

Documentation for the R-code to implement the methodology in “Moments Based Functional Synchronization”

GARETH M. JAMES*

The R directory contains six functions. The main function that calls the others is “curvesync”.

Description

This function takes a set of curves and attempts to synchronize them according to the three CAM and CAMP procedures outlined in the paper.

Installing

To install the software download the file curvesync, open R and type

```
> source(``filename``)
```

where filename is the name you saved the file under (don’t forget to give the directory structure in addition if needed.) Now if you type ls() you should see

```
> ls()
[1] "calcmoment"      "criterion.mix"  "criterion.mom" "curvesync"
[5] "curvesync.mix"  "curvesync.mom" "testY"
```

testY is a test set of curves. The other objects are all functions. Try the following commands one at a time.

```
> lincam <- curvesync(testY,linear=T,k=3,df=45,lambda=10000)
> nonlincam <- curvesync(testY,linear=F,k=3,df=45,lambda=10000)
> camp <- curvesync(testY,linear=T,k=3,df=45,lambda=1,lambda.p=100000)
```

After each command you should see three plots. The first corresponds to the original curves (the red line is the cross-sectional mean), the second plot is the synchronized curves and the final plot gives the synchronization functions. The first command implements the linear CAM method, the second line gives non-linear CAM and the final line CAMP. Try typing

```
> lincam$cam$sync
[1] 0.03323393
> nonlincam$cam$sync
[1] 0.009586423
> camp$camp$sync
[1] 0.002944886
```

*Marshall School of Business, University of Southern California

These numbers are measures of the quality of the synchronization (see below for details) with low numbers indicating a good synchronization. We see here that CAMP seems to be giving the best fit. Try changing the values of lambda and lambda.p and see how the quality of the fit changes. In practice you will probably want to try several different values of lambda and lambda.p on any given set of curves to find the optimal level of synchronization.

Note that curvesync turns off error messages. If you need these for debugging purposes you can turn them back on with the command

```
> options(show.error.messages=T)
```

Arguments

curvesync allows for up to seventeen inputs. They are

Y : This should be a T by N matrix of the observed curve values, where T is the number of time points that the curves have been observed over and N is the number of curves.

realtime : This is a vector of length T specifying the time points that the curves have been observed over. By default it is just a vector of evenly spaced points between 0 and 1.

k : This is the number of moments to synchronize on i.e. the procedure will attempt to synchronize the moments 1 through k . By default $k = 3$.

q : This controls the number of degrees of freedom used in the b-spline basis for parameterizing $X(t)$. The larger q is the more flexibility there is in the synchronization function. By default $q = 5$.

trans : This is an N by $q + 2$ matrix with each row corresponding to the coefficients parameterizing the corresponding function $X(t)$. This will generally not be fed to the function. But one of the outputs of curvesync is trans so if trans (computed from a previous run) is fed to the function then the function will not resynchronize the curves but instead plot the previously calculated synchronized curves and their synchronization functions.

plotgraphs : This is a true/false variable indicating whether the various plots (original curves, synchronized curves and synchronization functions) should be produced. The default is T.

lambda : This is a scalar or vector of λ_k 's for the moments criterion. The larger the λ_k 's the more the moments are forced to equate. lambda should either be a scalar, in which case the same weight is applied to all moments, or a vector of length $k - 1$ in which case different weights are applied to each of the second, third etc. moments (the first moment is always exactly equated). The default is lambda=100. One should be careful not to make the λ_k 's too large or there is a tendency to get large changes in the shape of the curves. Note that the lambda values are divided by the average moments over the curves. For example λ_2 is divided by the average of $\mu_{q_i}^2$ over all N curves. This has the effect of standardizing the weight according to the size of the moments.

diags : This is a true/false variable indicating whether to print the output code from the non linear optimization function. This will tell you whether there are any convergence problems with the function. The code will be an integer between 1 and 5 indicating why the optimization process terminated.

1. relative gradient is close to zero, current iterate is probably solution.
2. successive iterates within tolerance, current iterate is probably solution.
3. last global step failed to locate a point lower than 'estimate'. Either 'estimate' is an approximate local minimum of the function or 'stepol' is too small.
4. iteration limit exceeded.
5. maximum step size 'stepmax' exceeded five consecutive times. Either the function is unbounded below, becomes asymptotic to a finite value from above in some direction or 'stepmax' is too small.

See the nlm documentation for further details. The default is F.

- linear** : This is a true/false variable indicating whether the linear CAM synchronization procedure (the first of the three methods outlined in the paper) is to be used. The default is T.
- df** : As a preprocessing step every curve is smoothed using a smoothing spline. This is important to ensure reasonable first derivatives for the information function. df controls the degrees of freedom for the smoothing spline. Small numbers will smooth the curves towards a straight line while large numbers will leave the curves relatively unchanged. If the data is already fairly smooth you should use a large value for df. See the smooth.spline documentation for further details. The default is df=10.
- shift** : This is a true/false variable that allows for simply shifting the curves to equate the first moments. This is like linear except linear performs a linear stretch to also force the second moment to equate.
- CAM** : This is a true/false variable that indicates whether one of the two CAM methods will be fit. For example if CAM=T and linear=T then linear CAM is fit. If CAM=T and linear=F then the non-linear CAM is fit. If CAM=F then neither are fit. The default is CAM=T.
- lambda.p** : This is the value for λ_p the weight placed on the Procrustes part of the optimization criterion. If lambda.p=NULL then the linear or non-linear CAM procedure is fit (depending on whether linear equals T or F). If lambda.p is not NULL then the CAMP procedure is fit (possibly in addition to the linear or non-linear CAM approach depending on the value of CAM). If lambda=0 and cs.targe=T (see below for details) then CAMP reduces down to the standard Procrustes method of Ramsay and Lee (1998) and the target function is estimated as the cross-sectional mean. The default is NULL.
- target** : This allows for the manual specification of a target function for the CAMP or Procrustes methods. By default this is NULL in which case the target is estimated as outlined above.
- maxit** : This variable specifies the maximum number of iterations to use in the non-linear optimizer for each curve when performing CAMP. The CAMP algorithm can run a lot slower than the two CAM procedures so, if you have a lot of data, you may want to start with a low number and then increase this as feasible.
- maxrange** : Occasionally the non-linear CAM or CAMP optimizers get stuck in a local minimum where $X(t)$ essentially shrinks to a vertical line forcing the synchronized curve to a single point mass. To prevent this happening there is a penalty term in the optimizer. The penalty is calculated using

$$\text{penalty} = 10^3(\text{div} - \text{maxrange})I(\text{div} > \text{maxrange})$$

where

$$div = \frac{|X(t_T) - t_T| + |X(t_1) - t_1|}{t_T - t_1}.$$

div is a measure of the deviation of the starting and ending points of the synchronized curve from the unsynchronized curve. Hence if *maxrange* is small this forces the starting and ending points to be very similar. If *maxrange* is large then there is little constraint. The default is *maxrange*=0.8 which allows for an average of 40% deviation in each of the start and end times before applying any penalty. In practice I have found that this value prevents the convergence problems while allowing flexibility in the fit.

cs.target : This is a true/false variable that indicates whether to use the cross sectional mean of the observed curves (T) or use linear CAM first and take the cross sectional mean of the partially synchronized curves (F). The former option with *lambda*=0 essentially gives the straight Procrustes method. I have generally found better results using a partial synchronization first so the default is *cs.target*=F.

Value

curvesync returns two lists, *cam* and *camp*, with *cam* corresponding to the output from a linear or non-linear CAM fit and *camp* to a CAMP or Procrustes fit. Each list contains six components.

trans : These are the coefficients used to define the estimated synchronization functions $X(t)$. If you recall *curvesync* using these values the function will use *trans* to recompute the $X(t)$ and plot the synchronized curves and synchronization functions.

YofX : These are the smoothed and synchronized curves computed over the original vector *realtime*. *YofX* has the same dimension as the original matrix *Y*.

X : This is a T by N matrix with each column corresponding to the estimated synchronization function $X(t)$.

mom : This is a k by N matrix containing the first k moments for each of the original curves. This can provide a useful summary of the differences between the curves in terms of the time axis.

Y : Smoothed versions of the original curves.

sync : A cross-validated measure of the level of synchronization of the new curves versus the original data. For each curve we calculate

$$sync_i = \frac{\int [Y_i(X_i(t)) - \overline{Y(X(t))}]^2 dt}{\int [Y_i(t) - \overline{Y(t)}]^2 dt}$$

where $\overline{Y(X(t))}$ and $\overline{Y(t)}$ are respectively the cross-sectional means on the synchronized and on the unsynchronized curves excluding the i th curve. *sync* is then calculated by taking the average over $sync_i$. Values of *sync* close to zero indicate a high level of synchronization relative to the original curves. Values near one indicate similar levels of synchronization between the new and old curves.