IST772 Summary Template: Chapter 11 – Analyzing Change over Time
Originality Assertion: By submitting this file you affirm that this writing is your own.

Name: Hendi Kushta
Date: 04/11/2023

****Important** Copying and/or pasting anything from the textbook will not be acceptable for your chapter notes submissions. You must write your notes in your own words and generate your own code, results, and graphs in R. This is what forces your brain to process the material that you read.**

INTRODUCTION

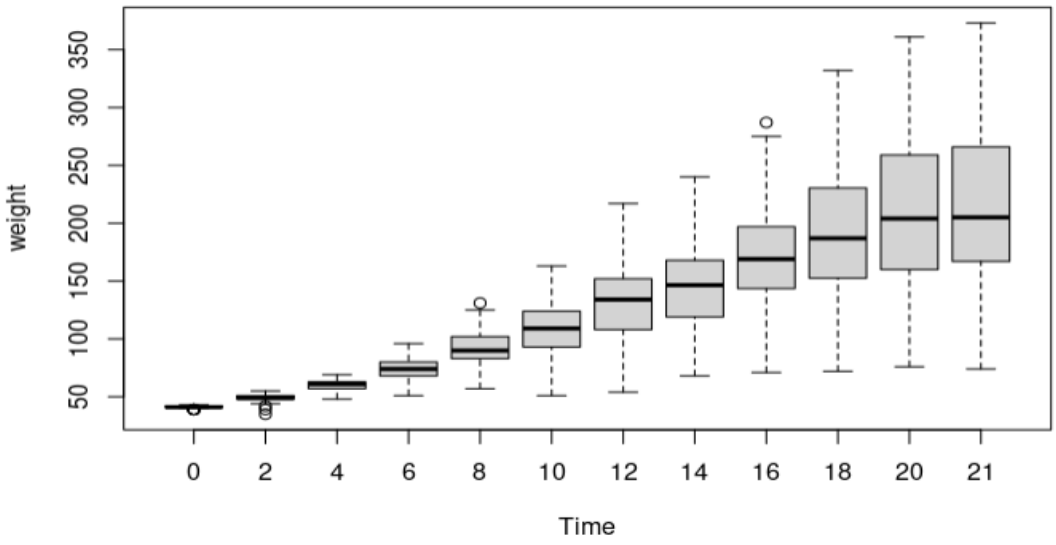
The data sets discussed so far were cross-sectional, meaning that all the data were collected at the same time. Time was not explicitly considered in the measurement of variables or analysis of the data. Even if time was implicit in the measurement process, it was not included in the data sets, and only one measurement was taken for each case. To understand changes over time, it is necessary to measure the same thing at least twice.
The chapter discusses the importance of understanding and analyzing data that are connected to time in various fields such as business, education, government, social sciences, engineering, and natural sciences. It introduces two types of data configurations used in research: repeated measures and time series. Repeated measures involve measuring subjects at two or more points in time, often with an intervention or activity in between. Time series, on the other hand, measure a single phenomenon or a small number of related phenomena over many time intervals, usually with a large number of data points. The chapter emphasizes that observations in time-related data are not independent, meaning they are connected and possibly correlated. This has implications for choosing appropriate analytical techniques and avoiding assumptions of independence that may not hold in these types of data sets. correlation between them.

REPEATED-MEASURES ANALYSIS

In a repeated-measures study, data is collected from the same subjects or cases at two or more points in time. This can be useful for studying changes over time within individuals. An example is measuring the resting heart rate of five individuals (A1, B1, C1, D1, E1) at the gym, then measuring their heart rates again after walking on a treadmill for 5 minutes (A2, B2, C2, D2, E2). These pairs of observations are not independent because they come from the same source of variance, the subjects themselves. The advantage of this design is that it allows for capturing the amount of change between the two measurements for each individual, which can eliminate the influence of individual differences. This is in contrast to treating the two sets of observations as independent groups, which

may result in inflated within-group variance and reduce the ability to detect changes.

```{r}
boxplot(weight ~ Time, data=ChickWeight)
```



The "ChickWeight" data set contains measurements of chick weights at different time points after hatching. A box plot of the weights is generated using the command "boxplot(weight ~ Time, data=ChickWeight)", which shows a pattern of growth over time with variability, particularly after day 6. The dependent measures t-test will be used to compare weights on neighboring days, specifically day 16 and day 18. The code to set up the data for analysis is provided.

```{r}
ch16index <- ChickWeight$Time == 16    # Chicks measured at time 16
ch18index <- ChickWeight$Time == 18    # Chicks measured at time 18
bothChicks <- ChickWeight[ch16index | ch18index,] # Both sets together
# Grab weights for t=16
time16weight <- bothChicks[bothChicks$Time == 16,"weight"]
# Grab weights for t=18
time18weight <- bothChicks[bothChicks$Time == 18,"weight"]
cor(time16weight,time18weight)
```

```
[1] 0.9789155
```

The code generates lists of cases with measurements at time 16 and time 18, and combines them to create a data set for analysis. The correlation between the two vectors of measurements is high (r = 0.97), reflecting the extent and stability of

individual differences among chicks over time. This dependency over time is the reason for using a repeated-measures analysis, as it takes into account the repeated measurements on the same subjects.

```r
library(BEST)
mean(time16weight)
mean(time18weight)
# Independent groups t-test
t.test(time18weight,time16weight,paired = FALSE)
BESTmcmc(time18weight,time16weight)    # Run the Bayesian equivalent
```

```
Loading required package: HDInterval
[1] 168.0851
[1] 190.1915

        Welch Two Sample t-test

data:  time18weight and time16weight
t = 2.0446, df = 88.49, p-value = 0.04386
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
  0.6216958 43.5910701
sample estimates:
mean of x mean of y
 190.1915  168.0851

Waiting for parallel processing to complete...done.
MCMC fit results for BEST analysis:
100002 simulations saved.
         mean     sd median   HDIlo  HDIup Rhat n.eff
mu1    189.86  8.606 189.85 173.172 207.05    1 62395
mu2    167.94  7.036 167.94 154.005 181.60    1 60544
nu      36.69 29.164  28.24   2.843  94.84    1 19625
sigma1  56.67  6.738  56.16  43.754  70.01    1 45781
sigma2  46.10  5.555  45.73  35.776  57.41    1 43248

'HDIlo' and 'HDIup' are the limits of a 95% HDI credible interval.
'Rhat' is the potential scale reduction factor (at convergence, Rhat=1).
'n.eff' is a crude measure of effective sample size.
```

The mean weight at 16 days is 168.1 grams, while the mean weight at 18 days is 190.2 grams, indicating a promising mean difference. To determine if the difference in means is statistically significant, a t-test is conducted with the argument "paired=FALSE" to treat the two groups as independent. The result shows a t-value of 2.05 on 88.49 degrees of freedom, which is significant at $p < .05$. However, the confidence interval, ranging from 0.62 to 43.6, suggests that the mean difference could be very small on the low end. Additionally, the Bayesian t-test shows that the 95% Highest Density Intervals (HDIs) for the two means overlap, indicating that there may not be a credible difference between the two means when each group is considered as an independent entity. To perform the appropriate analysis, the two groups should be treated as dependent, and a dependent groups t-test should be conducted.

```r
t.test(time18weight,time16weight,paired = TRUE)    # Dependent groups t-test
```

```
        Paired t-test

data:  time18weight and time16weight
t = 10.136, df = 46, p-value = 2.646e-13
alternative hypothesis: true mean difference is not equal to 0
95 percent confidence interval:
 17.71618 26.49658
sample estimates:
mean difference
       22.10638
```

The output confirms that the data have been treated as matched pairs in a dependent samples t-test, with the t-value of 10.1 reported on 46 degrees of freedom, $p < .001$. The confidence interval shows a much narrower band with a lower boundary mean difference of 17.7 and an upper boundary mean difference of 26.5. This suggests greater certainty about the result, with the lower bound nowhere near 0, indicating a likely significant difference between day 16 and day 18 measurements.
The stronger results in the dependent samples t-test compared to the incorrect independent samples t-test can be attributed to the high correlation ($r = 0.97$) between the day 16 and day 18 measurements. The dependent samples t-test eliminates individual differences among chicks, leaving only the amount of change for each case. This is demonstrated through the use of difference scores in the analysis, where the traditional and Bayesian analyses are conducted to ascertain if, on average, the difference scores are significantly larger than 0.

```{r}
weightDiffs <- time18weight - time16weight     # Make difference scores
t.test(weightDiffs)    # Run a one sample t-test on difference scores
# Run the Bayesian one-sample test on difference scores
BESTmcmc(weightDiffs)
```

```
        One Sample t-test

data:  weightDiffs
t = 10.136, df = 46, p-value = 2.646e-13
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 17.71618 26.49658
sample estimates:
mean of x
 22.10638

Waiting for parallel processing to complete...done.
MCMC fit results for BEST analysis:
100002 simulations saved.
       mean     sd median  HDIlo  HDIup  Rhat n.eff
mu    21.98  2.275  21.98 17.535  26.48 1.000 60541
nu    40.44 30.728  32.13  2.825 101.74 1.001 24622
sigma 14.97  1.721  14.84 11.728  18.37 1.000 51057

'HDIlo' and 'HDIup' are the limits of a 95% HDI credible interval.
'Rhat' is the potential scale reduction factor (at convergence, Rhat=1).
'n.eff' is a crude measure of effective sample size.
```

The paired t-test is essentially equivalent to a one-sample t-test, where the difference between paired observations is analyzed. In the ChickWeight data example, the paired t-test compares the measurements taken at day 16 and day 18 for the 47 chicks, treating them as one group measured at two different time points.
The output of the Bayesian estimation using the BESTmcmc(weightDiffs) command confirms the results from the paired t-test, showing a similar range for the mean difference between the two time points. However, it's important to note that confidence intervals and HDIs represent different approaches to inference

from sample data, with HDIs being based on posterior distributions obtained through Markov chain Monte Carlo techniques.
One limitation of the paired t-test is that it can only compare two time points at a time, which may not be sufficient for data with multiple time points. In such cases, repeated-measures ANOVA can be used to compare cases across two or more time points. However, repeated-measures ANOVA typically requires a balanced design, where each case has measurements at each time point. If the data is imbalanced, more sophisticated analytical techniques may be needed, but the aov() function works best with a balanced design.
Overall, the paired t-test is a useful tool for analyzing matched pairs of data, but it has limitations when comparing multiple time points or dealing with imbalanced data. Repeated-measures ANOVA may be a better choice in such cases, but it requires a balanced design for optimal results.

BOX ON P.244: USING EZANOVA

The aov() procedure in R is a simple and effective way to analyze data using ANOVA, including repeated-measures designs. However, it requires a balanced data set, meaning that all cases must have data at all time points. If the data set is unbalanced, the output from aov() may be difficult to interpret and could potentially lead to incorrect results without warning.
To address these limitations, alternative packages such as ez have been developed, which provide richer output and more advanced diagnostics to help analysts avoid missteps. The ezANOVA() function, written by Mike Lawrence, is one such alternative that can be used for repeated-measures ANOVA with unbalanced data sets.

```{r}
#install.packages("ez")
library("ez")
chwBal <- ChickWeight                    # Copy the dataset
chwBal$TimeFact <- as.factor(chwBal$Time)    # Convert Time to a factor
# Make a list of rows
list <- rowSums(table(chwBal$Chick,chwBal$TimeFact))==12
list <- list[list==TRUE]                # Keep only those with 12 observations
list <- as.numeric(names(list))         # Extract the row indices
chwBal <- chwBal[chwBal$Chick %in% list,]    # Match against the data
ezANOVA(data=chwBal,dv=.(weight),within=.(TimeFact),wid=.(Chick),
detailed=TRUE)
```

	Effect	DFn	DFd	SSn	SSd	F	p	p<.05	ges
	<chr>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<chr>	<dbl>
1	(Intercept)	1	44	8431251	429898.6	862.9362	1.504306e-30	*	0.9126857
2	TimeFact	11	484	1982388	376697.6	231.5519	7.554752e-185	*	0.7107921

IST772 Summary Template: Chapter 11 – Analyzing Change over Time
Originality Assertion: By submitting this file you affirm that this writing is your own.

	Effect <chr>	W <dbl>	p <dbl>	p<.05 <chr>
2	TimeFact	1.496988e-17	2.370272e-280	*

1 row

	Effect <chr>	GGe <dbl>	p[GG] <dbl>	p[GG]<.05 <chr>	HFe <dbl>	p[HF] <dbl>	p[HF]<.05 <chr>
2	TimeFact	0.1110457	7.816387e-23	*	0.1125621	4.12225e-23	*

1 row

The output from ezANOVA() includes the following information:

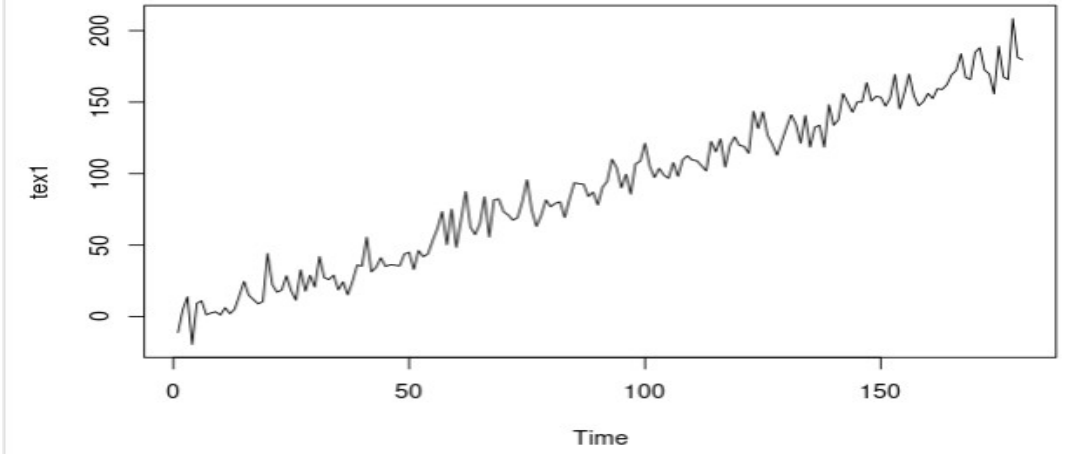
$ANOVA section: The calculated F-test statistic (F(11,484) = 231.6) indicates statistical significance (p < .001), allowing rejection of the null hypothesis of no change in weight over time. The generalized eta-squared value is also provided (0.71).
$Mauchly's Test for Sphericity section: If this test is significant, it indicates that the assumption of homogeneity of variance has been violated among pairs of time groups.
Additional tests section: Greenhouse-Geisser correction (p[GG]) and Huynh-Feldt correction (p[HF]) are provided, which apply adjustments to the degrees of freedom to counteract possible inflation of the F-ratio. If the associated p-values remain significant, it supports the decision to reject the null hypothesis.
Warning/error messages: ezANOVA() checks for proper data preparation and may indicate errors if an unbalanced data set is submitted for repeated measures analysis.
Additional functions: The ez package offers additional functions for plotting and diagnostics, and online resources and tutorials can be found using the search string "ez package R" to avoid confusion with other software programs called ezANOVA.

TIME-SERIES ANALYSIS

Time-series analysis is an alternative approach to repeated-measures designs that involves capturing and analyzing many measurements of one subject or phenomenon over time. This allows researchers to detect events or anomalies that may happen to the subject and examine trends and cycles in the data. However, trends and cycles in time-series data can interfere with the examination of other relationships in the data. Time-series analysis can be implemented using R code and line graphs to visualize the data.

IST772 Summary Template: Chapter 11 – Analyzing Change over Time
Originality Assertion: By submitting this file you affirm that this writing is your own.

```r
set.seed(1234)                    # Control random numbers
tslen <- 180                      # About half a year of daily points
ex1 <- rnorm(n=tslen,mean=0,sd=10)       # Make a random variable
tex1 <- ex1 + seq(from=1, to=tslen, by=1) # Add the fake upward trend
plot.ts(tex1)        # Plot the time series with a connected line
```



The code provided generates a fake time series data with a growth trend by creating 180 random data points with a standard deviation of 10. The data points are plotted using the plot.ts() command, which connects the dots with a line to represent the time sequence. The plot shows a general trend of increasing magnitude over time, with daily fluctuations. The code also generates a second random variable for illustrating an analytical challenge of correlating variables.

```r
ex2 <- rnorm(n=tslen,mean=0,sd=10)         # Make another random variable
tex2 <- ex2 + seq(from=1, to=tslen, by=1)    # Add the fake upward trend
cor(ex1, ex2)      # Correlation between the two random variables
cor(tex1, tex2)    # Correlation between the two time series
```

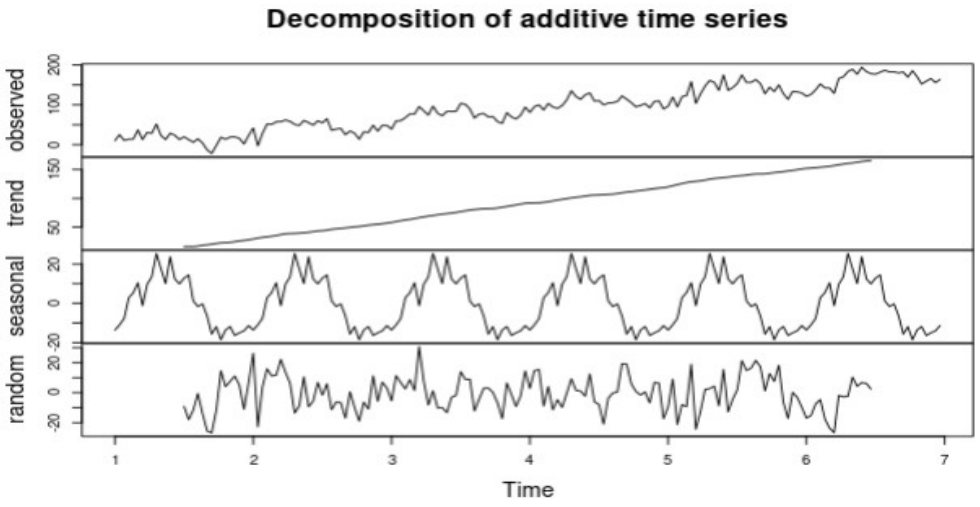
```
[1] -0.09385519
[1] 0.9634188
```

The code generates two sets of random variables, "tex1" and "tex2", with a normal distribution and a standard deviation of 10. Both sets of variables are then transformed into fake time series data by adding an upward trend. When correlating the original random normal variables, the correlation is low as expected. However, when correlating the fake time series variables, the correlation is significantly high due to the shared upward trend. This demonstrates the importance of considering trends in time series data when analyzing correlations, as trends can create artifacts and lead to incorrect conclusions about the relationship between variables. It is necessary to remove trends from time series data to avoid misinterpretation of correlation results.

IST772 Summary Template: Chapter 11 – Analyzing Change over Time
Originality Assertion: By submitting this file you affirm that this writing is your own.

Time-series analysis involves decomposing a time series data into its components, including trend, seasonality, cyclicality, and irregularity, in order to analyze them separately. Trend refers to the growth or decline of a time series over time, while seasonality represents regular fluctuations that occur in a specific time period, such as seasons of the year. Cyclicality refers to repeating fluctuations that do not have a regular time period, such as economic recessions. The irregular component, sometimes referred to as noise, is the unpredictable part of a time series. Time-series analysis involves analyzing and interpreting these components individually to gain insights from the data.
A random normal variable is created and a trend is added to it using a sequence of increasing integers. Additionally, a seasonal component in the form of a sine wave is added to the data. The resulting time-series plot shows the growth trend, irregular ups and downs, and six discernible cycles in the data. To decompose the time series into its components, R has a built-in function that can be used for this purpose.

```r
decOut <- decompose(ts(tex3,frequency=30))
plot(decOut)
```



The decompose() function in R is used to analyze time-series data, and it requires a time-series object as input. The ts() function is used to convert a vector of values into a time-series object, and the "frequency" parameter in ts() specifies the natural time scale on which seasonal variation may occur. The decomposition plot produced by plot(decOut) shows four panes stacked on top of each other: the original time-series line chart, the trend component, the seasonal component, and the irregular or noise component. It is important to correctly specify the "frequency" parameter to accurately detect seasonal fluctuations. The diagnostics from the resulting decomposition data object can provide further insights into the components of the time series.

IST772 Summary Template: Chapter 11 – Analyzing Change over Time
Originality Assertion: By submitting this file you affirm that this writing is your own.

```r
mean(decOut$trend, na.rm="TRUE")
mean(decOut$seasonal)
mean(decOut$random,na.rm="TRUE")
cor(ex3, decOut$random, use="complete.obs")
```

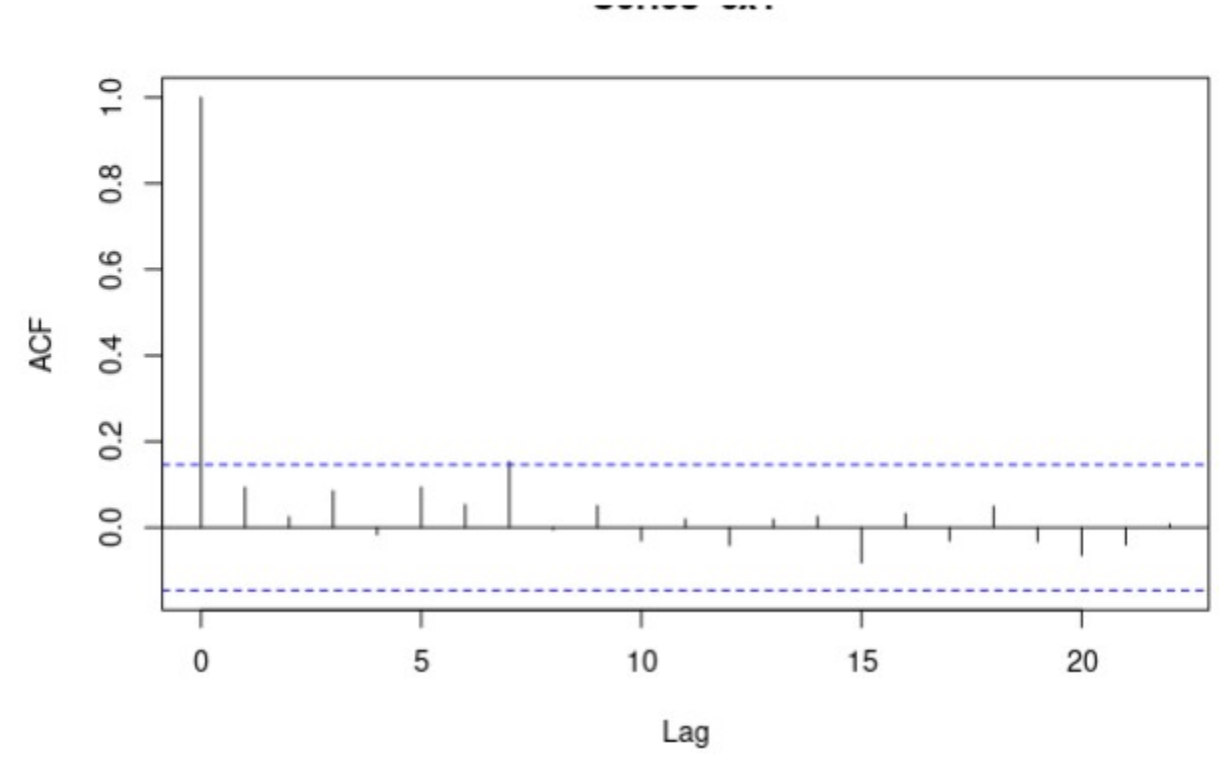
```
[1] NA
[1] 1.262782e-16
[1] NA
[1] 0.8304297
```

The passage discusses the decomposition of time series data using the decompose() function in R. It explains how the mean of the trend component reflects the pattern of the sequence used to create it, the seasonal component is a sine wave with artifacts around 0, and the irregular component accessed with $random is close to 0 with a slight negative mean. The passage also mentions that the missing values in the trend and random components are ignored during the calculation. The correlation between the extracted irregular component and the original random normal variable used to create the time series is calculated, showing a reasonably large correlation. The passage highlights that the decomposed data is likely to be stationary, but further diagnostics such as autocorrelation function (ACF) are needed to make better judgments about the properties of the time series data. The concept of ACF is briefly explained, where correlations between a variable and its lagged values are used to identify patterns in time series data with trends or seasonal variations.

The passage mentions that the first time series variable created earlier can be recreated using the set.seed() function to obtain the exact same results, providing a reminder of how it was originally done.

IST772 Summary Template: Chapter 11 – Analyzing Change over Time
Originality Assertion: By submitting this file you affirm that this writing is your own.

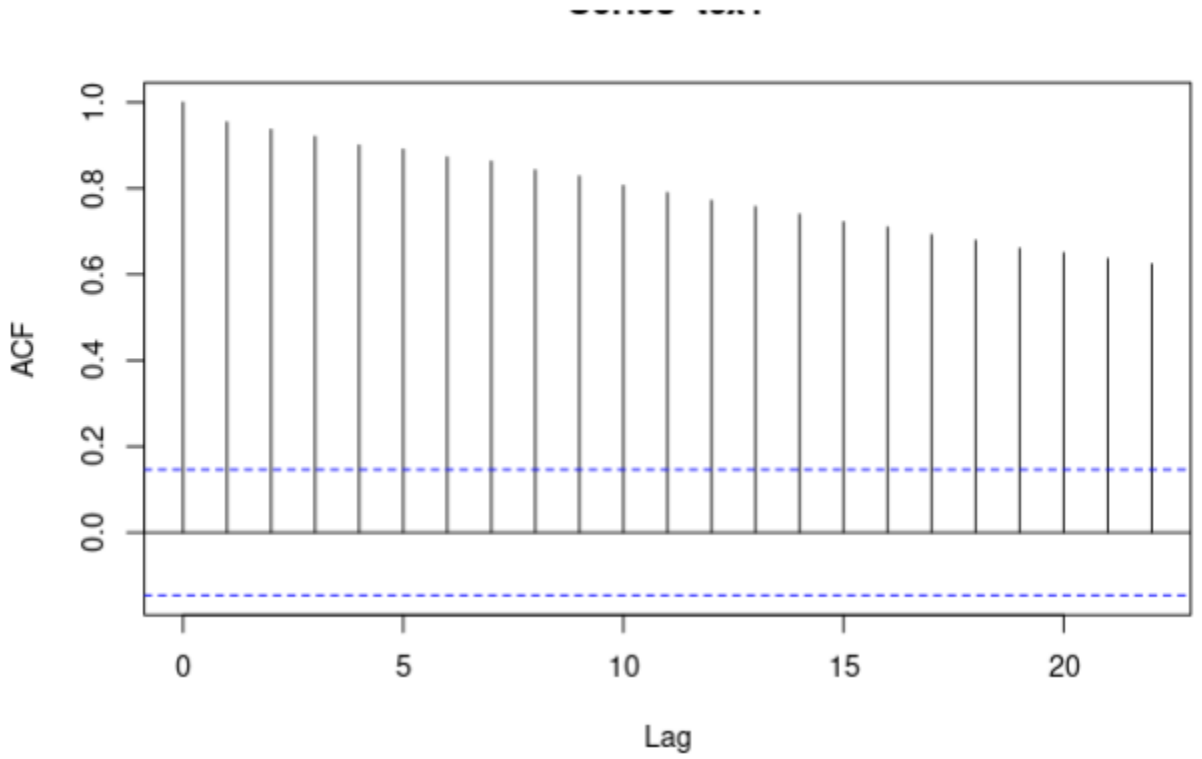
```r
set.seed(1234)
tslen <- 180
ex1 <- rnorm(n=tslen,mean=0,sd=10)      # Make a random variable
acf(ex1)
```



The autocorrelation plot obtained from the command "acf(ex1)" which shows a completely stationary process with no discernible pattern in the variation of correlations at different lags, except for the first lag which is perfectly correlated with itself. The passage also mentions the threshold of statistical significance for positive and negative correlations shown by horizontal dotted lines. The passage then introduces the addition of a trend to the time series data for further analysis.

IST772 Summary Template: Chapter 11 – Analyzing Change over Time
Originality Assertion: By submitting this file you affirm that this writing is your own.

```r
tex1 <- ex1 + seq(from=1, to=tslen, by=1)    # Add the fake upward trend
acf(tex1)
```



EXPLORING A TIME-SERIES WITH REAL DATA

The passage discusses the process of decomposing a time series by removing trend and seasonal components, along with a diagnostic strategy for analyzing the results. It then introduces the use of real data from a built-in data set called EuStockMarkets, which records the daily closing prices of four different European stock markets between 1991 and 1998. The passage suggests using R functions such as head(), tail(), and View() to review and analyze the data, which is stored in a multivariate time series (mts) structure in R.

```r
str(EuStockMarkets)
```

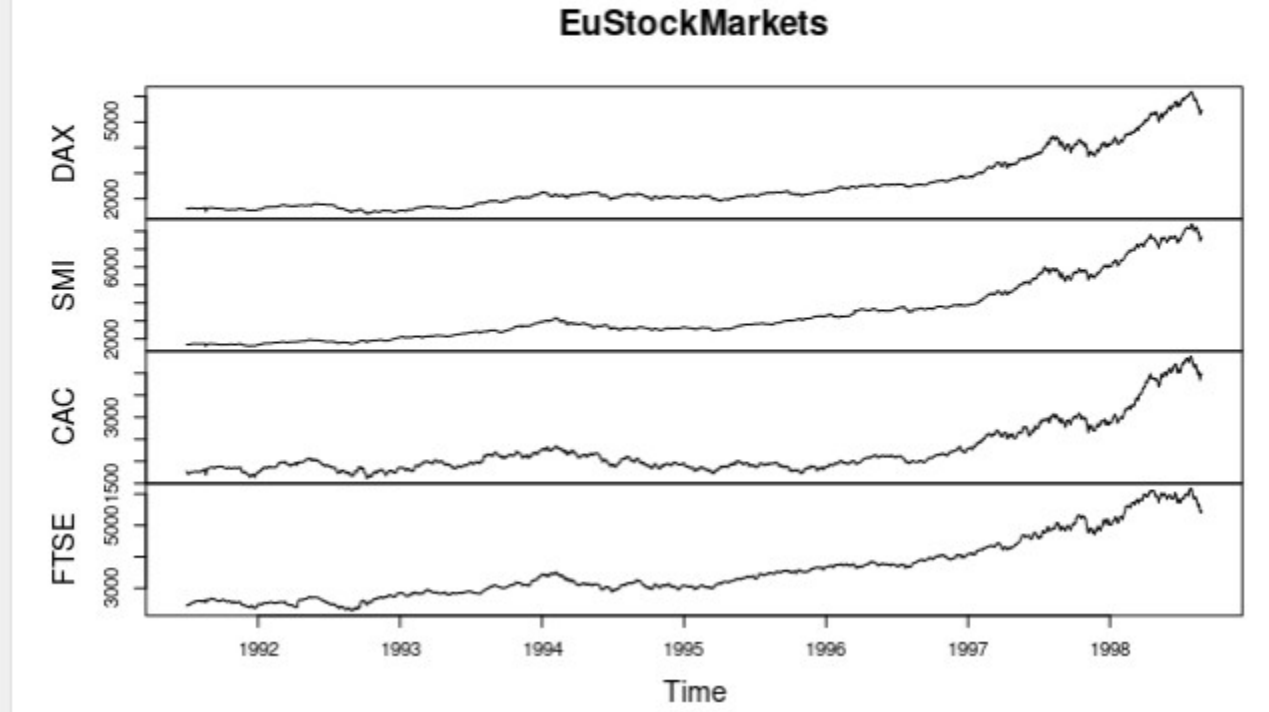
```
Time-Series [1:1860, 1:4] from 1991 to 1999: 1629 1614 1607 1621 1618 ...
- attr(*, "dimnames")=List of 2
 ..$ : NULL
 ..$ : chr [1:4] "DAX" "SMI" "CAC" "FTSE"
```

The EuStockMarkets data set, which contains daily closing prices of four different

IST772 Summary Template: Chapter 11 – Analyzing Change over Time
Originality Assertion: By submitting this file you affirm that this writing is your own.

European stock markets (DAX, SMI, CAC, and FTSE) between 1991 and 1998. It mentions that the data has 1,860 observations per vector and the "tsp" attribute indicates the time span of the data. The passage poses a research question about finding the two stock markets that are least correlated with each other to diversify holdings in index funds. It suggests using the plot() function to create a four-paned plot to examine the stock markets together after removing trend and seasonality.

```{r}
plot(EuStockMarkets)
```



The presence of an upward trend in the time series plots of the EuStockMarkets data set, which may cause spurious correlations among the variables. The author suggests using the technique of differencing, which involves subtracting the second element from the first element of a time series to remove the trend and flatten out any trends that occur over time. This technique is simple and effective for removing trends in time-series analysis.

IST772 Summary Template: Chapter 11 – Analyzing Change over Time
Originality Assertion: By submitting this file you affirm that this writing is your own.

The passage demonstrates the use of the diff() command in R to calculate the differences between neighboring pairs of elements in a sequence, which can be used for differencing in time series analysis. The resulting stationary series with no variance is shown using the example of the seq_len(10) sequence. The diff() command is then applied to the EuStockMarkets data set to remove the trend component. The resulting differenced series is tested for stationarity using the adf.test() procedure, starting with the DAX variable.

```{r}
library(tseries)
adf.test(diff(EuStockMarkets[,"DAX"]))
```

```
Warning: p-value smaller than printed p-value
        Augmented Dickey-Fuller Test

data:  diff(EuStockMarkets[, "DAX"])
Dickey-Fuller = -9.9997, Lag order = 12, p-value = 0.01
alternative hypothesis: stationary
```

The augmented Dickey-Fuller test is used to test the null hypothesis of nonstationarity in a time series. If the test is significant, indicating rejection of the null hypothesis, it provides evidence for stationarity. This can be further confirmed by examining the autocorrelation function (ACF) plot, which should have very few significant autocorrelations after differencing. In this case, the time series of the stock markets examined showed evidence of stationarity after differencing. However, differencing generally cannot remove seasonality from a time series. In this case, there was no evidence of seasonality affecting the tests of stationarity. To determine the weak relationship between two stock markets, a bivariate correlation matrix using the differenced time series can be used, for example by applying the cor() function.

```{r}
cor(diff(EuStockMarkets))
```

```
           DAX       SMI       CAC      FTSE
DAX  1.0000000 0.7468422 0.7449335 0.6769647
SMI  0.7468422 1.0000000 0.6414284 0.6169238
CAC  0.7449335 0.6414284 1.0000000 0.6707475
FTSE 0.6769647 0.6169238 0.6707475 1.0000000
```

The correlation matrix indicates that SMI and FTSE are weakly related, with a correlation coefficient of approximately r = 0.62. This translates to about 38% of shared variance between the two indices when squared, which is quite significant. However, this conclusion differs from the original correlation matrix of the

"undifferenced" time series, highlighting the importance of removing trend and cyclical components from a time series before conducting substantive analysis. This suggests that considering differenced time series can provide different insights and may be necessary for accurate analysis.

FINDING CHANGE POINTS IN TIME SERIES

Once the skills of testing for trends, differencing to remove trends and seasonality, and handling cyclical components in time series data are mastered, researchers can begin to ask substantive research questions. One method for exploring such questions is change-point analysis, where algorithms search for major transitions in the time series data, typically in the mean level of the data. Change-point analysis allows researchers to identify the point in time when a transition occurred and quantify the change in the mean level of the time series. For example, using the EuStockMarkets data in R, which represents European stock markets, researchers can use the changepoint package to explore if there was a specific inflection point when the average value of stocks substantially increased, beyond the gradual growth observed from 1991 to 1999.

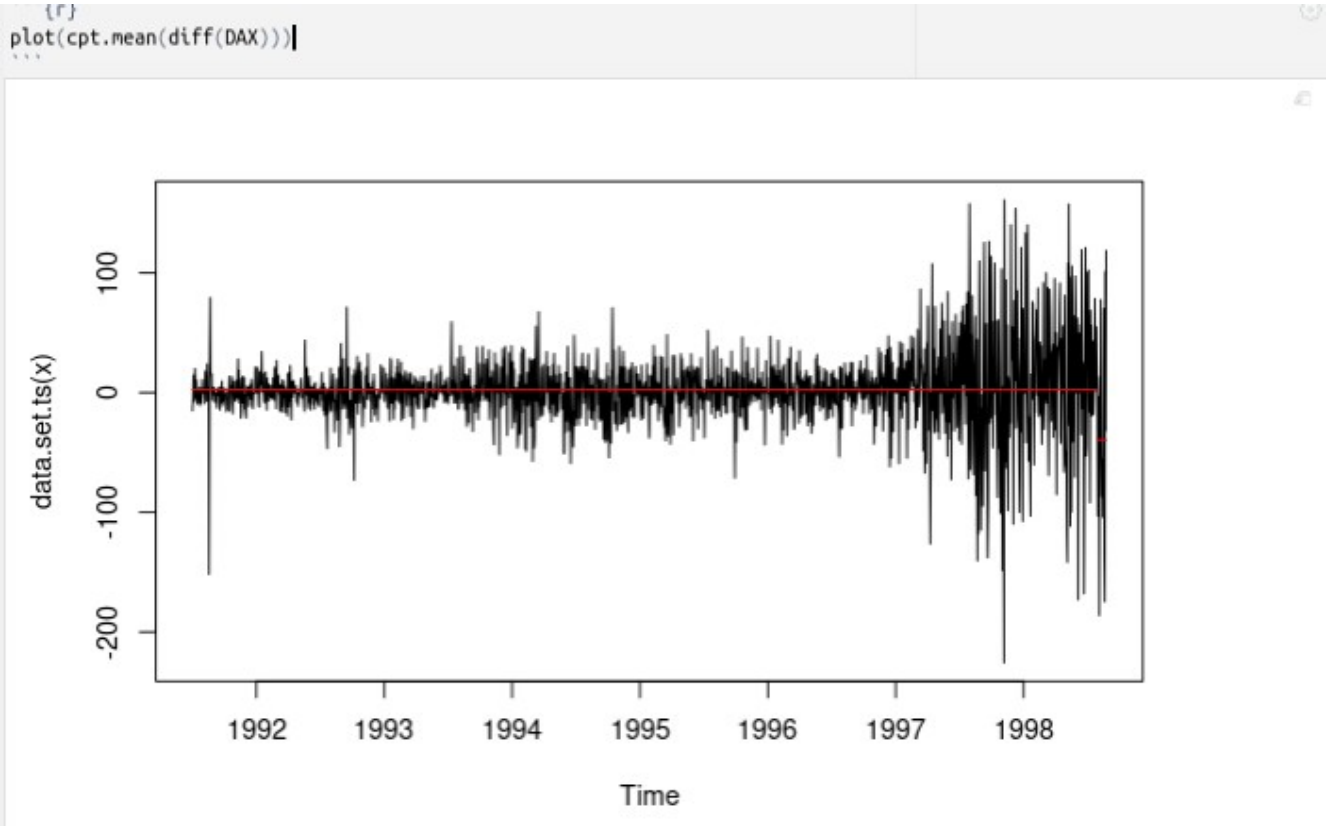
```{r}
#install.packages("changepoint")
library(changepoint)
DAX <- EuStockMarkets[,"DAX"]
DAXcp <- cpt.mean(DAX)
DAXcp
```

```
Class 'cpt' : Changepoint Object
       ~~   : S4 class containing 12 slots with names
              cpttype date version data.set method test.stat pen.type pen.value minseglen cpts ncpts.max param.est

Created on  : Tue Apr 11 18:56:17 2023

summary(.)  :
----------
Created Using changepoint version 2.2.4
Changepoint type      : Change in mean
Method of analysis    : AMOC
Test Statistic  : Normal
Type of penalty       : MBIC with value, 22.585
Minimum Segment Length : 1
Maximum no. of cpts   : 1
Changepoint Locations : 1467
```

IST772 Summary Template: Chapter 11 – Analyzing Change over Time
Originality Assertion: By submitting this file you affirm that this writing is your own.

The cpt.mean() function is used to detect transition points where the mean of a time series changes substantially. The output of the function includes information about the method of analysis, such as the algorithm used (AMOC) and the maximum number of change points allowed (in this case, 1). The type of penalty used in the analysis determines how sensitive the algorithm is to detecting changes, with higher penalty values resulting in fewer detected change points. The cpt.mean() function detects a change in the mean of the time series at a specific point in time, which can be visualized using a plot. The change point analysis can reveal mean shifts that may not be evident to the eye in data sets with more modest changes. The cpt.mean() function can also be used to detect multiple change points in a time series, allowing for the analysis of interventions or experiments. It is important to note that differencing or removing trends from the data before conducting the analysis may impact the detection of change points. A similar analytical procedure, cpt.var(), can be used to detect changes in the variability of a time series over time.

```
{r}
plot(cpt.mean(diff(DAX)))
```



A significant change point was detected at point 1480, which corresponds to a period partway through 1997, as shown in Figure 11.12. This suggests that during the rapid increase of the market in 1997, there was also an increase in volatility, characterized by substantially greater variance. It is recommended to apply the

cpt.var() command on other markets in the EuStockMarkets data and plot the results to obtain a change-point graph. It is also mentioned that the analysis was performed on the first order difference scores instead of the raw time-series data to avoid potential influence from trends on the change-point variance analysis.

PROBABILITIES IN CHANGE-POINT ANALYSIS

The cpt.mean() procedure does not conduct a statistical significance test, but it generates a confidence "level" expressed as a probability, which indicates the strength of belief about the detected change in mean of the time series. This confidence level ranges from 0 to 1, with values closer to 1 indicating a stronger effect and greater certainty that the change in mean is not due to chance. To obtain a simpler output from cpt.mean(), the class can be set to FALSE using a command.

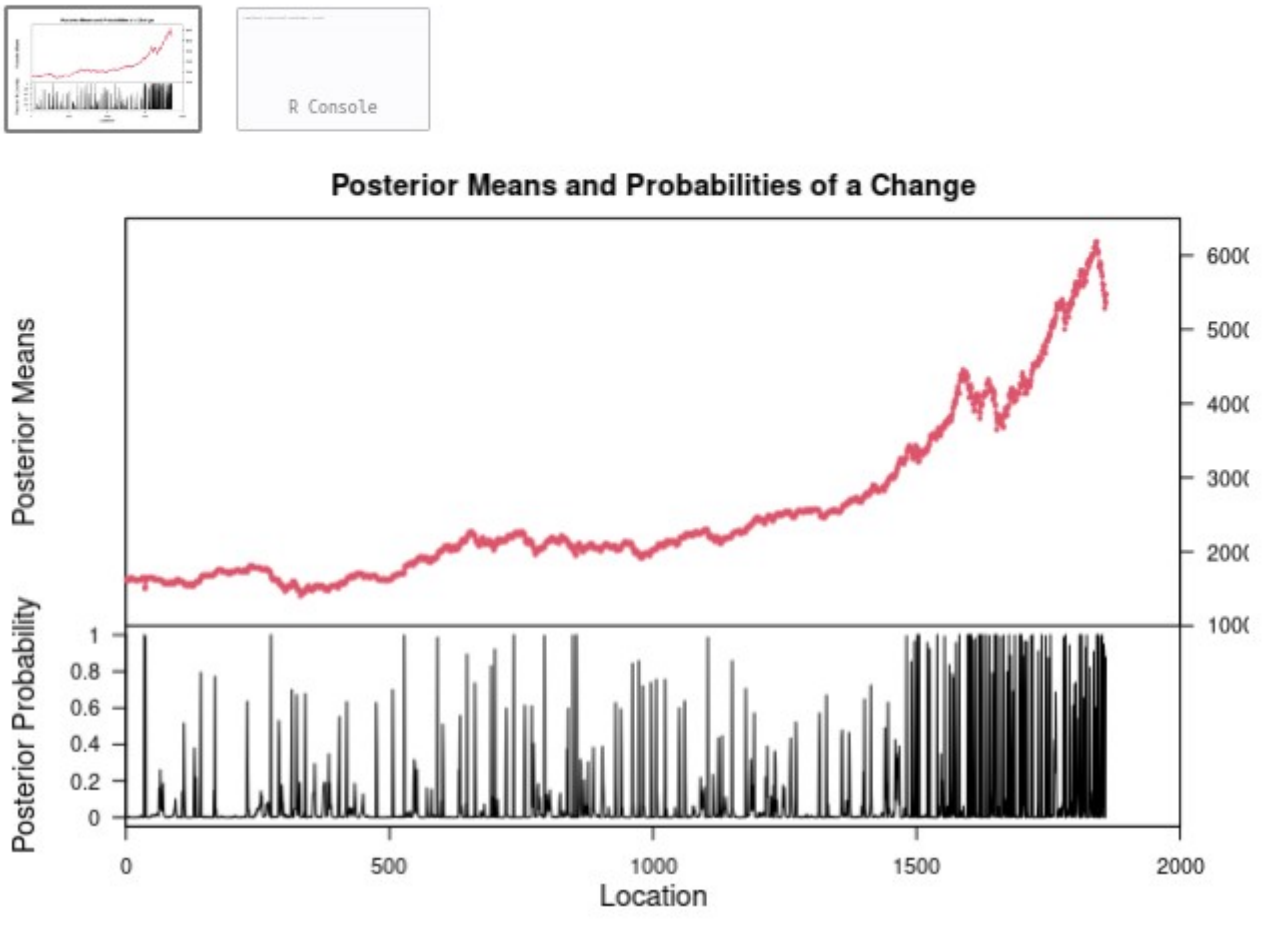
```{r}
DAXcp <- cpt.mean(DAX,class=FALSE)
DAXcp['conf.value"]
```

```
conf.value
         1
```

The confidence value obtained from cpt.mean() is 1.0, which signifies the strongest possible confidence in the detected change in the mean of the time series data.
Bayesian analysis, using the bcp package, provides a more detailed output with probabilities of mean changes at each point in time.
The bcp package requires converting the time series data to a vector using as.vector() before passing it to the bcp() function.
Plotting the output of bcp() creates a data display with two panes, one showing the original time series and the other showing the probabilities of mean changes at each point in time.
The probabilities of mean changes show isolated spikes, but near data point 1500 (beginning of the year 1997), there is a substantial density of probability values near 1.
Further analysis and visualization can provide a better understanding of the detected mean changes in the time series data.

IST772 Summary Template: Chapter 11 – Analyzing Change over Time
Originality Assertion: By submitting this file you affirm that this writing is your own.

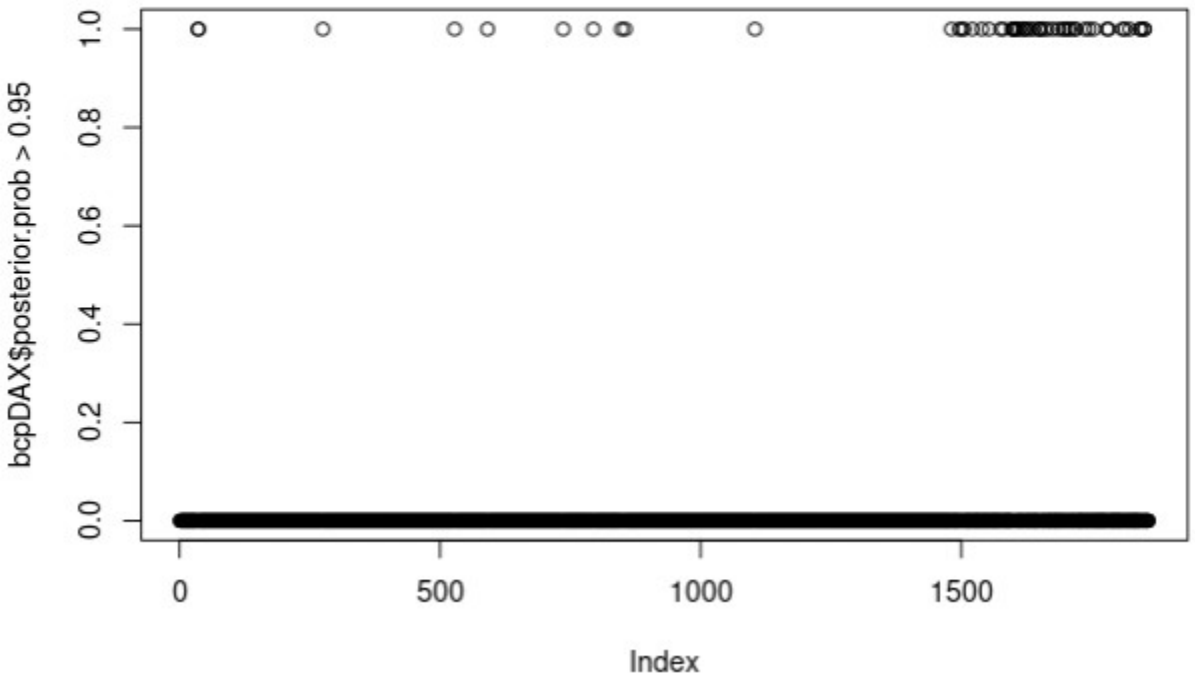
```r
#install.packages("bcp")
library(bcp)
bcpDAX <- bcp(as.vector(DAX))
plot(bcpDAX)
```



The resulting data display has two panes, showing the original time series data and probabilities of mean changes.
There are isolated spikes of probabilities near 1 at various points in the timeline.
Near data point 1500 (beginning of 1997), there is a substantial density of probabilities near 1.
Replotting probabilities using a programming trick can provide further insight into detected mean changes.

IST772 Summary Template: Chapter 11 – Analyzing Change over Time
Originality Assertion: By submitting this file you affirm that this writing is your own.

```{r}
plot(bcpDAX$posterior.prob >.95)
```



A logical test is used to separate low and medium probabilities from high probabilities.
Probabilities below or equal to 0.95 are recoded as FALSE (0) while probabilities above 0.95 are recoded as TRUE (1).
The resulting data display (Figure 11.14) shows multiple points with high probabilities of being change points, particularly around 500, 1000, and 1500.
These change points correspond to periods just before 1994 and just after 1996, indicating modest rises in the DAX index.
Most importantly, there is a sustained rise in the mean of the time series in early 1997, confirming the detection of a major shift in the mean of the time series using cpt.mean().
The density of probability estimates in this time region further supports the significant change in the mean of the time series.

IST772 Summary Template: Chapter 11 – Analyzing Change over Time
Originality Assertion: By submitting this file you affirm that this writing is your own.

BOX ON P.268: QUICK VIEW OF ARIMA

ARIMA (Auto-Regressive, Integrated, Moving Average) is a flexible method for analyzing time series data used for modeling and forecasting.
ARIMA consists of autoregressive (AR), integrated (I), and moving average (MA) components, which capture past history, growth/decline, and prediction errors, respectively.
ARIMA models are designated as arima(p,d,q), where p is the order of the autoregressive component, d is the order of the integrated component, and q is the order of the moving average component.
Identifying the appropriate values for p, d, and q requires careful work with diagnostic statistical procedures.
Once the values for p, d, and q are determined, ARIMA can be used to construct a prediction model for accurate forecasting, assuming the underlying phenomenon continues in a similar fashion.
Developing an appropriate ARIMA model may involve removing or modeling seasonality and achieving stationarity in the time series.
R contains an ARIMA procedure that can be used to model and predict time series data, such as water levels in the LakeHuron data set.

```{r}
# Run a model with p=1, d=0, and q=1; hold out the last ten values
tsFit <- arima(LakeHuron[1:88], order=c(1,0,1)) # Fit the model
predict(tsFit, n.ahead=10)
# Predict the next ten values
LakeHuron[89:98]
# Compare with the actual values
```

```
$pred
Time Series:
Start = 89
End = 98
Frequency = 1
 [1] 578.2125 578.4516 578.6239 578.7481 578.8376 578.9020 578.9485 578.9820 579.0061 579.0235

$se
Time Series:
Start = 89
End = 98
Frequency = 1
 [1] 0.6830779 1.0090854 1.1422332 1.2055827 1.2371976 1.2532988 1.2615781 1.2658558 1.2680712 1.2692200

 [1] 576.89 575.96 576.80 577.68 578.38 578.52 579.74 579.31 579.89 579.96
```

The code provided uses the LakeHuron data set to develop an ARIMA(1,0,1) model with the first 88 observations.
The predict() function is used to generate predictions for the next ten values based on the ARIMA model, including standard errors for each prediction.

IST772 Summary Template: Chapter 11 – Analyzing Change over Time
Originality Assertion: By submitting this file you affirm that this writing is your own.

The final 10 observations of the LakeHuron data set, which were held back from the analysis process, are shown.
ARIMA is not an inferential technique, but Bayesian thinking can be applied to provide greater flexibility and a richer view of uncertainty around future predictions.
To learn more about ARIMA and related diagnostic procedures, searching for "ARIMA tutorial R" can provide good introductory explanations.