Time Series Database Interface: R fame (TSfame)

February 1, 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 the fame "testvigFame.db" database.

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*.

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")
```

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]  0.1029549  0.3811046 -0.1951881  0.1829547  1.5452670 -0.5336026
 [7]  0.9950376 -0.1695151  1.0310811 -1.3809481
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] "2010-02-01 14:30:04 EST"

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

Slot "hasVintages":
```

```
[1] FALSE

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

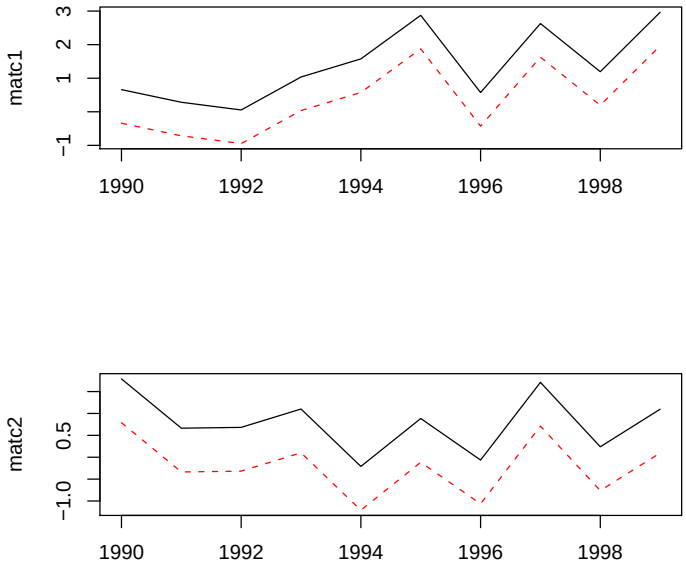
```
> TSget("matc2", con)
```

```
Time Series:
Start = 1990
End = 1999
Frequency = 1
 [1] -0.6247415  1.1392887  1.7353905 -0.3114494 -0.3433792 -2.4332421
 [7]  0.4979763  1.3765396 -0.1593961  0.4748521
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] "2010-02-01 14:30:04 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  0.1029549 -0.6247415
1991  0.3811046  1.1392887
1992 -0.1951881  1.7353905
1993  0.1829547 -0.3114494
1994  1.5452670 -0.3433792
1995 -0.5336026 -2.4332421
1996  0.9950376  0.4979763
1997 -0.1695151  1.3765396
1998  1.0310811 -0.1593961
1999 -1.3809481  0.4748521
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] "2010-02-01 14:30:05 EST"

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

Slot "hasVintages":
[1] FALSE

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

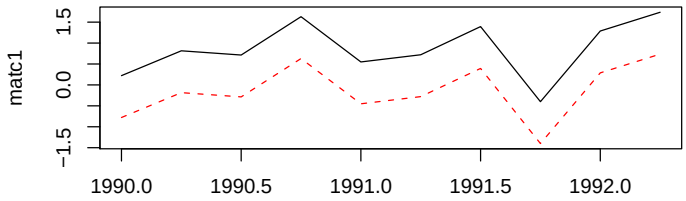
```
> tfplot(z + 1, TSget(c("matc1", "matc2"), con), lty = c("solid",
      "dashed"), col = c("black", "red"))
```

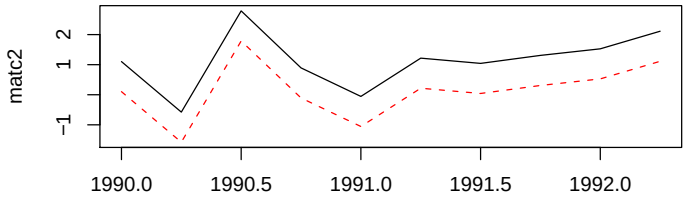



```
> z <- ts(matrix(rnorm(20), 10, 2), start = c(1990, 1), frequency = 4)
> seriesNames(z) <- c("matc1", "matc2")
> TSreplace(z, con)

[1] TRUE

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

             matc1      matc2
1990 Q1 -1.1577188  1.0844678
1990 Q2  0.1215080  1.3383568
1990 Q3 -0.3587615  0.6261051
1990 Q4  0.2022406 -0.7947190
1991 Q1 -0.3115706 -0.8616512
1991 Q2  0.4961037  0.1213843
1991 Q3  1.1233805  0.6255605
1991 Q4 -1.6964024 -1.2905053
1992 Q1 -0.6223771 -0.6609210
1992 Q2  1.0049164  1.0077474
attr(,"TSmeta")
```

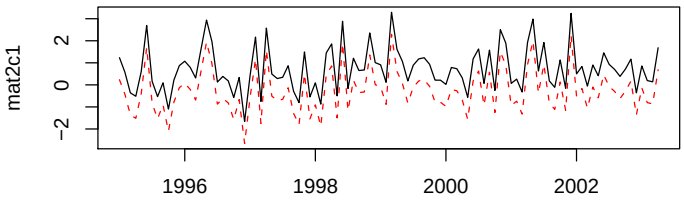
```
An object of class "TSmeta"
Slot "TSdescription":
[1] NA

Slot "TSdoc":
[1] NA

Slot "TSlabel":
[1] NA

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

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

Slot "DateStamp":
[1] "2010-02-01 14:30:05 EST"

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

Slot "hasVintages":
[1] FALSE

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

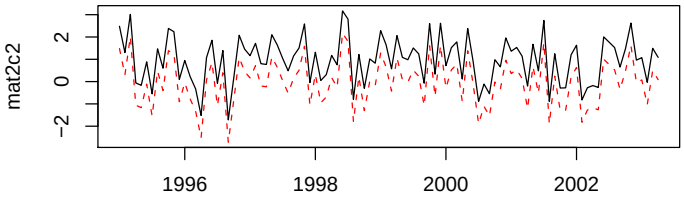
```
> tfplot(z + 1, TSget(c("matc1", "matc2"), con), lty = c("solid",
      "dashed"), col = c("black", "red"))
```






```
> 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 exampoles 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.

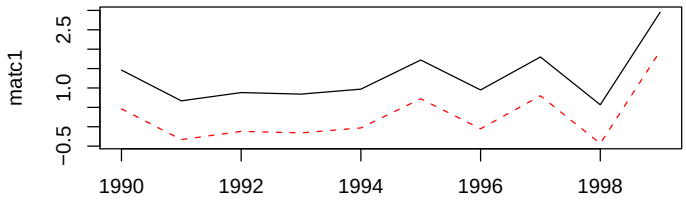
```
> z <- TSget("mat2c1")
> TSmeta("mat2c1")

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

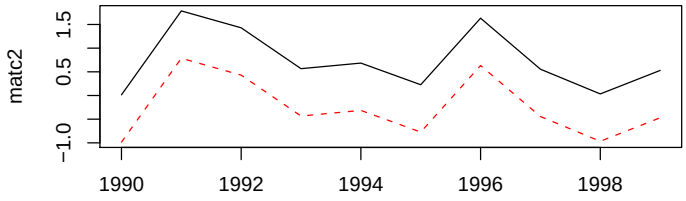
```
Slot "TSdoc":
[1] "NA"

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*.

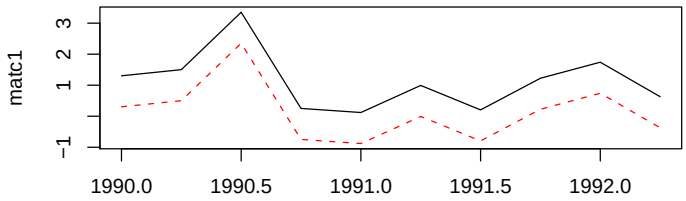
```
> z <- ts(matrix(rnorm(10), 10, 1), start = c(1990, 1), frequency = 1)
> TSreplace(z, serIDs = "Series1", con)

[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
```

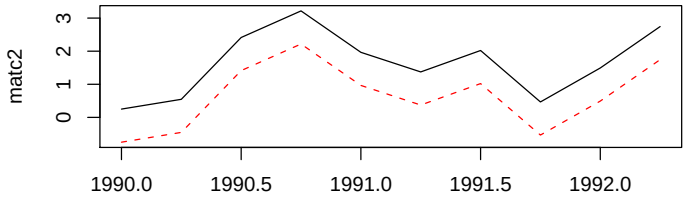
```
> zz <- TSget("Series1", con, TSdescription = TRUE, TSdoc = TRUE)
> start(zz)

[1] 1990    1

> end(zz)

[1] 1999    1

> TSdescription(zz)

[1] "short rnorm series  from  testvigFame.db retrieved  2010-02-01 14:30:07"

> 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"))
```



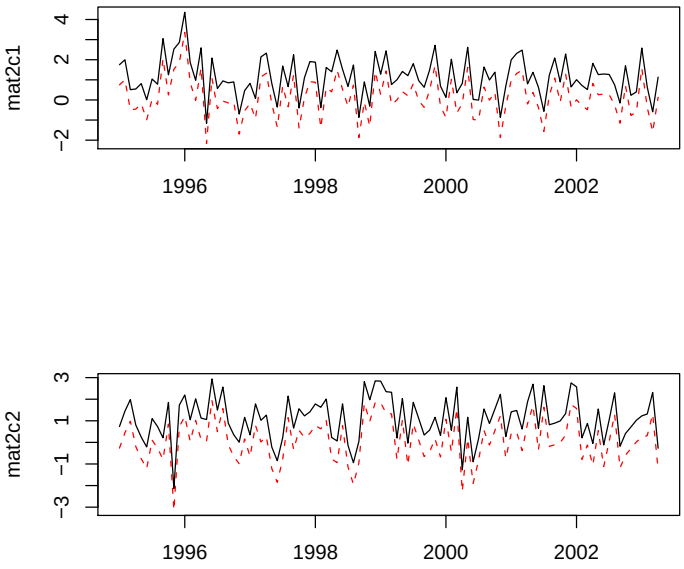



```
> z <- ts(matrix(rnorm(20), 10, 2), start = c(1990, 1), frequency = 4)
> seriesNames(z) <- c("matc1", "matc2")
> TSreplace(z, con)

[1] TRUE

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



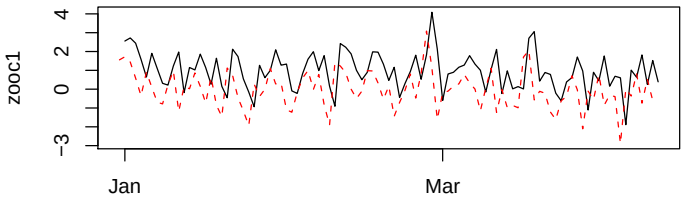


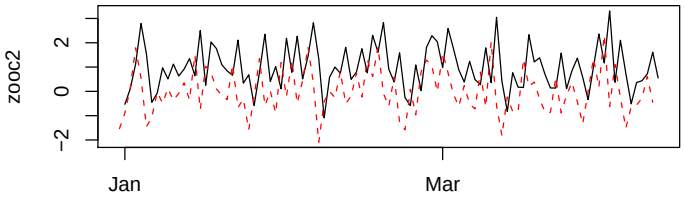

```
> z <- ts(matrix(rnorm(200), 100, 2), start = c(1995, 1), frequency = 12)
> seriesNames(z) <- c("mat2c1", "mat2c2")
> TSreplace(z, con)

[1] TRUE

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



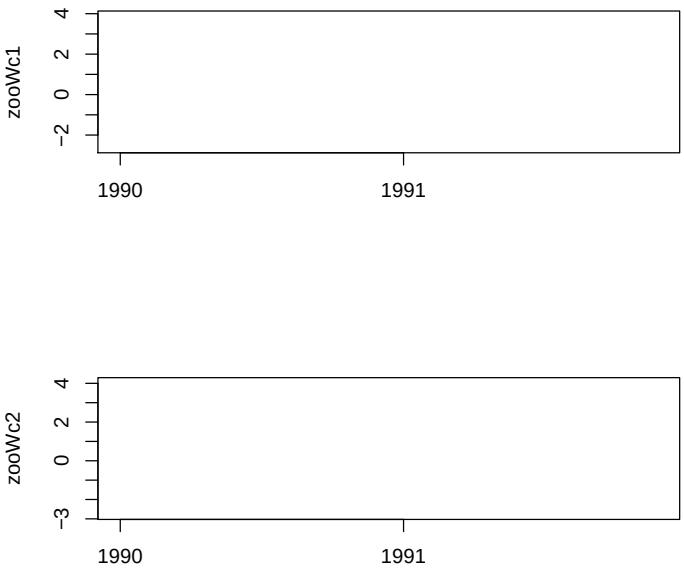
The following examples use dates and times which are not handled by *ts*, so the *zoo* time representation is used.

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

[1] TRUE

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






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

[1] TRUE

> tfplot(z + 1, TSget(c("zooWc1", "zooWc2"), con), 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)
```

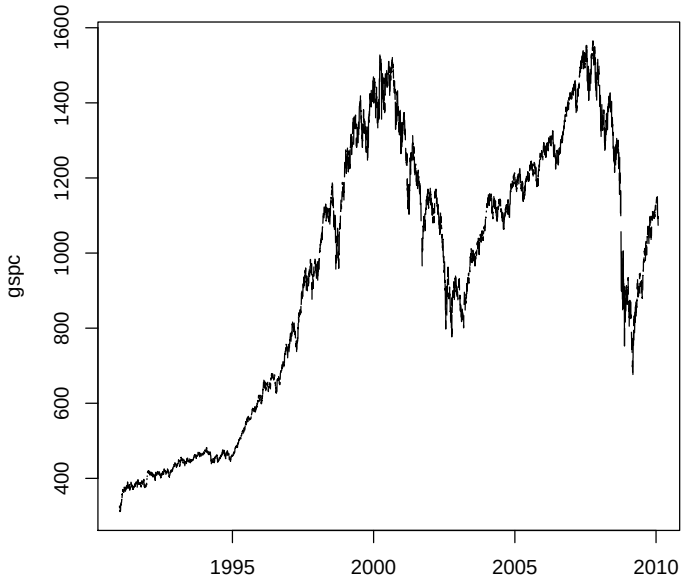
```
[1] "Close"

> TSdescription(x)

[1] "^gspc Close  from  yahoo"

> TSdoc(x)

[1] "^gspc Close  from  yahoo retrieved  2010-02-01 14:30:13"
```

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):

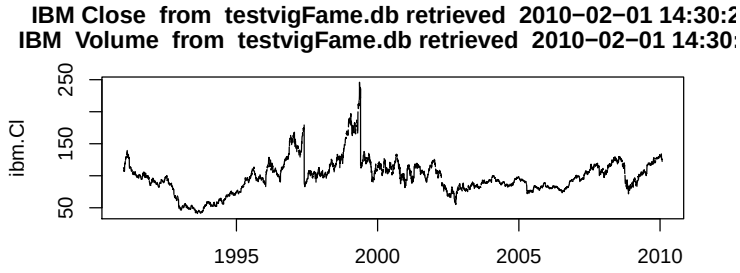
```
> TSreplace(x, serIDs = "gspc", Table = "B", con = con)

[1] TRUE
```

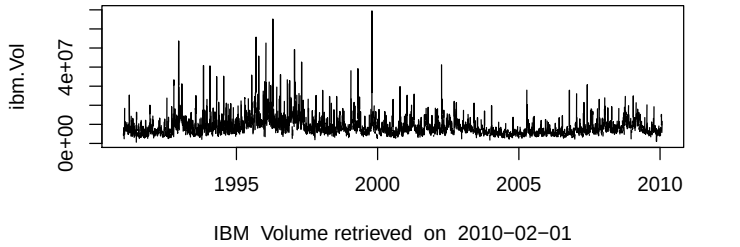
and check the saved version:

```
> 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  2010-02-01 14:30:18"
[2] "IBM  Volume  from  testvigFame.db retrieved  2010-02-01 14:30:18"

> TSdoc(z)

[1] "IBM  Close   retrieved on  2010-02-01"
[2] "IBM  Volume retrieved  on  2010-02-01"

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



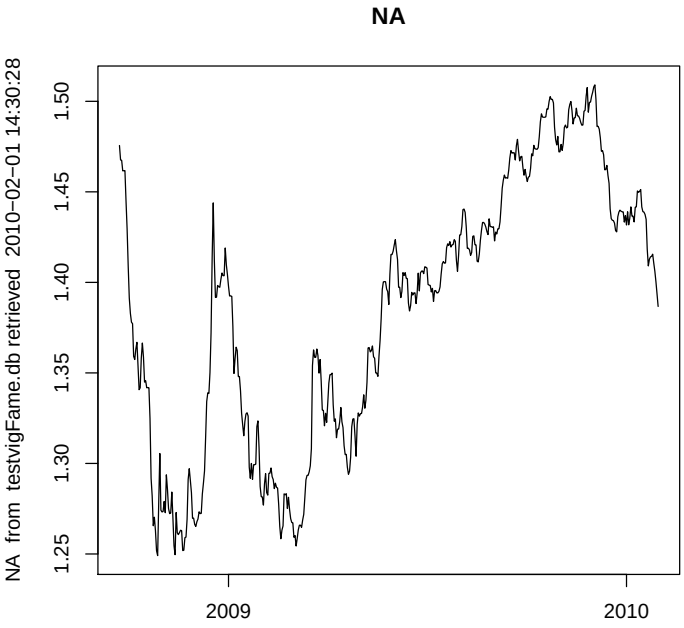


Oanda has maximum of 500 days, so the start date is specified here so as to not exceed that.

```
> Oanda <- TSconnect("histQuote", dbname = "oanda")
> x <- TSget("EUR/USD", start = Sys.Date() - 495, con = Oanda)
> TSreplace(x, serIDs = "EURUSD", con = con)

[1] TRUE
```

and check the saved version:

```
> z <- TSget(serIDs = "EURUSD", TSdoc = TRUE, TSdescription = TRUE,
      con = con)
> tfplot(z, Title = TSdoc(z), ylab = TSdescription(z))
> tfplot(z, Title = "EUR/USD", start = "2007-01-01")
> tfplot(z, Title = "EUR/USD", start = "2007-03-01")
> tfplot(z, Title = "EUR/USD", start = Sys.Date() - 14, end = Sys.Date(),
      xlab = format(Sys.Date(), "%Y"))
```




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

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(conets, "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.

THESE EXAMPLES ARE TEMPORARLY DISABLED BECAUSE OF A BUG (NOT YET SUPPORTED FEATURE) TO ACCES REMOTE FAME SERVERS.

```
> m <- dbDriver("fame")
> conets <- try(TSconnect(m, dbname = "ets /home/ets/db/etsintoecd.db",
      accessMode = "read"))
```

```
> if (!inherits(conets, "try-error")) {
      options(TSconnection = conets)
      print(TSmeta("M.SDR.CCUSMA02.ST"))
      z <- getfame("M.SDR.CCUSMA02.ST", "ets /home/ets/db/etsintoecd.db",
          save = FALSE, envir = parent.frame(), start = NULL, end = NULL,
          getDoc = FALSE)
      id <- fameDbOpen("ets /home/ets/db/etsintoecd.db", "read")
      fameWhat(id, "M.SDR.CCUSMA02.ST")
      fameDbClose(id)
      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 = "V122646", conets))
  }

> if (!inherits(conets, "try-error")) {
      print(TSdescription(TSget("V122646", TSdescription = TRUE)))
      print(TSdescription("V122646"))
      print(TSdoc(TSget("V122646", TSdoc = TRUE)))
      print(TSdoc("V122646"))
      tfplot(TSget("V122646", names = "V122646", conets))
  }

> if (!inherits(conets, "try-error")) {
      z <- TSget("V122646", TSdescription = TRUE)
      tfplot(z, Title = strsplit(TSdescription(z), ","))
  }

> if (!inherits(conets, "try-error")) {
      z <- TSget("SDSP500", TSdescription = TRUE)
      tfplot(z, Title = TSdescription(z))
      plot(z)
  }

> if (!inherits(conets, "try-error")) {
      z <- TSget(c("DSP500", "SDSP500"), TSdescription = TRUE)
      tfplot(z, xlab = TSdescription(z))
  }
```

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

> if (!inherits(conets, "try-error")) {
      ETSgdp <- annualizedGrowth(aggregate(TSget("V1992067"), nfrequency = 4,
          FUN = mean), lag = 4, names = "GDP Y/Y Growth")
      tfplot(ETSgdp)
  }

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

Finally, dbDisconnect does nothing in TSfame, but is provided for compatability with other packages.

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