FUNCTIONAL DATA ANALYSIS: INTRO TO R's FDA

EXAMPLE IN R...fda.txt

DOCUMENTATION: found on the fda website, link under software (http://www.psych.mcgill.ca/misc/fda/)
Nice 2005 document with examples, explanations.

CODE: available from R-CRAN or the above website (in R: use "package" menu to download and install).

ERRORS IN FDA: Please let me know about errors, for posting to Jim Ramsay.

STEPS IN USING FDA

- choose basis and "set up basis functions": might depend on range of t values, but not on y values or specific t-values
$\Rightarrow$ basisfd (a basis object)

- turn vectors into functions using data (t's and y's) and basisfd.
This can be done by least squares or by "lightly smoothing" the data.
$\Rightarrow$ datafd (a functional data object)

- plot, summarize with pointwise means and standard deviations using datafd

- align (warp), functional principal components, linear discriminant analysis, derivatives, ...

basisfd: Basis Object (class=bs)

Example:

```
basisfd <- create.bspline.basis(rangeval=c(0, 1),
   nbasis=NULL, norder=4, breaks=NULL)
```

You must specify

- type of basis: Fourier, B-spline, power, constant (one function $\phi(t) \equiv 1$), exponential ($\phi_j(t) = \exp(\alpha_j t)$), polygonal (piecewise linear), polynomial
- range $= c(a, b)$: t values are in interval [a,b]
- number of basis functions (or something that determines this, like number of knots)
- some parameters (depends on basis chosen): eg knot values, degree, period (for Fourier series)
- Miscellaneous: dropind : leave out basis functions; quadvals and values: used for integrals and derivatives of basis functions

BSPLINE BASIS: parameters are similar, but not identical to, *bs*

```
create.bspline.basis(rangeval=c(0, 1), nbasis=NULL,
  norder=4, breaks=NULL)
```

RANGEVAL – range for independent variable, default is [0,1]
BREAKS=KNOTS
– include interior knots and boundary knots, in increasing order, boundary knots must equal RANGEVAL
– default: equally spaced with RANGEVAL as boundary knots
NORDER (must be between 1 and 20)
– is the degree + 1 (order = 4 means piecewise cubic)
NBASIS = number of basis functions to use

```
create.bspline.basis(rangeval=c(0, 1), nbasis=NULL,
  norder=4, breaks=NULL)
```

NOTE: we must have
nbasis = degree + # knots + 1 = norder + length(breaks) - 2

We needn't specify all three: nbasis, order, breaks

We won't always get errors if we specify all, but with
nbasis not equal to norder + length(breaks) - 2.

R code...

datafd: Functional Data Object

- turn vectors into functions using data (t's and y's) and basisfd.

This can be done by 'lightly smoothing" the data:

```
datafdPar <- fdPar(basisfd, 2, lambda)  ## info on smoothing
datalist <- smooth.basis(x, y, datafdPar)  ## data
datalist$fd  # this is the functional data object
```

or by least squares:

```
data2fd(y, argvals=seq(0, 1, len = n), basisfd,
   fdnames=defaultnames, argnames=c("time", "reps", "values")
```

'

Turn vectors into functions using data (t's and y's) and basisfd (continued)

data2fd can handle NA's, can handle different t-values for each individual.
Since it's least squares (no penalty), we can get some undesirable results.

smooth.basis: can't have NA's, must have same t's for each individual.

POSSIBLE ACTIONS:

- Use smooth.basis.
- Check data2fd fit - looks OK? Great.
- Lightly smooth data "outside of" fda library. Then use smooth.basis.
- "Outside of" fda library: interpolate to fill in missing data values.

GOAL: do not change data much at all. This is initial processing.

Output: a basis class object, coeff, and fdnames
(fdnames is optional input, too - for plotting)

FDNAMES: list of length 3 used for labelling plots

– first: argument value (eg ‘age’), default = ‘time’

– second: a vector for replication (eg ‘mouse id’), default = ‘reps1, reps2..’

– third: response (eg ‘body mass’), default = ‘values’

Difference between fdnames and argnames in data2fd??

- fdnames[2] is a vector (one entry per curve)

- argnames[2] is the name of fdnames[2] eg “mouse”

data2fd and smooth.basis are also used for turning differential operator into a function.

smooth.basis

```
datafdPar <- fdPar(basisfd, 2, lambda)  ## info on smoothing
datalist <- smooth.basis(x, y, datafdPar)  ## data
datalist$fd  # this is the functional data object

fdPar(fdobj=fd(), Lfdobj=int2Lfd(0), lambda=0, estimate=TRUE,
          penmat=NULL)
```

Lfdobj = integer or differential operator (gives penalty on function)
Lfdobj = 2 $\Rightarrow$ penalty = $\int [f'']^2$

```
smooth.basis(argvals, y, fdParobj, wtvec=rep(1,n),
       dffactor=1,fdnames=list(NULL, dimnames(y)[2], NULL)
```

R code ...

data2fd

```
data2fd(y, argvals=seq(0, 1, len = n), basisfd,
   fdnames=defaultnames, argnames=c("time", "reps", "values")
```

This fits by ordinary least squares, not penalized least squares.

Y, ARGVALS (NA's permitted - only in y??); $nrep$ = number of reps/curves

- if all individuals are observed at same argument values, $t_1, \ldots, t_n$,

y is the n by $nrep$ matrix of responses and argvals can be $(t_1, \ldots, t_n)$

- if all individuals are observed at same argument values, $t_1, \ldots, t_n$, but

with some missing values, do as above but use NA's at missing y-values

- if individual i is observed at $(t_{i1}, \ldots, t_{in})$ (same length for all individu-

als): both y and argvals are n by $nrep$.

- if individual i is observed at $(t_{i1}, \ldots, t_{in_i})$:

- let $T_, \ldots, T_K$ be the union of all distinct t-values.

- y is K by $nrep$ with many NA's, argvals is K by $nrep$.

Some Other Objects/Classes

- bivariate functional data class *bifd* for functions of two variables
- Linear differential operator object (via Lfd)
- functional parameter object (via fdPar)

SUMMARY/GRAPHICS COMMANDS FOR FD OBJECTS

PLOT

```
plotFd(fd,Lfd,matplt=TRUE,href=TRUE,nex,...)
```

- fd is a functional data object to be plotted
- Lfd: what derivative do you want to plot? (0 = function, 1, 2, ...). We can make this a complicated differential operator: eg plot $m''(t) - sin(2\pi t) * m(t)$ ....
- matplt = T plot all curves at once, = F plot one at a time
- href = T plot x-axis, = F omit x-axis

MEAN, SD, CENTER, VAR

- meanFd(fd): point-wise mean
- std.fd(fd): point-wise sd's
- center.fd(fd): subtract point-wise average from all curves
- var.fd(fd1,fd2): a bivariate functional data object

R code ....