eL Reference Card
for version 0.1 (Purple Line, Linden)

eLtime

The eL toolbox stores date information in `eLtime` objects. Create one with the `eLtime` command.

`eLtime(`*year*`,`*part*`,`*frequency*`)`	Complete syntax
`eLtime(`*year*`,`*month*`,`*abbreviation*`)`	Monthly syntax
`eLtime(`*year*`)`	Annual syntax
`eLtime`	Default syntax

Complete syntax: Both *year* and *part* must be integers, and *frequency* must be either `'a'` (annual), `'s'` (biannual), `'q'` (quarterly), or `'m'` (monthly). Valid combinations of frequency and part are

`'a'`	`NaN`
`'s'`	$1, 2$
`'q'`	$1, 2, 3, 4$
`'m'`	$1, 2, \ldots, 12$

Monthly syntax: *month abbreviation* must be one of `'jan'`,`'feb'`,...,`'dec'`.

Annual syntax: A single integer argument returns an `eLtime` object with annual frequency.

Default syntax: Calling `eLtime` with no argument returns a default value equal to January, 2001.

Field Manipulation

`year(`*t*`)`	Retrieves integer-valued *year*
`year(`*t*`,`*value*`)`	Sets *year* to *value*
`part(`*t*`)`	Retrieves integer-valued *part*
`part(`*t*`,`*value*`)`	Sets *year* to *value*
`frequency(`*t*`)`	Retrieves character-valued frequency

Other commands for retrieving and setting an `eLtime`'s *part* are `month`, `quarter`, and `semester`. These have the same syntax as `part`.

Vectors and Matrices

`eLtime` objects can be vector and matrix valued. All elements must have the same frequency.

Creation

If *year* and *part* arguments to `eLtime` are identically-sized integer-valued vectors or matrices, then the resulting object inherits their dimensions.
Vectors can also be created with the colon operator.

t1`:`*t2*	Row vector from *t1* to *t2¿t1*
t1`:`*step*`:`*t2*	increment by *step* instead of 1
t2`:`*t1*	Row vector backwards from *t2* to *t1¡t2*.
t2`:-`*step*`:`*t1*	increment by -*step* instead of -1.

Concatination and Transposition

Concatination	
`[`*t1 t2*`]`	Horizontal concatination
`[`*t1*`;`*t2*`]`	Vertical concatination
Transposition	
t`'`	Transpose of *t*
t`.'`	Transpose of *t*

Indexing

Individual elements of a matrix-valued `eLtime` can be retrieved with ordinary Matlab indexing. To retrieve the appropriate range for indices, use

`size(`*t*`)`	Returns $(rows, columns)$ of *t*.
`length(`*t*`)`	Returns $\max\{rows, columns\}$ of *t*.

Arithmetic

Subtraction	
t2`-`*t1*	Number of periods from *t1* to *t2*
t1`-`*j*	`eLtime` *j* periods before *t1*
Addition	
t1`+`*j*	`eLtime` *j* periods after *t1*

Here *t1* and *t2* are conformable and *j* is either an integer or an integer-valued matrix conformable with *t1*. In the later case, addition and subtraction are carried out element-by-element.

Comparisons

t1`==`*t2*	1 if the *year*s and *part*s match; 0 otherwise
t1`≅`*t2*	$1 - (t1\texttt{==}t2)$
t1`>=`*t2*	1 if *t1* equals or follows *t2*; 0 otherwise
t1`>`*t2*	1 if *t1* follows *t2*; 0 otherwise
t1`<=`*t2*	$1 - (t1\texttt{>=}t2)$
t1`<`*t2*	$1 - (t1\texttt{>}t2)$

All comparisons require the operands' *frequency*s to match.

Frequency Conversions

`m2q(`*tm*`)`	Quarterly `eLtime` containing *tm*
`m2s(`*tm*`)`	Biannual `eLtime` containing *tm*
`m2a(`*tm*`)`	Annual `eLtime` containing *tm*
`q2s(`*tq*`)`	Biannual `eLtime` containing *tq*
`q2a(`*tq*`)`	Annual `eLtime` containing *tq*
`s2a(`*ts*`)`	Annual `eLtime` containing *ts*

Here, *tm*, *tq*, and *ts* are monthly, quarterly, and biannual `eLtime`s.

Class Conversions

`double(`*t*`)`	Returns $year + (part - 1)/(parts\ per\ year)$.
`int32(`*t*`)`	Returns *year* for annual *t*; returns $year \times parts\ per\ year + part$ otherwise.
`char(`*t*`)`	Returns string representation of *t*.

Miscellaneous

`ismember(`*t1*`,`*t2*`)`	Indicators for membership in *t2*
`sort(`*t1*`)`	Sort columns in ascending order
`unique(`*t*`)`	*t*'s unique elements
`nowm`	The current monthly `eLtime`
`nowq`	The current quarterly `eLtime`
`nowa`	The current annual `eLtime`

eLdata

The toolbox stores time-series observations in `eLdata` objects.

`eLdata(`*value*`,`*date*`,`*orientation*`)`	Complete syntax
`eLdata(`*value*`,`*date*`)`	Short syntax
`eLdata`	Default syntax

Complete syntax: *value* can be real or complex-valued matrix, *date* must be a vector-valued `eLtime` object and *orientation* must equal `'r'` or `'c'` indicating that time runs along *value*'s rows or columns. *date* and *value* must have the same size on the dimension *orientation*.

Short syntax: Infers *orientation* from the dimension of conformity between *value* and *date*. If *value* is square, it sets *orientation* to `'r'`.

Default syntax: Returns a monthly `eLtime` for the most recent eight quarters with unit values.

Indexing

Index the time dimension with `eLtime` objects and the non-time dimension with integers.

`x[`*t1*`:`*t2*`,`*i*`:`*j*`]`

extracts columns *i* through *j* from date *t1* though *t2*.

`size(`*x*`)`	$[rows, columns]$ of *x*'s value
`length(`*x*`)`	$\max\{rows, columns\}$ of *x*'s value
`Tsize(`*x*`)`	Size of time dimension
`Ksize(`*x*`)`	Number of scalar time series in *x*

Missing Values

Operators marked with ✓ ignore all data from any date with a missing value (`Inf` or `NaN`) in any operand.

Arithmetic

Element-by-Element operations	
x`+`*y*	*x* with values summed with *y*'s
`-`*x*	*x* with negative of values
x`-`*y*	*x* with values differenced with *y*'s
x`.*`*y*	*x* with values multiplied by *y*'s
x`./`*y*	*x* with values divided by *y*'s
x`.^`*y*	*x* with values exponentiated by *y*'s
Scalar operations	
x`+`*s*	*x* with values summed with scalar *s*
s`*`*x*	*x* with values multiplied by scalar *s*
x`/`*s*	*x* with values divided by scalar *s*

The scalar *s* may be complex. All arithmetic operators require their `eLdata` operands' *date*s to equal each other.

Linear Algebra

x'	x with changed *orientation* and transposed values
x.'	x' with complex-conjugate values
[x y]	Horizontal concatination of values ('c')
	Horizontal concatination of dates and values ('r')
[x;y]	Vertical concatination of dates and values ('c')
	Vertical concatination of values ('r')
x'*y	Matrix product of the eLdatas' values ✓
m'*x	Linear combinations of x(*orientation*='r')
x*m	Linear combinations of x (*orientation*='c')

Concatination's operands may have different *dates* if it operates on both dates and values. All other linear algebraic operators require their eLdata operands' *dates* to equal each other.

Data Analysis

Elementary Time Series

zeros(x)	x with values = zeros
ones(x)	x with values = ones
rand(x)	x with values = $U[0,1]$ draws
randn(x)	x with values = $\mathcal{N}(0,1)$ draws
trend(x)	x with values = double(*date of* x)

Sums and Products

sum(x)	sum of x's values. ✓
sum(x,'eLdata')	x with values=sum(x)✓
cumsum(x)	x with values=running sums
prod(x)	product of x's values.✓
prod(x,'eLdata')	x with values=prod(x)✓
cumprod(x)	x with values=running products

Summary Statistics

mean(x)	sum(x)/Tsize(x) ✓
mean(x,'eLdata')	x with values = mean(x) ✓
std(x)	row vector with standard deviations normalized by Tsize(x)−1 ✓
std(x,0)	row vector with standard deviations normalized by Tsize(x) ✓
std(x,1)	same as std(x)✓
std(x,j,'eLdata')	x with values = std(x,j) ✓
cov(x)	Variance-covariance matrix of x normalized by Tsize(x)−1 ✓
cov(x,0)	Variance-covariance matrix of x normalized by Tsize(x) ✓
corrcoef(x)	Correlation coefficient matrix of x normalized by Tsize(x)−1 ✓
corrcoef(x,0)	Correlation coefficient matrix of x normalized by Tsize(x) ✓

Filters

hptrend(x)	x with values = HP trend ($\lambda = 1600$)
hptrend(x,*lambda*)	Changes λ from 1600 to *lambda*

Plotting

plot(x) creates a figure with the individual time series of x plotted in separate panels.

'nber'	Shade NBER-defined recesions
'single'	Place all plots into a single panel
'xticks',t	Place x-tick marks on dates in t
'yticks',y	Place y-tick marks on values in y
'ylim',[*ylow*,,*yhigh*]	Set y-axis limits to *ylow* and *yhigh*
'forecast',t	Plot data after t with dashed line to indicate forecast

plot also accepts options valid for Matlab's standard plot command.

Mathematical Functions

All of the trigonometric, complex, and exponential functions operate on eLdata objects by passing through to their *values*.

L

The toolbox stores rational lag polynomials as L objects. The set of matrix-valued polynomials with a scalar-valued denominator is closed under addition, multiplication, concatination, and matrix inversion, so each L object has

top	its matrix-valued numerator,
bottom	its scalar-valued denominator.

These polynomials' coefficients may be complex values, and negative powers of L are permitted.

Creation and Scalar Arithmetic

The function L called with no arguments creates the scalar lag-polynomial with numerator L and denominator 1. Use arithmetic operations to build more complex scalar polynomials.

Polynomial -Constant (A-c) operations

c*A	Multiplication
A.^j	Exponentiation (j integer)
A/c	Division
c+A	Addition
-A	Unary minus

Polynomial-Polynomial operations

A*B	Polynomial multiplication
A/B	Polynomial division
A+B	Polynomial addition

Linear Algebra

[A B]	Horizontal concatination
[A;B]	Vertical concatination
A'	Transposition
A.'	Conjugate transposition
A*B	Matrix multiplication
det(A)	Determinant
inv(A)	Matrix inverse

Polynomial-constant and polynomial-polynomial addition and subtraction are also defined for matrix-valued polynomials.

Data Multiplication

Time-series analysis defines

$$(a_{-M}L^{-M} + \cdots + a_0 + \cdots + a_M L^M) \times x_t = a_{-M}x_{t+M} + \cdots + a_0 x_t + \cdots + a_M x_{t-m}$$

- In the eL toolbox A*x creates an eLdata object with the linear combination of x and its lags and leads given by the polynomial coefficients in A.
- If A's denominator differs from 1, then the appropriate linear combination uses infinite leads and lags. In this case A*x yields an eLdata object with NaN values.
- If x has orientation r, then A must premultiply it and either be scalar or have columns=Ksize(x).
- If x has orientation 'c', then A must postmultiply it and be either scalar or have rows=Ksize(x).

Indexing

A(i,j)	Submatrix polynomial from rows i and columns j
A{'T'}	Numerator of A
A{'B'}	Denominator of A
A.v(z)	Evaluation of A at complex scalar z
A.c(j)	Matrix of coefficients multiplying L^j after denominator expansion

Denominator expansion fails if the denominator has one or more roots on the unit circle.

Starting and Stopping eL

eLstart	Start the toolbox
eLstop	Stop the toolbox
eLstatus	Determine whether the toolbox is started or stopped.

These commands help avoid name space conflicts with other programs that might name a variable or function "L".

Example

This example downloads NIPA data, calculates real GDP's growth rate and HP cycle, plots them, and calculates their standard deviations.

```
tstart=eLtime(1953,4,'q'); tend=eLtime(1983,4,'q');
urlwrite('http://www.bea.gov/bea/dn/nipaweb/...
    GetCSV.asp?GetWhat=SS_Data/Section1All_xls.xls...&
    Section=2','Section1All_xls.xls');
[Nums Wrds]=xlsread('Section1All_xls.xls','10106 Qtr');
t=eLtime(Nums(1,4:end),Nums(2,4:end),'q');
d=eLdata(Nums(3:end,4:end),t,'r');
loggdp=100*log(d(1,tstart:tend));
dlgdp=(1-L)*loggdp;
hpgdp=loggdp-hptrend(loggdp);
plot([dlgdp; hpgdp],'nber','single',...
    'legend',strvcat('Growth Rate','HP Cycle'));
legend('boxoff');
std(dlgdp)
std(hpgdp)
```

Copyright © 2008 Federal Reserve Bank of Chicago