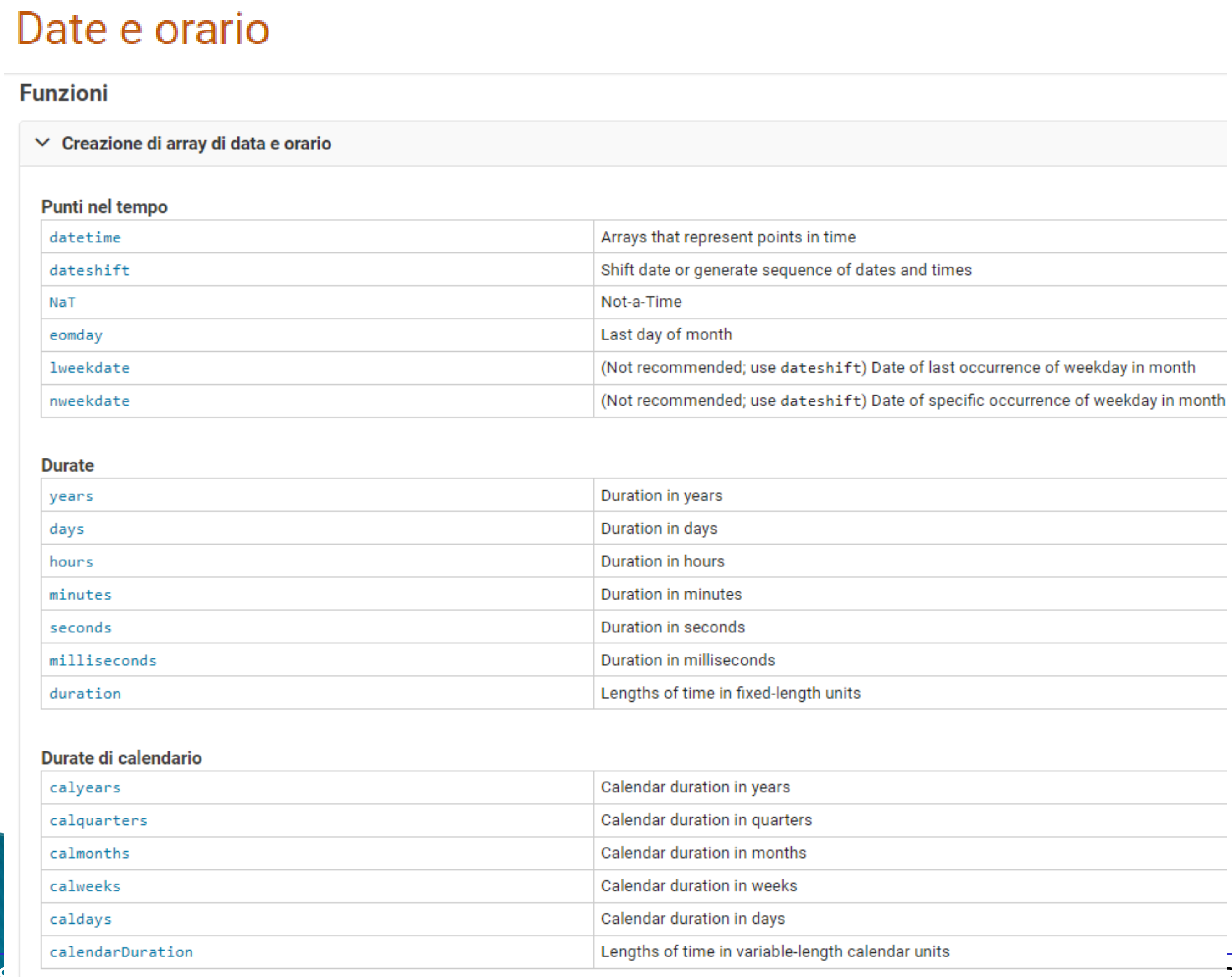
ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Funzioni

Creazione di array di data e orario

Punti nel tempo

`datetime`	Arrays that represent points in time
`dateshift`	Shift date or generate sequence of dates and times
`NaT`	Not-a-Time
`eomday`	Last day of month
`lweekdate`	(Not recommended; use `dateshift`) Date of last occurrence of weekday in month
`nweekdate`	(Not recommended; use `dateshift`) Date of specific occurrence of weekday in month

Durate

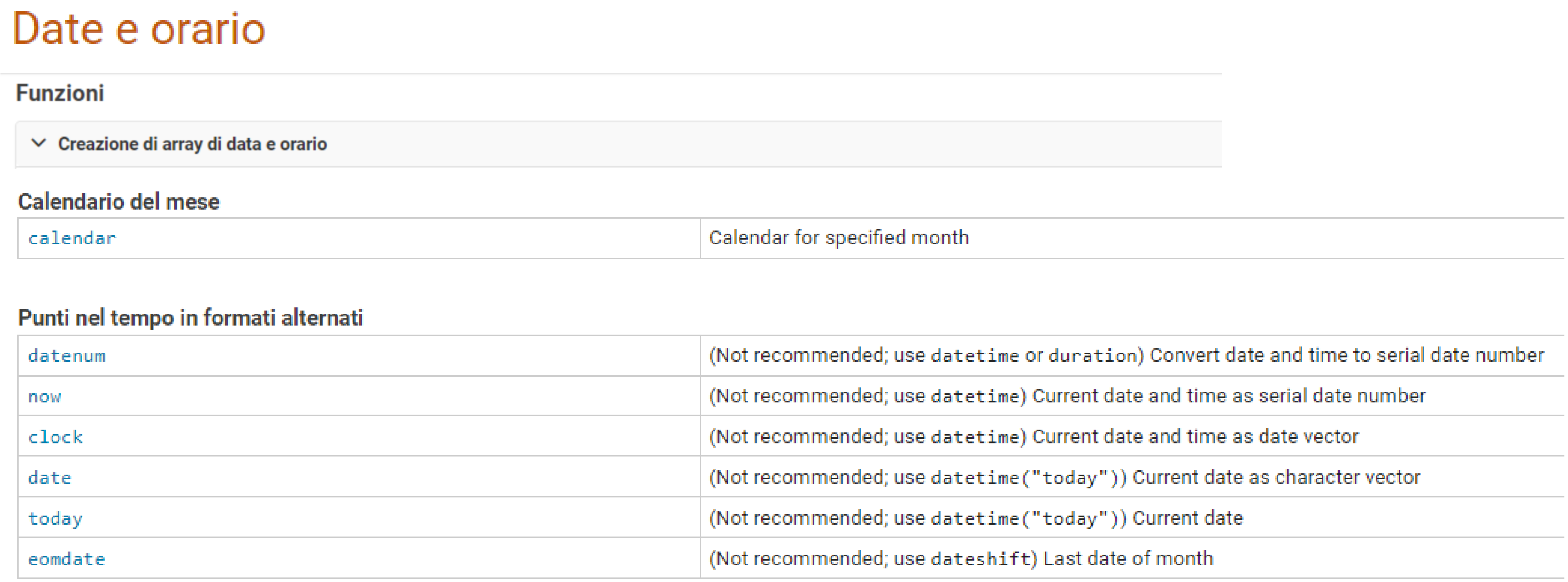
`years`	Duration in years
`days`	Duration in days
`hours`	Duration in hours
`minutes`	Duration in minutes
`seconds`	Duration in seconds
`milliseconds`	Duration in milliseconds
`duration`	Lengths of time in fixed-length units

Durate di calendario

`calyears`	Calendar duration in years
`calquarters`	Calendar duration in quarters
`calmonths`	Calendar duration in months
`calweeks`	Calendar duration in weeks
`caldays`	Calendar duration in days
`calendarDuration`	Lengths of time in variable-length calendar units

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Funzioni

˅ Creazione di array di data e orario

Calendario del mese

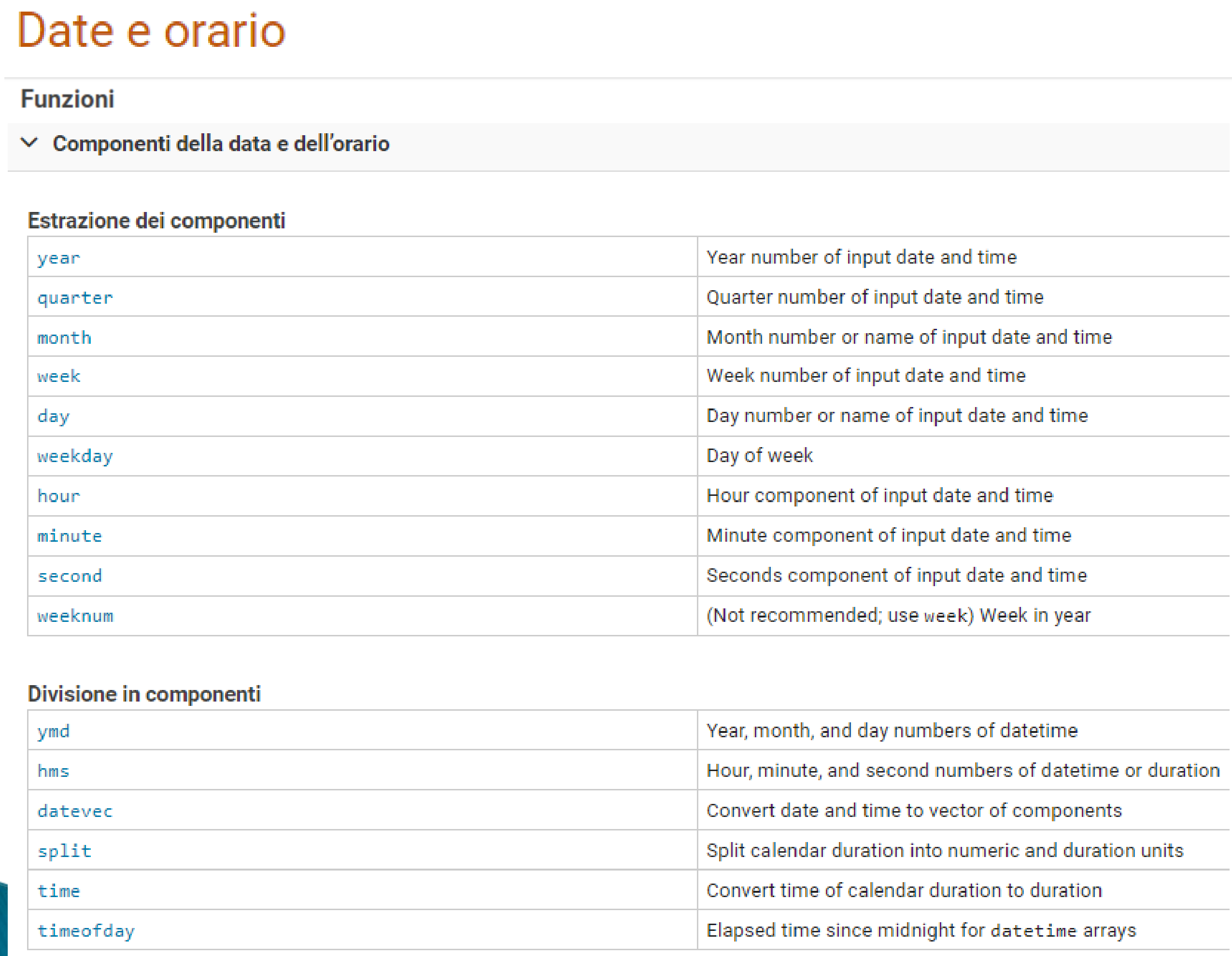
`calendar`	Calendar for specified month

Punti nel tempo in formati alternati

`datenum`	(Not recommended; use `datetime` or `duration`) Convert date and time to serial date number
`now`	(Not recommended; use `datetime`) Current date and time as serial date number
`clock`	(Not recommended; use `datetime`) Current date and time as date vector
`date`	(Not recommended; use `datetime("today")`) Current date as character vector
`today`	(Not recommended; use `datetime("today")`) Current date
`eomdate`	(Not recommended; use `dateshift`) Last date of month

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Funzioni

Componenti della data e dell'orario

Estrazione dei componenti

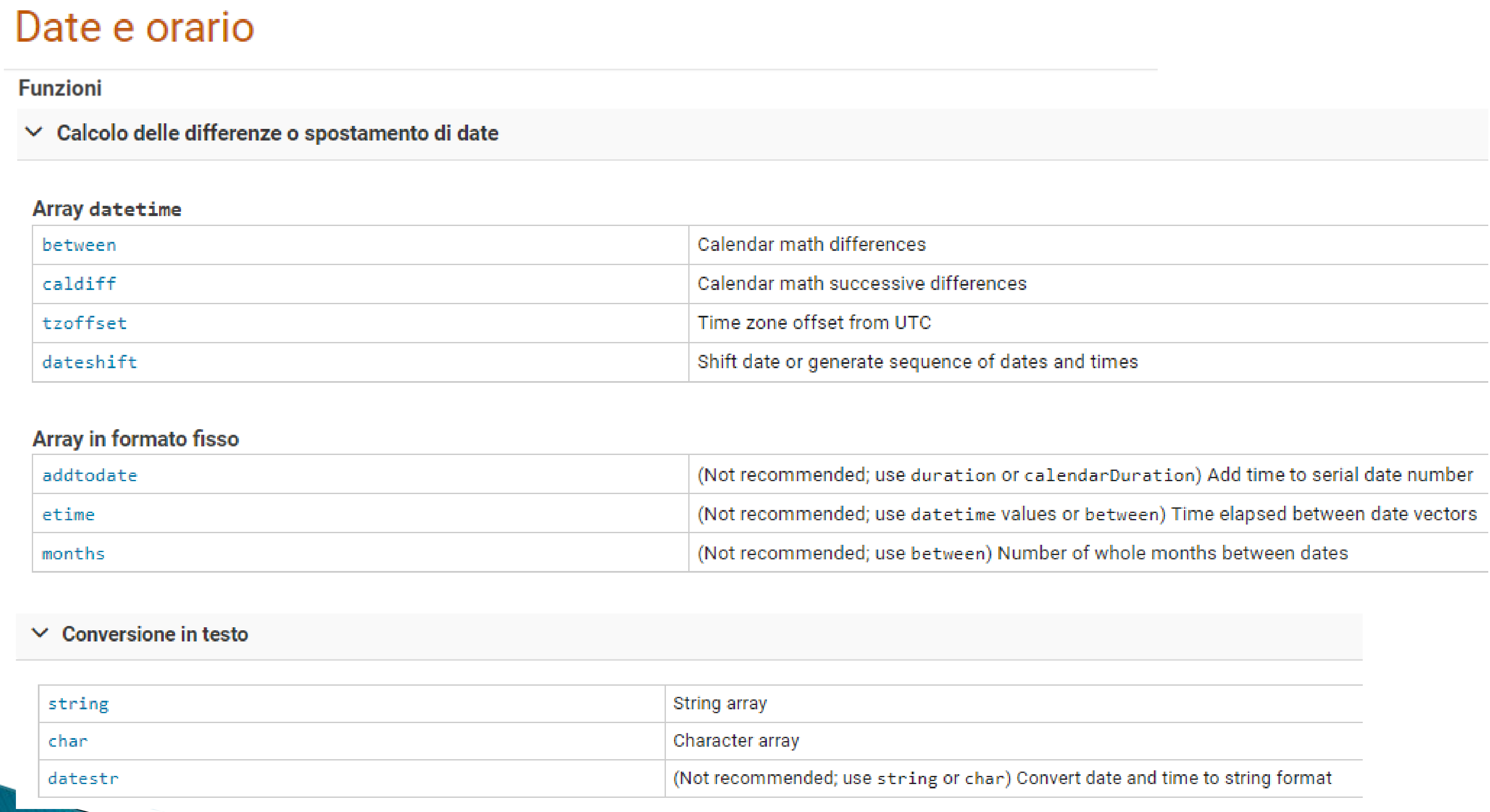
`year`	Year number of input date and time
`quarter`	Quarter number of input date and time
`month`	Month number or name of input date and time
`week`	Week number of input date and time
`day`	Day number or name of input date and time
`weekday`	Day of week
`hour`	Hour component of input date and time
`minute`	Minute component of input date and time
`second`	Seconds component of input date and time
`weeknum`	(Not recommended; use `week`) Week in year

Divisione in componenti

`ymd`	Year, month, and day numbers of datetime
`hms`	Hour, minute, and second numbers of datetime or duration
`datevec`	Convert date and time to vector of components
`split`	Split calendar duration into numeric and duration units
`time`	Convert time of calendar duration to duration
`timeofday`	Elapsed time since midnight for `datetime` arrays

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Funzioni

Calcolo delle differenze o spostamento di date

Array `datetime`

`between`	Calendar math differences
`caldiff`	Calendar math successive differences
`tzoffset`	Time zone offset from UTC
`dateshift`	Shift date or generate sequence of dates and times

Array in formato fisso

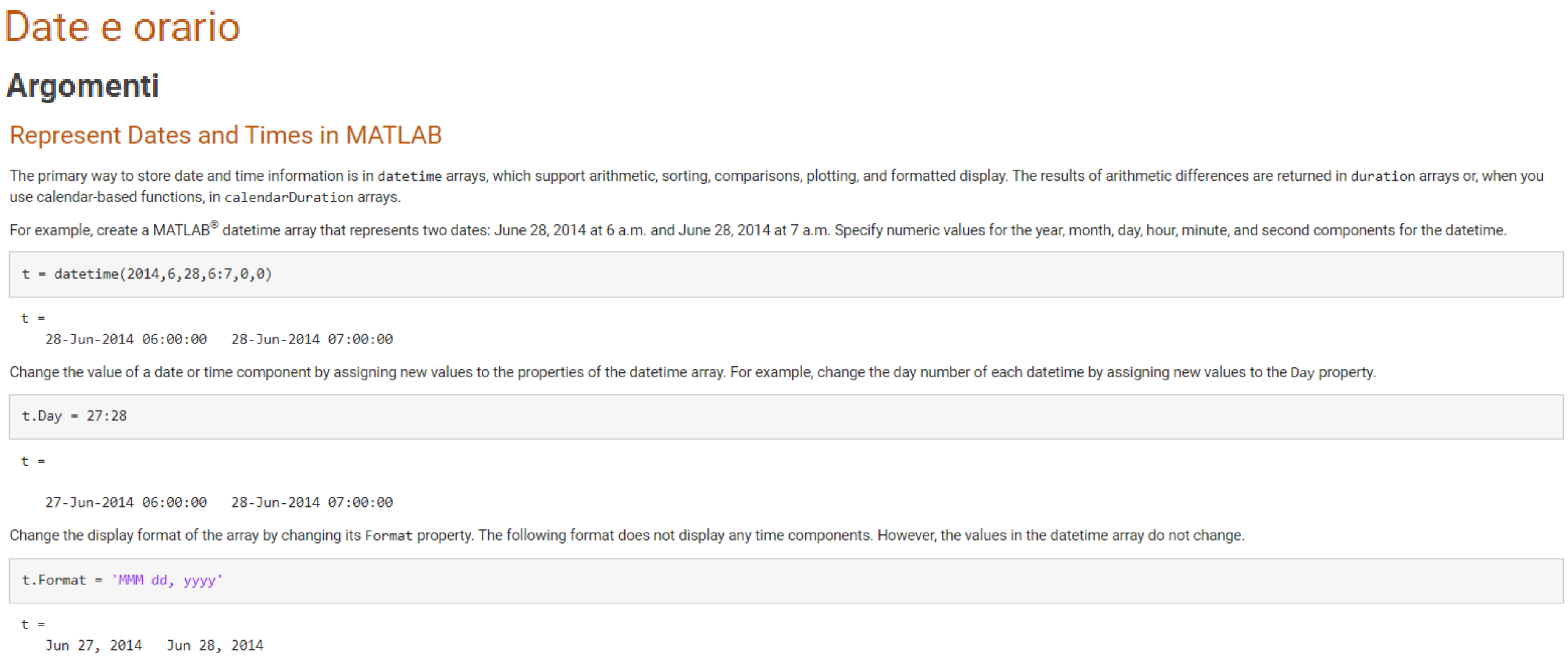
`addtodate`	(Not recommended; use `duration` or `calendarDuration`) Add time to serial date number
`etime`	(Not recommended; use `datetime` values or `between`) Time elapsed between date vectors
`months`	(Not recommended; use `between`) Number of whole months between dates

Conversione in testo

`string`	String array
`char`	Character array
`datestr`	(Not recommended; use `string` or `char`) Convert date and time to string format

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

The primary way to store date and time information is in `datetime` arrays, which support arithmetic, sorting, comparisons, plotting, and formatted display. The results of arithmetic differences are returned in `duration` arrays or, when you use calendar-based functions, in `calendarDuration` arrays.

For example, create a MATLAB® datetime array that represents two dates: June 28, 2014 at 6 a.m. and June 28, 2014 at 7 a.m. Specify numeric values for the year, month, day, hour, minute, and second components for the datetime.

```
t = datetime(2014,6,28,6:7,0,0)
```

```
t =
   28-Jun-2014 06:00:00   28-Jun-2014 07:00:00
```

Change the value of a date or time component by assigning new values to the properties of the datetime array. For example, change the day number of each datetime by assigning new values to the `Day` property.

```
t.Day = 27:28
```

```
t =

   27-Jun-2014 06:00:00   28-Jun-2014 07:00:00
```

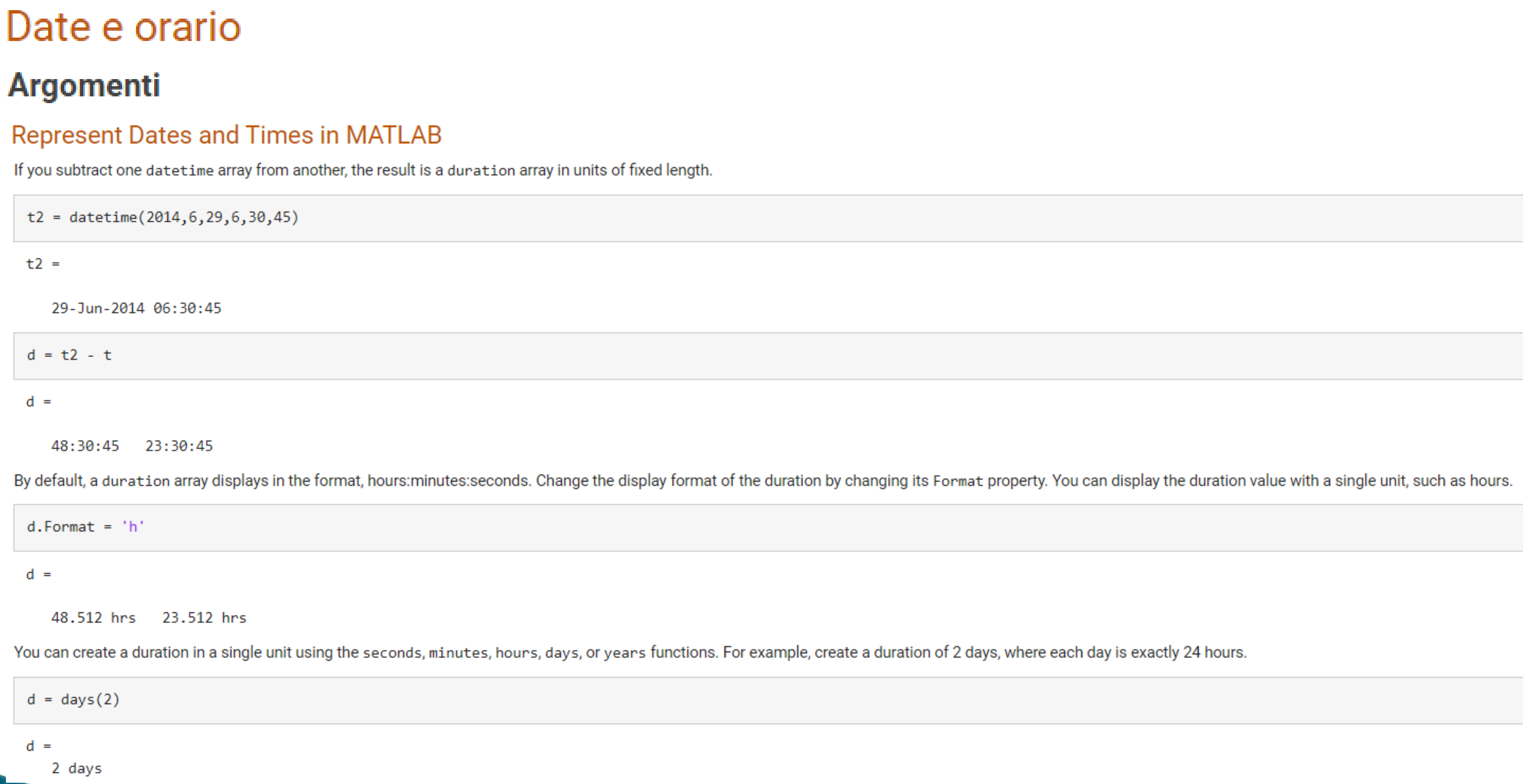
Change the display format of the array by changing its `Format` property. The following format does not display any time components. However, the values in the datetime array do not change.

```
t.Format = 'MMM dd, yyyy'
```

```
t =
   Jun 27, 2014   Jun 28, 2014
```

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

If you subtract one `datetime` array from another, the result is a `duration` array in units of fixed length.

```
t2 = datetime(2014,6,29,6,30,45)
```

```
t2 =

   29-Jun-2014 06:30:45
```

```
d = t2 - t
```

```
d =

   48:30:45   23:30:45
```

By default, a `duration` array displays in the format, hours:minutes:seconds. Change the display format of the duration by changing its `Format` property. You can display the duration value with a single unit, such as hours.

```
d.Format = 'h'
```

```
d =

   48.512 hrs   23.512 hrs
```

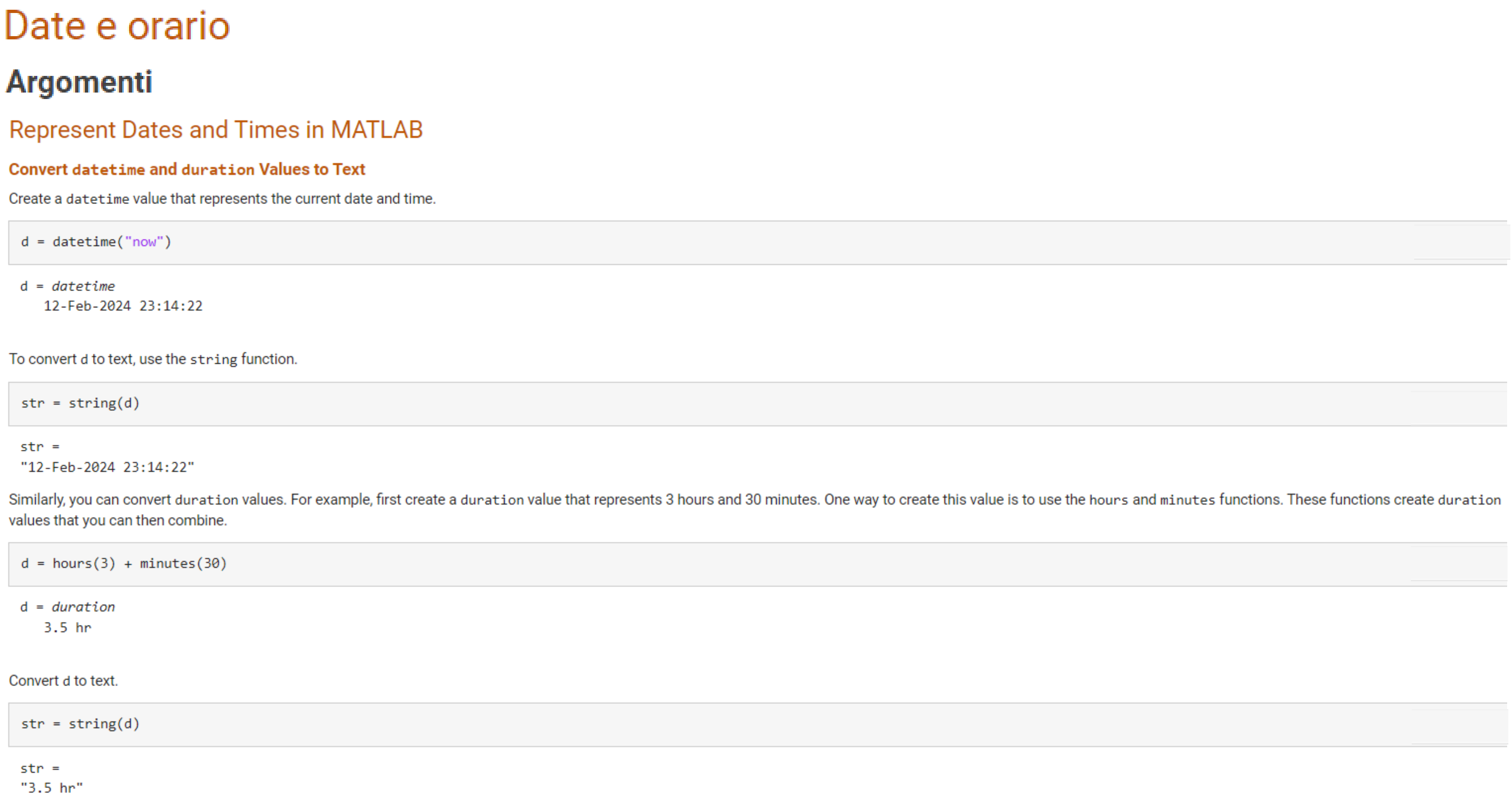
You can create a duration in a single unit using the `seconds`, `minutes`, `hours`, `days`, or `years` functions. For example, create a duration of 2 days, where each day is exactly 24 hours.

```
d = days(2)
```

```
d =
   2 days
```

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Convert `datetime` and `duration` Values to Text

Create a `datetime` value that represents the current date and time.

```
d = datetime("now")
```

```
d = datetime
   12-Feb-2024 23:14:22
```

To convert d to text, use the `string` function.

```
str = string(d)
```

```
str = 
"12-Feb-2024 23:14:22"
```

Similarly, you can convert `duration` values. For example, first create a `duration` value that represents 3 hours and 30 minutes. One way to create this value is to use the `hours` and `minutes` functions. These functions create `duration` values that you can then combine.

```
d = hours(3) + minutes(30)
```

```
d = duration
   3.5 hr
```

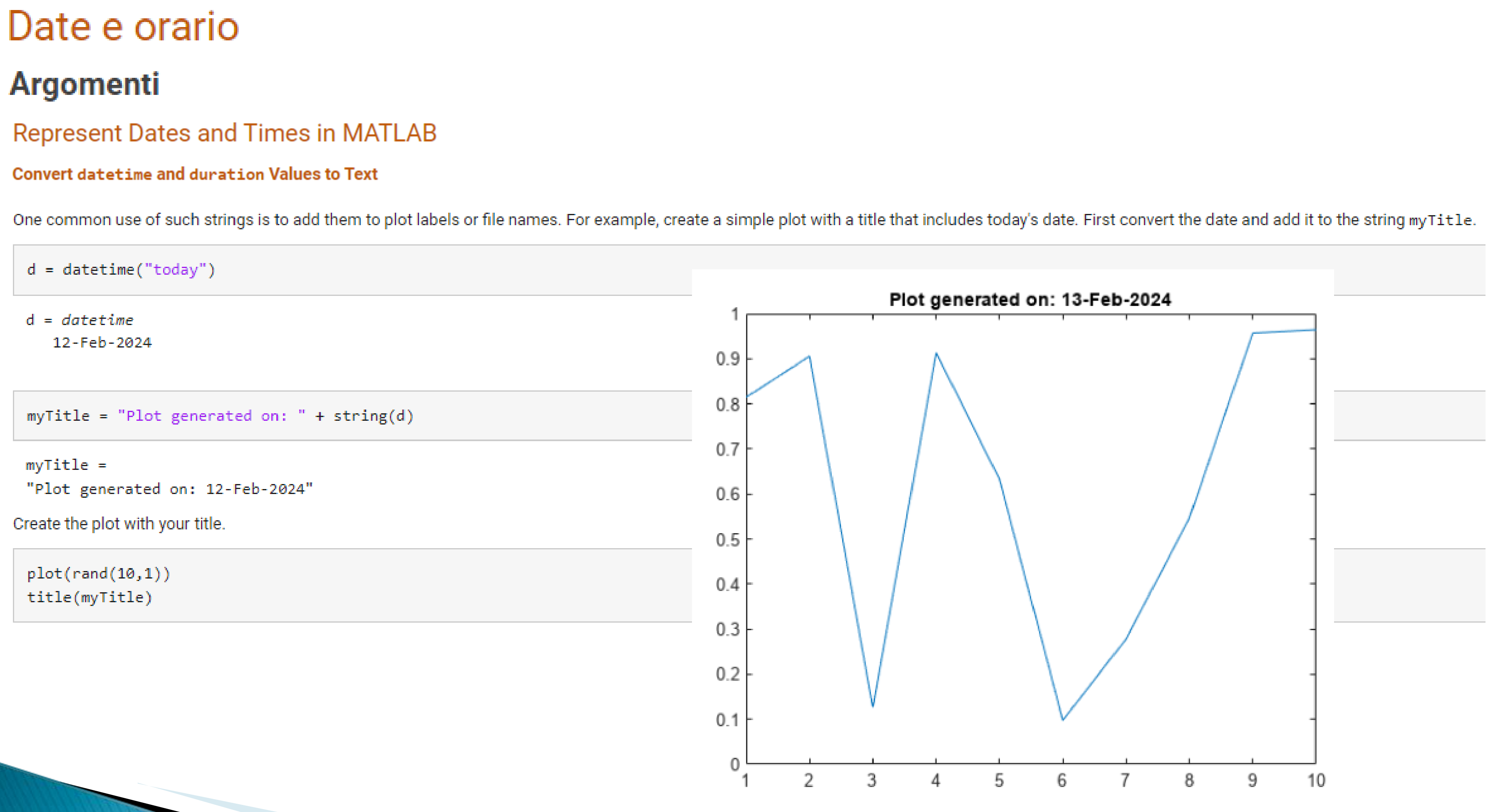
Convert d to text.

```
str = string(d)
```

```
str = 
"3.5 hr"
```

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Convert `datetime` and `duration` Values to Text

One common use of such strings is to add them to plot labels or file names. For example, create a simple plot with a title that includes today's date. First convert the date and add it to the string `myTitle`.

```
d = datetime("today")
```

```
d = datetime
   12-Feb-2024
```

```
myTitle = "Plot generated on: " + string(d)
```

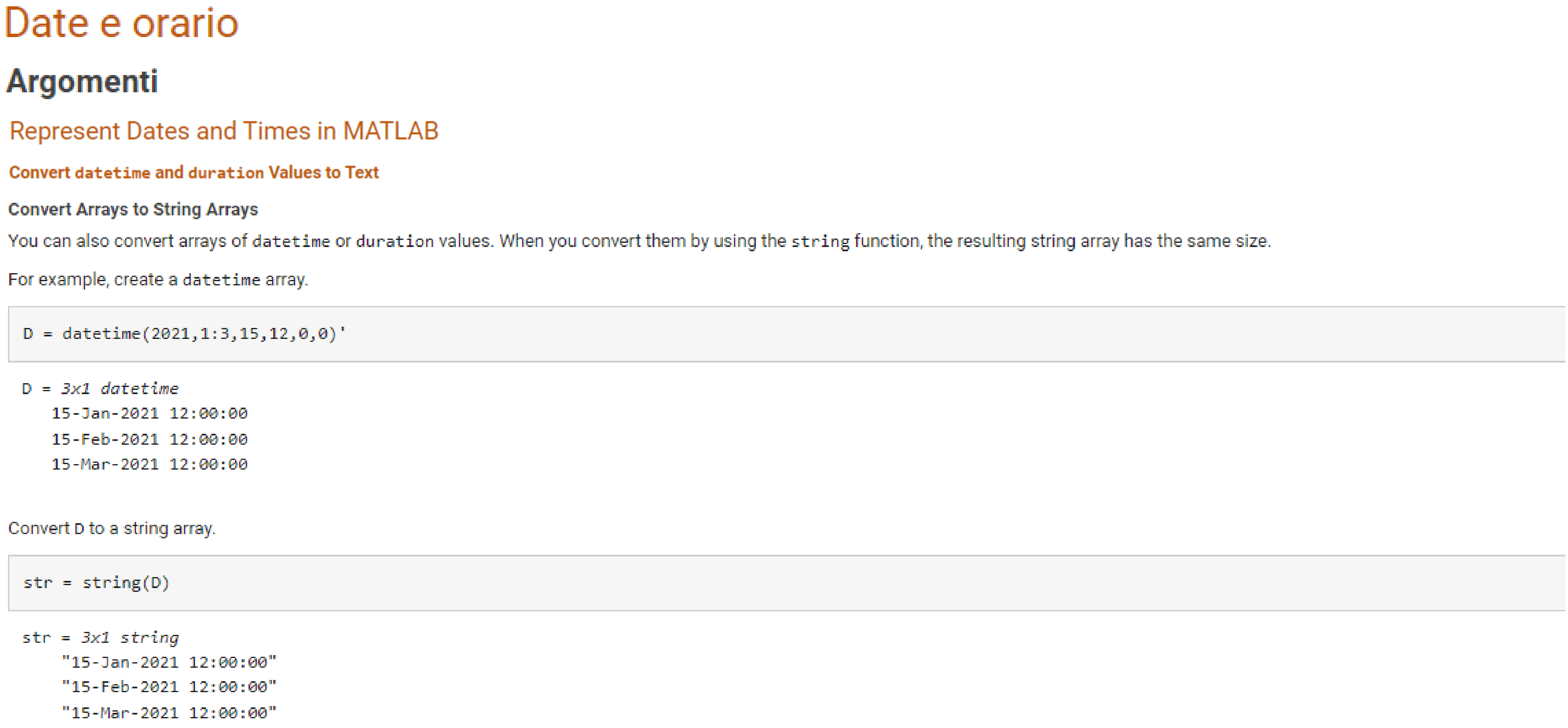
```
myTitle = 
"Plot generated on: 12-Feb-2024"
```

Create the plot with your title.

```
plot(rand(10,1))
title(myTitle)
```

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Convert `datetime` and `duration` Values to Text

Convert Arrays to String Arrays

You can also convert arrays of `datetime` or `duration` values. When you convert them by using the `string` function, the resulting string array has the same size.

For example, create a `datetime` array.

```
D = datetime(2021,1:3,15,12,0,0)'
```

```
D = 3x1 datetime
   15-Jan-2021 12:00:00
   15-Feb-2021 12:00:00
   15-Mar-2021 12:00:00
```

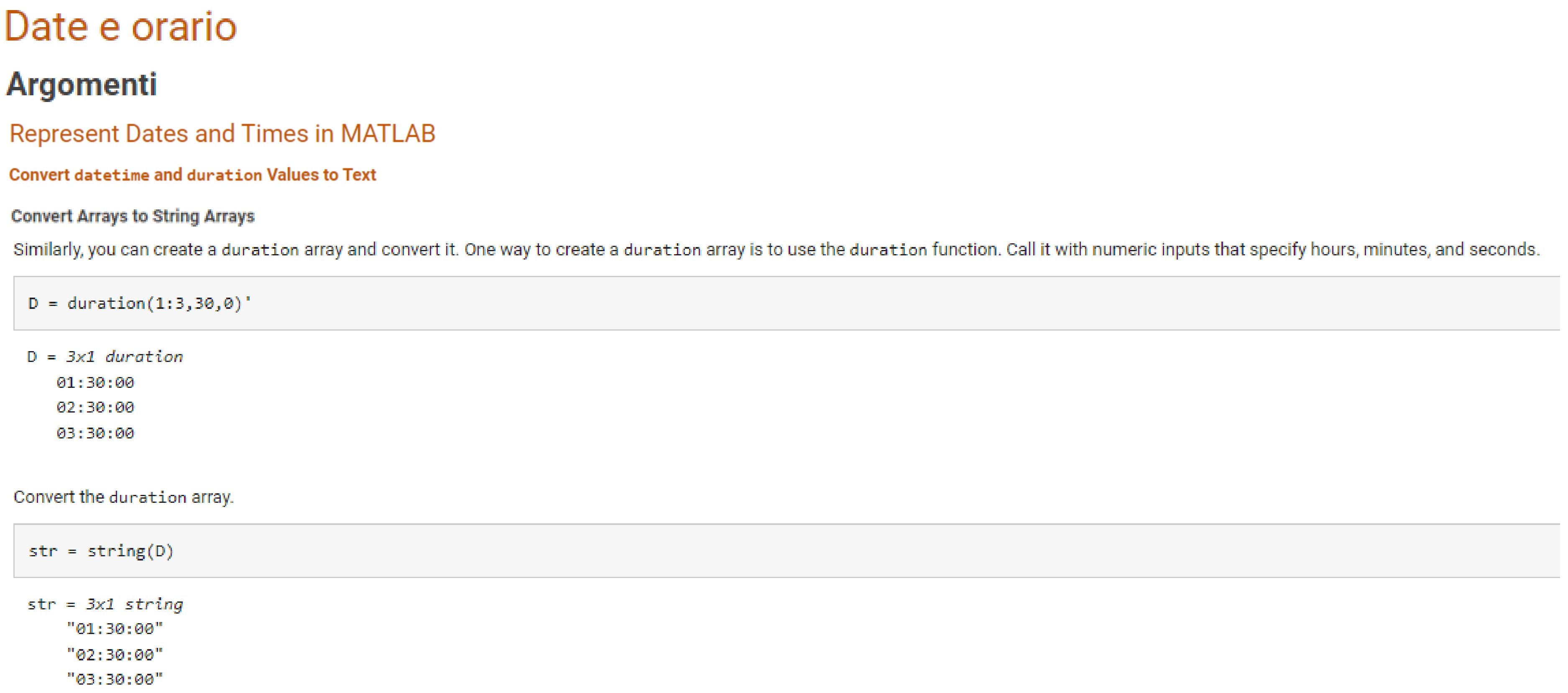
Convert `D` to a string array.

```
str = string(D)
```

```
str = 3x1 string
    "15-Jan-2021 12:00:00"
    "15-Feb-2021 12:00:00"
    "15-Mar-2021 12:00:00"
```

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Convert `datetime` and `duration` Values to Text

Convert Arrays to String Arrays

Similarly, you can create a `duration` array and convert it. One way to create a `duration` array is to use the `duration` function. Call it with numeric inputs that specify hours, minutes, and seconds.

```
D = duration(1:3,30,0)'
```

```
D = 3x1 duration
   01:30:00
   02:30:00
   03:30:00
```

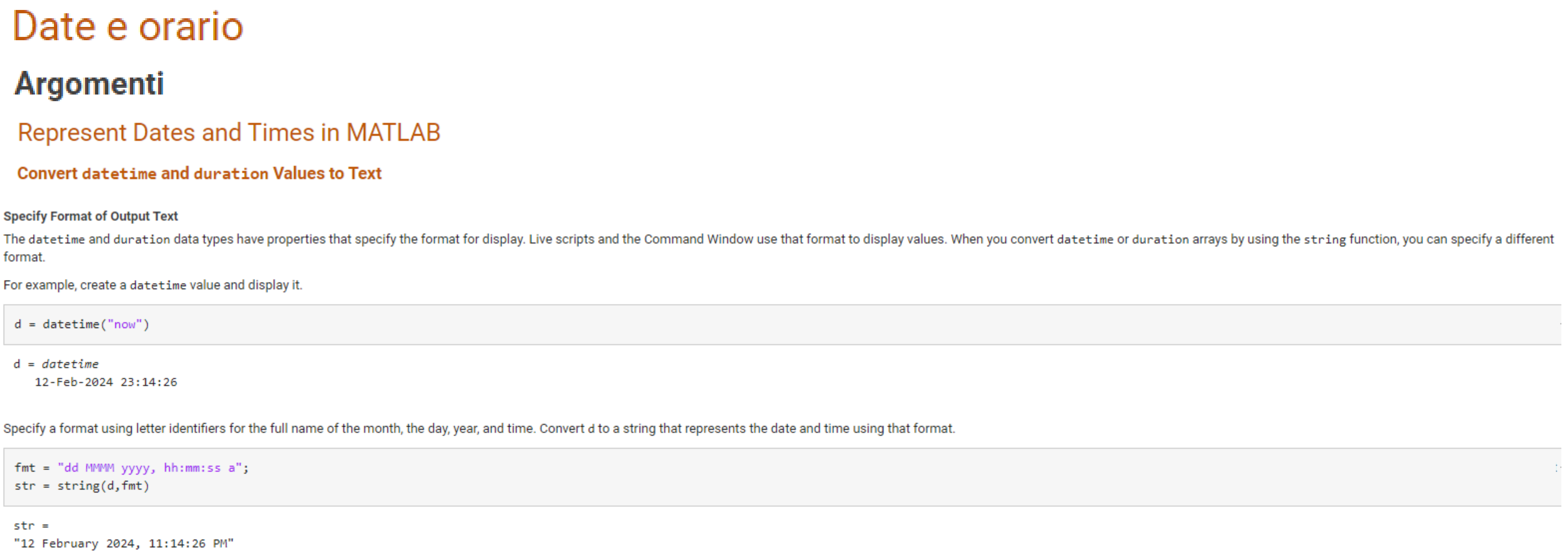
Convert the `duration` array.

```
str = string(D)
```

```
str = 3x1 string
    "01:30:00"
    "02:30:00"
    "03:30:00"
```

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Convert `datetime` and `duration` Values to Text

Specify Format of Output Text

The `datetime` and `duration` data types have properties that specify the format for display. Live scripts and the Command Window use that format to display values. When you convert `datetime` or `duration` arrays by using the `string` function, you can specify a different format.

For example, create a `datetime` value and display it.

```
d = datetime("now")
```

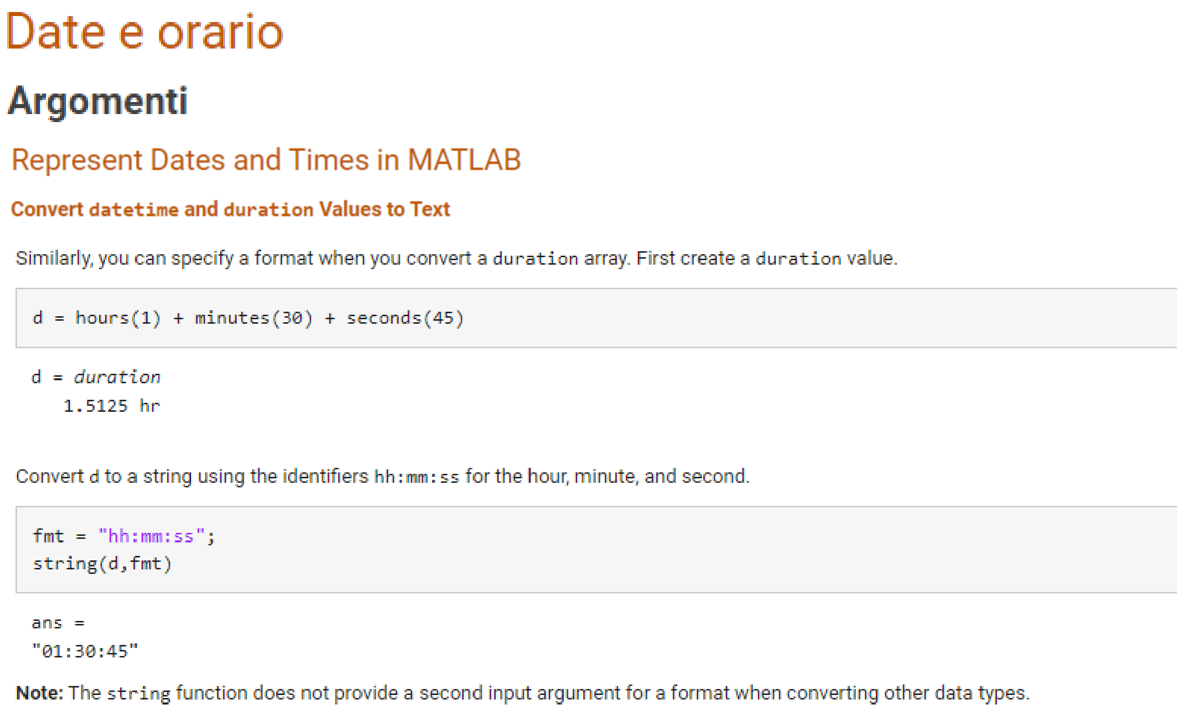
```
d = datetime
   12-Feb-2024 23:14:26
```

Specify a format using letter identifiers for the full name of the month, the day, year, and time. Convert `d` to a string that represents the date and time using that format.

```
fmt = "dd MMMM yyyy, hh:mm:ss a";
str = string(d,fmt)
```

```
str =
"12 February 2024, 11:14:26 PM"
```

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Convert `datetime` and `duration` Values to Text

Similarly, you can specify a format when you convert a `duration` array. First create a `duration` value.

```
d = hours(1) + minutes(30) + seconds(45)
```

```
d = duration
   1.5125 hr
```

Convert `d` to a string using the identifiers `hh:mm:ss` for the hour, minute, and second.

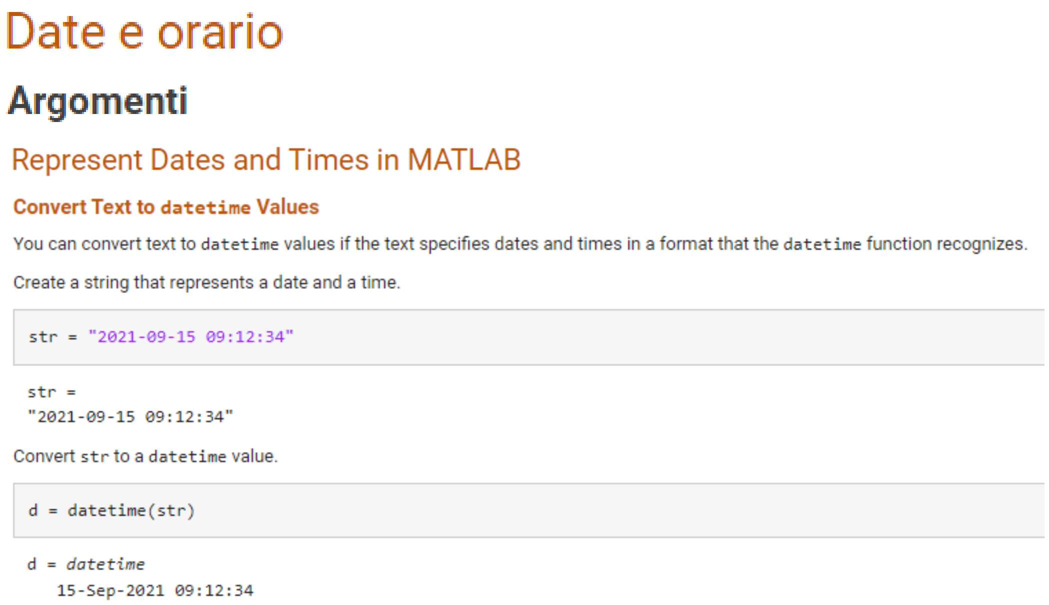
```
fmt = "hh:mm:ss";
string(d,fmt)
```

```
ans = 
"01:30:45"
```

Note: The `string` function does not provide a second input argument for a format when converting other data types.

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Convert Text to `datetime` Values

You can convert text to `datetime` values if the text specifies dates and times in a format that the `datetime` function recognizes.

Create a string that represents a date and a time.

```
str = "2021-09-15 09:12:34"
```

```
str = 
"2021-09-15 09:12:34"
```

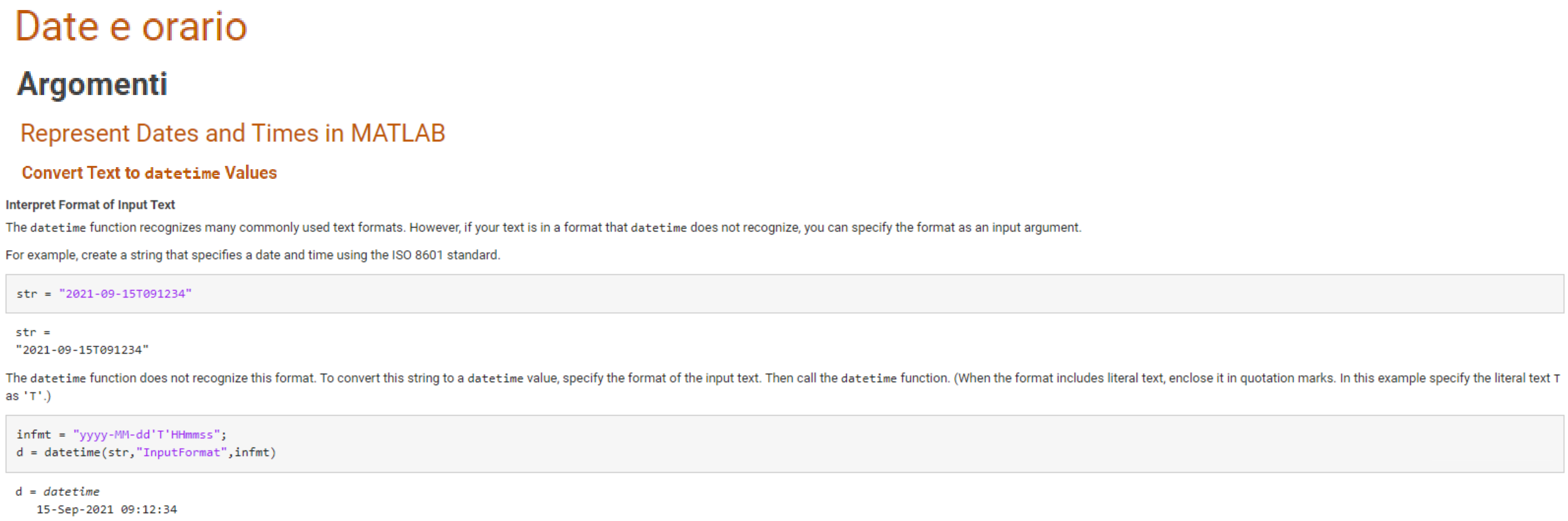
Convert `str` to a `datetime` value.

```
d = datetime(str)
```

```
d = datetime
   15-Sep-2021 09:12:34
```

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Convert Text to `datetime` Values

Interpret Format of Input Text

The `datetime` function recognizes many commonly used text formats. However, if your text is in a format that `datetime` does not recognize, you can specify the format as an input argument.

For example, create a string that specifies a date and time using the ISO 8601 standard.

```
str = "2021-09-15T091234"
```

```
str = 
"2021-09-15T091234"
```

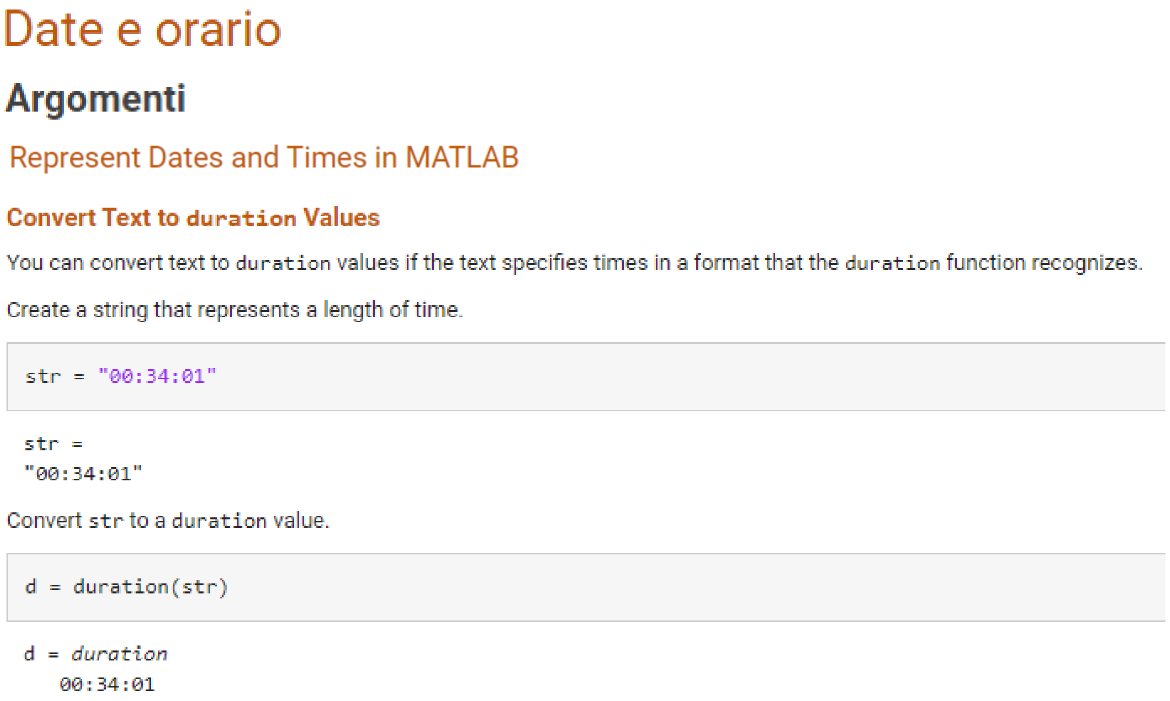
The `datetime` function does not recognize this format. To convert this string to a `datetime` value, specify the format of the input text. Then call the `datetime` function. (When the format includes literal text, enclose it in quotation marks. In this example specify the literal text `T` as `'T'`.)

```
infmt = "yyyy-MM-dd'T'HHmmss";
d = datetime(str,"InputFormat",infmt)
```

```
d = datetime
   15-Sep-2021 09:12:34
```

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Convert Text to `duration` Values

You can convert text to `duration` values if the text specifies times in a format that the `duration` function recognizes.

Create a string that represents a length of time.

```
str = "00:34:01"
```

```
str = 
"00:34:01"
```

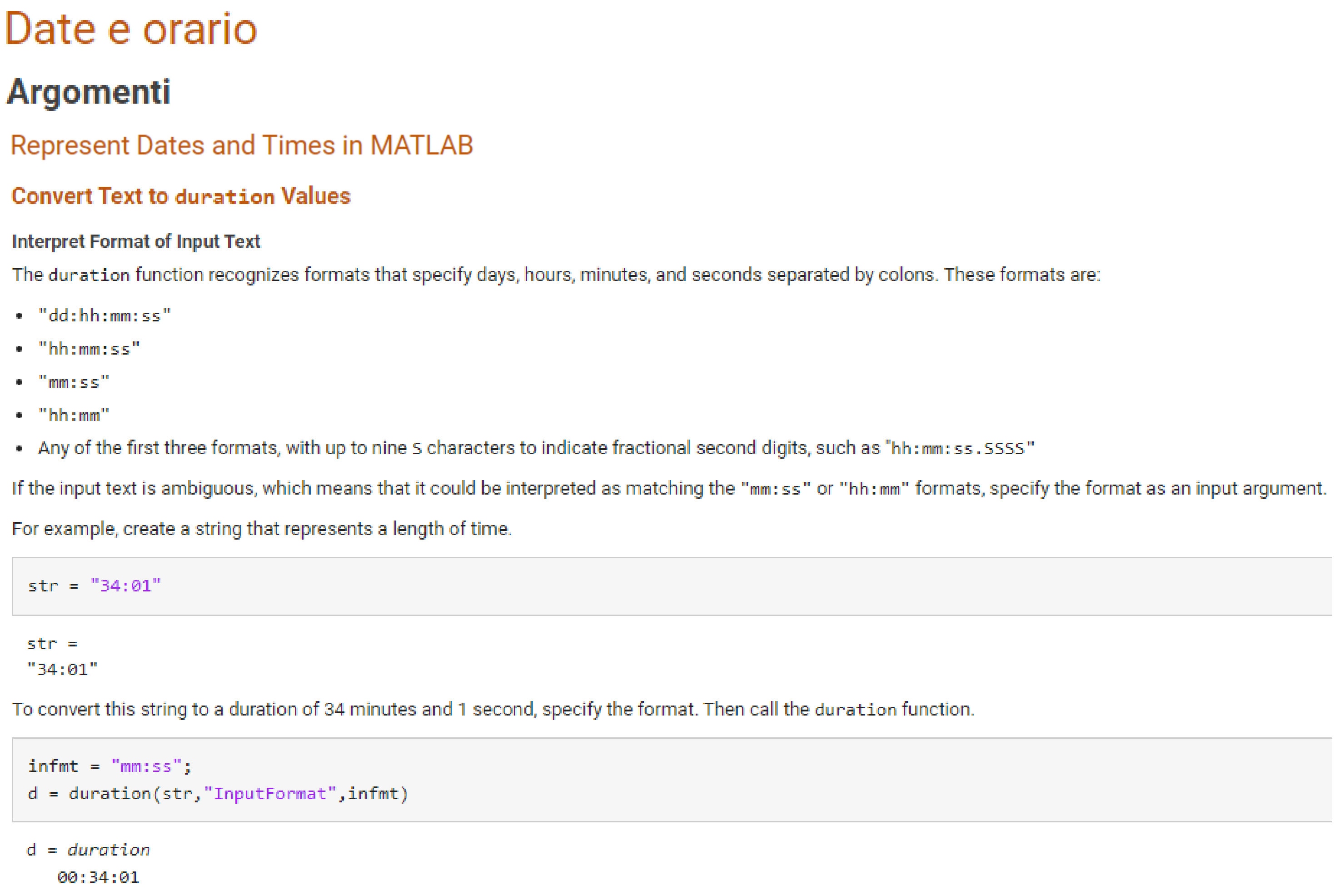
Convert `str` to a `duration` value.

```
d = duration(str)
```

```
d = duration
   00:34:01
```

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Convert Text to `duration` Values

Interpret Format of Input Text

The `duration` function recognizes formats that specify days, hours, minutes, and seconds separated by colons. These formats are:

- `"dd:hh:mm:ss"`
- `"hh:mm:ss"`
- `"mm:ss"`
- `"hh:mm"`
- Any of the first three formats, with up to nine `S` characters to indicate fractional second digits, such as `"hh:mm:ss.SSSS"`

If the input text is ambiguous, which means that it could be interpreted as matching the `"mm:ss"` or `"hh:mm"` formats, specify the format as an input argument.

For example, create a string that represents a length of time.

```
str = "34:01"
```

```
str = 
"34:01"
```

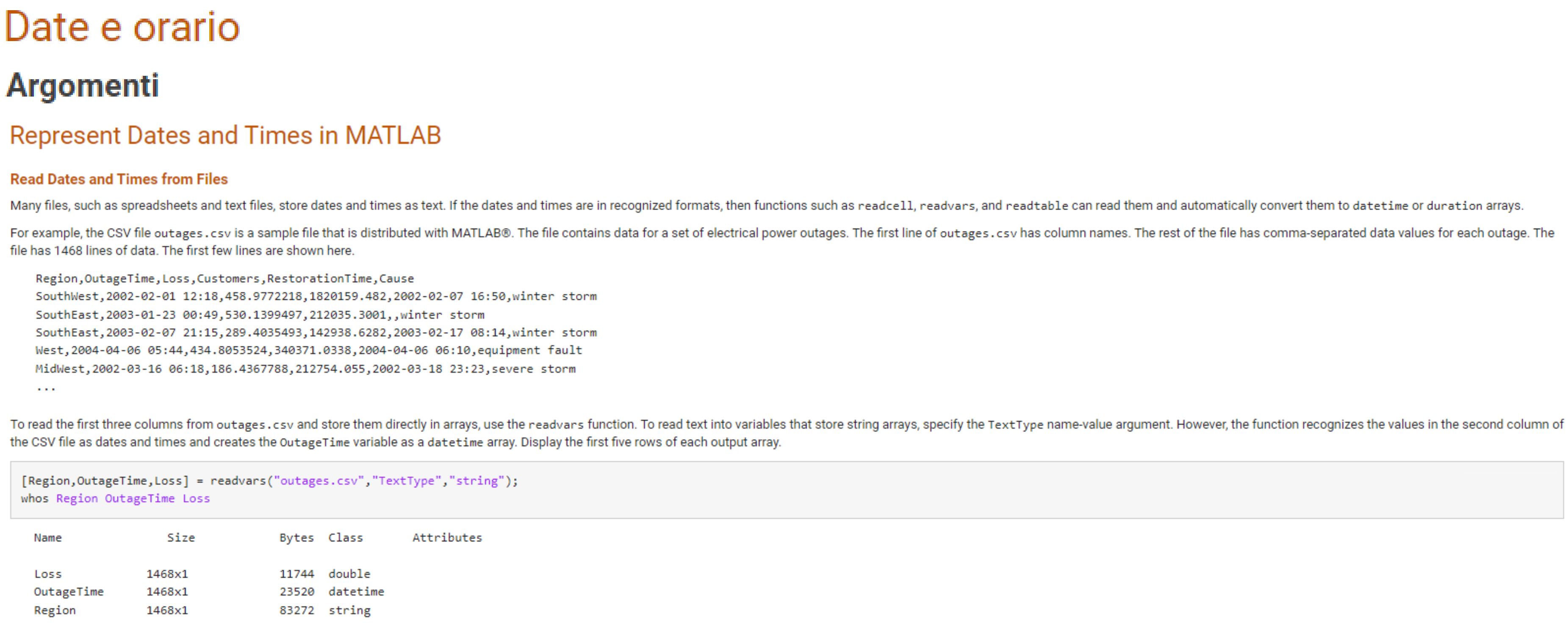
To convert this string to a duration of 34 minutes and 1 second, specify the format. Then call the `duration` function.

```
infmt = "mm:ss";
d = duration(str,"InputFormat",infmt)
```

```
d = duration
   00:34:01
```

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Read Dates and Times from Files

Many files, such as spreadsheets and text files, store dates and times as text. If the dates and times are in recognized formats, then functions such as `readcell`, `readvars`, and `readtable` can read them and automatically convert them to `datetime` or `duration` arrays.

For example, the CSV file `outages.csv` is a sample file that is distributed with MATLAB®. The file contains data for a set of electrical power outages. The first line of `outages.csv` has column names. The rest of the file has comma-separated data values for each outage. The file has 1468 lines of data. The first few lines are shown here.

```
Region,OutageTime,Loss,Customers,RestorationTime,Cause
SouthWest,2002-02-01 12:18,458.9772218,1820159.482,2002-02-07 16:50,winter storm
SouthEast,2003-01-23 00:49,530.1399497,212035.3001,,winter storm
SouthEast,2003-02-07 21:15,289.4035493,142938.6282,2003-02-17 08:14,winter storm
West,2004-04-06 05:44,434.8053524,340371.0338,2004-04-06 06:10,equipment fault
MidWest,2002-03-16 06:18,186.4367788,212754.055,2002-03-18 23:23,severe storm
...
```

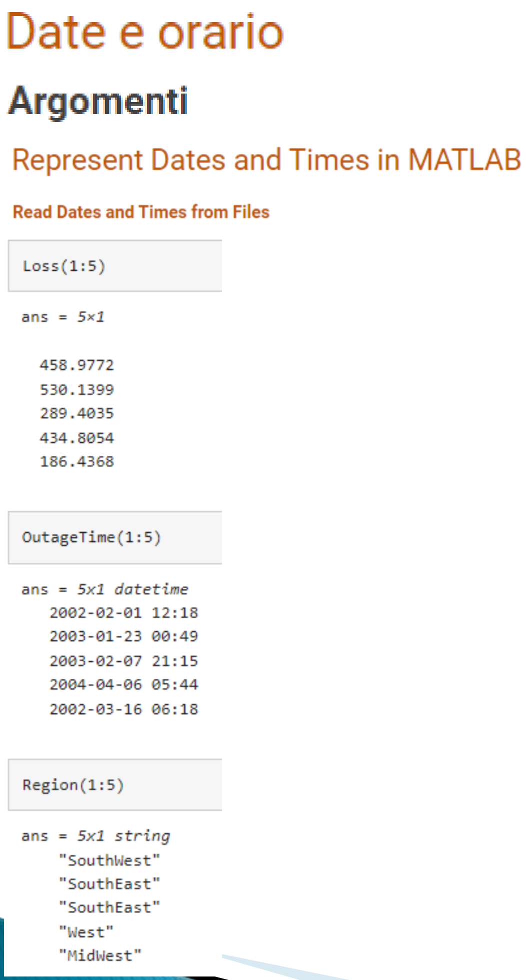
To read the first three columns from `outages.csv` and store them directly in arrays, use the `readvars` function. To read text into variables that store string arrays, specify the `TextType` name-value argument. However, the function recognizes the values in the second column of the CSV file as dates and times and creates the `OutageTime` variable as a `datetime` array. Display the first five rows of each output array.

```
[Region,OutageTime,Loss] = readvars("outages.csv","TextType","string");
whos Region OutageTime Loss
```

```
  Name                Size             Bytes  Class       Attributes

  Loss             1468x1              11744  double
  OutageTime       1468x1              23520  datetime
  Region           1468x1              83272  string
```

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Read Dates and Times from Files

```
Loss(1:5)
```

```
ans = 5×1

   458.9772
   530.1399
   289.4035
   434.8054
   186.4368
```

```
OutageTime(1:5)
```

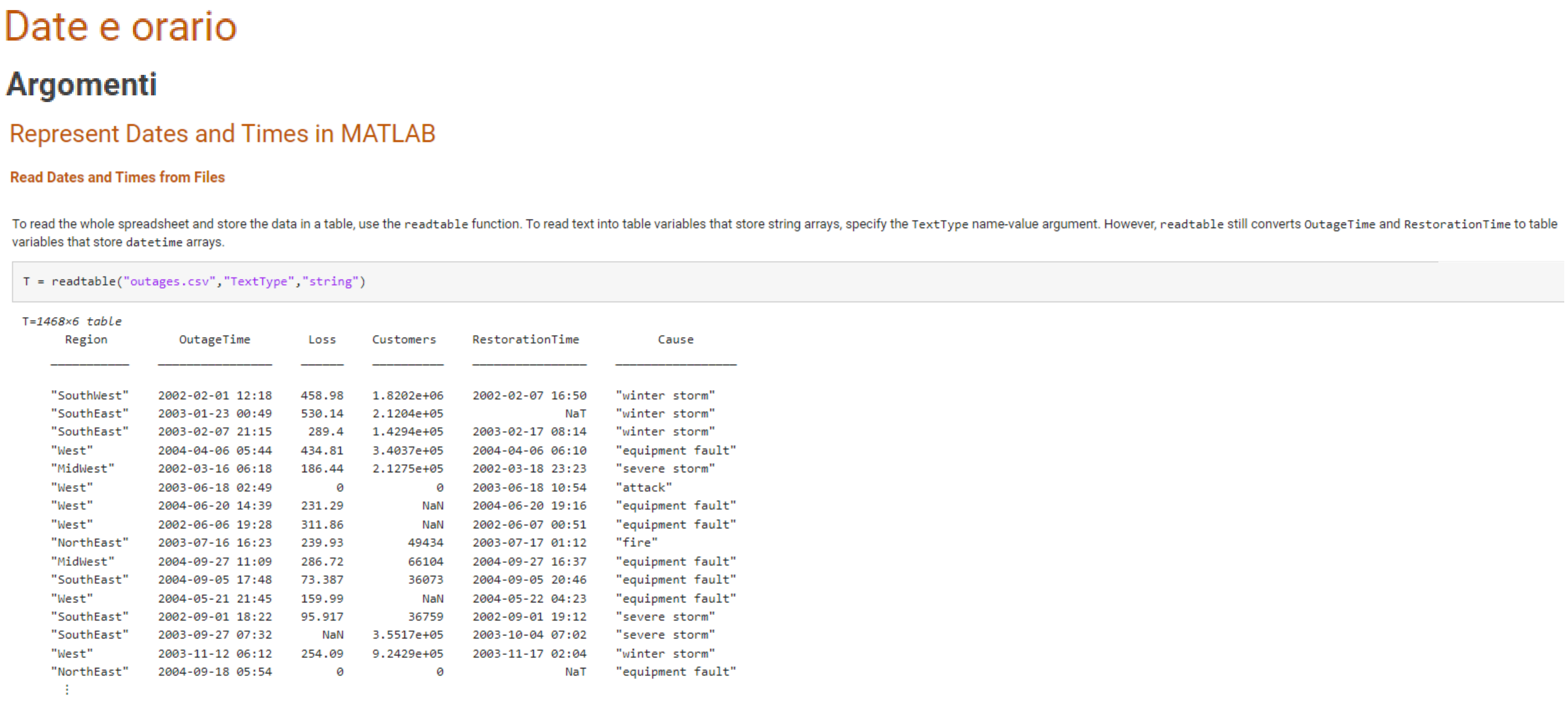
```
ans = 5x1 datetime
    2002-02-01 12:18
    2003-01-23 00:49
    2003-02-07 21:15
    2004-04-06 05:44
    2002-03-16 06:18
```

```
Region(1:5)
```

```
ans = 5x1 string
     "SouthWest"
     "SouthEast"
     "SouthEast"
     "West"
     "MidWest"
```

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

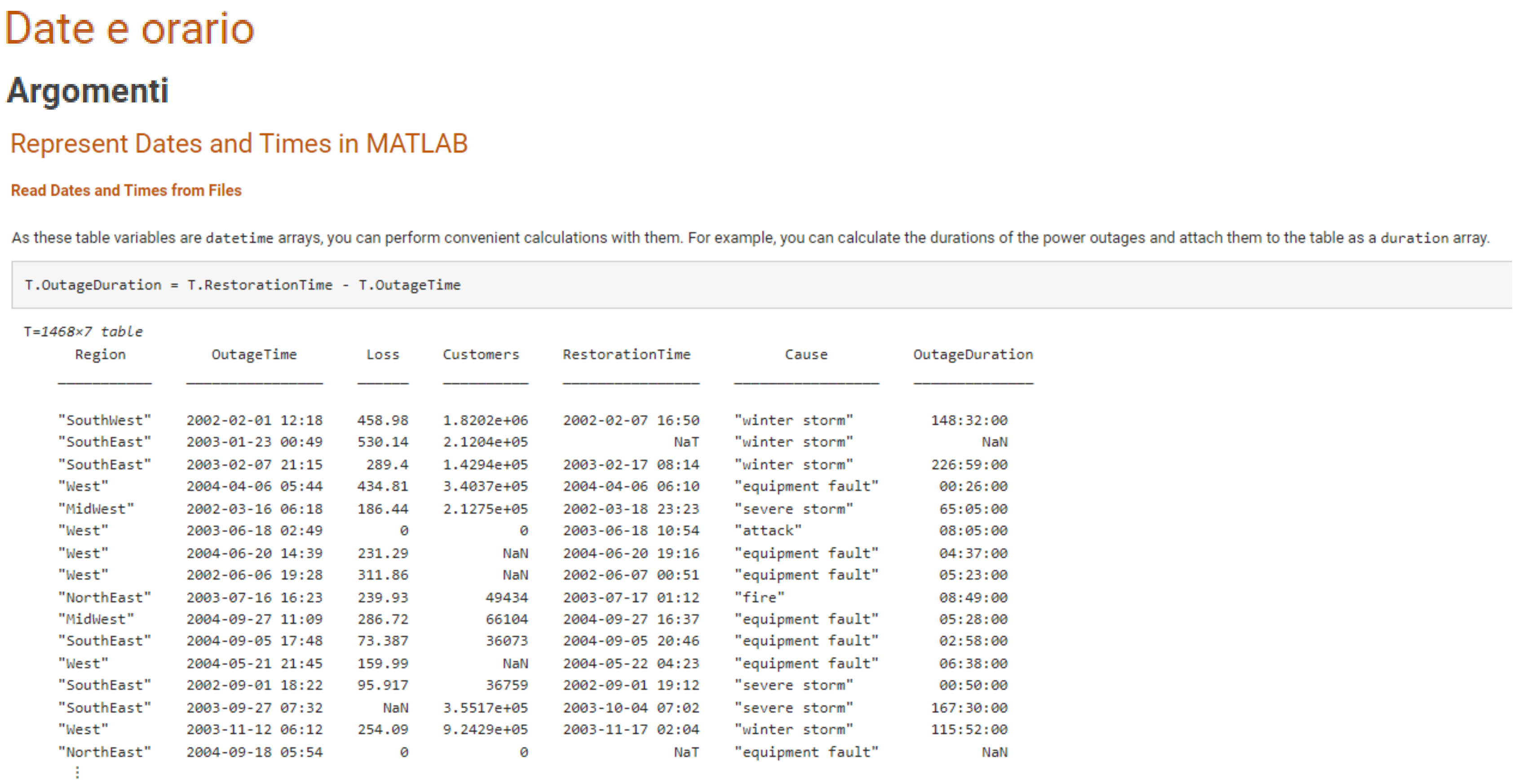
Read Dates and Times from Files

To read the whole spreadsheet and store the data in a table, use the `readtable` function. To read text into table variables that store string arrays, specify the `TextType` name-value argument. However, `readtable` still converts `OutageTime` and `RestorationTime` to table variables that store `datetime` arrays.

```
T = readtable("outages.csv","TextType","string")
```

```
T=1468×6 table
       Region          OutageTime        Loss     Customers      RestorationTime            Cause
    ___________    ________________    ______    __________    ________________    _________________

    "SouthWest"    2002-02-01 12:18    458.98    1.8202e+06    2002-02-07 16:50    "winter storm"
    "SouthEast"    2003-01-23 00:49    530.14    2.1204e+05                 NaT    "winter storm"
    "SouthEast"    2003-02-07 21:15     289.4    1.4294e+05    2003-02-17 08:14    "winter storm"
    "West"         2004-04-06 05:44    434.81    3.4037e+05    2004-04-06 06:10    "equipment fault"
    "MidWest"      2002-03-16 06:18    186.44    2.1275e+05    2002-03-18 23:23    "severe storm"
    "West"         2003-06-18 02:49         0             0    2003-06-18 10:54    "attack"
    "West"         2004-06-20 14:39    231.29           NaN    2004-06-20 19:16    "equipment fault"
    "West"         2002-06-06 19:28    311.86           NaN    2002-06-07 00:51    "equipment fault"
    "NorthEast"    2003-07-16 16:23    239.93         49434    2003-07-17 01:12    "fire"
    "MidWest"      2004-09-27 11:09    286.72         66104    2004-09-27 16:37    "equipment fault"
    "SouthEast"    2004-09-05 17:48    73.387         36073    2004-09-05 20:46    "equipment fault"
    "West"         2004-05-21 21:45    159.99           NaN    2004-05-22 04:23    "equipment fault"
    "SouthEast"    2002-09-01 18:22    95.917         36759    2002-09-01 19:12    "severe storm"
    "SouthEast"    2003-09-27 07:32       NaN    3.5517e+05    2003-10-04 07:02    "severe storm"
    "West"         2003-11-12 06:12    254.09    9.2429e+05    2003-11-17 02:04    "winter storm"
    "NorthEast"    2004-09-18 05:54         0             0                 NaT    "equipment fault"
          ⋮
```

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Read Dates and Times from Files

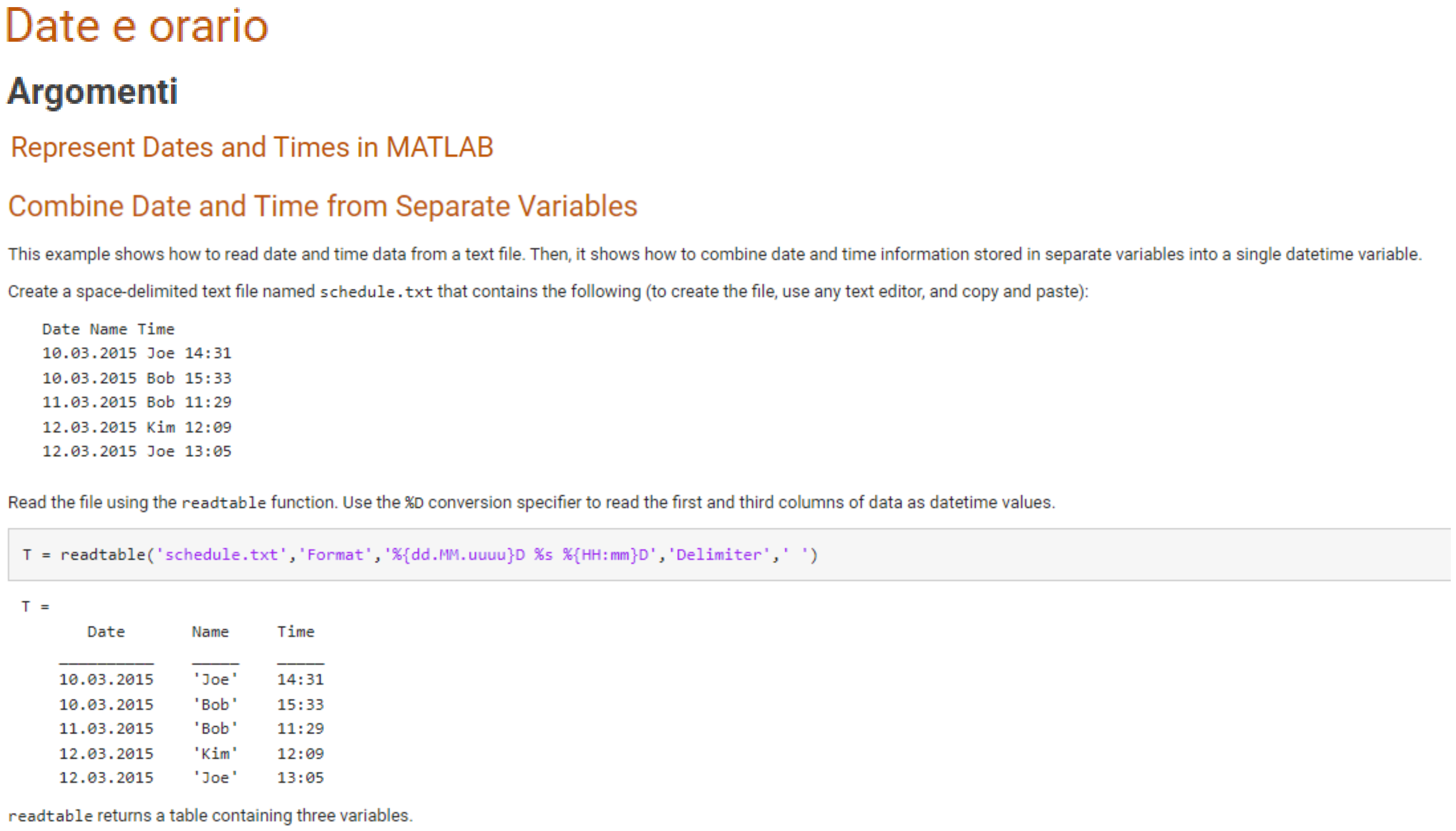
As these table variables are `datetime` arrays, you can perform convenient calculations with them. For example, you can calculate the durations of the power outages and attach them to the table as a `duration` array.

```
T.OutageDuration = T.RestorationTime - T.OutageTime
```

```
T=1468×7 table
      Region         OutageTime        Loss     Customers     RestorationTime          Cause          OutageDuration
    ___________    ________________    ______    __________    ________________    _________________    ______________

    "SouthWest"    2002-02-01 12:18    458.98    1.8202e+06    2002-02-07 16:50    "winter storm"         148:32:00
    "SouthEast"    2003-01-23 00:49    530.14    2.1204e+05                 NaT    "winter storm"               NaN
    "SouthEast"    2003-02-07 21:15     289.4    1.4294e+05    2003-02-17 08:14    "winter storm"         226:59:00
    "West"         2004-04-06 05:44    434.81    3.4037e+05    2004-04-06 06:10    "equipment fault"       00:26:00
    "MidWest"      2002-03-16 06:18    186.44    2.1275e+05    2002-03-18 23:23    "severe storm"          65:05:00
    "West"         2003-06-18 02:49         0             0    2003-06-18 10:54    "attack"                08:05:00
    "West"         2004-06-20 14:39    231.29           NaN    2004-06-20 19:16    "equipment fault"       04:37:00
    "West"         2002-06-06 19:28    311.86           NaN    2002-06-07 00:51    "equipment fault"       05:23:00
    "NorthEast"    2003-07-16 16:23    239.93         49434    2003-07-17 01:12    "fire"                  08:49:00
    "MidWest"      2004-09-27 11:09    286.72         66104    2004-09-27 16:37    "equipment fault"       05:28:00
    "SouthEast"    2004-09-05 17:48    73.387         36073    2004-09-05 20:46    "equipment fault"       02:58:00
    "West"         2004-05-21 21:45    159.99           NaN    2004-05-22 04:23    "equipment fault"       06:38:00
    "SouthEast"    2002-09-01 18:22    95.917         36759    2002-09-01 19:12    "severe storm"          00:50:00
    "SouthEast"    2003-09-27 07:32       NaN    3.5517e+05    2003-10-04 07:02    "severe storm"         167:30:00
    "West"         2003-11-12 06:12    254.09    9.2429e+05    2003-11-17 02:04    "winter storm"         115:52:00
    "NorthEast"    2004-09-18 05:54         0             0                 NaT    "equipment fault"            NaN
         ⋮
```

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Combine Date and Time from Separate Variables

This example shows how to read date and time data from a text file. Then, it shows how to combine date and time information stored in separate variables into a single datetime variable.

Create a space-delimited text file named `schedule.txt` that contains the following (to create the file, use any text editor, and copy and paste):

```
Date Name Time
10.03.2015 Joe 14:31
10.03.2015 Bob 15:33
11.03.2015 Bob 11:29
12.03.2015 Kim 12:09
12.03.2015 Joe 13:05
```

Read the file using the `readtable` function. Use the `%D` conversion specifier to read the first and third columns of data as datetime values.

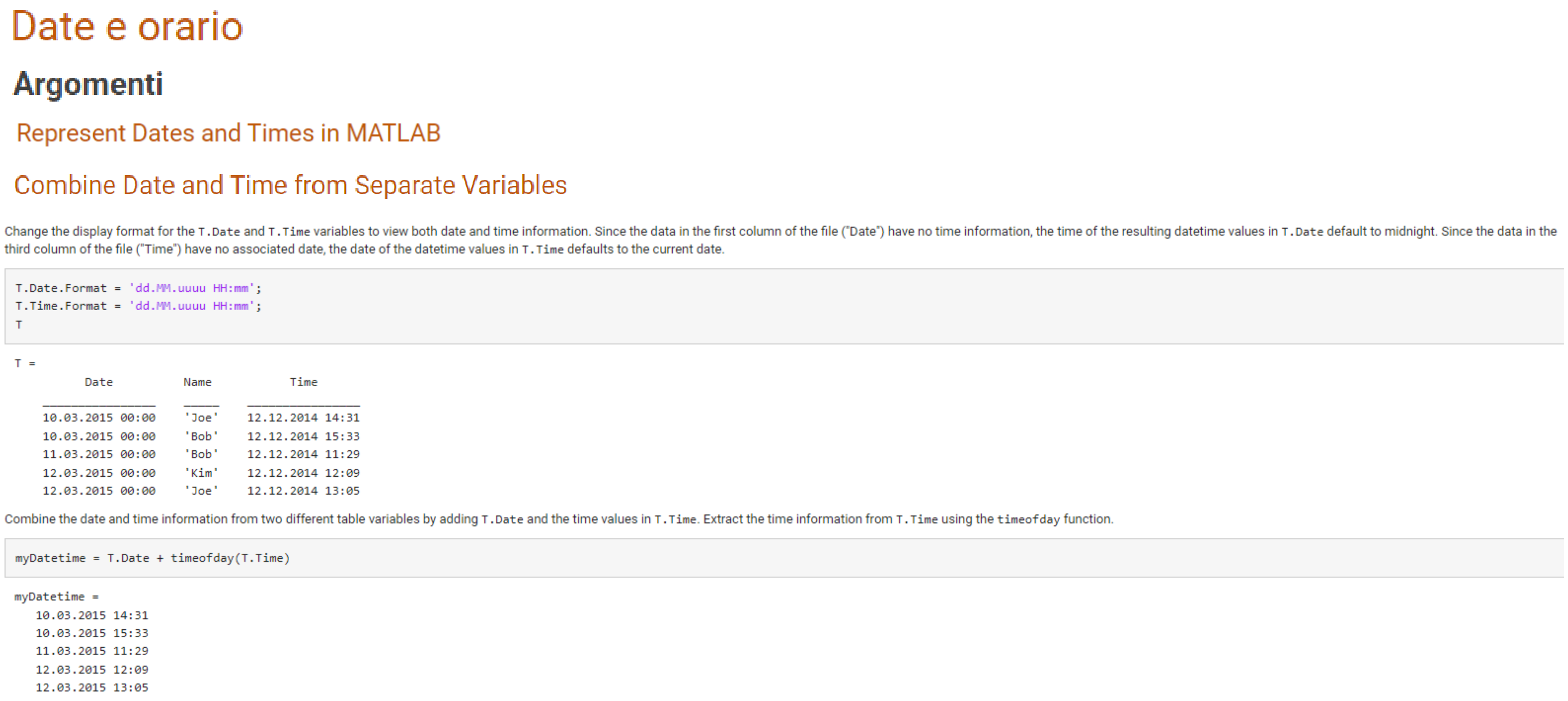
```
T = readtable('schedule.txt','Format','%{dd.MM.uuuu}D %s %{HH:mm}D','Delimiter',' ')
```

```
T = 

       Date        Name     Time 
    __________    _____    _____

    10.03.2015    'Joe'    14:31
    10.03.2015    'Bob'    15:33
    11.03.2015    'Bob'    11:29
    12.03.2015    'Kim'    12:09
    12.03.2015    'Joe'    13:05
```

`readtable` returns a table containing three variables.

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Combine Date and Time from Separate Variables

Change the display format for the `T.Date` and `T.Time` variables to view both date and time information. Since the data in the first column of the file ("Date") have no time information, the time of the resulting datetime values in `T.Date` default to midnight. Since the data in the third column of the file ("Time") have no associated date, the date of the datetime values in `T.Time` defaults to the current date.

```
T.Date.Format = 'dd.MM.uuuu HH:mm';
T.Time.Format = 'dd.MM.uuuu HH:mm';
T
```

```
T = 
          Date            Name            Time      

    ________________    _____    ________________

    10.03.2015 00:00    'Joe'    12.12.2014 14:31
    10.03.2015 00:00    'Bob'    12.12.2014 15:33
    11.03.2015 00:00    'Bob'    12.12.2014 11:29
    12.03.2015 00:00    'Kim'    12.12.2014 12:09
    12.03.2015 00:00    'Joe'    12.12.2014 13:05
```

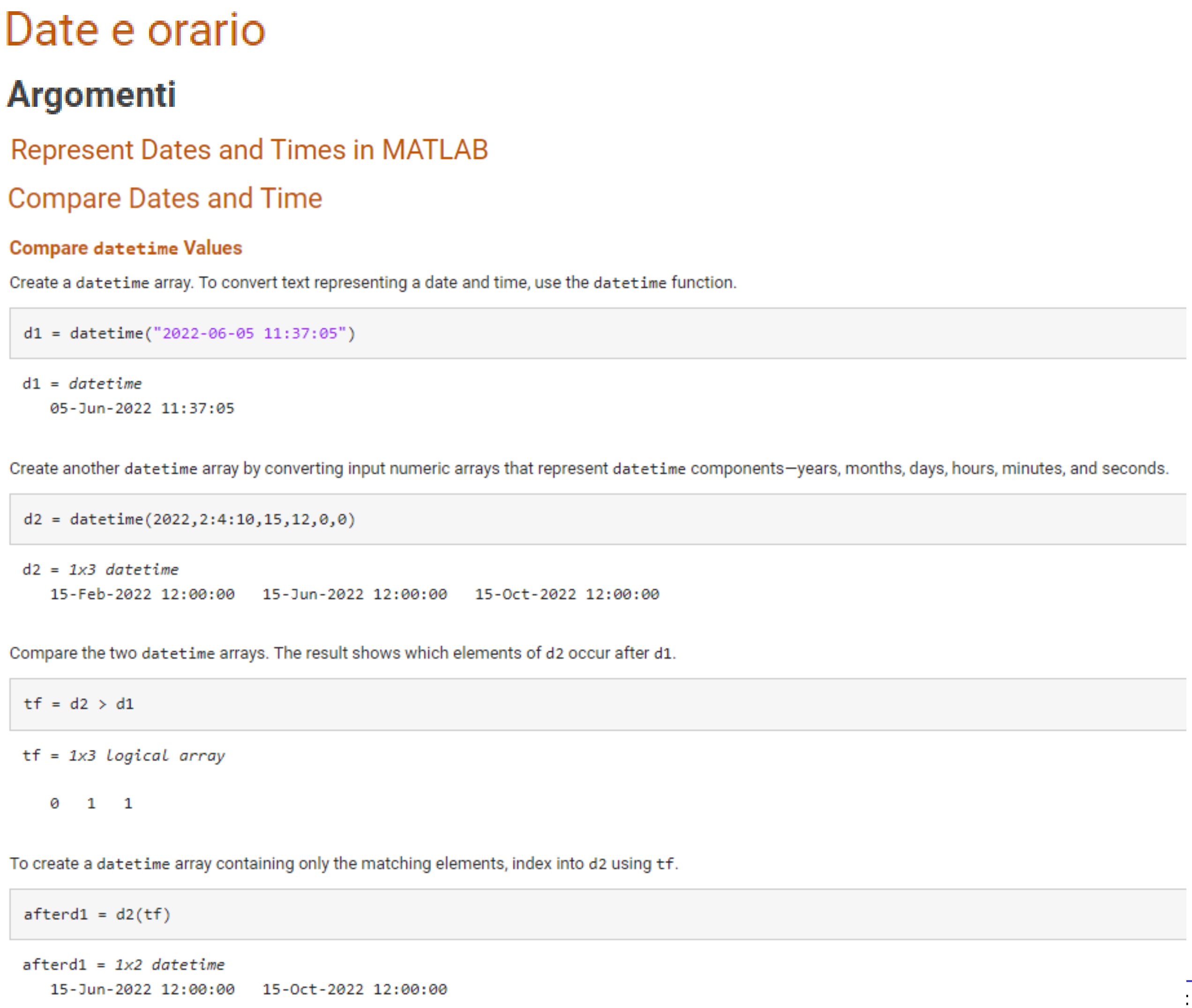
Combine the date and time information from two different table variables by adding `T.Date` and the time values in `T.Time`. Extract the time information from `T.Time` using the `timeofday` function.

```
myDatetime = T.Date + timeofday(T.Time)
```

```
myDatetime = 
   10.03.2015 14:31
   10.03.2015 15:33
   11.03.2015 11:29
   12.03.2015 12:09
   12.03.2015 13:05
```

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Compare Dates and Time

Compare datetime Values

Create a datetime array. To convert text representing a date and time, use the datetime function.

```
d1 = datetime("2022-06-05 11:37:05")
```

```
d1 = datetime
   05-Jun-2022 11:37:05
```

Create another datetime array by converting input numeric arrays that represent datetime components—years, months, days, hours, minutes, and seconds.

```
d2 = datetime(2022,2:4:10,15,12,0,0)
```

```
d2 = 1x3 datetime
   15-Feb-2022 12:00:00   15-Jun-2022 12:00:00   15-Oct-2022 12:00:00
```

Compare the two datetime arrays. The result shows which elements of d2 occur after d1.

```
tf = d2 > d1
```

```
tf = 1x3 logical array

   0   1   1
```

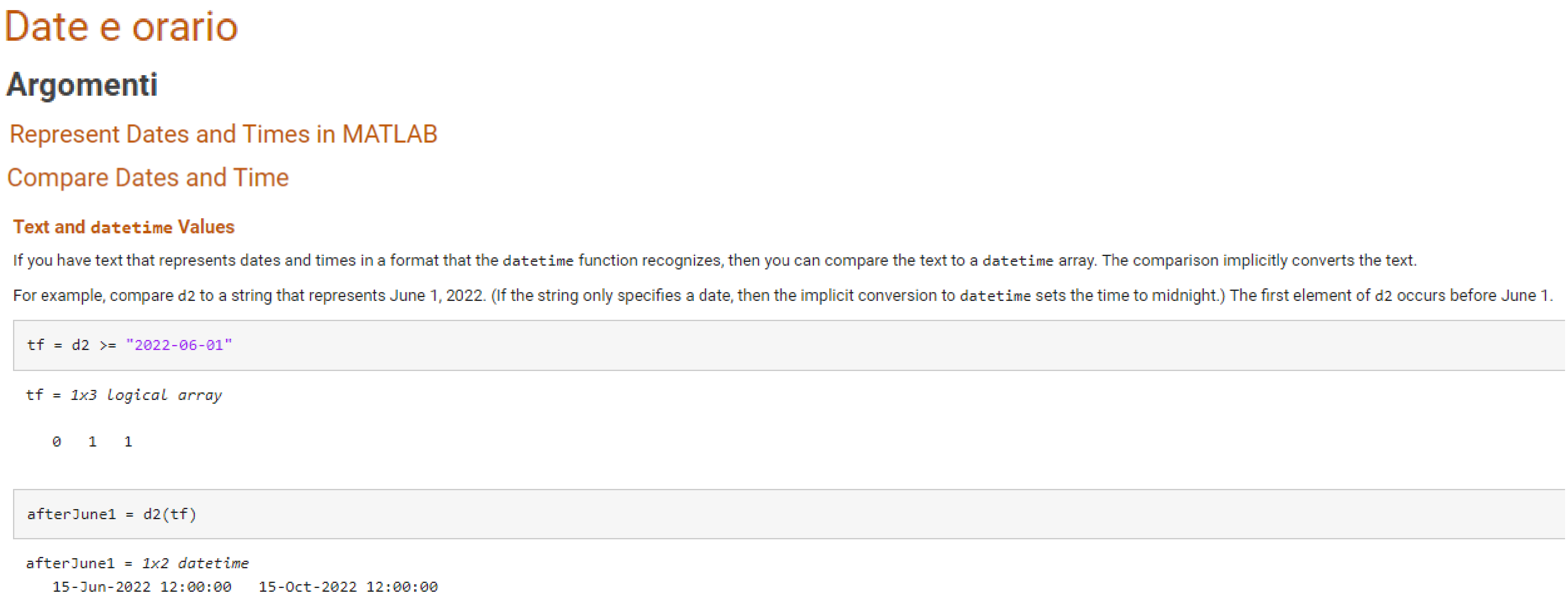
To create a datetime array containing only the matching elements, index into d2 using tf.

```
afterd1 = d2(tf)
```

```
afterd1 = 1x2 datetime
   15-Jun-2022 12:00:00   15-Oct-2022 12:00:00
```

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Compare Dates and Time

Text and `datetime` Values

If you have text that represents dates and times in a format that the `datetime` function recognizes, then you can compare the text to a `datetime` array. The comparison implicitly converts the text.

For example, compare `d2` to a string that represents June 1, 2022. (If the string only specifies a date, then the implicit conversion to `datetime` sets the time to midnight.) The first element of `d2` occurs before June 1.

```
tf = d2 >= "2022-06-01"
```

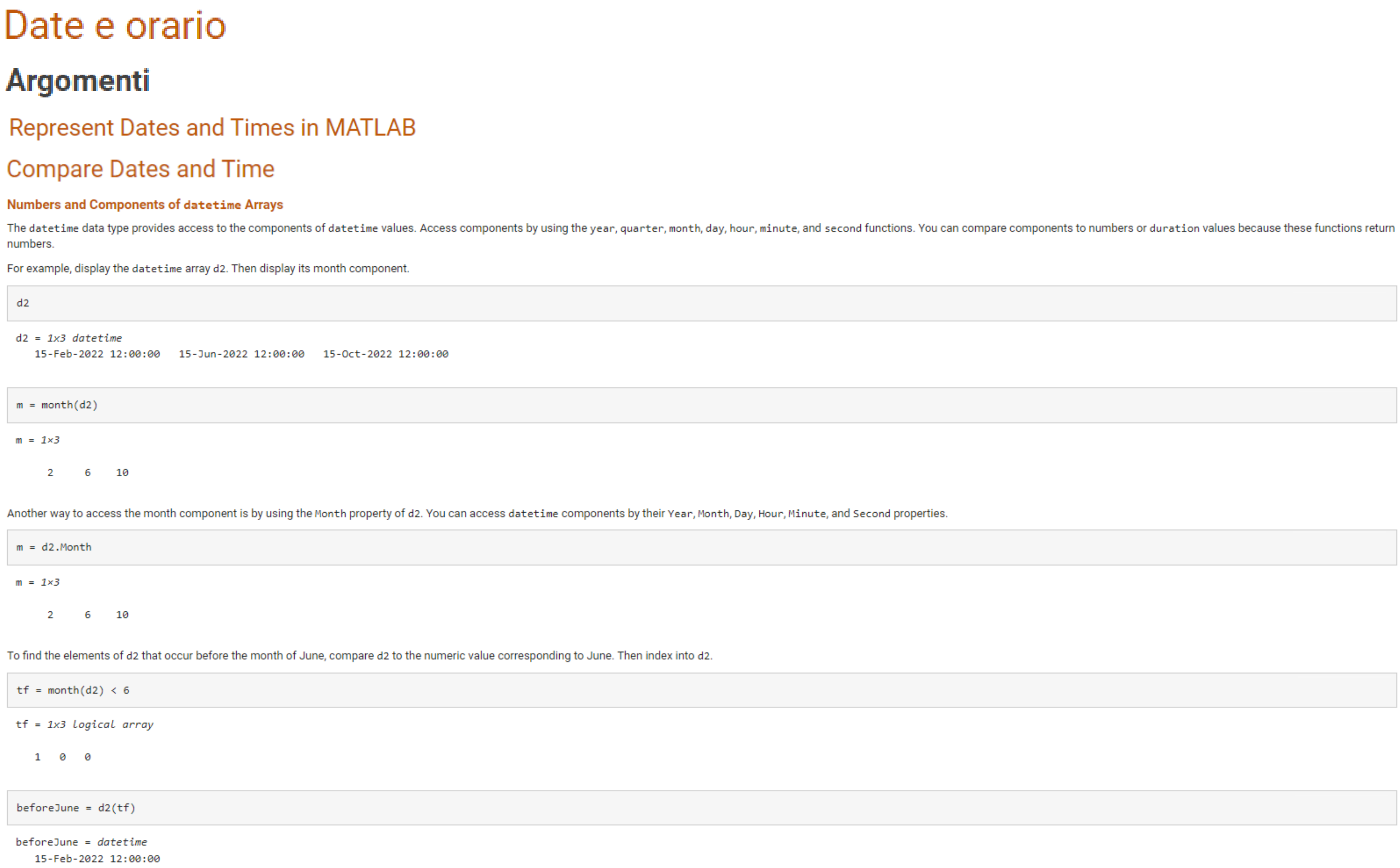
```
tf = 1x3 logical array

   0   1   1
```

```
afterJune1 = d2(tf)
```

```
afterJune1 = 1x2 datetime
   15-Jun-2022 12:00:00   15-Oct-2022 12:00:00
```

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Compare Dates and Time

Numbers and Components of `datetime` Arrays

The `datetime` data type provides access to the components of `datetime` values. Access components by using the `year`, `quarter`, `month`, `day`, `hour`, `minute`, and `second` functions. You can compare components to numbers or `duration` values because these functions return numbers.

For example, display the `datetime` array `d2`. Then display its month component.

```
d2
```

```
d2 = 1x3 datetime
   15-Feb-2022 12:00:00   15-Jun-2022 12:00:00   15-Oct-2022 12:00:00
```

```
m = month(d2)
```

```
m = 1×3

     2     6    10
```

Another way to access the month component is by using the `Month` property of `d2`. You can access `datetime` components by their `Year`, `Month`, `Day`, `Hour`, `Minute`, and `Second` properties.

```
m = d2.Month
```

```
m = 1×3

     2     6    10
```

To find the elements of `d2` that occur before the month of June, compare `d2` to the numeric value corresponding to June. Then index into `d2`.

```
tf = month(d2) < 6
```

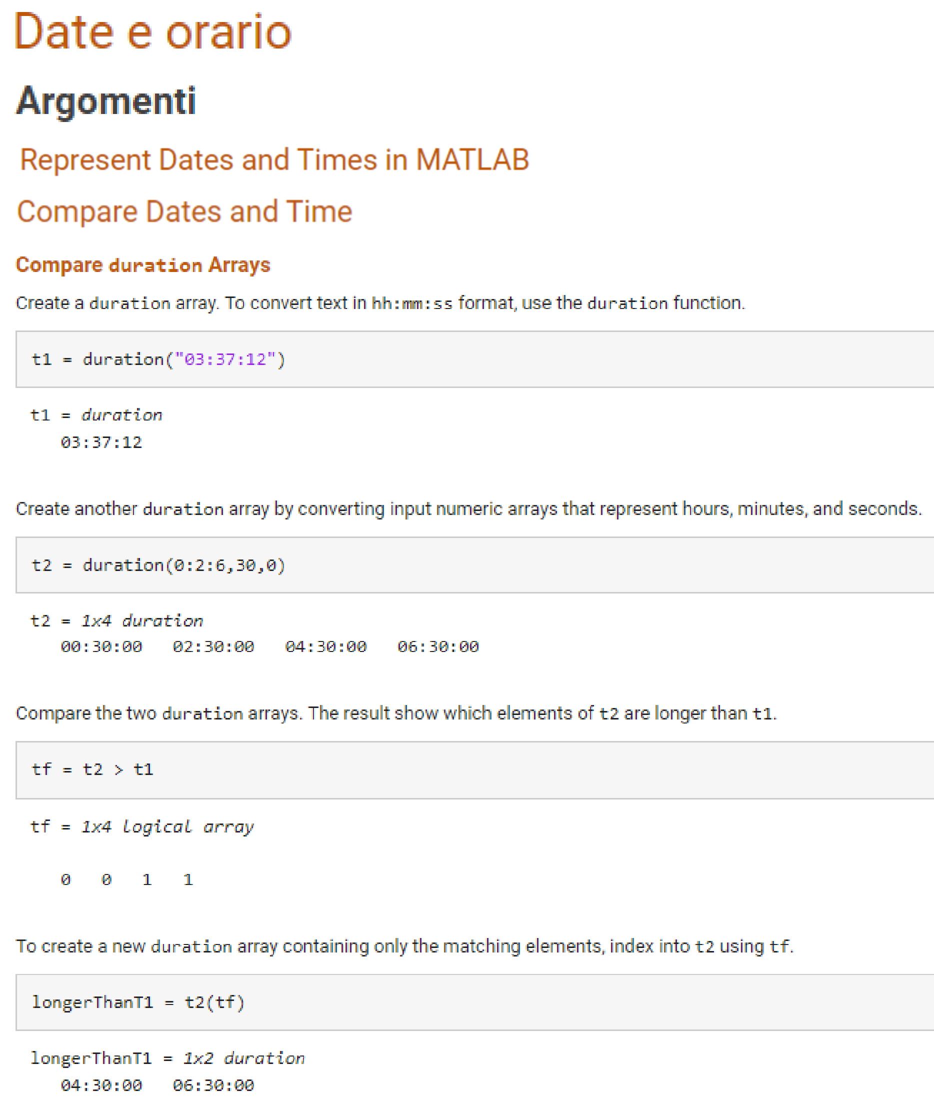
```
tf = 1x3 logical array

   1   0   0
```

```
beforeJune = d2(tf)
```

```
beforeJune = datetime
   15-Feb-2022 12:00:00
```

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Compare Dates and Time

Compare duration Arrays

Create a duration array. To convert text in hh:mm:ss format, use the duration function.

```
t1 = duration("03:37:12")
```

```
t1 = duration
   03:37:12
```

Create another duration array by converting input numeric arrays that represent hours, minutes, and seconds.

```
t2 = duration(0:2:6,30,0)
```

```
t2 = 1x4 duration
   00:30:00   02:30:00   04:30:00   06:30:00
```

Compare the two duration arrays. The result show which elements of t2 are longer than t1.

```
tf = t2 > t1
```

```
tf = 1x4 logical array

   0   0   1   1
```

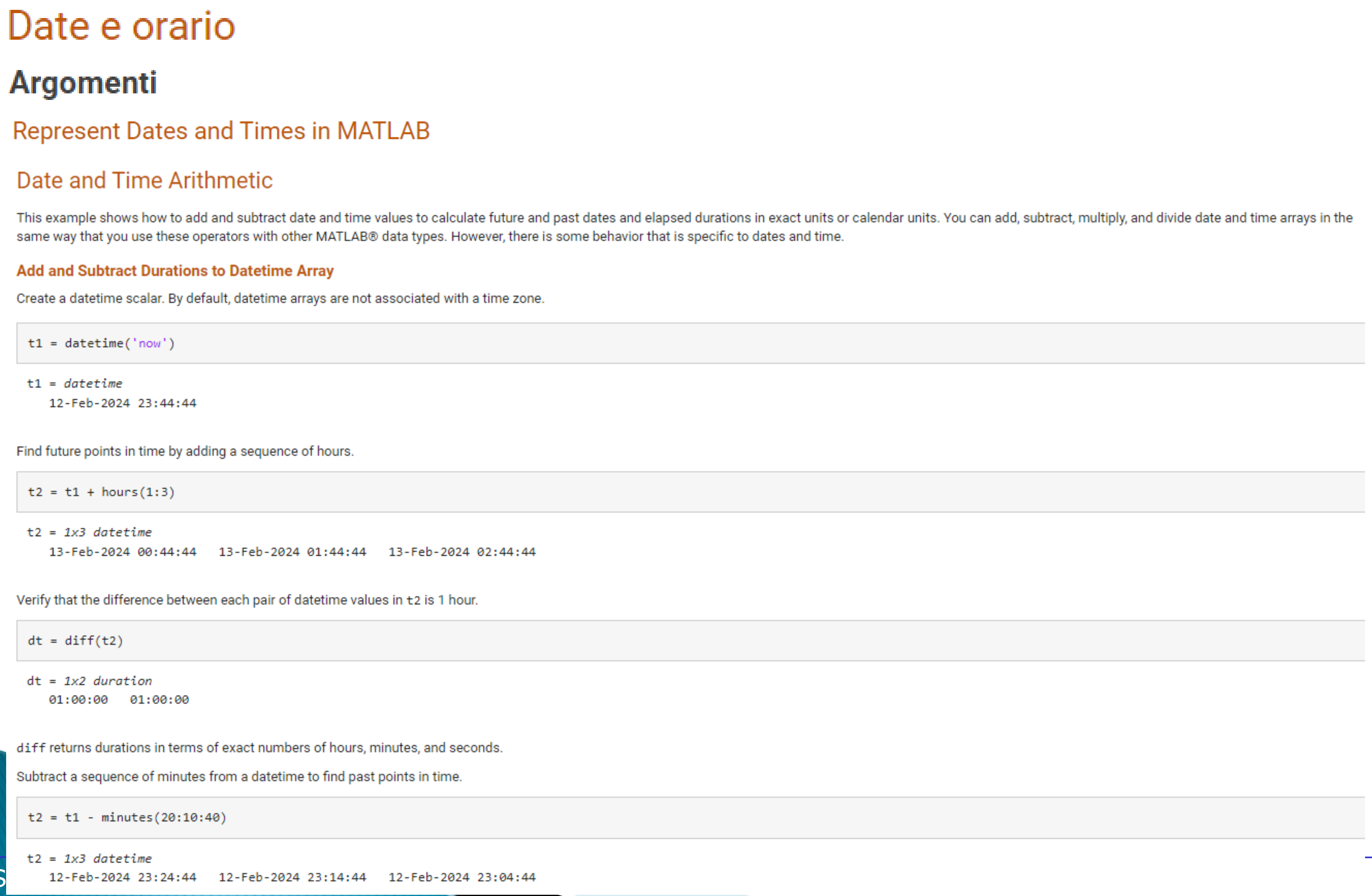
To create a new duration array containing only the matching elements, index into t2 using tf.

```
longerThanT1 = t2(tf)
```

```
longerThanT1 = 1x2 duration
   04:30:00   06:30:00
```

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Date and Time Arithmetic

This example shows how to add and subtract date and time values to calculate future and past dates and elapsed durations in exact units or calendar units. You can add, subtract, multiply, and divide date and time arrays in the same way that you use these operators with other MATLAB® data types. However, there is some behavior that is specific to dates and time.

Add and Subtract Durations to Datetime Array

Create a datetime scalar. By default, datetime arrays are not associated with a time zone.

```
t1 = datetime('now')
```

```
t1 = datetime
   12-Feb-2024 23:44:44
```

Find future points in time by adding a sequence of hours.

```
t2 = t1 + hours(1:3)
```

```
t2 = 1x3 datetime
   13-Feb-2024 00:44:44   13-Feb-2024 01:44:44   13-Feb-2024 02:44:44
```

Verify that the difference between each pair of datetime values in `t2` is 1 hour.

```
dt = diff(t2)
```

```
dt = 1x2 duration
   01:00:00   01:00:00
```

`diff` returns durations in terms of exact numbers of hours, minutes, and seconds.

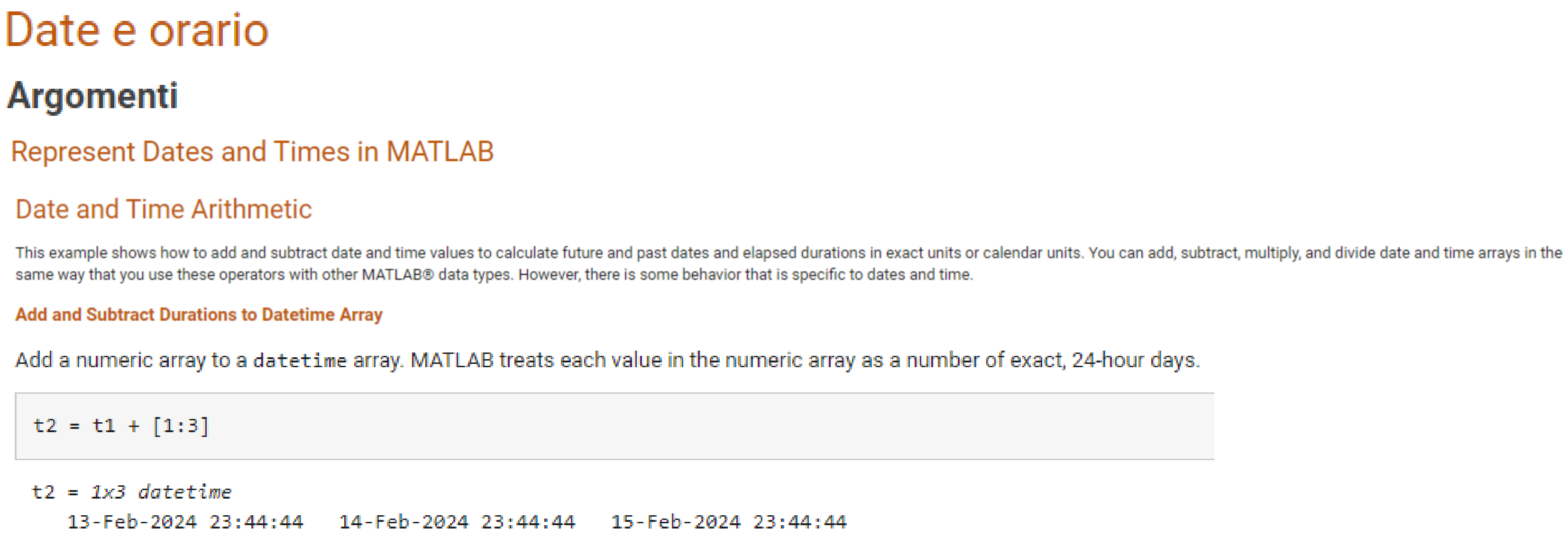
Subtract a sequence of minutes from a datetime to find past points in time.

```
t2 = t1 - minutes(20:10:40)
```

```
t2 = 1x3 datetime
   12-Feb-2024 23:24:44   12-Feb-2024 23:14:44   12-Feb-2024 23:04:44
```

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Date and Time Arithmetic

This example shows how to add and subtract date and time values to calculate future and past dates and elapsed durations in exact units or calendar units. You can add, subtract, multiply, and divide date and time arrays in the same way that you use these operators with other MATLAB® data types. However, there is some behavior that is specific to dates and time.

Add and Subtract Durations to Datetime Array

Add a numeric array to a `datetime` array. MATLAB treats each value in the numeric array as a number of exact, 24-hour days.

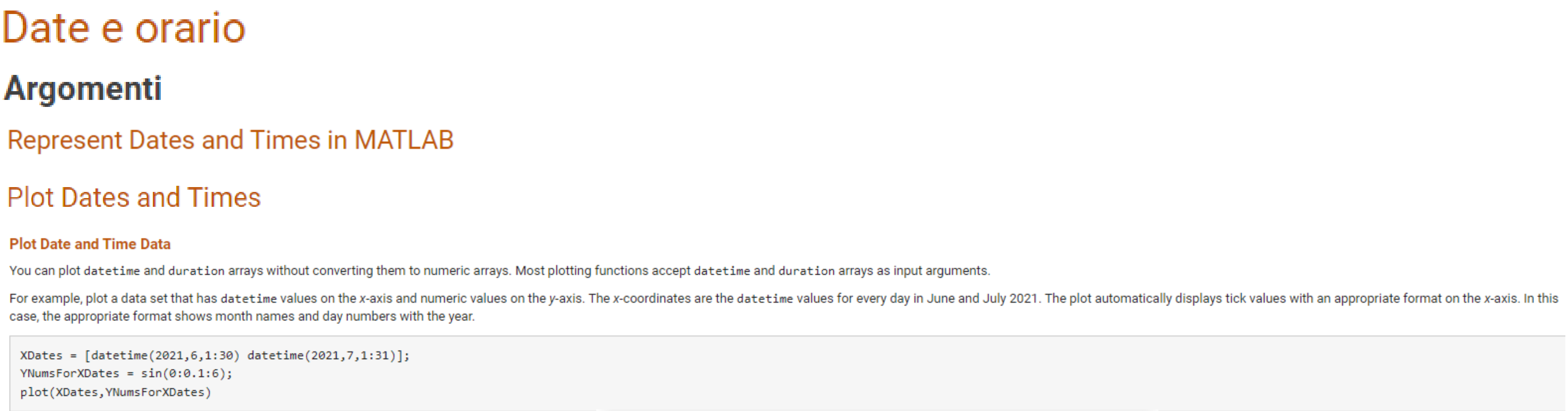
```
t2 = t1 + [1:3]
```

```
t2 = 1x3 datetime
   13-Feb-2024 23:44:44   14-Feb-2024 23:44:44   15-Feb-2024 23:44:44
```

ESERCITAZIONI
MATLAB – Nozioni fondamentali

MathWorks®

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

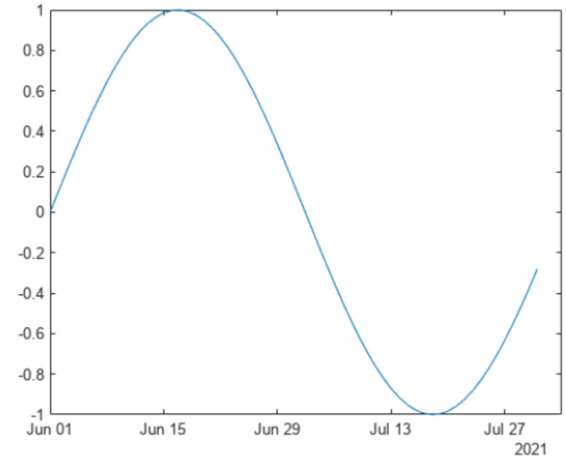
Plot Dates and Times

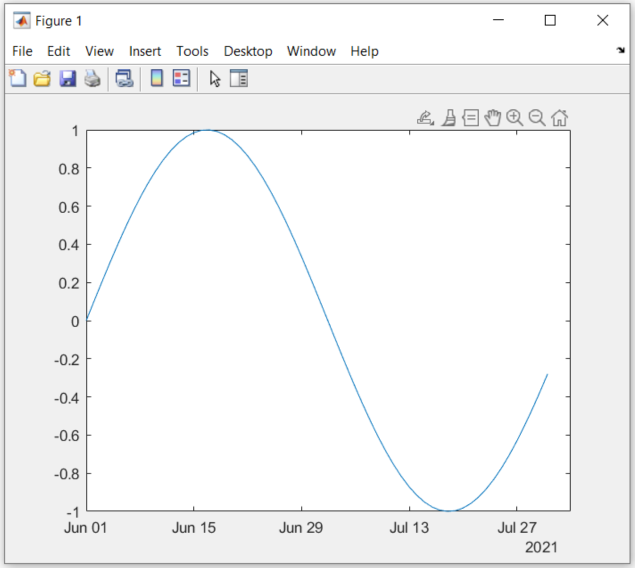
Plot Date and Time Data

You can plot `datetime` and `duration` arrays without converting them to numeric arrays. Most plotting functions accept `datetime` and `duration` arrays as input arguments.

For example, plot a data set that has `datetime` values on the *x*-axis and numeric values on the *y*-axis. The *x*-coordinates are the `datetime` values for every day in June and July 2021. The plot automatically displays tick values with an appropriate format on the *x*-axis. In this case, the appropriate format shows month names and day numbers with the year.

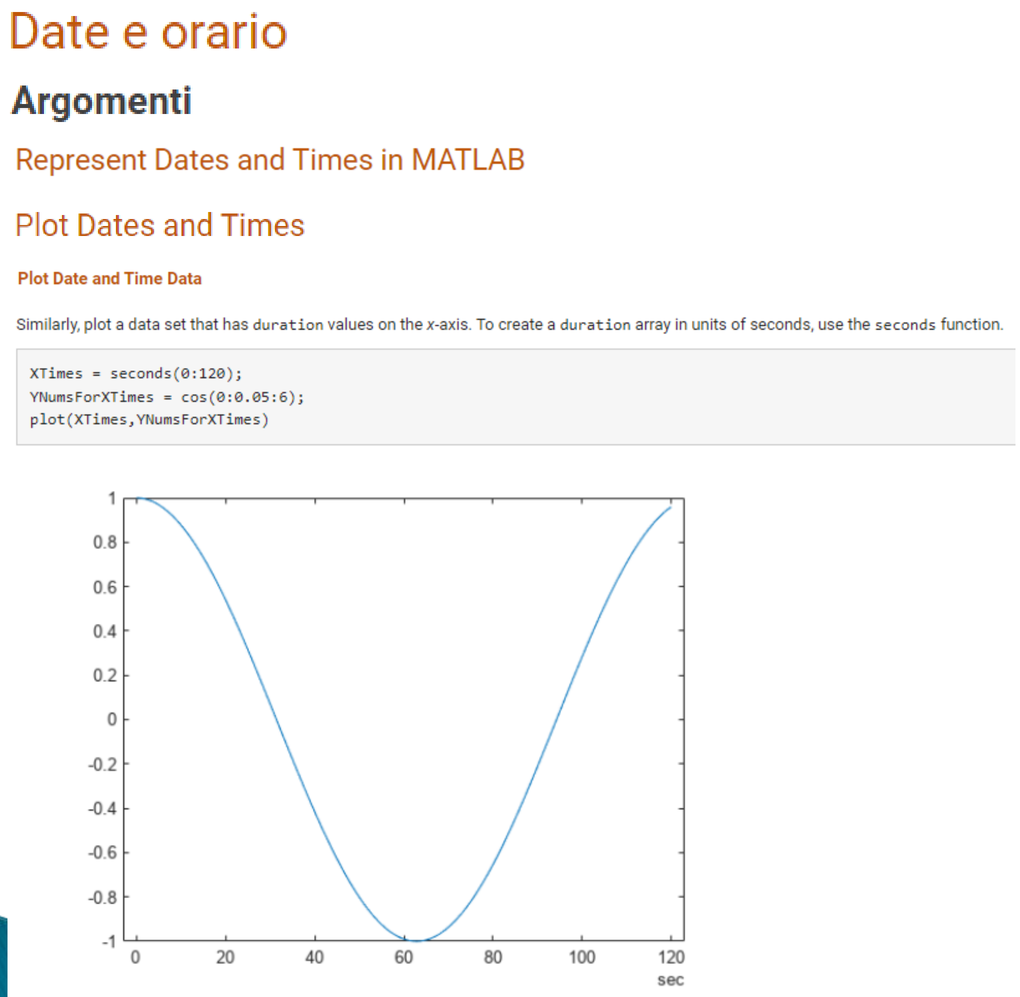
```
XDates = [datetime(2021,6,1:30) datetime(2021,7,1:31)];
YNumsForXDates = sin(0:0.1:6);
plot(XDates,YNumsForXDates)
```





ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Plot Dates and Times

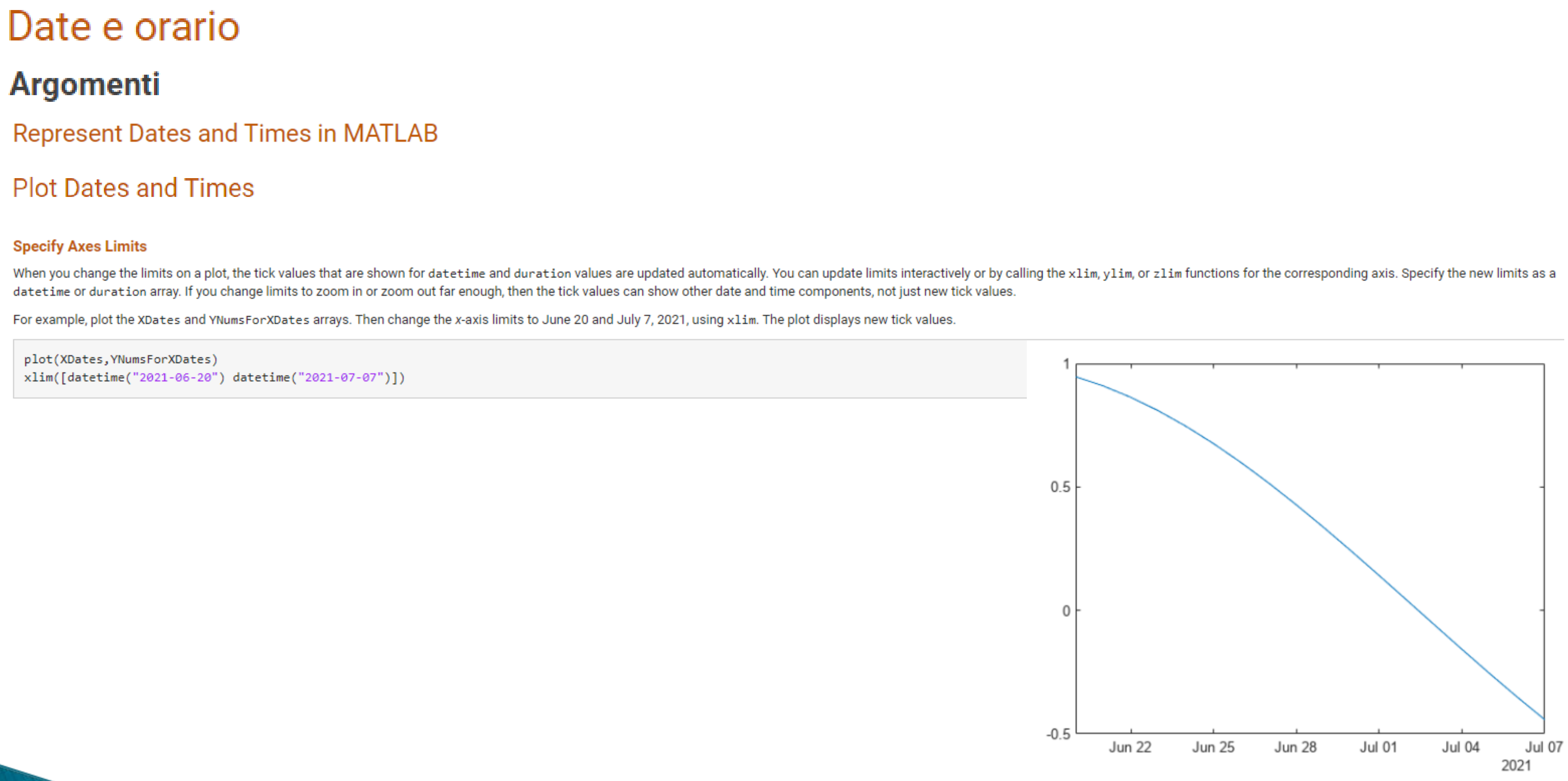
Plot Date and Time Data

Similarly, plot a data set that has `duration` values on the *x*-axis. To create a `duration` array in units of seconds, use the `seconds` function.

```
XTimes = seconds(0:120);
YNumsForXTimes = cos(0:0.05:6);
plot(XTimes,YNumsForXTimes)
```

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Plot Dates and Times

Specify Axes Limits

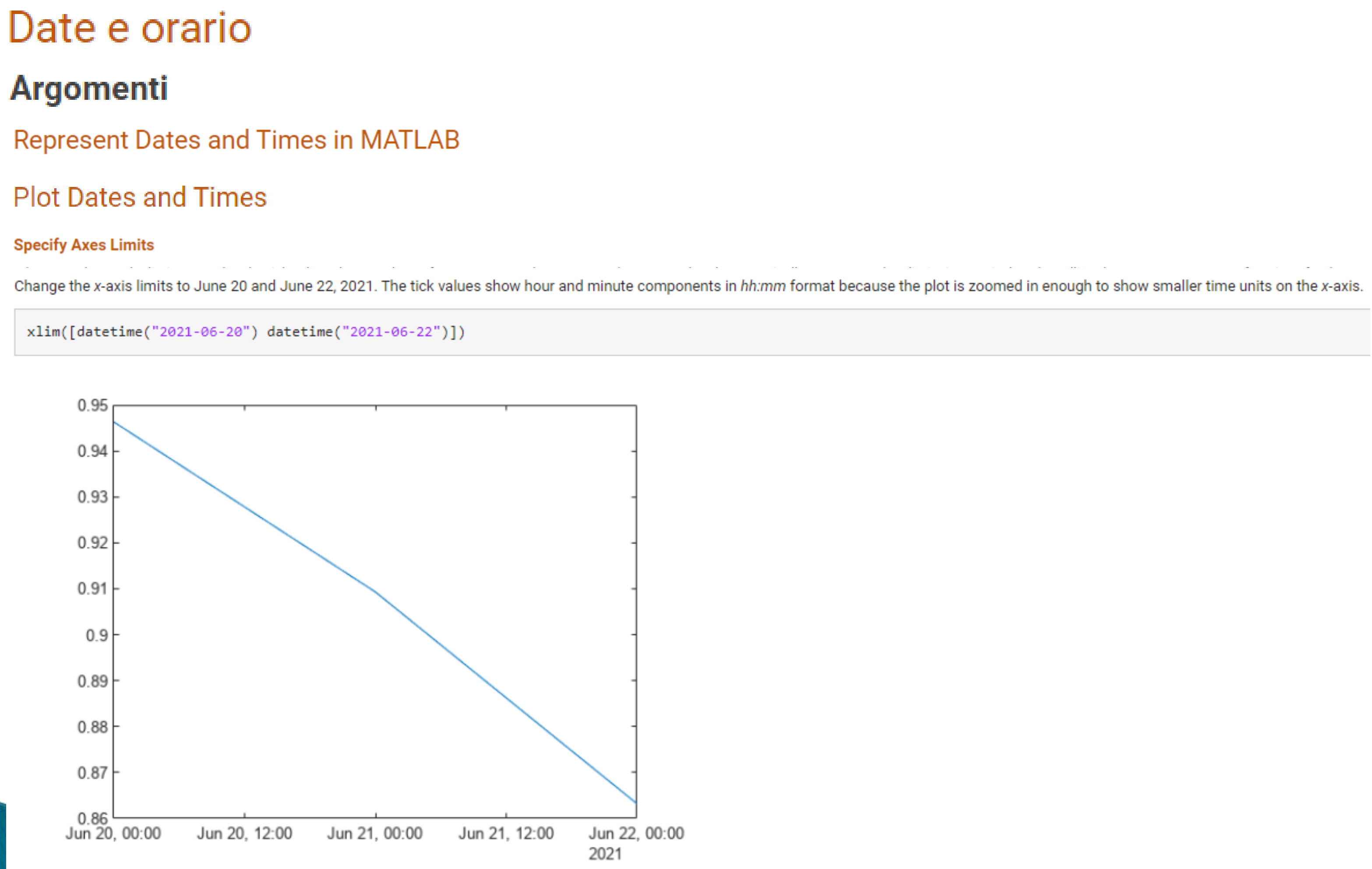
When you change the limits on a plot, the tick values that are shown for `datetime` and `duration` values are updated automatically. You can update limits interactively or by calling the `xlim`, `ylim`, or `zlim` functions for the corresponding axis. Specify the new limits as a `datetime` or `duration` array. If you change limits to zoom in or zoom out far enough, then the tick values can show other date and time components, not just new tick values.

For example, plot the `XDates` and `YNumsForXDates` arrays. Then change the *x*-axis limits to June 20 and July 7, 2021, using `xlim`. The plot displays new tick values.

```
plot(XDates,YNumsForXDates)
xlim([datetime("2021-06-20") datetime("2021-07-07")])
```

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

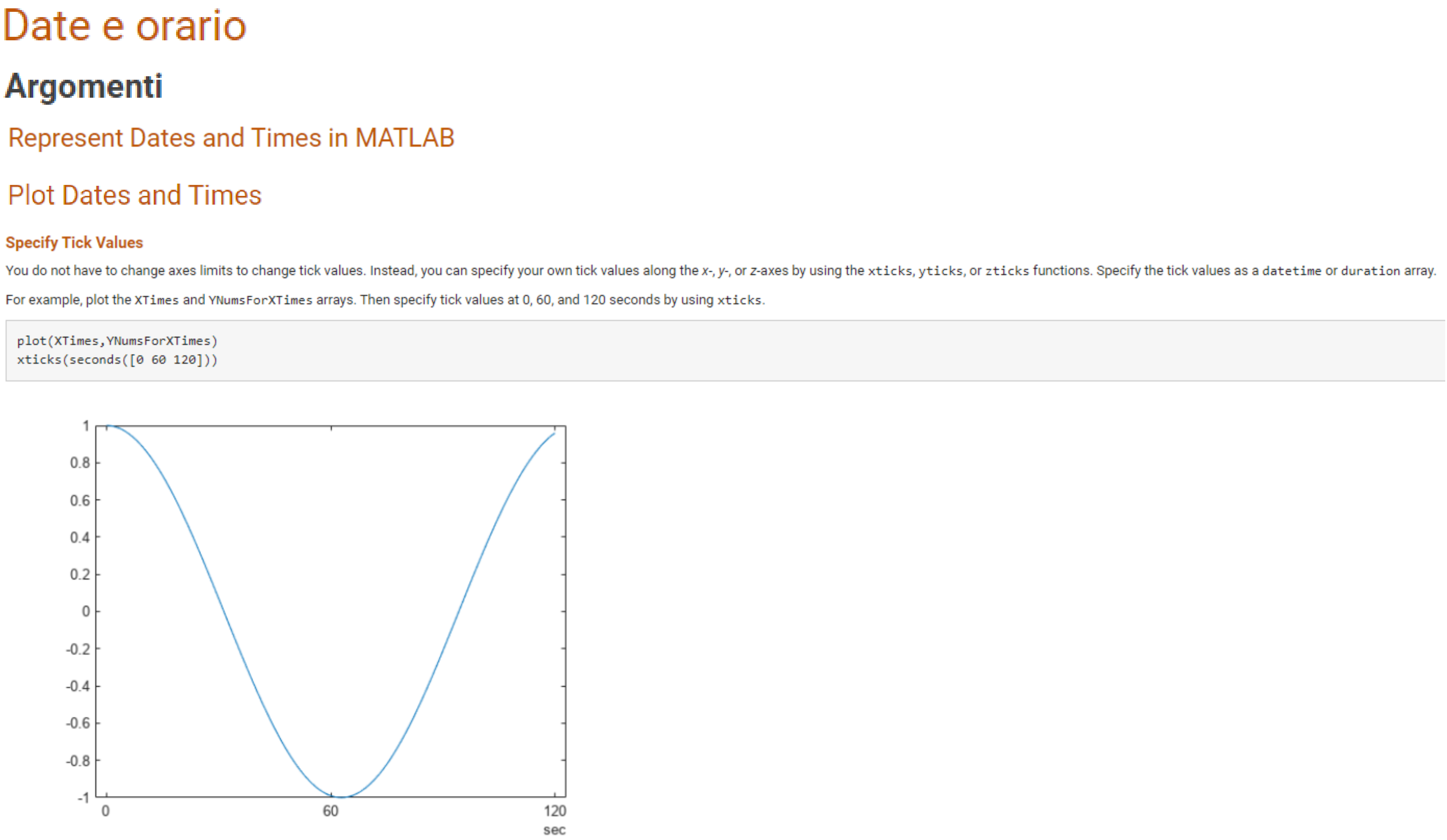
Plot Dates and Times

Specify Axes Limits

Change the *x*-axis limits to June 20 and June 22, 2021. The tick values show hour and minute components in *hh:mm* format because the plot is zoomed in enough to show smaller time units on the *x*-axis.

```
xlim([datetime("2021-06-20") datetime("2021-06-22")])
```

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Plot Dates and Times

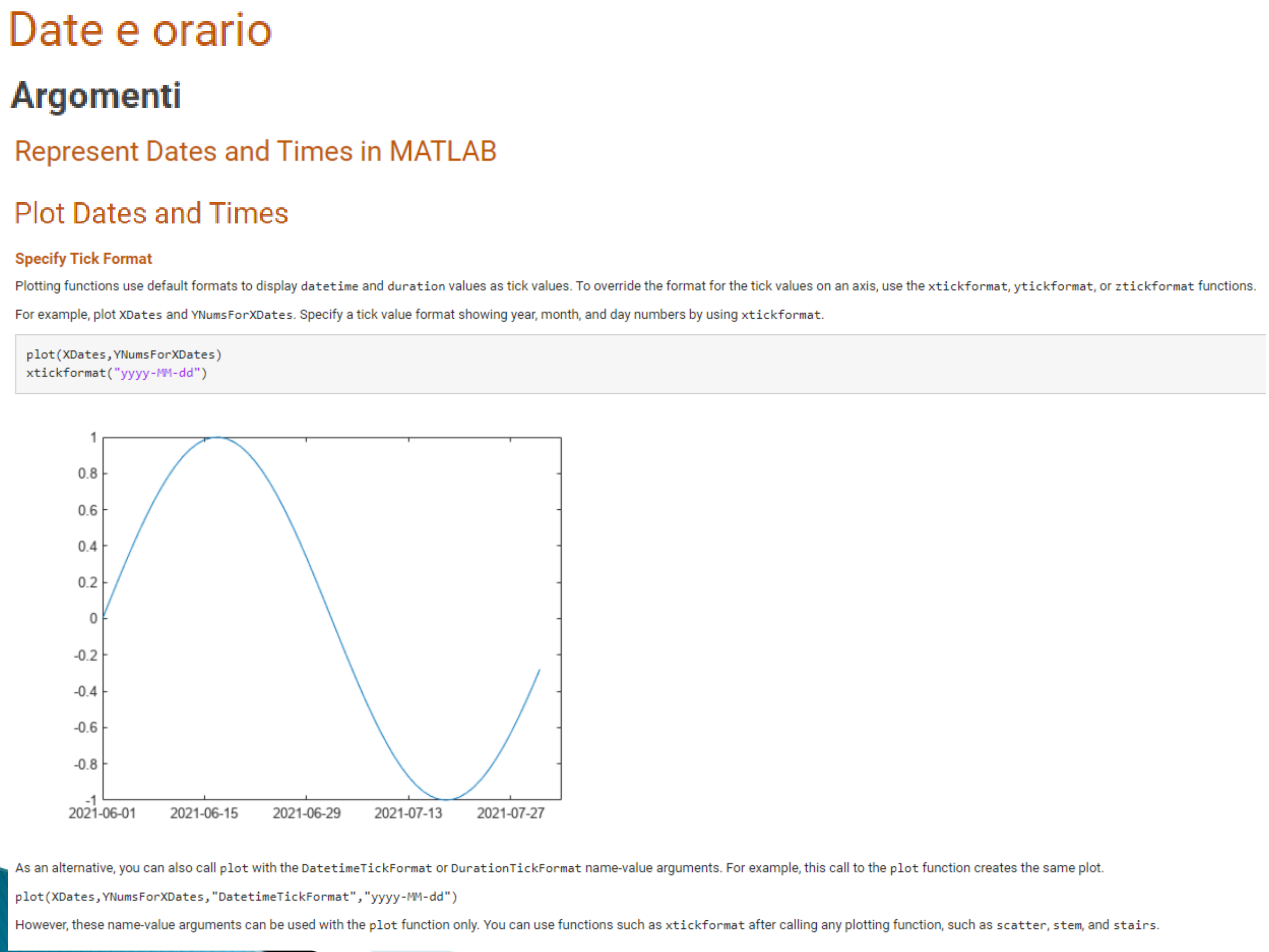
Specify Tick Values

You do not have to change axes limits to change tick values. Instead, you can specify your own tick values along the *x*-, *y*-, or *z*-axes by using the `xticks`, `yticks`, or `zticks` functions. Specify the tick values as a `datetime` or `duration` array.

For example, plot the `XTimes` and `YNumsForXTimes` arrays. Then specify tick values at 0, 60, and 120 seconds by using `xticks`.

```
plot(XTimes,YNumsForXTimes)
xticks(seconds([0 60 120]))
```

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Plot Dates and Times

Specify Tick Format

Plotting functions use default formats to display `datetime` and `duration` values as tick values. To override the format for the tick values on an axis, use the `xtickformat`, `ytickformat`, or `ztickformat` functions.

For example, plot `XDates` and `YNumsForXDates`. Specify a tick value format showing year, month, and day numbers by using `xtickformat`.

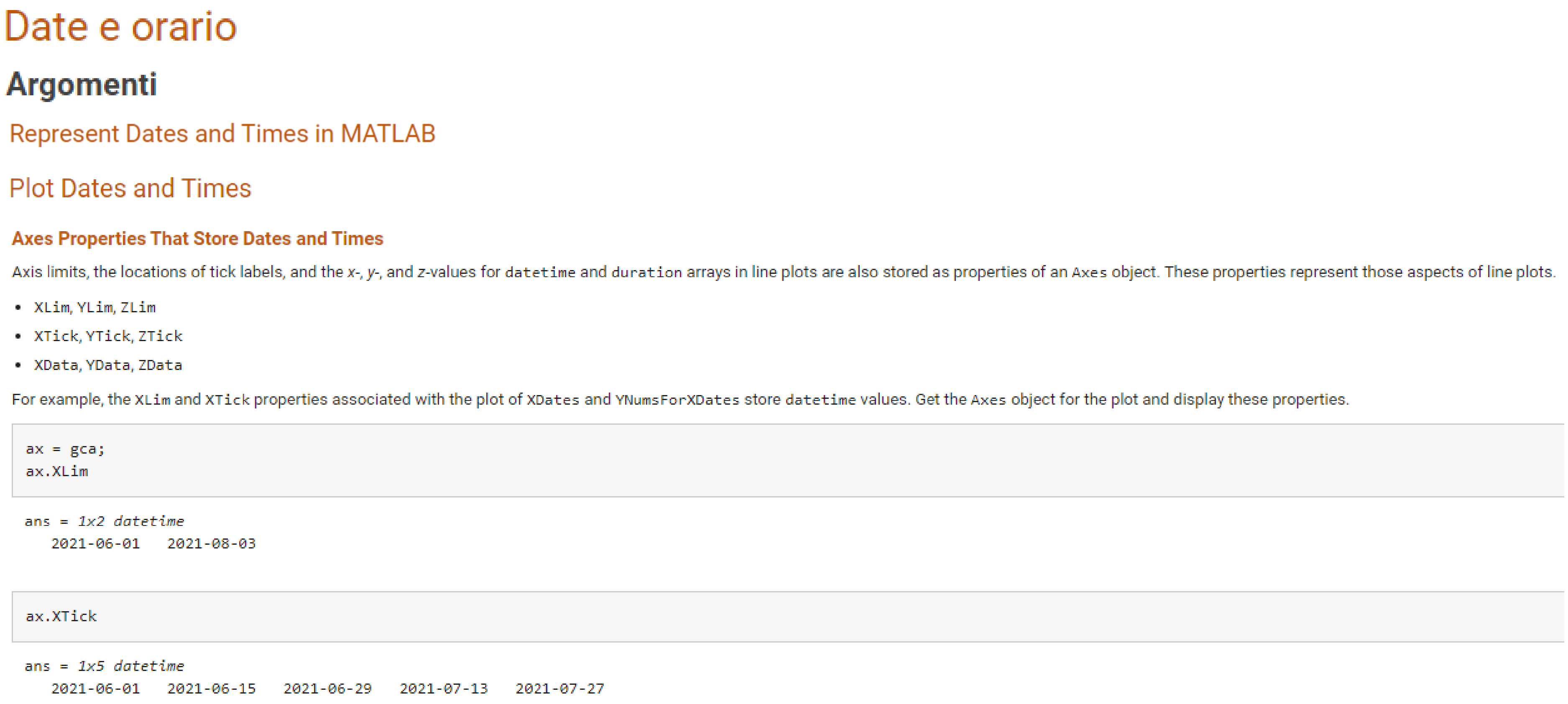
```
plot(XDates,YNumsForXDates)
xtickformat("yyyy-MM-dd")
```

As an alternative, you can also call `plot` with the `DatetimeTickFormat` or `DurationTickFormat` name-value arguments. For example, this call to the `plot` function creates the same plot.

```
plot(XDates,YNumsForXDates,"DatetimeTickFormat","yyyy-MM-dd")
```

However, these name-value arguments can be used with the `plot` function only. You can use functions such as `xtickformat` after calling any plotting function, such as `scatter`, `stem`, and `stairs`.

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Plot Dates and Times

Axes Properties That Store Dates and Times

Axis limits, the locations of tick labels, and the *x*-, *y*-, and *z*-values for `datetime` and `duration` arrays in line plots are also stored as properties of an `Axes` object. These properties represent those aspects of line plots.

- `XLim`, `YLim`, `ZLim`
- `XTick`, `YTick`, `ZTick`
- `XData`, `YData`, `ZData`

For example, the `XLim` and `XTick` properties associated with the plot of `XDates` and `YNumsForXDates` store `datetime` values. Get the `Axes` object for the plot and display these properties.

```
ax = gca;
ax.XLim
```

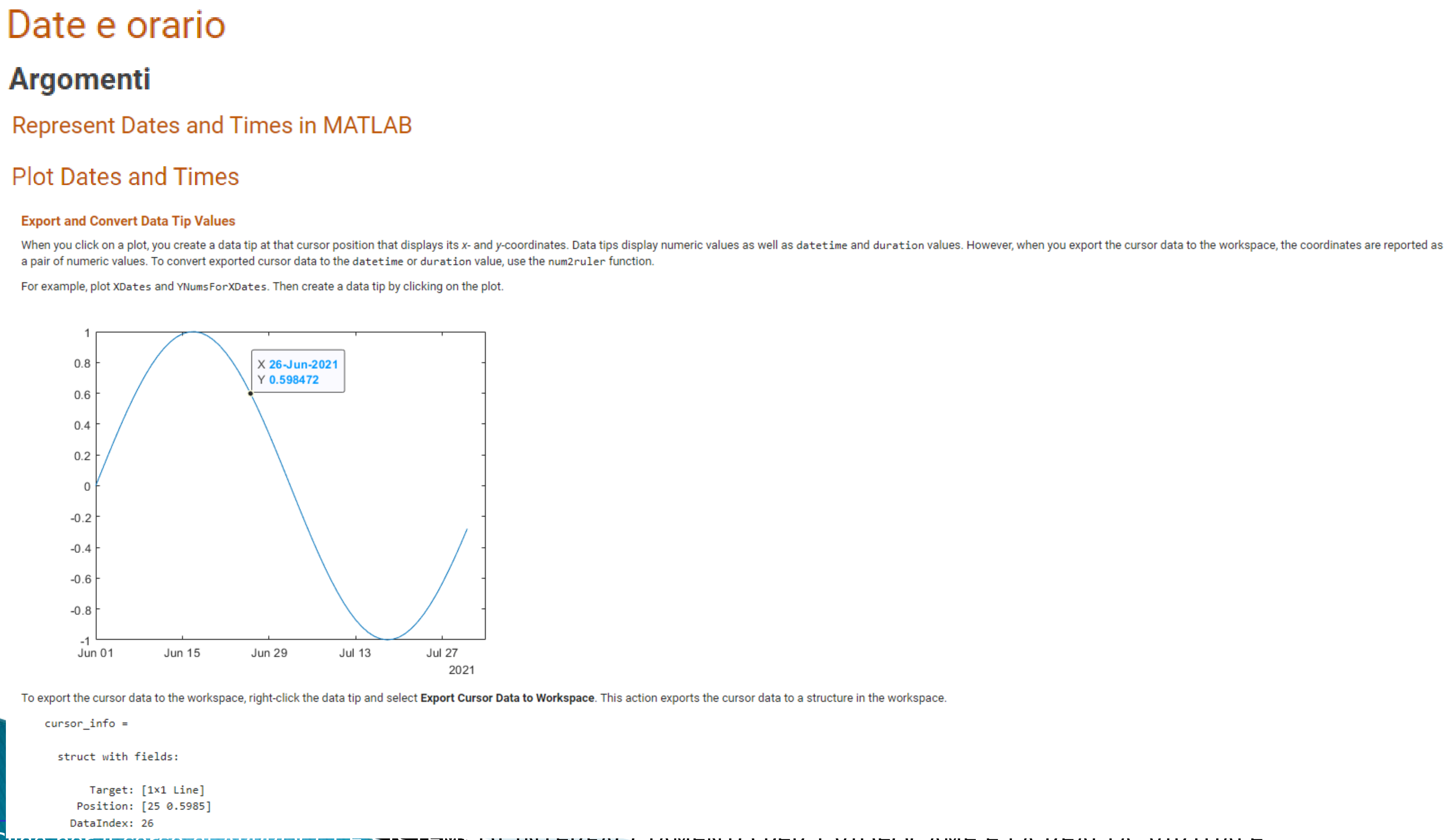
```
ans = 1x2 datetime
   2021-06-01   2021-08-03
```

```
ax.XTick
```

```
ans = 1x5 datetime
   2021-06-01   2021-06-15   2021-06-29   2021-07-13   2021-07-27
```

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Plot Dates and Times

Export and Convert Data Tip Values

When you click on a plot, you create a data tip at that cursor position that displays its *x*- and *y*-coordinates. Data tips display numeric values as well as `datetime` and `duration` values. However, when you export the cursor data to the workspace, the coordinates are reported as a pair of numeric values. To convert exported cursor data to the `datetime` or `duration` value, use the `num2ruler` function.

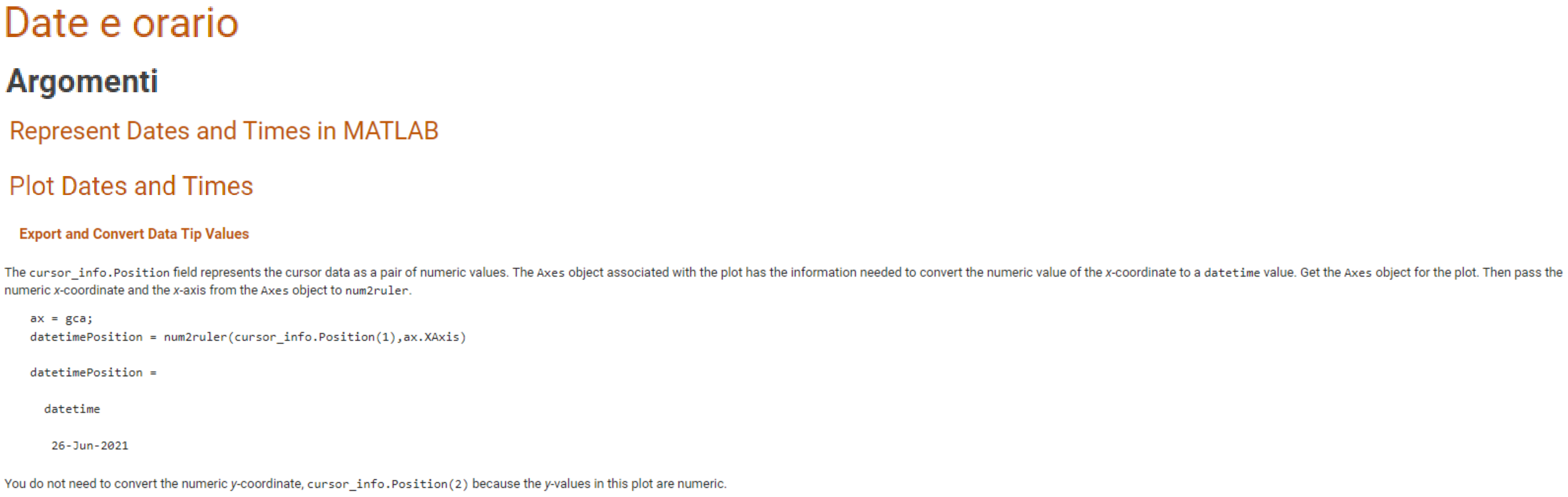
For example, plot `XDates` and `YNumsForXDates`. Then create a data tip by clicking on the plot.

To export the cursor data to the workspace, right-click the data tip and select **Export Cursor Data to Workspace**. This action exports the cursor data to a structure in the workspace.

```
cursor_info = 

  struct with fields:

       Target: [1×1 Line]
     Position: [25 0.5985]
    DataIndex: 26
```

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Plot Dates and Times

Export and Convert Data Tip Values

The `cursor_info.Position` field represents the cursor data as a pair of numeric values. The `Axes` object associated with the plot has the information needed to convert the numeric value of the *x*-coordinate to a `datetime` value. Get the `Axes` object for the plot. Then pass the numeric *x*-coordinate and the *x*-axis from the `Axes` object to `num2ruler`.

```
ax = gca;
datetimePosition = num2ruler(cursor_info.Position(1),ax.XAxis)

datetimePosition = 

  datetime

   26-Jun-2021
```

You do not need to convert the numeric *y*-coordinate, `cursor_info.Position(2)` because the *y*-values in this plot are numeric.

ESERCITAZIONI
MATLAB – Nozioni fondamentali

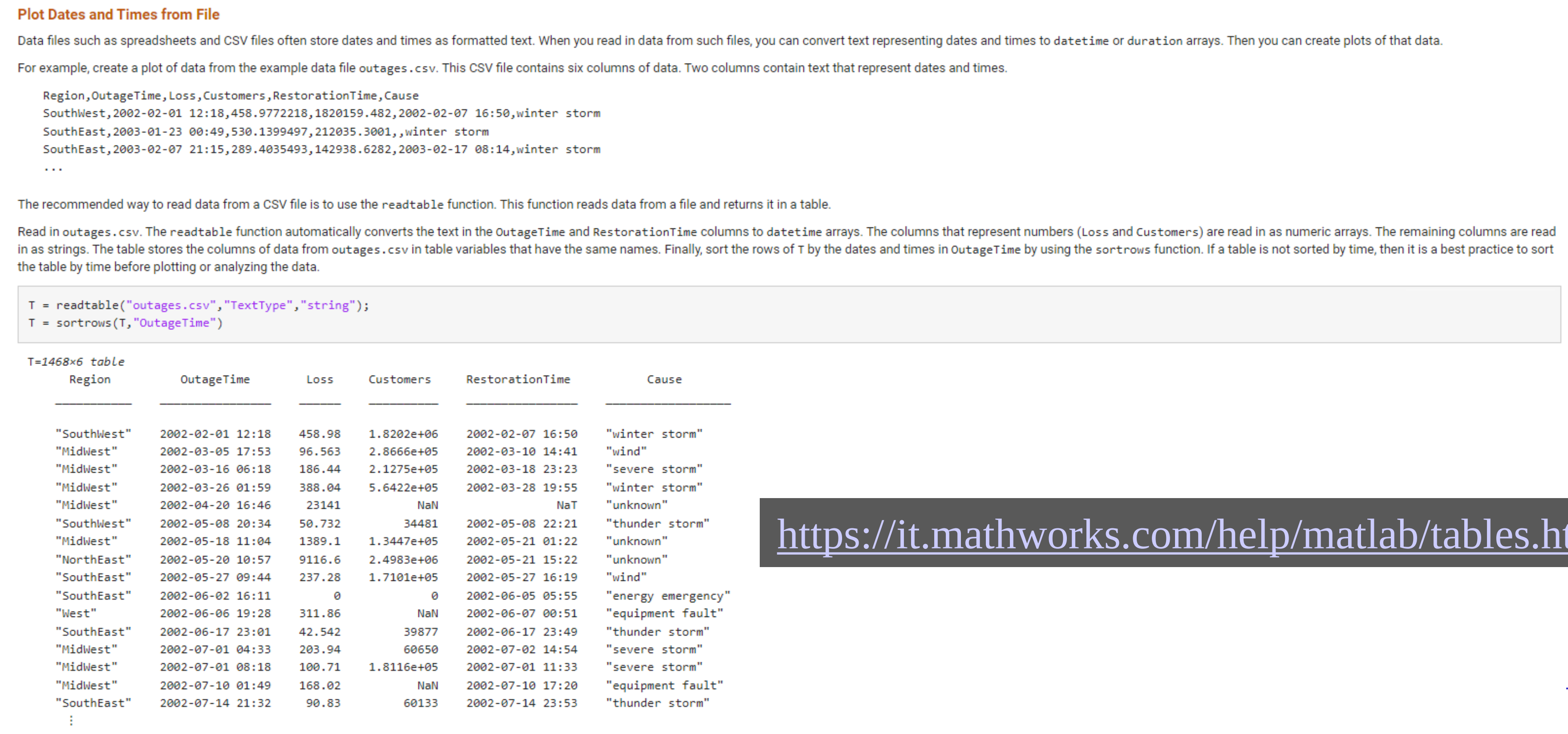
Tipi di dati

Date e orario

Argomenti

Represent Dates and Times in MATLAB

Plot Dates and Times



Plot Dates and Times from File

Data files such as spreadsheets and CSV files often store dates and times as formatted text. When you read in data from such files, you can convert text representing dates and times to `datetime` or `duration` arrays. Then you can create plots of that data.

For example, create a plot of data from the example data file `outages.csv`. This CSV file contains six columns of data. Two columns contain text that represent dates and times.

```
Region,OutageTime,Loss,Customers,RestorationTime,Cause
SouthWest,2002-02-01 12:18,458.9772218,1820159.482,2002-02-07 16:50,winter storm
SouthEast,2003-01-23 00:49,530.1399497,212035.3001,,winter storm
SouthEast,2003-02-07 21:15,289.4035493,142938.6282,2003-02-17 08:14,winter storm
...
```

The recommended way to read data from a CSV file is to use the `readtable` function. This function reads data from a file and returns it in a table.

Read in `outages.csv`. The `readtable` function automatically converts the text in the `OutageTime` and `RestorationTime` columns to `datetime` arrays. The columns that represent numbers (`Loss` and `Customers`) are read in as numeric arrays. The remaining columns are read in as strings. The table stores the columns of data from `outages.csv` in table variables that have the same names. Finally, sort the rows of `T` by the dates and times in `OutageTime` by using the `sortrows` function. If a table is not sorted by time, then it is a best practice to sort the table by time before plotting or analyzing the data.

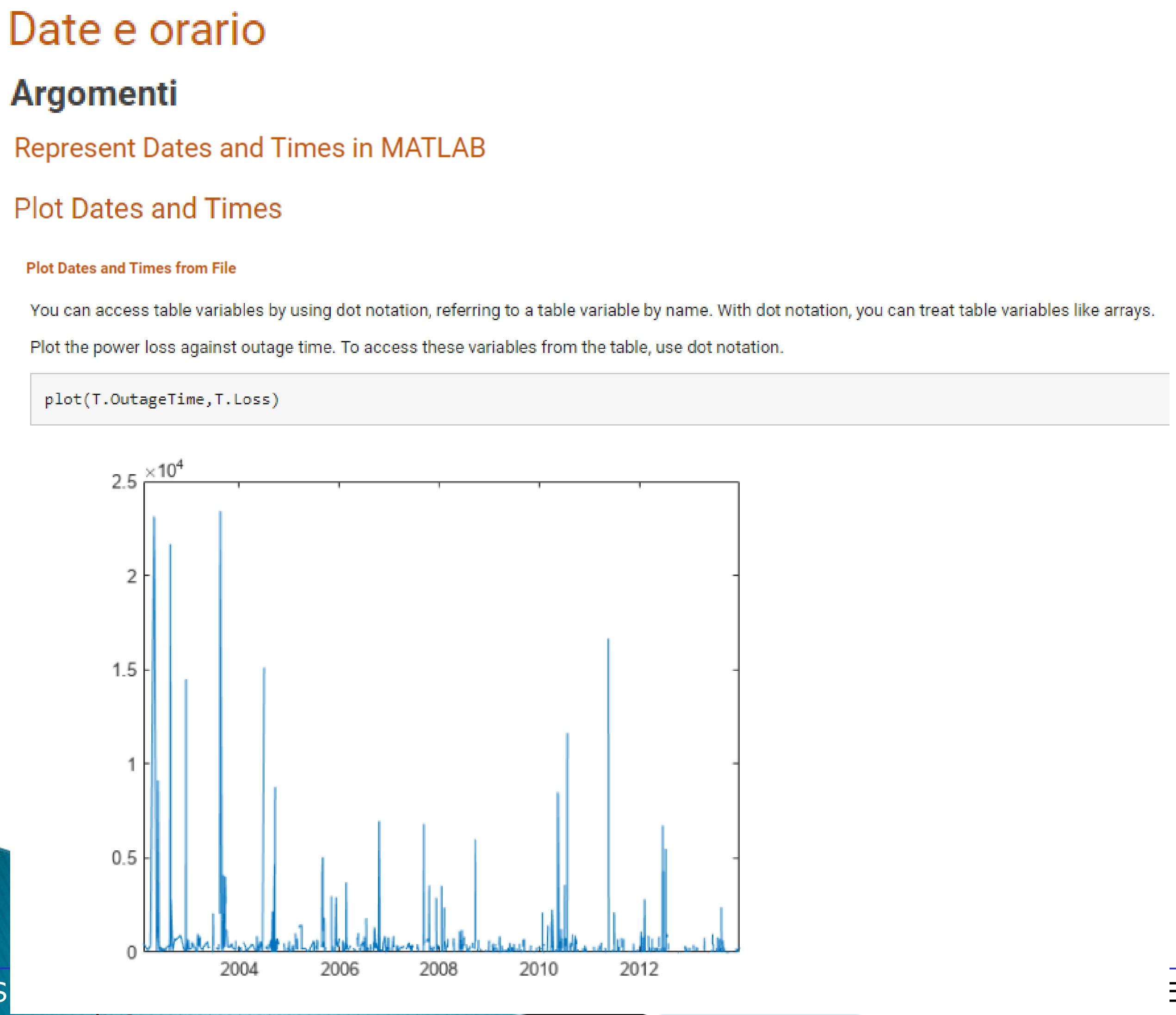
```
T = readtable("outages.csv","TextType","string");
T = sortrows(T,"OutageTime")
```

```
T=1468×6 table
       Region          OutageTime        Loss     Customers       RestorationTime            Cause
    ___________    ________________    ______    __________    ________________    __________________

    "SouthWest"    2002-02-01 12:18    458.98    1.8202e+06    2002-02-07 16:50    "winter storm"
    "MidWest"      2002-03-05 17:53    96.563    2.8666e+05    2002-03-10 14:41    "wind"
    "MidWest"      2002-03-16 06:18    186.44    2.1275e+05    2002-03-18 23:23    "severe storm"
    "MidWest"      2002-03-26 01:59    388.04    5.6422e+05    2002-03-28 19:55    "winter storm"
    "MidWest"      2002-04-20 16:46     23141           NaN                 NaT    "unknown"
    "SouthWest"    2002-05-08 20:34    50.732         34481    2002-05-08 22:21    "thunder storm"
    "MidWest"      2002-05-18 11:04    1389.1    1.3447e+05    2002-05-21 01:22    "unknown"
    "NorthEast"    2002-05-20 10:57    9116.6    2.4983e+06    2002-05-21 15:22    "unknown"
    "SouthEast"    2002-05-27 09:44    237.28    1.7101e+05    2002-05-27 16:19    "wind"
    "SouthEast"    2002-06-02 16:11         0             0    2002-06-05 05:55    "energy emergency"
    "West"         2002-06-06 19:28    311.86           NaN    2002-06-07 00:51    "equipment fault"
    "SouthEast"    2002-06-17 23:01    42.542         39877    2002-06-17 23:49    "thunder storm"
    "MidWest"      2002-07-01 04:33    203.94         60650    2002-07-02 14:54    "severe storm"
    "MidWest"      2002-07-01 08:18    100.71    1.8116e+05    2002-07-01 11:33    "severe storm"
    "MidWest"      2002-07-10 01:49    168.02           NaN    2002-07-10 17:20    "equipment fault"
    "SouthEast"    2002-07-14 21:32     90.83         60133    2002-07-14 23:53    "thunder storm"
      ⋮
```

https://it.mathworks.com/help/matlab/tables.html

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Plot Dates and Times

Plot Dates and Times from File

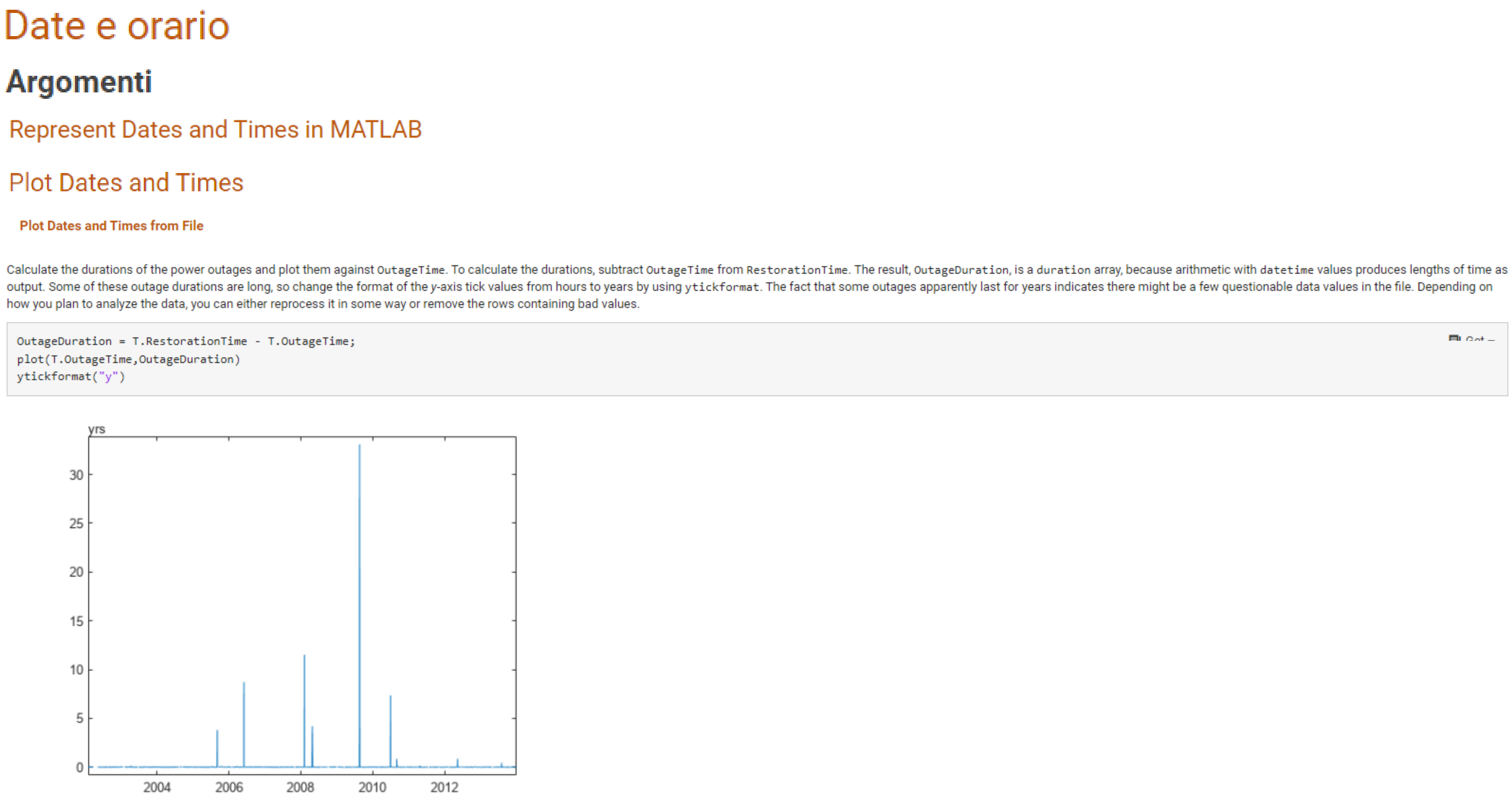
You can access table variables by using dot notation, referring to a table variable by name. With dot notation, you can treat table variables like arrays.

Plot the power loss against outage time. To access these variables from the table, use dot notation.

```
plot(T.OutageTime,T.Loss)
```

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Tipi di dati



Date e orario

Argomenti

Represent Dates and Times in MATLAB

Plot Dates and Times

Plot Dates and Times from File

Calculate the durations of the power outages and plot them against `OutageTime`. To calculate the durations, subtract `OutageTime` from `RestorationTime`. The result, `OutageDuration`, is a `duration` array, because arithmetic with `datetime` values produces lengths of time as output. Some of these outage durations are long, so change the format of the *y*-axis tick values from hours to years by using `ytickformat`. The fact that some outages apparently last for years indicates there might be a few questionable data values in the file. Depending on how you plan to analyze the data, you can either reprocess it in some way or remove the rows containing bad values.

```
OutageDuration = T.RestorationTime - T.OutageTime;
plot(T.OutageTime,OutageDuration)
ytickformat("y")
```

https://it.mathworks.com/help/matlab/matlab_prog/set-display-format-of-date-and-time-arrays.html

ESERCITAZIONI
MATLAB – Nozioni fondamentali

Operatori e operazioni elementari

https://it.mathworks.com/help/matlab/operators-and-elementary-operations.html

Loop e dichiarazioni condizionali

https://it.mathworks.com/help/matlab/control-flow.html

Riferimenti Bibliografici

[1] https://it.mathworks.com