap=myColMap);

plt.text(2019.2, 31.5,
         "[$\degree$C]", color='black', size = 23)
plt.title("Hovmoller␣diagram␣of␣the␣\n␣\
NOAAGlobalTemp␣Anomalies",
          fontweight = "bold", size = 25, pad = 20)
plt.xlabel("Time", size = 25, labelpad = 20)
plt.ylabel("Latitude", size = 25, labelpad = 12)
colbar = plt.colorbar(contf, drawedges=False,
                      ticks = [-2,-1,0,1,2])
```

## 1.5 4D netCDF File and Its Map Plotting

The 4D climate data means 3D spatial dimensions and 1D time. For example, the NCEP Global Ocean Data Assimilation System (GODAS) monthly water temperature data are at 40 depth levels ranging from 5 meters to 4478 meters and at 1/3 degree latitude by 1 degree longitude horizontal resolution and are from January 1980. The NOAA-CIRES 20th Century Reanalysis (20CR) monthly air temperature data are at 24 different pressure levels ranging from 1,000 mb to 10 mb and at 2-degree latitude and longitude horizontal resolution and are from January 1851.

GODAS data can be downloaded from NOAA ESRL,
`www.esrl.noaa.gov/psd/data/gridded/data.godas.html`.
The data for each year is a netCDF file and is about 140MB. The following R code can read the GODAS 2015 data into R.

```
#R read a 4D netCDF file: lon, lat, level, time
setwd("/Users/sshen/climstats")
library(ncdf4)
# read GODAS data 1-by-1 deg, 40 levels, Jan-Dec 2015
nc=ncdf4::nc_open("data/godas2015.nc")
nc
Lat <- ncvar_get(nc, "lat")
Lon <- ncvar_get(nc, "lon")
Level <- ncvar_get(nc, "level")
Time <- ncvar_get(nc, "time")
head(Time)
#[1] 78527 78558 78586 78617 78647 78678
library(chron)
month.day.year(78527,c(month = 1, day = 1, year = 1800))
# 2015-01-01
# potential temperature pottmp[lon, lat, level, time]
godasT <- ncvar_get(nc, "pottmp")
dim(godasT)
#[1] 360 418  40  12,
#i.e., 360 lon, 418 lat, 40 levels, 12 months=2015
t(godasT[246:250, 209:210, 2, 12])
#Dec level 2 (15-meter depth) water temperature [K] of
#a few grid boxes over the eastern tropical Pacific
#        [,1]     [,2]     [,3]     [,4]     [,5]
#[1,] 300.0655 299.9831 299.8793 299.7771 299.6641
#[2,] 300.1845 300.1006 299.9998 299.9007 299.8045
```

```
#Python read a netCDF data file
import netCDF4 as nc
# go to your working directory
os.chdir('/Users/sshen/climstats')
# read a .nc file from the folder named "data"
nc1 = nc.Dataset('data/godas2015.nc')
# get the detailed description of the dataset
print(nc1)
# dimensions of the 2015 pottem data array
godasT = nc1.variables['pottmp'][:]
print(godasT.shape)
# (12, 40, 418, 360)
```

Figure 1.11 shows the 2015 annual mean water temperature at 195 meters depth based on the GODAS data. At this depth level, the equatorial upwelling appears: The deep ocean cooler water in the equatorial region upwells and makes the equatorial water cool. The equatorial water is not the hottest anymore at this level, and is cooler than the water in some subtropical regions as shown in Figure 1.11.

Figure 1.11 can be generated by the following computer code.

```
#R plot Fig. 1.11: The ocean potential temperature
# the 20th layer from surface: 195 meters depth
# compute 2015 annual mean temperature at 20th layer
library(maps)
climmat=matrix(0,nrow=360,ncol=418)
```

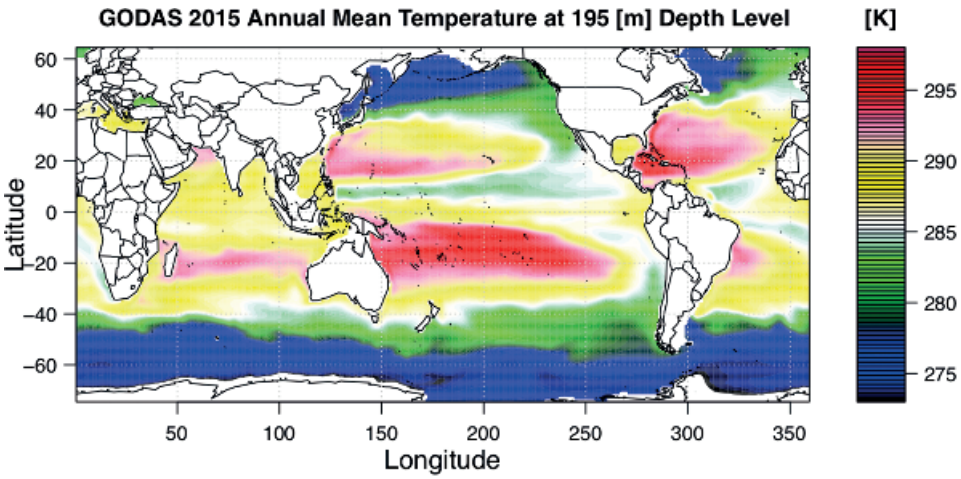


**Figure 1.11** The 2015 annual mean water temperature at 195 meters depth based on GODAS data.

```
sdmat=matrix(0,nrow=360,ncol=418)
Jmon<-1:12
for (i in 1:360){
  for (j in 1:418){
    climmat[i,j] = mean(godasT[i,j,20,Jmon]);
    sdmat[i,j]=sd(godasT[i,j,20,Jmon])
                  }
                }
int=seq(273,298,length.out=81)
rgb.palette=colorRampPalette(c('black','blue',
    'darkgreen','green', 'white','yellow',
    'pink','red','maroon'), interpolate='spline')
par(mar=c(3.5, 3.5, 2.5, 0), mgp=c(2, 0.8, 0))
filled.contour(Lon, Lat, climmat,
     color.palette=rgb.palette, levels=int,
    plot.title=title(main=
  "GODAS 2015 Annual Mean Temperature at 195[m] Depth Level",
            xlab="Longitude",ylab="Latitude",
            cex.lab=1.3, cex.axis=1.3),
plot.axes={axis(1); axis(2); map('world2', add=TRUE);grid()},
            key.title=title(main="[K]"))
```

```
#Python plot Fig. 1.11: A spatial map from 4D data
climmat = np.zeros((360, 418))
for i in range(360):
    for j in range(418):
      climmat[i,j] = np.mean(godasT[:, 20, j, i])
climmat = np.transpose(climmat)
lat4 = np.linspace(-75, 65, 418)
long = np.linspace(0, 360, 360)
```

```
myColMap = LinearSegmentedColormap.from_list(
    name='my_list',
    colors=['black','blue','darkgreen','green',
            'white','yellow','pink','red','maroon'],
         N=100)
plt.figure(figsize=(18,12));
clev3 = np.arange(godasT.min(), godasT.max(), 0.1)
contf = plt.contourf(long, lat4, climmat,
                     clev3, cmap=myColMap);
plt.text(382, 66, "[K]", fontsize=23, color='black')
plt.tick_params(length=6, width=2, labelsize=20)
m = Basemap(projection='cyl', llcrnrlon=0,
    urcrnrlon=360, resolution='l', fix_aspect=False,
    suppress_ticks=False, llcrnrlat=-75, urcrnrlat=65)
m.drawcoastlines(linewidth=1)
plt.title("GODAS␣2015␣Annual␣Mean␣Temperature␣\n␣\
at␣195[m]␣Depth␣Level",
          size = 25, fontweight = "bold", pad = 20)
plt.xlabel("Longitude", size = 25, labelpad = 20)
plt.ylabel("Latitude", size = 25, labelpad = 15)
plt.tick_params(length=6, width=2, labelsize=30)
colbar = plt.colorbar(contf, drawedges=False,
                      ticks = [275,280,285,290,295])
```

## 1.6 Paraview, 4DVD, and Other Tools

Besides using R, Python, and Panoply to plot climate data, other software packages may also be used to visualize data for some specific purposes, such as Paraview for 3D visualization and 4DVD for fast climate diagnostics and data delivery.

### 1.6.1 Paraview

Paraview is an open-source data visualization software package, available at `www.paraview.org/download`. There are many online tutorials, such as `www.paraview.org/tutorials`.

You may use the software ParaView to plot the GODAS data in a 3D view as shown in Figure 1.12 for the December 2015 water temperature.

### 1.6.2 4DVD

4DVD (4-dimensional visual delivery of big climate data) is a fast data visualization and delivery software system at `www.4dvd.org`. It optimally harnesses the power of distributed computing, databases, and data storage to allow a large number of general public users to quickly visualize climate data. For example, teachers and students can use 4DVD to

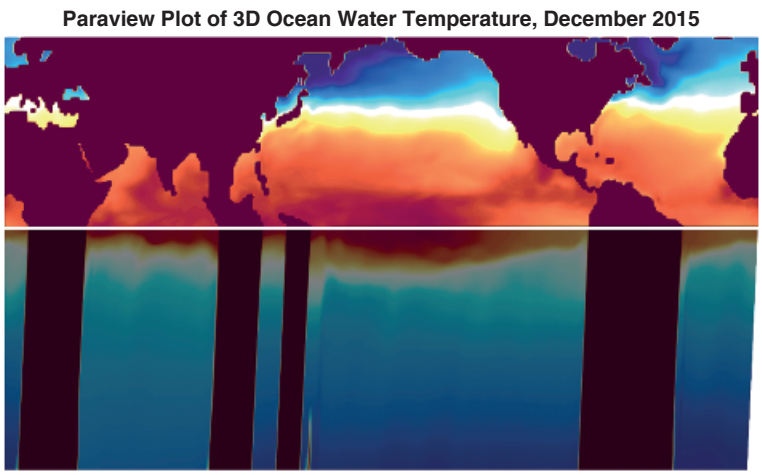


**Figure 1.12** A 3D view of the December 2015 annual mean water temperature based on the GODAS data. The surface of the Northern Hemisphere and the cross-sectional map along the equator from surface to the 2,000 meters depth. The dark color indicates land; the red, yellow, and blue colors indicate temperature.

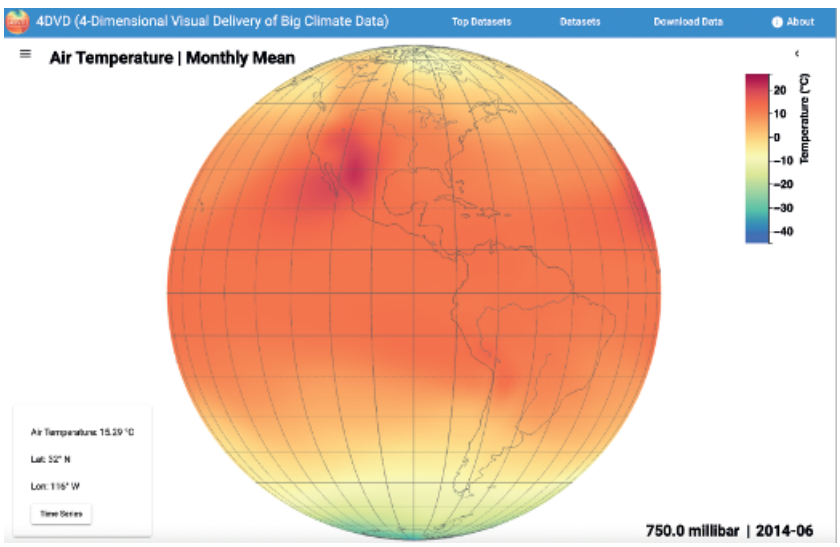


**Figure 1.13** The 4DVD screenshot for the NOAA-CIRES 20th Century Reanalysis temperature at 750 millibar height level for June 2014.

visualize climate model data in classrooms and download the visualized data instantly. The 4DVD website has a tutorial for users.

Here, we provide an example of 4DVD visualization of the NOAA-CIRES 20CR climate model data for atmosphere. The 4DVD can display a temperature map at a given time and given pressure level, as shown in Figure 1.13 for January 1851 and 750 millibar pressure level, or one can obtain a time series of the monthly air temperature for a specific grid box

from January 1851. In fact, 4DVD can show multiple time series for the same latitude-longitude location but at different pressure levels. The 4DVD not only allows a user to visualize the data but also to download the data for the figure shown in the 4DVD system. In this sense, 4DVD is like a machine that plays data, while the regular DVD player machine, popular for about 30 years since the 1980s, plays DVD discs for music and movies.

### 1.6.3 Other Online Climate Data Visualization Tools

Besides R, Python, Panoply, ParaView, and 4DVD, there are many other data visualization and delivery software systems. A few popular and free ones are listed as follows.

Nullschool `https://nullschool.net` is a beautiful data visualizer of wind, ocean flows, and many climate parameters. It is supported by the data from a global numerical weather prediction system, named the Global Forecast System (GFS) run by the United States' National Weather Service (NWS).

Ventusky `www.ventusky.com` has both website and smartphone app. It has an attractive and user-friendly interface that allows users to get digital weather information instantly around the globe.

Climate Reanalyzer `climatereanalyzer.org` is a comprehensive tool for climate data plotting and download. It has a user-friendly interface for reanalysis and historical station data. The data can be exported in either CSV (comma-separated values) format or JSON (JavaScript object notation) format.

Google Earth Engine `earthengine.google.com` provides visualization tools together with a huge multi-petabytes storage of climate data. Its modeling and satellite data sources are from multiple countries.

Giovanni `giovanni.gsfc.nasa.gov` is an online climate data plotting tool with an interface. It allows users to download the plotted figures in different formats, such as png. It is supported by various kinds of NASA climate datasets.

Climate Engine `climateengine.org` is a web application for plotting climate and remote sensing data. Similar to Giovanni, it also has a tabular interface for a user to customized data and maps.

NOAA Climate at a Glance `www.ncdc.noaa.gov/cag` is a data visualization tool mainly for visualizing the observed climate data over the United States. It has functions for both spatial maps and historical time series.

Web-based Reanalyses Intercomparison Tools (WRIT) `psl.noaa.gov/data/writ` is similar to Giovanni, in that WRIT also has an interface table for a user to enter plot parameters. The plot (in postscript format) and its data (in netCDF format) can be downloaded. WRIT is designed for the data from climate reanalysis models.

### 1.6.4 Use ChatGPT as a Study Assistant

ChatGPT (generative pre-trained transformer) is an artificial intelligence (AI) system that can help us develop computer codes, write essays, draft course syllabi, and more. For example, in your box of the ChatGPT interface, you can enter "Write a Python code to plot

the cosine function as a blue thick dashed curve." ChatGPT will give you the following Python code:

```
import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(-np.pi, np.pi, 100)
y = np.cos(x)

plt.plot(x, y, linestyle='--', linewidth=3, color='blue')
plt.xlabel('x')
plt.ylabel('cos(x)')
plt.title('Plot␣of␣the␣cosine␣function')
plt.show()
```

You can copy the above code and paste it to your Jupyter Notebook cell, then run the code to produce the curve of the cosine function in the interval $[-\pi, \pi]$. With this sample and according to your needs, you can modify the code for different functions, intervals, colors, thicknesses, titles, axis labels, and more.

You can also ask ChatGPT to work on a dataset, such as "Write an R code to plot the NOAAGlobalTempts.csv data and its trend line." ChatGPT will give you an R code. You copy and paste the code to your RStudio. Set your R code in the correct directory that has the NOAAGlobalTemp annual time series data file, `NOAAGlobalTempts.csv`. Run your R code to plot the data and the linear trend line. Of course, you can modify the code according to your requirements. The R code is as follows.

```
setwd('/Users/sshen/climmath/data')
library(tidyverse)

# Load the data
data <- read.csv("NOAAGlobalTempts.csv")

# Plot the data
ggplot(data, aes(x = Year, y = Tempts)) +
  geom_line() +
  geom_smooth(method = "lm", se = FALSE) +
  ggtitle("NOAA␣Global␣Temperature␣Data") +
  xlab("Year") +
  ylab("Temperature␣(Celsius)")
```

Another example is that you enter "Write an essay on global warming." ChatGPT will write an essay for you. Most likely, it will not reflect your ideas, but it will give you a few tips, such as the definition of global warming, evidences and data, consequences, main causes, how to slow down the warming, and a conclusion. Again, you can use this ChatGPT essay as a sample and modify the essay using your own data and ideas, such as, for example, green technologies, solar farms, microclimate resources for smart cities, trends of the oil business, lifestyles, AI in self-driving electric cars, and education.

ChatGPT was released by the OpenAI Lab in November 2022. You may use this tool as your assistant while learning and working. It is only your assistant. It is not you! You can

efficiently use it to give you hints and inspire your ideas. You still have to produce your own work.

## 1.7 Chapter Summary

This chapter has provided a brief introduction to useful statistical concepts and methods for climate science and has included the following material.

(i) Formulas to compute the most commonly used statistical indices:

- Mean as a simple average, median as a datum whose value is in the middle of the sorted data sequence, i.e., the median is larger than 50% of the data and smaller than the remaining 50%,
- Standard deviation as a measure of the width of the probability distribution,
- Variance as the square of the standard deviation,
- Skewness as a measure of the degree of asymmetry in distribution, and
- Kurtosis as a measure of the peakedness of the data distribution compared to that of the normal distribution.

(ii) The commonly used statistical plots:

- Histogram for displaying the probability distribution of data,
- Linear regression line for providing a linear model for data,
- Box plot for quantifying the probability distribution of data, and
- Q-Q plot for checking whether the data are normally distributed.

(iii) Read and plot the netCDF file for plotting a 2D map by R and Python. Other data visualization tools, such as Panoply, Paraview, 4DVD, Plotly, and Nullschool, were briefly introduced. The online tools like 4DVD and Nullschool may be used for classroom teaching and learning.

Computer codes and climate data examples are given to demonstrate these concepts and the use of the relevant tools and formulas. With this background, plus some R or Python programming skill, you will have sufficient knowledge to meet the needs of the basic statistical analysis for climate data.

# References and Further Reading


[1] B. Hennemuth, S. Bender, K. Bulow, et al., 2013: *Statistical Methods for the Analysis of Simulated and Observed Climate Data, Applied in Projects and Institutions Dealing with Climate Change Impact and Adaptation.* CSC Report 13, Climate Service Center, Germany.

`www.climate-service-center.de/imperia/md/content/csc/projekte/csc-report13_englisch_final-mit_umschlag.pdf`

This free statistics recipe book outlines numerous methods for climate data analysis, collected and edited by climate science professionals, and has examples of real climate data.

[2] IPCC, 2021: *AR6 Climate Change 2021: The Physical Science Basis*. Contribution of Working Group I to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change [V. Masson-Delmotte, P. Zhai, A. Pirani et al. (eds.)]. Cambridge University Press.

This is the famous IPCC report available for free at the IPCC website, `www.ipcc.ch/report/ar6/wg1/`. It includes many high-quality figures plotted from climate data.

[3] NCEI, 2021: *Anomalies vs. Temperature*. Last accessed on 12 May 2021.

`www.ncdc.noaa.gov/monitoring-references/dyk/anomalies-vs-temperature`

This is an excellent site for education and gives an easy-to-understand justification of the use of anomalies. From this site, you can find many other educational resources for climate science, such as the concise description of climate extreme index and global precipitation percentile maps.

[4] S. S. P. Shen and R. C. J. Somerville, 2019: *Climate Mathematics: Theory and Applications*. Cambridge University Press.

This book attempts to modernize mathematical education for undergraduate students majoring in atmospheric and oceanic sciences, or related fields. The book integrates six traditional mathematics courses (i.e., Calculus I, II, and III, Linear Algebra, Statistics, and Computer Programming) into a single course named Climate Mathematics. The course uses real climate data and can be taught in three semesters for freshmen and sophomores, or in one semester as an upper level or graduate class. Computer codes of R and Python and other learning resources are available at the book website `www.climatemathematics.org`.

[5] T. M. Smith, R. W. Reynolds, T. C. Peterson, and J. Lawrimore, 2008: Improvements to NOAA's historical merged land-ocean surface temperatures analysis (1880-2006). *Journal of Climate*, 21, 2283–2296, https://doi.org10.1175/ 2007 JCLI2100.1.

These authors are among the main contributors who reconstructed sea surface temperature (SST). They began their endeavor in the early 1990s.

[6] R. S. Vose, D. Arndt, V. F. Banzon et al., 2012: NOAA's merged land-ocean surface temperature analysis. *Bulletin of the American Meteorological Society*, 93, 1677–1685.

These authors are experts on the reconstruction of both SST and the land surface air temperature, and have published many papers in data quality control and uncertainty quantifications.

## Exercises

**1.1** The NOAA Merged Land Ocean Global Surface Temperature Analysis (NOAAGlobalTemp) V5 includes the global land-spatial average annual mean temperature anomalies as shown here:

```
1880   -0.843351   0.031336   0.009789   0.000850   0.020698
1881   -0.778600   0.031363   0.009789   0.000877   0.020698
1882   -0.802413   0.031384   0.009789   0.000897   0.020698
......
```

The first column is for time in years, and the second is the temperature anomalies in [Kelvin]. Columns 3–6 are data errors. This data file for the land temperature can be downloaded from the NOAAGlobalTemp website
`www.ncei.noaa.gov/data/noaa-global-surface-temperature/v5/access/timeseries`

The data file named `aravg.ann.land.90S.90N.v5.0.0.202104.asc.txt` is also included in `data.zip` that can be downloaded from the book website `www.climatestatistics.org`

(a) Plot the anomalies against time from 1880 to 2020 using the point-line graph like Figure 1.1.
(b) Plot the anomalies against time from 1880 to 2020 using the staircase chart like Figure 1.2.
(c) Plot the anomalies against time from 1880 to 2020 using the color bar chart like Figure 1.3.

**1.2** NOAAGlobalTemp V5 also has the global **ocean**-spatial average annual mean temperature anomalies contained in a data file named
`aravg.ann.ocean.90S.90N.v5.0.0.202104.asc.txt`
For this dataset, please plot the anomaly data in the same three styles (a), (b) and (c) as in the previous problem. You may download the data file from the NOAAGlobalTemp V5 website or from `data.zip` for this book.

**1.3** NOAAGlobalTemp V5 also has the global **land**-spatial average monthly mean temperature anomalies from January 1880 to April 2021. The data file is named
`aravg.mon.land.90S.90N.v5.0.0.202104.asc.txt`
For this dataset, please plot the January anomaly data from January 1880 to January 2021 in the same three styles (a) - (c) as in the previous problem.

**1.4** For the monthly global land data of NOAAGlobalTemp V5 in the previous problem,

(a) For the January data from 1880 to 2020, compute the following statistical indices: mean, standard deviation, variance, skewness, kurtosis, max, 75%-percentile, median, 25th percentile, and min.
(b) Do the same for February, March, ..., December.
(c) Aggregate all the results in (a) and (b) into a $10 \times 12$ matrix with the 10 statistical indices in rows and the 12 months in columns. Add proper column names, such as `Jan Feb Mar ...`, and also row names.
(d) Use 100–300 words to describe the differences of the statistical indices for different months.

**1.5** For the monthly data CRUTEM4 (Climate Research Unit surface air temperature anomalies) in 4DVD, download the historical time series data for January of a location of your interest, and compute the following statistical indices: mean, standard deviation, variance, skewness, kurtosis, max, 75th percentile, median, 25th percentile, and min.

**1.6** Do the same as the previous problem but for July. Comment on the differences between the statistical indices in January and July.

**1.7** (a) Plot a histogram of the January global land temperature anomalies using the data file in the previous problem, i.e.,
`aravg.mon.land.90S.90N.v5.0.0.202104.asc.txt`
(b) Do the same for July.
(c) Use 100–200 words to describe the comparison of the two histograms.

**1.8** (a) Plot a box plot for the January global land temperature anomalies
`aravg.mon.land.90S.90N.v5.0.0.202104.asc.txt`

(b) Do the same but for the July data.
(c) Put the two box plots next to each other in the same figure.
(d) Use 100–200 words to describe the comparison of the two box plots.

**1.9** (a) Use the monthly data in the above problem to generate 12 Q-Q plots relative to the standard normal distribution for every month from January to December.
(b) From the 12 Q-Q plots, discuss which month's temperature anomalies are the closest to the normal distribution.

**1.10** (a) Using the monthly data in the previous problem
`aravg.mon.land.90S.90N.v5.0.0.202104.asc.txt`
compute the linear trend of January temperature anomalies from 1880 to 2020. Output your result in the unit: [°C/century].
(b) Repeat (a) for each of the other 11 months.
(c) Generate a list for the 12 linear trends.
(d) Use 100–200 words to discuss the numerical values of the list.

**1.11** Plot the December 1997 surface temperature anomalies using the NOAAGlobalTemp V5 data on a $5^\circ \times 5^\circ$ grid in the netCDF format:

`NOAAGlobalTemp_v5.0.0_gridded_s188001_e202104_c20210509T133251.nc`

The data can be downloaded from NOAAGlobalTemp V5 or extracted from `data.zip` for this book at the website `www.climatestatistics.org`.

**1.12** Use Panoply to make the same December 1997 surface temperature anomalies plot as the previous problem, but with the *Robinson* map projection.

**1.13** Use Panoply to make the same December 1997 surface temperature anomalies plot as the previous problem, but with the *Mercator* map projection.

**1.14** Use Panoply to make the same December 1997 surface temperature anomalies plot as the previous problem, but with the *Orthographic* map projection.

**1.15** Use Panoply to make the same December 1997 surface temperature anomalies plot as the previous problem, but with the *Stereographic (Two-Hemisphere)* map projection.

**1.16** Plot a Hovmöller diagram like Figure 1.10 for the gridded NOAAGlobalTemp monthly anomalies at longitude 150° W and a latitude interval [50° S, 50° N] from January 1989 to December 2018 (i.e., 240 months).

**1.17** Plot a Hovmöller diagram like Figure 1.10 for the gridded NOAAGlobalTemp monthly anomalies at longitude 140° W and a latitude interval [40° S, 40° N] from January 1971 to December 2000 (i.e., 360 months).

**1.18** Plot a Hovmöller diagram like Figure 1.10 for the gridded NOAAGlobalTemp monthly anomalies on the equator with longitude from 160° E to 90° W from January 1971 to December 2000 (i.e., 360 months).

**1.19** Plot a Hovmöller diagram like Figure 1.10 for the gridded NOAAGlobalTemp monthly anomalies at latitude 5° S with longitude from 160° E to 90° W from January 1971 to December 2000 (i.e., 360 months).

**1.20** Use R or Python to plot four December 2015 potential temperature maps at the depth layers 5, 25, 105, and 195 meters based on the GODAS data. You can use the netCDF

data file `godas2015.nc` in `data.zip` for this book or download the netCDF GODAS data from a NOAA website, such as
`www.esrl.noaa.gov/psd/data/gridded/data.godas.html`

**1.21** Use Panoply to plot the same maps as the previous problem but with the Robinson projection.

**1.22** Use Plotly in R or Python to plot four December 2015 potential temperature maps at the depth layers 5, 25, 105, and 195 meters based on the GODAS data. Try to place the four maps together to make a 3D visualization.

**1.23** Plot four June 2015 potential temperature maps at the depth layers 5, 25, 105, and 195 meters using the same netCDF GODAS data file as the previous problem.

**1.24** Use Plotly in R or Python to plot four June 2015 potential temperature maps at the depth layers 5, 25, 105, and 195 meters based on the GODAS data. Try to place the four maps together to make a 3D visualization.

**1.25** Plot three cross-sectional maps of the December 2015 potential temperature at 10° S, equator, and 10° N using the same netCDF GODAS data file as the previous problem. Each map is on a $40 \times 360$ grid, with 40 depth levels and 1-deg longitude resolution for the equator.

**1.26** Do the same as the previous problem but for the June 2015 GODAS potential temperature data.

**1.27** Use R or Python to plot a cross-sectional map of the December 2015 potential temperature along a meridional line at longitude at 170° E using the same netCDF GODAS data file as the previous problem.

**1.28** Use R or Python to plot a cross-sectional map of the June 2015 potential temperature along a meridional line at longitude at 170° E using the same netCDF GODAS data file as the previous problem.