Time Series Database Interface: R fame (TSfame)

December 23, 2010

1 Introduction

The code from the vignette that generates this guide can be loaded into an editor with *edit(vignette("TSfame"))*. This uses the default editor, which can be changed using *options()*. It should be possible to view the pdf version of the guide for this package with *print(vignette("TSfame"))*.

WARNING: Running these example will overwrite a fame database called "testvigFame.db". Beware, if by any chance you have a database with this name.

Once R is started, the functions in this package are made available with

```
> library("TSfame")
```

This will also load required packages *TSdbi*, *DBI*, *fame*, *methods*, and *tframe*. Some examples below also require *zoo*, and *tseries*.

The package *fame* may be installed but not functional because the Fame HLI code is not available. A warning will be issues and the vignette example will not work,

2 Using the Database - TSdbi Functions

This section gives several simple examples of putting series on and reading them from the database. (If a large number of series are to be loaded into a database, one would typically do this with a batch process in Fame.) The first thing to do is to establish a connection to the database:

```
> con <- TSconnect("fame", dbname = "testvigFame.db")
```

(It is also possible to establish connections to Fame databases using Fame server. See the section "Examples Using TSdbi with ets" below for more details.)

This puts a series called *vec* on the database and then reads is back.

```
> z <- ts(rnorm(10), start = c(1990, 1), frequency = 1)
> seriesNames(z) <- "vec"
> if (TSexists("vec", con)) TSdelete("vec", con)
> TSput(z, con)
> z <- TSget("vec", con)
```

If the series is printed it is seen to be a ”ts” time series with some extra attributes.

TSput fails if the series already exists on the *con*, so the above example checks and deletes the series if it already exists. *TSreplace* does not fail if the series does not yet exist, so examples below use it instead. Several plots below show original data and the data retrieved after it is written to the database. One is added to the original data so that both lines are visible.

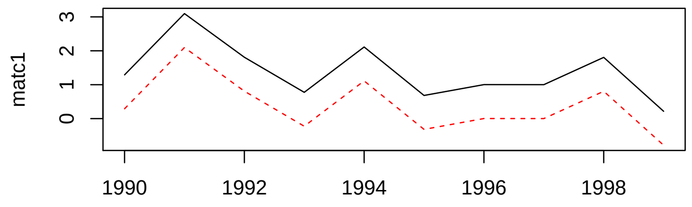
And now more examples:

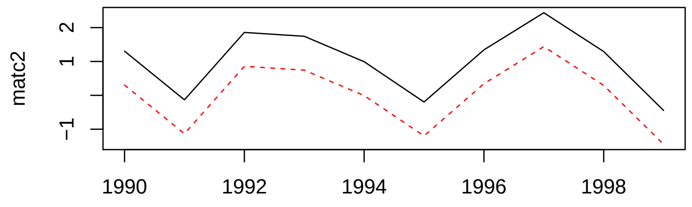
```
> z <- ts(matrix(rnorm(20), 10, 2), start = c(1990, 1), frequency = 1)
> seriesNames(z) <- c("matc1", "matc2")
> TSreplace(z, con)

[1] TRUE

> TSget("matc1", con)

Time Series:
Start = 1990
End = 1999
Frequency = 1
 [1]  1.0240633  0.4888207  0.3633597 -0.8991316  0.9061040 -0.6132410
 [7] -1.0724845 -0.7418611  1.4220151  1.4358306
attr(,"seriesNames")
[1] matc1
attr(,"TSmeta")
An object of class "TSmeta"
Slot "TSdescription":
[1] NA

Slot "TSdoc":
[1] NA

Slot "TSlabel":
[1] NA

Slot "serIDs":
[1] "matc1"

Slot "conType":
[1] "TSfameConnection"
attr(,"package")
[1] "TSfame"

Slot "DateStamp":
[1] "2010-12-23 14:47:18 EST"

Slot "dbname":
```

```
[1] "testvigFame.db"

Slot "hasVintages":
[1] FALSE

Slot "hasPanels":
[1] FALSE

> TSget("matc2", con)

Time Series:
Start = 1990
End = 1999
Frequency = 1
 [1]  0.09871655  0.18798772 -0.15826256 -0.62380571  1.09548344 -1.21222309
 [7]  1.36903761 -0.24566358 -0.77642758  2.05981831
attr(,"seriesNames")
[1] matc2
attr(,"TSmeta")
An object of class "TSmeta"
Slot "TSdescription":
[1] NA

Slot "TSdoc":
[1] NA

Slot "TSlabel":
[1] NA

Slot "serIDs":
[1] "matc2"

Slot "conType":
[1] "TSfameConnection"
attr(,"package")
[1] "TSfame"

Slot "DateStamp":
[1] "2010-12-23 14:47:18 EST"

Slot "dbname":
[1] "testvigFame.db"

Slot "hasVintages":
[1] FALSE
```

```
Slot "hasPanels":
[1] FALSE

> TSget(c("matc1", "matc2"), con)

Time Series:
Start = 1990
End = 1999
Frequency = 1
          matc1       matc2
1990  1.0240633  0.09871655
1991  0.4888207  0.18798772
1992  0.3633597 -0.15826256
1993 -0.8991316 -0.62380571
1994  0.9061040  1.09548344
1995 -0.6132410 -1.21222309
1996 -1.0724845  1.36903761
1997 -0.7418611 -0.24566358
1998  1.4220151 -0.77642758
1999  1.4358306  2.05981831
attr(,"TSmeta")
An object of class "TSmeta"
Slot "TSdescription":
[1] NA

Slot "TSdoc":
[1] NA

Slot "TSlabel":
[1] NA

Slot "serIDs":
[1] "matc1" "matc2"

Slot "conType":
[1] "TSfameConnection"
attr(,"package")
[1] "TSfame"

Slot "DateStamp":
[1] "2010-12-23 14:47:18 EST"

Slot "dbname":
[1] "testvigFame.db"

Slot "hasVintages":
```

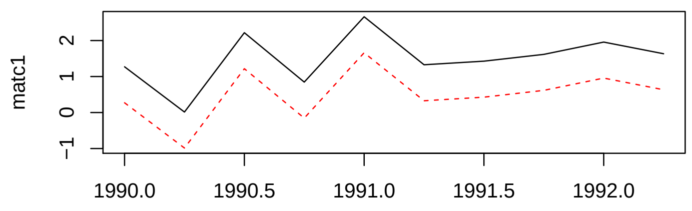
```
[1] FALSE

Slot "hasPanels":
[1] FALSE

> tfplot(z + 1, TSget(c("matc1", "matc2"), con), lty = c("solid",
      "dashed"), col = c("black", "red"))
```

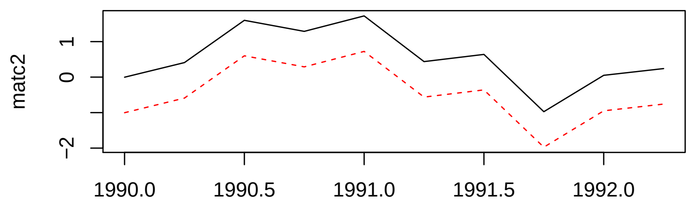




```
> z <- ts(matrix(rnorm(20), 10, 2), start = c(1990, 1), frequency = 4)
> seriesNames(z) <- c("matc1", "matc2")
> TSreplace(z, con)

[1] TRUE

> TSget(c("matc1", "matc2"), con)

             matc1      matc2
1990 Q1  0.5594204 -0.2011508
1990 Q2 -0.6536674  0.6705651
1990 Q3 -1.3384250  0.5032841
1990 Q4  0.5945044 -0.7981909
1991 Q1 -0.6964748 -1.1230537
1991 Q2 -0.1513224  0.9387143
```

```
1991 Q3 -0.4232585  0.1409405
1991 Q4 -0.2178601 -2.3670386
1992 Q1  0.5235157  0.4574788
1992 Q2 -1.9195393 -1.9619522
attr(,"TSmeta")
An object of class "TSmeta"
Slot "TSdescription":
[1] NA

Slot "TSdoc":
[1] NA

Slot "TSlabel":
[1] NA

Slot "serIDs":
[1] "matc1" "matc2"

Slot "conType":
[1] "TSfameConnection"
attr(,"package")
[1] "TSfame"

Slot "DateStamp":
[1] "2010-12-23 14:47:19 EST"

Slot "dbname":
[1] "testvigFame.db"

Slot "hasVintages":
[1] FALSE

Slot "hasPanels":
[1] FALSE

> tfplot(z + 1, TSget(c("matc1", "matc2"), con), lty = c("solid",
      "dashed"), col = c("black", "red"))
```

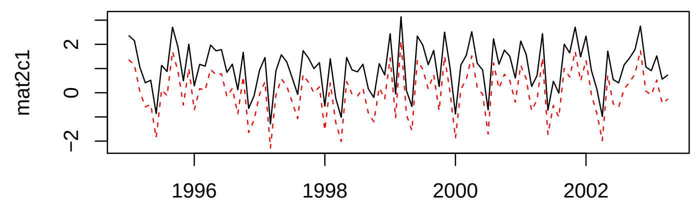


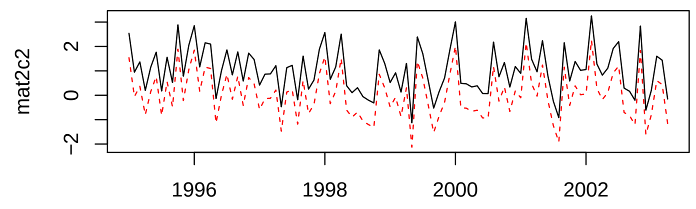


```
> z <- ts(matrix(rnorm(200), 100, 2), start = c(1995, 1), frequency = 12)
> seriesNames(z) <- c("mat2c1", "mat2c2")
> TSreplace(z, con)
```

```
[1] TRUE
```

```
> tfplot(z + 1, TSget(c("mat2c1", "mat2c2"), con), lty = c("solid",
      "dashed"), col = c("black", "red"))
```





The following extract information about the series from the database, although not much information has been added for these examples.

```
> TSmeta("mat2c1", con)
> TSmeta("vec", con)
> TSdates("vec", con)
> TSdescription("vec", con)
> TSdoc("vec", con)
```

Below are examples that make more use of *TSdescription* and codeTSdoc.
Often it is convenient to set the default connection:

```
> options(TSconnection = con)
```

and then the *con* specification can be omitted from the function calls unless another connection is needed. The *con* can still be specified, and some examples below do specify it, just to illustrate the alternative syntax.

```
> z <- TSget("mat2c1")
> TSmeta("mat2c1")
```

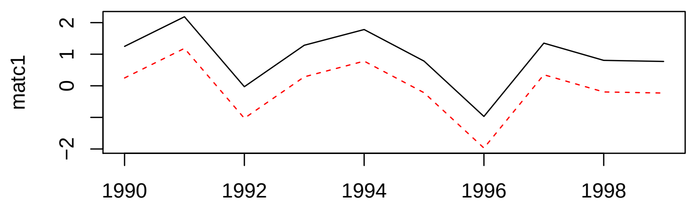
```
An object of class "TSmeta"
Slot "TSdescription":
[1] "NA"
```

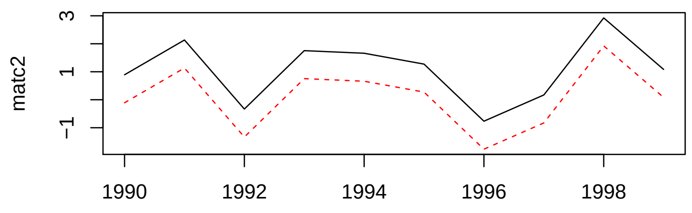
```
Slot "TSdoc":
[1] "NA"

Slot "TSlabel":
[1] NA

Slot "serIDs":
[1] "mat2c1"

Slot "conType":
[1] "TSfameConnection"
attr(,"package")
[1] "TSfame"

Slot "DateStamp":
[1] NA

Slot "dbname":
[1] "testvigFame.db"

Slot "hasVintages":
[1] FALSE

Slot "hasPanels":
[1] FALSE
```

Data documentation can be in two forms, a description specified by *TSdescription* or longer documentation specified by *TSdoc*. These can be added to the time series object, in which case they will be written to the database when *TSput* or *TSreplace* is used to put the series on the database. Alternatively, they can be specified as arguments to *TSput* or *TSreplace*. The description or documentation will be retrieved as part of the series object with *TSget* only if this is specified with the logical arguments *TSdescription* and *TSdoc*. They can also be retrieved directly from the database with the functions *TSdescription* and *TSdoc*.

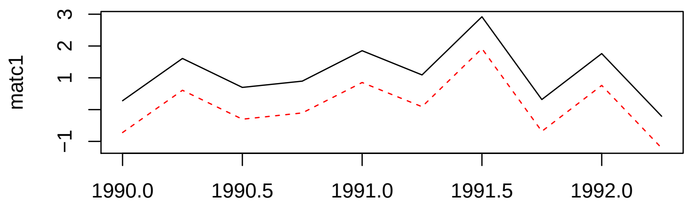
```
> z <- ts(matrix(rnorm(10), 10, 1), start = c(1990, 1), frequency = 1)
> TSreplace(z, serIDs = "Series1", con)

[1] TRUE

> zz <- TSget("Series1", con)
> TSreplace(z, serIDs = "Series1", con, TSdescription = "short rnorm series",
      TSdoc = "Series created as an example in the vignette.")

[1] TRUE
```

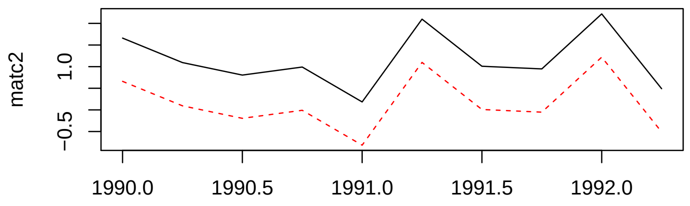
```
> zz <- TSget("Series1", con, TSdescription = TRUE, TSdoc = TRUE)
> start(zz)

[1] 1990    1

> end(zz)

[1] 1999    1

> TSdescription(zz)

[1] "short rnorm series  from  testvigFame.db retrieved  2010-12-23 14:47:20"

> TSdoc(zz)

[1] "Series created as an example in the vignette."

> TSdescription("Series1", con)

[1] "short rnorm series"

> TSdoc("Series1", con)

[1] "Series created as an example in the vignette."

> z <- ts(rnorm(10), start = c(1990, 1), frequency = 1)
> seriesNames(z) <- "vec"
> TSreplace(z, con)

[1] TRUE

> zz <- TSget("vec", con)
> z <- ts(matrix(rnorm(20), 10, 2), start = c(1990, 1), frequency = 1)
> seriesNames(z) <- c("matc1", "matc2")
> TSreplace(z, con)

[1] TRUE

> tfplot(z + 1, TSget(c("matc1", "matc2"), con), lty = c("solid",
      "dashed"), col = c("black", "red"))
```



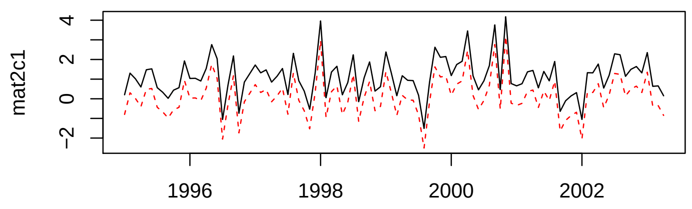


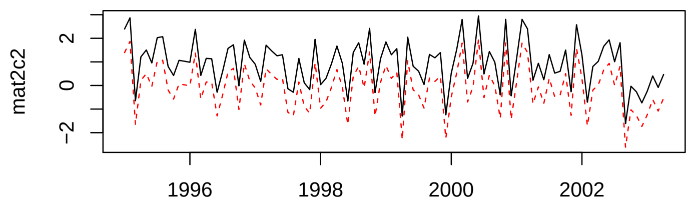
```
> z <- ts(matrix(rnorm(20), 10, 2), start = c(1990, 1), frequency = 4)
> seriesNames(z) <- c("matc1", "matc2")
> TSreplace(z, con)

[1] TRUE

> tfplot(z + 1, TSget(c("matc1", "matc2"), con), lty = c("solid",
      "dashed"), col = c("black", "red"))
```





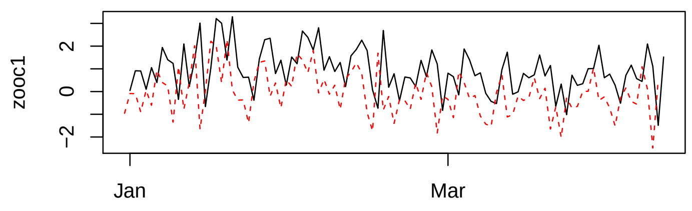
```
> z <- ts(matrix(rnorm(200), 100, 2), start = c(1995, 1), frequency = 12)
> seriesNames(z) <- c("mat2c1", "mat2c2")
> TSreplace(z, con)

[1] TRUE

> tfplot(z + 1, TSget(c("mat2c1", "mat2c2"), con), lty = c("solid",
      "dashed"), col = c("black", "red"))
```

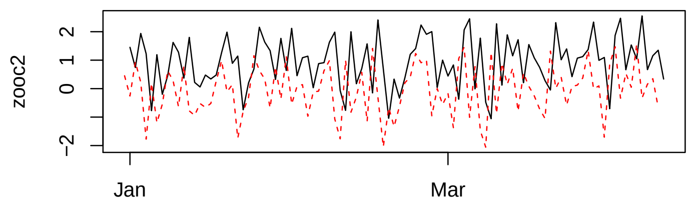




The following examples use dates and times which are not handled by *ts*, so the *zoo* time representation is used.

```
> require("zoo")
> z <- zoo(matrix(rnorm(200), 100, 2), as.Date("1990-01-01") +
      0:99)
> seriesNames(z) <- c("zooc1", "zooc2")
> TSreplace(z, con)

[1] TRUE

> tfplot(z + 1, TSget(c("zooc1", "zooc2"), con), lty = c("solid",
      "dashed"), col = c("black", "red"))
```





Beware that (as of Dec, 2010) there is a bug with weekly dates:

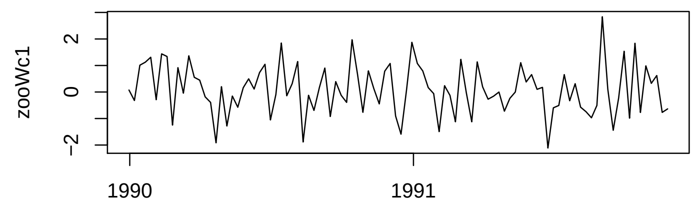
```
> z <- zoo(matrix(rnorm(200), 100, 2), as.Date("1990-01-01") +
      0:99 * 7)
> seriesNames(z) <- c("zooWc1", "zooWc2")
> TSreplace(z, con)

[1] TRUE

> z2 <- TSget(c("zooWc1", "zooWc2"), con)
> time(z)

  [1] "1990-01-01" "1990-01-08" "1990-01-15" "1990-01-22" "1990-01-29"
  [6] "1990-02-05" "1990-02-12" "1990-02-19" "1990-02-26" "1990-03-05"
 [11] "1990-03-12" "1990-03-19" "1990-03-26" "1990-04-02" "1990-04-09"
 [16] "1990-04-16" "1990-04-23" "1990-04-30" "1990-05-07" "1990-05-14"
 [21] "1990-05-21" "1990-05-28" "1990-06-04" "1990-06-11" "1990-06-18"
 [26] "1990-06-25" "1990-07-02" "1990-07-09" "1990-07-16" "1990-07-23"
 [31] "1990-07-30" "1990-08-06" "1990-08-13" "1990-08-20" "1990-08-27"
 [36] "1990-09-03" "1990-09-10" "1990-09-17" "1990-09-24" "1990-10-01"
 [41] "1990-10-08" "1990-10-15" "1990-10-22" "1990-10-29" "1990-11-05"
 [46] "1990-11-12" "1990-11-19" "1990-11-26" "1990-12-03" "1990-12-10"
 [51] "1990-12-17" "1990-12-24" "1990-12-31" "1991-01-07" "1991-01-14"
 [56] "1991-01-21" "1991-01-28" "1991-02-04" "1991-02-11" "1991-02-18"
 [61] "1991-02-25" "1991-03-04" "1991-03-11" "1991-03-18" "1991-03-25"
 [66] "1991-04-01" "1991-04-08" "1991-04-15" "1991-04-22" "1991-04-29"
 [71] "1991-05-06" "1991-05-13" "1991-05-20" "1991-05-27" "1991-06-03"
 [76] "1991-06-10" "1991-06-17" "1991-06-24" "1991-07-01" "1991-07-08"
 [81] "1991-07-15" "1991-07-22" "1991-07-29" "1991-08-05" "1991-08-12"
 [86] "1991-08-19" "1991-08-26" "1991-09-02" "1991-09-09" "1991-09-16"
 [91] "1991-09-23" "1991-09-30" "1991-10-07" "1991-10-14" "1991-10-21"
 [96] "1991-10-28" "1991-11-04" "1991-11-11" "1991-11-18" "1991-11-25"
```

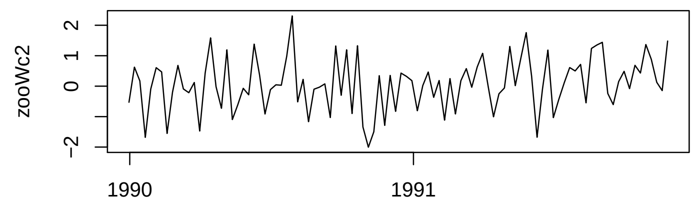
```
> time(z2)
```

```
  [1] "1989-12-31" "1990-01-07" "1990-01-14" "1990-01-21" "1990-01-28"
  [6] "1990-02-04" "1990-02-11" "1990-02-18" "1990-02-25" "1990-03-04"
 [11] "1990-03-11" "1990-03-18" "1990-03-25" "1990-04-01" "1990-04-08"
 [16] "1990-04-15" "1990-04-22" "1990-04-29" "1990-05-06" "1990-05-13"
 [21] "1990-05-20" "1990-05-27" "1990-06-03" "1990-06-10" "1990-06-17"
 [26] "1990-06-24" "1990-07-01" "1990-07-08" "1990-07-15" "1990-07-22"
 [31] "1990-07-29" "1990-08-05" "1990-08-12" "1990-08-19" "1990-08-26"
 [36] "1990-09-02" "1990-09-09" "1990-09-16" "1990-09-23" "1990-09-30"
 [41] "1990-10-07" "1990-10-14" "1990-10-21" "1990-10-28" "1990-11-04"
 [46] "1990-11-11" "1990-11-18" "1990-11-25" "1990-12-02" "1990-12-09"
 [51] "1990-12-16" "1990-12-23" "1990-12-30" "1991-01-06" "1991-01-13"
 [56] "1991-01-20" "1991-01-27" "1991-02-03" "1991-02-10" "1991-02-17"
 [61] "1991-02-24" "1991-03-03" "1991-03-10" "1991-03-17" "1991-03-24"
 [66] "1991-03-31" "1991-04-07" "1991-04-14" "1991-04-21" "1991-04-28"
 [71] "1991-05-05" "1991-05-12" "1991-05-19" "1991-05-26" "1991-06-02"
 [76] "1991-06-09" "1991-06-16" "1991-06-23" "1991-06-30" "1991-07-07"
 [81] "1991-07-14" "1991-07-21" "1991-07-28" "1991-08-04" "1991-08-11"
 [86] "1991-08-18" "1991-08-25" "1991-09-01" "1991-09-08" "1991-09-15"
 [91] "1991-09-22" "1991-09-29" "1991-10-06" "1991-10-13" "1991-10-20"
 [96] "1991-10-27" "1991-11-03" "1991-11-10" "1991-11-17" "1991-11-24"
```

and while this works:

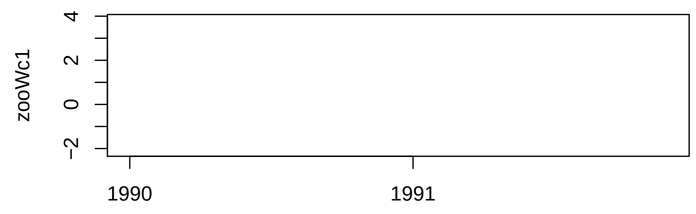
```
> tfplot(z2)
```





this may not:

```
> tfplot(z + 1, z2, col = c("black", "red"), lty = c("dashed",
      "solid"))
```



3 Examples Using Web Data

This section illustrates fetching data from a web server and loading it into the database. This would be a very slow way to load a database, but provides examples of different kinds of time series data. The fetching is done with *TShistQuote* which provides a wrapper for *get.hist.quote* from package *tseries* to give syntax consistent with the *TSdbi*.

Fetching data may fail due to lack of an Interenet connection or delays.

The connection *con* established above to the database will be used to save data but, to make the use of the two connections more obvious, neither will be set as the default:

```
> options(TSconnection = NULL)
```

Now connect to the web server and fetch data:

```
> require("TShistQuote")
> Yahoo <- TSconnect("histQuote", dbname = "yahoo")
> x <- TSget("^gspc", quote = "Close", con = Yahoo)
> plot(x)
> tfplot(x)
> TSrefperiod(x)
```

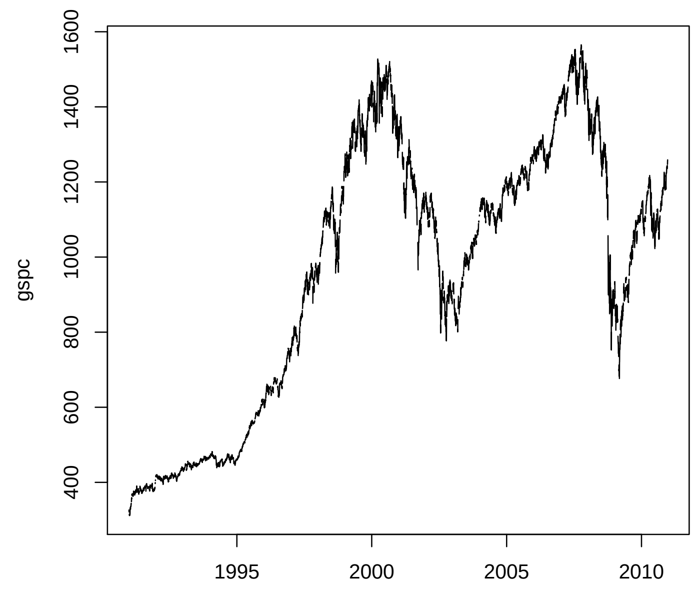
```
[1] "Close"

> TSdescription(x)

[1] "^gspc Close  from  yahoo"

> TSdoc(x)

[1] "^gspc Close  from  yahoo retrieved  2010-12-23 14:47:25"
```

Then write the data to the local server, specifying table B for business day data (using TSreplace in case the series is already there from running this example previously):

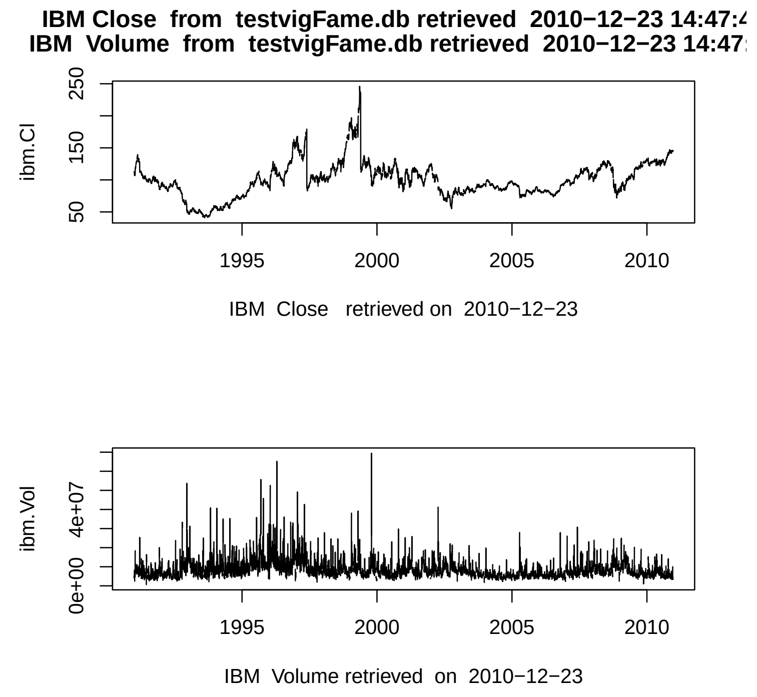
```
> TSreplace(x, serIDs = "gspc", Table = "B", con = con)

[1] TRUE
```

and check the saved version:

```
> TSrefperiod(TSget(serIDs = "gspc", con = con))

[1] "daily"

> TSdescription("gspc", con = con)

[1] "NA"

> TSdoc("gspc", con = con)

[1] "NA"

> tfplot(TSget(serIDs = "gspc", con = con))
```



```
> x <- TSget("ibm", quote = c("Close", "Vol"), con = Yahoo)
> TSreplace(x, serIDs = c("ibm.Cl", "ibm.Vol"), con = con, Table = "B",
      TSdescription. = c("IBM Close", "IBM  Volume"), TSdoc. = paste(c("IBM  Close   retriev
          "IBM  Volume retrieved  on "), Sys.Date()))

[1] TRUE

> z <- TSget(serIDs = c("ibm.Cl", "ibm.Vol"), TSdescription = TRUE,
      TSdoc = TRUE, con = con)
> TSdescription(z)

[1] "IBM Close  from  testvigFame.db retrieved  2010-12-23 14:47:32"
[2] "IBM  Volume  from  testvigFame.db retrieved  2010-12-23 14:47:32"

> TSdoc(z)

[1] "IBM  Close   retrieved on  2010-12-23"
[2] "IBM  Volume retrieved  on  2010-12-23"

> tfplot(z, xlab = TSdoc(z), Title = TSdescription(z))
> tfplot(z, Title = "IBM", start = "2007-01-01")
```



```
> dbDisconnect(con)
> dbDisconnect(Yahoo)
```

3.1 Examples Using TSdbi with ets

These examples use a database called "ets" which is available at the Bank of Canada. This set of examples illustrates how the programs might be used if a larger database is available. Typically a large database would be installed using database scripts directly rather than from R with *TSput* or *TSreplace*.

The following are wrapped in *if (!inherits(con, "try-error"))* so that the vignette will build even when the database is not available. This seems to require an explicit call to *print()*, but that is not usually needed to display results below. Another artifact of this is that results printed in the *if* block do not display until the end of the block.

```
> conets <- try(TSconnect("fame", dbname = "ets /home/ets/db/etsintoecd.db",
      accessMode = "read"))
> if (!inherits(conets, "try-error")) {
      print(TSmeta("M.SDR.CCUSMA02.ST", con = conets))
  }
```

```
Assuming 'ets /home/ets/db/etsintoecd.db' is a Fame Server path due to white space.
Assuming 'ets /home/ets/db/etsintoecd.db' is a Fame Server path due to white space.
```

```
An object of class "TSmeta"
Slot "TSdescription":
[1] "Special Drawing Right---Currency Conversions/US$ exchange rate/Average of daily rates/N

Slot "TSdoc":
[1] "Special Drawing Right---Currency Conversions/US$ exchange rate/Average of daily rates/N

Slot "TSlabel":
[1] NA

Slot "serIDs":
[1] "M.SDR.CCUSMA02.ST"

Slot "conType":
[1] "TSfameConnection"
attr(,"package")
[1] "TSfame"

Slot "DateStamp":
[1] NA

Slot "dbname":
[1] "ets /home/ets/db/etsintoecd.db"

Slot "hasVintages":
[1] FALSE

Slot "hasPanels":
[1] FALSE
```

The above connection is recognized to be a server because of the white space between ”ets” and the database name. This produces a warning message from the *fame* package call, because it is guessing that a Fame server call is intended. An alternative way to establish the connnection is to explicitely indicate that the server functionality is to be used:

```
> conServer <- try(TSconnect("fameServer", dbname = "/home/ets/db/etsintoecd.db",
      service = "2959", host = "ets", user = "", password = "",
      stopOnFail = TRUE))
> if (!inherits(conServer, "try-error")) {
      print(TSmeta("M.SDR.CCUSMA02.ST", con = conServer))
  }
```

```
An object of class "TSmeta"
Slot "TSdescription":
[1] "Special Drawing Right---Currency Conversions/US$ exchange rate/Average of daily rates/N
```

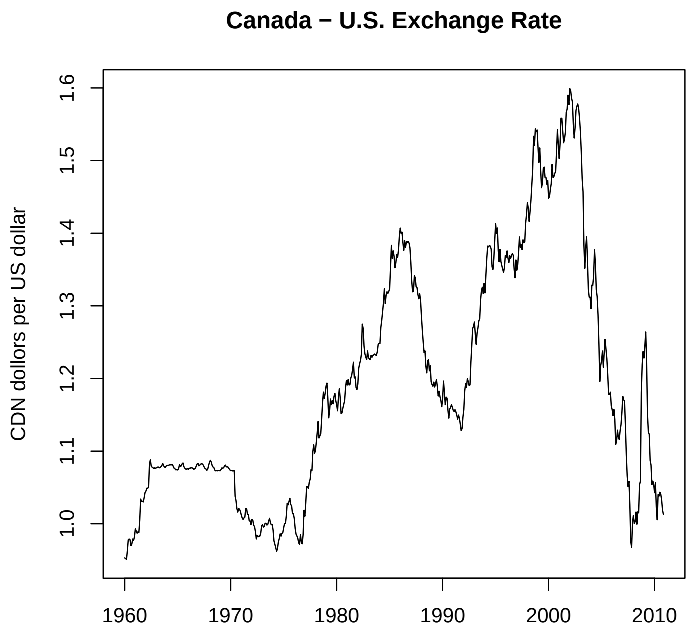
```
Slot "TSdoc":
[1] "Special Drawing Right---Currency Conversions/US$ exchange rate/Average of daily rates/N

Slot "TSlabel":
[1] NA

Slot "serIDs":
[1] "M.SDR.CCUSMA02.ST"

Slot "conType":
[1] "TSfameServerConnection"
attr(,"package")
[1] "TSfame"

Slot "DateStamp":
[1] NA

Slot "dbname":
[1] "/home/ets/db/etsintoecd.db"

Slot "hasVintages":
[1] FALSE

Slot "hasPanels":
[1] FALSE
```

This does not give the warning message, and for that reason will be used below, but they are interchangable.

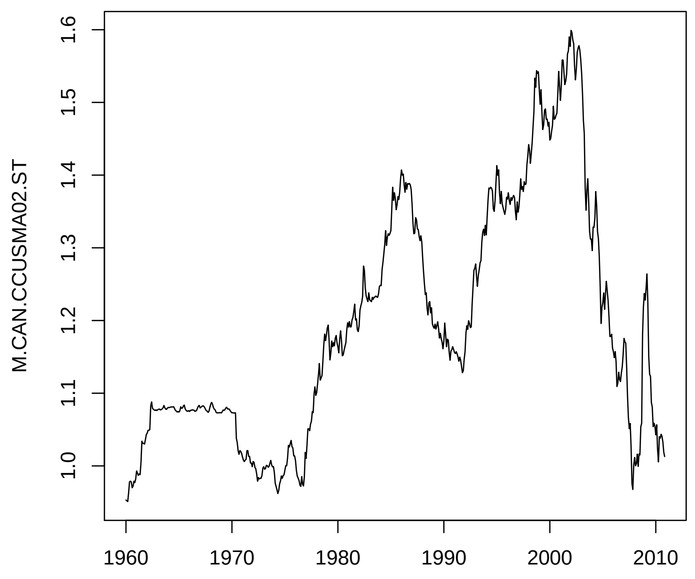
```
> if (!inherits(conServer, "try-error")) {
      options(TSconnection = conServer)
      print(TSmeta("M.SDR.CCUSMA02.ST"))
      EXCH.IDs <- t(matrix(c("M.SDR.CCUSMA02.ST", "SDR/USD exchange rate",
          "M.CAN.CCUSMA02.ST", "CAN/USD exchange rate", "M.MEX.CCUSMA02.ST",
          "MEX/USD exchange rate", "M.JPN.CCUSMA02.ST", "JPN/USD exchange rate",
          "M.EMU.CCUSMA02.ST", "Euro/USD exchange rate", "M.OTO.CCUSMA02.ST",
          "OECD /USD exchange rate", "M.G7M.CCUSMA02.ST", "G7   /USD exchange rate",
          "M.E15.CCUSMA02.ST", "Euro 15. /USD exchange rate"),
          2, 8))
      print(TSdates(EXCH.IDs[, 1]))
      z <- TSdates(EXCH.IDs[, 1])
      print(start(z))
      print(end(z))
      tfplot(TSget(serIDs = "M.CAN.CCUSMA02.ST", conServer), ylab = "CDN dollors per US doll
          Title = "Canada - U.S. Exchange Rate")
  }
```

```
An object of class "TSmeta"
Slot "TSdescription":
[1] "Special Drawing Right---Currency Conversions/US$ exchange rate/Average of daily rates/N

Slot "TSdoc":
[1] "Special Drawing Right---Currency Conversions/US$ exchange rate/Average of daily rates/N

Slot "TSlabel":
[1] NA

Slot "serIDs":
[1] "M.SDR.CCUSMA02.ST"

Slot "conType":
[1] "TSfameServerConnection"
attr(,"package")
[1] "TSfame"

Slot "DateStamp":
[1] NA

Slot "dbname":
[1] "/home/ets/db/etsintoecd.db"

Slot "hasVintages":
[1] FALSE

Slot "hasPanels":
[1] FALSE

     [,1]
[1,] "M.SDR.CCUSMA02.ST from 1960 1 to 2010 11       12"
[2,] "M.CAN.CCUSMA02.ST from 1960 1 to 2010 11       12"
[3,] "M.MEX.CCUSMA02.ST from 1963 1 to 2010 11       12"
[4,] "M.JPN.CCUSMA02.ST from 1960 1 to 2010 11       12"
[5,] "M.EMU.CCUSMA02.ST from 1979 1 to 2010 11       12"
[6,] "M.OTO.CCUSMA02.ST  not available"
[7,] "M.G7M.CCUSMA02.ST  not available"
[8,] "M.E15.CCUSMA02.ST  not available"
[[1]]
[1] 1960    1

[[2]]
[1] 1960    1

[[3]]
```

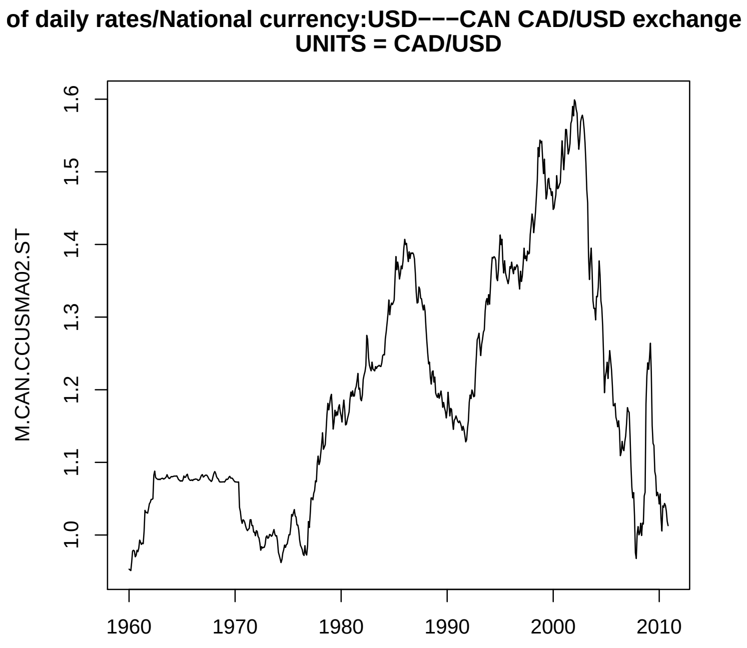
```
[1] 1963    1

[[4]]
[1] 1960    1

[[5]]
[1] 1979    1

[[6]]
[1] NA

[[7]]
[1] NA

[[8]]
[1] NA

[[1]]
[1] 2010   11

[[2]]
[1] 2010   11

[[3]]
[1] 2010   11

[[4]]
[1] 2010   11

[[5]]
[1] 2010   11

[[6]]
[1] NA

[[7]]
[1] NA

[[8]]
[1] NA
```



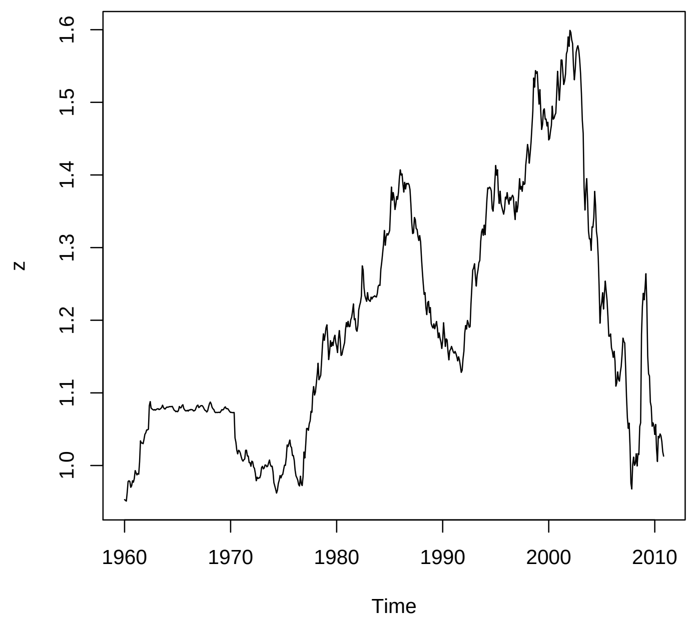
```
> if (!inherits(conServer, "try-error")) {
      print(TSdescription(TSget("M.CAN.CCUSMA02.ST", TSdescription = TRUE)))
      print(TSdescription("M.CAN.CCUSMA02.ST"))
      print(TSdoc(TSget("M.CAN.CCUSMA02.ST", TSdoc = TRUE)))
      print(TSdoc("M.CAN.CCUSMA02.ST"))
      tfplot(TSget("M.CAN.CCUSMA02.ST", names = "M.CAN.CCUSMA02.ST",
          conServer))
  }

[1] "CANADA---Currency Conversions/US$ exchange rate/Average of daily rates/National currenc
[1] "CANADA---Currency Conversions/US$ exchange rate/Average of daily rates/National currenc
[1] "CANADA---Currency Conversions/US$ exchange rate/Average of daily rates/National currenc
[1] "CANADA---Currency Conversions/US$ exchange rate/Average of daily rates/National currenc
```



```
> if (!inherits(conServer, "try-error")) {
      z <- TSget("M.CAN.CCUSMA02.ST", TSdescription = TRUE)
      tfplot(z, Title = strsplit(TSdescription(z), "//")[[1]][1:2])
  }
```



```
> if (!inherits(conServer, "try-error")) {
      plot(z)
  }
```



```
> if (!inherits(conServer, "try-error")) {
      options(TSconnection = NULL)
  }
```

Finally, dbDisconnect closes the connection if it is a Fame Server connect, and does nothing otherwise, but is provided for compatability with other connections.

```
> dbDisconnect(conets)
> dbDisconnect(conServer)
> dbDisconnect(con)
> options(TSconnection = NULL)
```