

# Package ‘juice’

April 13, 2006

**Title** Dynamic Systems Estimation - juice extensions

**Description** experimental dimension reduction techniques. See ?juice.Intro for more details

**Depends** R (>= 2.0.0), setRNG (>= 2004.4-1), tframe (>= 2006.4-1), dse1 (>= 2006.4-1)

**Version** 2006.4-1

**LazyLoad** yes

**License** Free. See the LICENCE file for details.

**Author** Paul Gilbert <pgilbert@bank-banque-canada.ca>

**Maintainer** Paul Gilbert <pgilbert@bank-banque-canada.ca>

**URL** <http://www.bank-banque-canada.ca/pgilbert>

## R topics documented:

00.juice.Intro . . . . .	2
TSdata.TSdataconcentrate . . . . .	2
canonical.prediction . . . . .	3
checkConsistentDimensions.TSmodelconcentrate . . . . .	4
checkResiduals.TSdataconcentrate . . . . .	4
concentrate . . . . .	5
concentrateOnly . . . . .	6
concentrateOriginal . . . . .	7
concentrated.checkResiduals . . . . .	8
concentrated.nseriesInput . . . . .	8
concentrated.tfplot . . . . .	9
concentratedDimension . . . . .	10
concentratedSeriesNames . . . . .	10
concentrator . . . . .	11
end.TScanonicalPrediction . . . . .	12
estConcentratedModel . . . . .	13
estProjection . . . . .	14
l.TSmodelconcentrate . . . . .	15
nseriesInput.TSmodelconcentrate . . . . .	15
percentChange.TScanonicalPrediction . . . . .	16
plot2by2 . . . . .	16
print.concentrate . . . . .	17

reconstitute . . . . .	17
selectSeries.concentrate . . . . .	18
tfplot.concentrate . . . . .	19
tfprint.concentrate . . . . .	20
tframed.concentrate . . . . .	20
tfwindow.concentrate . . . . .	21

<b>Index</b>	<b>22</b>
--------------	-----------

---

00.juice.Intro	<i>Juice</i>
----------------	--------------

---

## Description

The attempted approach is that a model of type TSmodelconcentrate should work like KF and ARMA model, so that plot, residual, etc., produce results for the full (reconstituted) data set. Special methods (eg. concentrated.tfplot, concentrated.checkResiduals) do the equivalent thing using the reduced dimensional data as the TSdata and dropping the fact that the model is of class TSmodelconcentrate.

The outputData for a concentrated object is the original data, but for a reconstituted object it is the reconstituted data. So, for example, tfplot(ConcentratedDataObject) will plot the original data (as well as the reconstructed data) whereas tfplot(reconstitute(ConcentratedDataObject)) plots only the reconstituted data. (To plot the concentrated series use concentrated.tfplot(ConcentratedDataObject)).

Typically one should not work with a reconstituted object unless it is explicitly needed, as the original data is "hidden".

The concentrated data is extracted from both TSdataconcentrate and TSdatareconstitute using the function concentrated.outputData.

---

TSdata.TSdataconcentrate	
	<i>TSdata Specific Methods</i>

---

## Description

See the generic function description.

## Usage

```
## S3 method for class 'TSdataconcentrate':
TSdata(data, names=NULL, ...)
```

## Arguments

data	a TSdataconcentrate object from which to get TSdata.
names	series names for the result.
...	arguments to be passed to other methods.

## Details

Uses reconstitute to build TSdata.

**See Also**

[reconstitute TSdata](#)

---

canonical.prediction

*Canonical Prediction*

---

**Description**

Use canonical correlation with input data as the independent variables used to predict output data.

**Usage**

```
canonical.prediction(d, conc=concentrator(d),
  q=min(concentrated.nseriesInput(d),
    concentrated.nseriesOutput(d)))
is.TScanonicalPrediction(x)
```

**Arguments**

d	a TSdataconcentrate object as returned by concentrate.
conc	a concentrator.
q	integer indicating the number of canonical variates to keep.
x	any object.

**Details**

Data d as returned by concentrate. Alternately, a different conc (proj) can be used. Use q canonical variates from input data as predictors of q canonical variates from output data and then use these to reconstruct output data. (ref T.W. Anderson p491) q cannot exceed min(concentrated.nseriesInput(d), concentrated.nseriesOutput(d))

**Value**

A TScanonicalPrediction object.

**See Also**

[concentrate concentrator](#)

**Examples**

```
data("egl.DSE.data.diff", package="dse1")
z <- canonical.prediction(concentrate(egl.DSE.data.diff))
is.TScanonicalPrediction(z)
```

---

```
checkConsistentDimensions.TSmodelconcentrate
  checkConsistentDimensions Specific Methods
```

---

### Description

See the generic function description.

### Usage

```
## S3 method for class 'TSmodelconcentrate':
checkConsistentDimensions(obj1, obj2=NULL)
```

### Arguments

obj1            a TSmodelconcentrate object.  
obj2            a matrix of time series or a TSdata object.

### See Also

[checkConsistentDimensions](#)

---

```
checkResiduals.TSdataconcentrate
  checkResiduals Specific Methods
```

---

### Description

See the generic function description.

### Usage

```
## S3 method for class 'TSdataconcentrate':
checkResiduals(obj, ...)
## S3 method for class 'TSdatareconstitute':
checkResiduals(obj, ...)
## S3 method for class 'concentrated':
checkResiduals(obj, ...)
```

### Arguments

obj            a TSdataconcentrate object.  
...            arguments passed to checkResiduals.

### See Also

[checkResiduals concentrated.checkResiduals](#)

concentrate

*Concentrate Series in a TSdata Object***Description**

Calculate a reduced dimesion version of the data using principal components (or cannonical correlation for TSdata with input and output).

**Usage**

```
concentrate(d, conc=NULL, center=TRUE, scale=TRUE, ...)
## Default S3 method:
concentrate(d, conc=NULL, center=TRUE, scale=TRUE, n=1, ...)
## S3 method for class 'TSdata':
concentrate(d,conc=NULL, center=TRUE, scale=TRUE, m=1, p=1, ...)
is.concentrate(x)
is.TSdataconcentrate(x)
is.TSmodelconcentrate(x)
```

**Arguments**

d	a matrix or TSdata object.
...	arguments to be passed to other methods.
conc	object containing the concentrator (projection) matrix used for the reduction
center	center the observations to mean zero first (passed to estProjection).
scale	scale the observations to SD one first (passed to estProjection).
n	dimension of the concentrated series (passed to estProjection).
m	dimension of the concentrated input series (passed to estProjection).
p	dimension of the concentrated output series (passed to estProjection).
x	any object.

**Value**

A matrix or TSdata object.

**See Also**

[estProjection](#) [reconstitute](#) [prcomp](#)

**Examples**

```
data("egl.DSE.data", package="dse1")
require("stats")
z <- concentrate(egl.DSE.data)
is.concentrate(z)
```

---

concentrateOnly	<i>Extract Concentrate</i>
-----------------	----------------------------

---

## Description

concentrateOnly

## Usage

```
concentrateOnly(d)
## S3 method for class 'concentrate':
concentrateOnly(d)
## S3 method for class 'TSdataconcentrate':
concentrateOnly(d)
## S3 method for class 'TSdatareconstitute':
concentrateOnly(d)
## S3 method for class 'TSestModel':
concentrateOnly(d)
## S3 method for class 'TSmodelconcentrate':
concentrateOnly(d)
```

## Arguments

d                      a concentrate object.

## Details

The concentrated data set is returned as a TSdata object, stripped of the fact that it is a concentrate.

## Value

A TSdata object.

## See Also

[concentrate](#) [concentrator](#) [concentrateOriginal](#)

## Examples

```
data("egl.DSE.data", package="dse1")
require("stats")
z <- concentrate(egl.DSE.data)
z <- concentrateOnly(z)
```

---

concentrateOriginal  
*Extract Original Series*

---

## Description

concentrateOriginal

## Usage

```
concentrateOriginal(d)
## S3 method for class 'TSdataconcentrate':
concentrateOriginal(d)
## S3 method for class 'TSdatareconstitute':
concentrateOriginal(d)
## S3 method for class 'concentrate':
concentrateOriginal(d)
## S3 method for class 'TScanonicalPrediction':
concentrateOriginal(d)
```

## Arguments

d                      A concentrate object.

## Details

The original data set is returned as a TSdata object, stripped of the fact that it is a concentrate.

## Value

A TSdata object.

## See Also

[concentrate](#) [concentrator](#) [concentrateOnly](#)

## Examples

```
data("egl.DSE.data", package="dse1")
require("stats")
z <- concentrate(egl.DSE.data)
z <- concentrateOriginal(z)
```

---

```
concentrated.checkResiduals
```

*Check Residuals of Concentrated Data*

---

### Description

The TSdataconcentrate is passed to checkResiduals as simple TSdata in the reduced dimension space (not expanded).

### Usage

```
concentrated.checkResiduals(data, ...)
```

### Arguments

data	a TSdataconcentrate object.
...	arguments passed to checkResiduals.

### Value

x

### See Also

[checkResiduals concentrate](#)

---

```
concentrated.nseriesInput
```

*Concentrated Dimension of TSdata*

---

### Description

The dimension (number of series) in concentrate data. This is the dimension onto which the original series has been projected.

### Usage

```
concentrated.nseriesInput(x)
concentrated.nseriesOutput(x)
```

### Arguments

x	A concentrated TSdata object.
---	-------------------------------

### Value

An integer.

### See Also

[concentratedDimension concentrate](#)



**Examples**

```
data("egl.DSE.data", package="dse1")
require("stats")
z <- concentrate(egl.DSE.data)
concentrated.nseriesOutput(z)
```

---

`concentrated.tfplot`*Plot Concentrated Series*

---

**Description**

The concentrate data is plotted.

**Usage**

```
concentrated.tfplot(x, ...)
```

**Arguments**

<code>x</code>	A concentrated data object.
<code>...</code>	arguments to be passed to other tfplot.

**Value**

Depends on the argument. For a simple concentrated data object the result is a vector of strings.

**Side Effects**

A plot is generated

**See Also**

[tfplot](#)

**Examples**

```
data("egl.DSE.data", package="dse1")
require("stats")
z <- concentrate(egl.DSE.data)
concentrated.tfplot(z)
```

---

concentratedDimension  
*Concentrated Dimension*

---

**Description**

The dimension (number of series) in concentrate data. This is the dimension onto which the original series has been projected.

**Usage**

```
concentratedDimension(x)
## S3 method for class 'concentrate':
concentratedDimension(x)
```

**Arguments**

`x`                      a concentrated data object.

**Value**

Depends on the argument. For a simple concentrated data object the result is an integer.

**See Also**

[concentrated.nseriesInput](#) [concentrated.nseriesOutput](#) [concentrate](#)

**Examples**

```
data("egl.DSE.data", package="dse1")
require("stats")
z <- concentrate(egl.DSE.data)
concentratedDimension(outputData(z))
concentrated.nseriesOutput(z)
```

---

concentratedSeriesNames  
*Concentrated Series Names*

---

**Description**

The names of series in concentrate data.

**Usage**

```
concentratedSeriesNames(x)
## S3 method for class 'concentrate':
concentratedSeriesNames(x)
## S3 method for class 'TSdata':
concentratedSeriesNames(x)
concentratedSeriesNamesInput(x)
concentratedSeriesNamesOutput(x)
```

**Arguments**

`x` A concentrated data object.

**Value**

Depends on the argument. For a simple concentrated data object the result is a vector of strings.

**See Also**

[seriesNames](#) [seriesNamesInput](#) [seriesNamesOutput](#) [concentratedDimension](#)  
[concentrate](#)

**Examples**

```
data("egl.DSE.data", package="dse1")
require("stats")
z <- concentrate(egl.DSE.data)
concentratedSeriesNames(z)
```

---

concentrator

*Data Concentrator*


---

**Description**

concentrator

**Usage**

```
concentrator(d)
## S3 method for class 'concentrate':
concentrator(d)
## S3 method for class 'concentrator':
concentrator(d)
## S3 method for class 'TSdata':
concentrator(d)
## S3 method for class 'TSdataconcentrator':
concentrator(d)
## S3 method for class 'TSmodelconcentrate':
concentrator(d)
is.concentrator(x)
is.TSdataconcentrator(x)
```

**Arguments**

`d` a concentrate or concentrator object.  
`x` any object.

**Details**

The concentrator is extracted from a concentrated data object.

**Value**

A concentrator.

**See Also**

[concentrate](#) [concentrateOnly](#) [concentrateOriginal](#)

**Examples**

```
data("egl.DSE.data", package="dse1")
require("stats")
z <- concentrate(egl.DSE.data)
concentrator(z)
is.concentrator(concentrator(z))
```

---

end.TScanonicalPrediction

*Specific Methods for TScanonicalPrediction*

---

**Description**

See the generic function description.

**Usage**

```
## S3 method for class 'TScanonicalPrediction':
end(x, ...)
## S3 method for class 'TScanonicalPrediction':
start(x, ...)
## S3 method for class 'TScanonicalPrediction':
periods(x)
## S3 method for class 'TScanonicalPrediction':
frequency(x, ...)
```

**Arguments**

<code>x</code>	An object containing TSdata.
<code>...</code>	(further arguments, currently disregarded).

**Value**

Depends.

**See Also**

[end](#) [start](#) [periods](#) [frequency](#)

---

estConcentratedModel

*Estimate a Concentrated Model*


---

## Description

estConcentratedModel

## Usage

```
estConcentratedModel(data, estimation="estVARXls",
                      estimation.args=NULL, ...)
## S3 method for class 'TSdata':
estConcentratedModel(data, estimation="estVARXls",
                      estimation.args=NULL, m=1, p=1, center=TRUE, scale=TRUE, ...)
## S3 method for class 'TSdataconcentrate':
estConcentratedModel(data,
                      estimation="estVARXls", estimation.args=NULL, warn=TRUE, ...)
```

## Arguments

data	A TSdata or TSdataconcentrate object.
estimation	Estimation method.
estimation.args	Estimation method arguments.
m, p	dimension of the concentrated series.
center	center the observations to mean zero first.
scale	scale the observations to SD one first.
warn	logical indicating if certain warning messages should be printed.
...	arguments to be passed to other methods.

## Details

A concentrated version of the data (reduced dimension) is used to estimate a reduced dimension model. The projections for concentrating the data are retained so that model predictions can be expanded to the full dimension data space.

If data is TSdataconcentrate then the concentrator with that data is used and m, p, center and scale are not used. For TSdata these arguments are used to first estimate a concentrated version of the data.

## Value

A TSmodelconcentrate.

## See Also

[concentrate](#) [concentrator](#) [estProjection](#)

**Examples**

```
data("egl.DSE.data.diff", package="dse1")
model <- estConcentratedModel(egl.DSE.data.diff)
```

---

estProjection

---

*Calculate Projection from Concentrating Series.*


---

**Description**

Calculate the projection to use for a reduced dimension version of the data using principal components (or canonical correlation for TSdata with input and output).

**Usage**

```
estProjection(data, center=TRUE, scale=TRUE, ...)
## Default S3 method:
estProjection(data, center=TRUE, scale=TRUE, n=1, ...)
## S3 method for class 'TSdata':
estProjection(data, center=TRUE, scale=TRUE, m=1,p=1, ...)
```

**Arguments**

data	a matrix for the default method or TSdata object.
n	dimension of the concentrated series.
m	dimension of the concentrated input series.
p	dimension of the concentrated output series.
center	logical indicating center the observations to mean zero first.
scale	logical indicating scale the observations to SD one first.
...	arguments passed to other methods.

**Value**

An object containing matrix (conc) to use to concentrate the data.

**See Also**

[estConcentratedModel](#) [concentrate](#) [reconstitute](#) [prcomp](#)

**Examples**

```
data("egl.DSE.data", package="dse1")
require("stats")
z <- estProjection(egl.DSE.data)
```

---

l.TSmodelconcentrate

*Specific Methods for l*


---

### Description

See the generic function description.

### Usage

```
## S3 method for class 'TSmodelconcentrate':
l(obj1, obj2, sampleT=nrow(outputData(obj2)),
  predictT=sampleT, result=NULL, warn=TRUE, ...)
```

### Arguments

obj1	a TSmodelconcentrate model object.
obj2	a TSdataconcentrate data object.
sampleT	an integer indicating the number of periods of data to use.
predictT	an integer to what period forecasts should be extrapolated.
result	if non-NULL then the returned value is only the sub-element indicated by result. result can be a character string or integer.
warn	if FALSE then certain warning messages are turned off.
...	arguments passed to other methods.

### See Also

[l 1.ARMA 1.SS](#)

---

nseriesInput.TSmodelconcentrate

*Specific Methods for input/nseriesOutput*


---

### Description

See the generic function description.

### Usage

```
## S3 method for class 'TSmodelconcentrate':
nseriesInput(x)
## S3 method for class 'TSmodelconcentrate':
nseriesOutput(x)
```

### Arguments

x	a TSdata or TSmodelconcentrate object.
---	--

### See Also

[nseriesInput nseriesOutput](#)

---

```
percentChange.TScanonicalPrediction
```

*Specific Methods for percentChange*

---

**Description**

See the generic function description.

**Usage**

```
## S3 method for class 'TScanonicalPrediction':
percentChange(obj,
               base=NULL, lag=1, cumulate=FALSE, e=FALSE, ...)
```

**Arguments**

obj	see the generic function.
e	see the generic function.
base	see the generic function.
lag	see the generic function.
cumulate	see the generic function.
...	arguments passed to other methods.

**See Also**

[percentChange](#)

---

plot2by2	<i>plot2by2</i>
----------	-----------------

---

**Description**

plot data series one vs another, two at a time (that is, data[,i] vs data[,j] for all i,j (not on time axis)).

**Usage**

```
plot2by2(data, ...)
## Default S3 method:
plot2by2(data, pch=".", ...)
## S3 method for class 'TSdata':
plot2by2(data, ...)
```

**Arguments**

data	a matrix of time series or a TSdata object.
pch	character to be used for plotting.
...	arguments passed to tfplot.

**Value**

None.



---

print.concentrate    *Print Specific Methods*


---

**Description**

See the generic function description.

**Usage**

```
## S3 method for class 'concentrate':
print(x, ...)
```

**Arguments**

`x`                    a concentrate object.  
`...`                  arguments to be passed to other methods.

**See Also**

[print](#)

---

reconstitute            *Reconstitute*


---

**Description**

reconstitute

**Usage**

```
reconstitute(d, conc=NULL, names=NULL)
## Default S3 method:
reconstitute(d, conc=NULL, names=seriesNames(d))
## S3 method for class 'concentrate':
reconstitute(d, conc=concentrator(d),
              names=seriesNames(d))
## S3 method for class 'TSdataconcentrate':
reconstitute(d, conc=concentrator(d),
              names=seriesNames(d))
is.TSdatareconstitute(x)
```

**Arguments**

`d`                    a concentrated data object.  
`conc`                a concentrator.  
`names`              series names for the result.  
`x`                    any object.

**Details**

A concentrated data object is used to reconstruct the full dimension data. Thus the result has the same dimension as the original data, but will not be exactly the same because some information is lost when the data is concentrated (unless the concentrate has the full dimension of the original data, which would usually be pointless).

**Value**

Depends on the argument.

**See Also**

[concentrate](#)

**Examples**

```
data("egl.DSE.data", package="dse1")
require("stats")
z <- concentrate(egl.DSE.data)
z <- reconstitute(z)
is.TSdatareconstitute(z)
```

---

`selectSeries.concentrate`

*Specific Methods for selectSeries*

---

**Description**

See the generic function description.

**Usage**

```
## S3 method for class 'concentrate':
selectSeries(x,
             series = seq(nrow(concentrator(x)$proj)))
```

**Arguments**

<code>x</code>	a concentrate data object.
<code>series</code>	vector of strings or integers indicating series to select.

**See Also**

[selectSeries](#)

---

 tfplot.concentrate *tfplot Specific Methods*


---

## Description

See the generic function description.

## Usage

```
## S3 method for class 'concentrate':
tfplot(x,
       tf=NULL, start=tfstart(tf), end=tfend(tf),
       series=seq(nseries(x)),
       Title=NULL, xlab=NULL, ylab=NULL,
       graphs.per.page=5, mar=par()$mar, reset.screen=TRUE, ...)
## S3 method for class 'TScanonicalPrediction':
tfplot(x,
       tf=NULL, start=tfstart(tf), end=tfend(tf),
       series=seq(nseries(x)),
       Title=NULL, xlab=NULL, ylab=NULL,
       graphs.per.page=5, mar=par()$mar, reset.screen=TRUE, ...)
## S3 method for class 'TSdataconcentrate':
tfplot(x,
       tf=NULL, start=tfstart(tf), end=tfend(tf),
       select.inputs = seq(length = nseriesInput(x)),
       select.outputs = seq(length = nseriesOutput(x)),
       Title = NULL, xlab = NULL, ylab = NULL,
       graphs.per.page = 5, mar=par()$mar, reset.screen =TRUE, ...)
## S3 method for class 'TSdatareconstitute':
tfplot(x, ...)
```

## Arguments

<code>x</code>	an object to plot.
<code>start</code>	see the generic <code>tfplot</code> .
<code>end</code>	see the generic <code>tfplot</code> .
<code>tf</code>	see the generic <code>tfplot</code> .
<code>series</code>	see the generic <code>tfplot</code> .
<code>select.inputs</code>	see the generic <code>tfplot</code> .
<code>select.outputs</code>	see the generic <code>tfplot</code> .
<code>Title</code>	see the generic <code>tfplot</code> .
<code>xlab</code>	see the generic <code>tfplot</code> .
<code>ylab</code>	see the generic <code>tfplot</code> .
<code>graphs.per.page</code>	see the generic <code>tfplot</code> .
<code>mar</code>	see the generic <code>tfplot</code> .
<code>reset.screen</code>	see the generic <code>tfplot</code> .
<code>...</code>	arguments to be passed to other methods.

**See Also**[tfplot](#)

---

`tfprint.concentrate`*Tfprint Specific Methods*

---

**Description**

See the generic function description.

**Usage**

```
## S3 method for class 'concentrate':
tfprint(x, ...)
```

**Arguments**

<code>x</code>	a concentrate object to print.
<code>...</code>	arguments to be passed to other methods.

**See Also**[tfprint](#)

---

`tframed.concentrate`*Construct a Tframed Object*

---

**Description**

Create a tframed object or set the tframe of an object.

**Usage**

```
## S3 method for class 'concentrate':
tframed(x, tf=NULL, names = NULL)
## S3 method for class 'concentrate':
tframe(x) <- value
```

**Arguments**

<code>x</code>	a (tframed) concentrate object or a concentrate object to be tframed.
<code>tf</code>	a tframe attribute to be applied to x.
<code>names</code>	optional (new) series names to be applied to x.

**Details**

See the generic.

**Value**

A tframed object.

**See Also**

[tframe](#)

---

tfwindow.concentrate  
*tfwindow Specific Methods*

---

**Description**

See the generic function description.

**Usage**

```
## S3 method for class 'concentrate':  
tfwindow(x, tf=NULL,  
         start=tfstart(tf), end=tfend(tf), warn=TRUE)
```

**Arguments**

x	a concentrate object to truncate.
start	A start date of a format compatible with the time series
end	An end date of a format compatible with the time series
tf	A tframe or tframed object
warn	A logical indicating if warning should be produced

**See Also**

[tfwindow](#)

# Index

\*Topic **internal**  
     tframed.concentrate, 19  
 \*Topic **programming**  
     tframed.concentrate, 19  
 \*Topic **ts**  
     00.juice.Intro, 1  
     canonical.prediction, 2  
     checkConsistentDimensions.TSmodelconcentrate, 3  
     checkResiduals.TSdataconcentrate, 4  
     concentrate, 4  
     concentrated.checkResiduals, 7  
     concentrated.nseriesInput, 7  
     concentrated.tfplot, 8  
     concentratedDimension, 9  
     concentratedSeriesNames, 9  
     concentrateOnly, 5  
     concentrateOriginal, 6  
     concentrator, 10  
     end.TScanonicalPrediction, 11  
     estConcentratedModel, 12  
     estProjection, 13  
     l.TSmodelconcentrate, 14  
     nseriesInput.TSmodelconcentrate, 14  
     percentChange.TScanonicalPrediction, 15  
     plot2by2, 15  
     print.concentrate, 16  
     reconstitute, 16  
     selectSeries.concentrate, 17  
     tfplot.concentrate, 18  
     tfprint.concentrate, 19  
     tfwindow.concentrate, 20  
     TSdata.TSdataconcentrate, 2  
 \*Topic **utilities**  
     tframed.concentrate, 19  
     00.juice.Intro, 1  
  
     canonical.prediction, 2  
     checkConsistentDimensions, 3  
     checkConsistentDimensions.TSmodelconcentrate, 3  
     checkResiduals, 4, 7  
     checkResiduals.concentrated  
         (*checkResiduals.TSdataconcentrate*), 4  
     checkResiduals.TSdataconcentrate, 4  
     checkResiduals.TSdatareconstitute  
         (*checkResiduals.TSdataconcentrate*), 4  
     concentrate, 3, 4, 6–13, 17  
     concentrated.checkResiduals, 4, 7  
     concentrated.nseriesInput, 7, 9  
     concentrated.nseriesOutput, 9  
     concentrated.nseriesOutput  
         (*concentrated.nseriesInput*), 7  
     concentrated.tfplot, 8  
     concentratedDimension, 8, 9, 10  
     concentratedSeriesNames, 9  
     concentratedSeriesNamesInput  
         (*concentratedSeriesNames*), 9  
     concentratedSeriesNamesOutput  
         (*concentratedSeriesNames*), 9  
     concentrateOnly, 5, 6, 11  
     concentrateOriginal, 6, 6, 11  
     concentrator, 3, 6, 10, 12  
  
     end, 11  
     end.TScanonicalPrediction, 11  
     estConcentratedModel, 12, 13  
     estProjection, 5, 12, 13  
  
     frequency, 11  
     frequency.TScanonicalPrediction  
         (*end.TScanonicalPrediction*), 11  
  
     is.concentrate(*concentrate*), 4  
     is.concentrator(*concentrator*), 10

`is.TScanonicalPrediction`  
     (`canonical.prediction`), 2  
`is.TSdataconcentrate`  
     (`concentrate`), 4  
`is.TSdataconcentrator`  
     (`concentrator`), 10  
`is.TSdatareconstitute`  
     (`reconstitute`), 16  
`is.TSmodelconcentrate`  
     (`concentrate`), 4  
  
`juice.Intro` (`00.juice.Intro`), 1  
  
`l`, 14  
`l.ARMA`, 14  
`l.SS`, 14  
`l.TSmodelconcentrate`, 14  
  
`nseriesInput`, 14  
`nseriesInput.TSmodelconcentrate`,  
     14  
`nseriesOutput`, 14  
`nseriesOutput.TSmodelconcentrate`  
     (`nseriesInput.TSmodelconcentrate`),  
     14  
  
`percentChange`, 15  
`percentChange.TScanonicalPrediction`,  
     15  
`periods`, 11  
`periods.TScanonicalPrediction`  
     (`end.TScanonicalPrediction`),  
     11  
`plot2by2`, 15  
`prcomp`, 5, 13  
`print`, 16  
`print.concentrate`, 16  
  
`reconstitute`, 2, 5, 13, 16  
  
`selectSeries`, 17  
`selectSeries.concentrate`, 17  
`seriesNames`, 10  
`seriesNamesInput`, 10  
`seriesNamesOutput`, 10  
`start`, 11  
`start.TScanonicalPrediction`  
     (`end.TScanonicalPrediction`),  
     11  
  
`tfplot`, 8, 19  
`tfplot.concentrate`, 18  
`tfplot.TScanonicalPrediction`  
     (`tfplot.concentrate`), 18  
  
`tfplot.TSdataconcentrate`  
     (`tfplot.concentrate`), 18  
`tfplot.TSdatareconstitute`  
     (`tfplot.concentrate`), 18  
`tfprint`, 19  
`tfprint.concentrate`, 19  
`tframe`, 20  
`tframe<- .concentrate`  
     (`tframed.concentrate`), 19  
`tframed.concentrate`, 19  
`tfwindow`, 20  
`tfwindow.concentrate`, 20  
`TSdata`, 2  
`TSdata.TSdataconcentrate`, 2