November 24, 2020

# Computing Elastic Shape Distances between Curves in d-dimensional Space

This README file describes the zip file by the name ESD_alldim.zip which contains a software package that incorporates the methods presented in the manuscript entitled

> *On Computing Elastic Shape Distances between Curves in d-dimensional Space*
> by Javier Bernal, James Lawrence, Gunay Dogan, Charles Hagwood; currently in preparation.

This software computes the elastic registration of two simple curves in d-dimensional space, d a positive integer, and the elastic shape distance between them.

Javier Bernal
National Institute of Standards and Technology (NIST)
Gaithersburg, MD
Javier.Bernal@nist.gov

---



---

The implementation is in Matlab with the exception of the Dynamic Programming routine which is written in Fortran but is executed as a Matlab mex file.

With the exception of the Matlab driver routine, ESD_driv_3dim.m, which is designed for the case d = 3, all Matlab routines in the package can be executed for any d if the current driver routine is adjusted or replaced to handle the value of d.

However, parameter dimx in the Dynamic Programing Fortran routine DP_MEX_WNDSTRP_ALLDIM.F may have to be modified so that instead of having a value of 3, it has the value of d. The Fortran routine must then be processed to obtain a new mex file for the routine by typing in the Matlab window:

```
mex -compatibleArrayDims DP_MEX_WNDSTRP_ALLDIM.F
```

Since ESD_driv_3dim.m is the driver routine for the 3-dimensional case of the package, the package can be executed for this case by simply typing in the Matlab window:

Of course, before doing this, ESD_driv_3dim.m must be edited if necessary, so that when executed, it reads the appropriate data corresponding to the two curves under consideration.

As will be described below, values that are assigned to certain variables in this routine may have to be modified in the routine as well before the execution of the package.

Given curves beta1 and beta2 in d-dimensional space, and positive numbers T1 and T2, such that beta1 and beta2 can be interpreted to be continuous functions from [0,T1] and [0,T2], respectively, into the d-dimensional Euclidean space, and assuming beta1 and beta2 are the curves under consideration, discretizations of the two curves are first read by the driver routine ESD_driv_3dim.m once the package is executed.

Irrespective of the value of d, the program then proceeds to Matlab routine ESD_comp_alldim.m which computes an approximation of the length of each curve by computing the length of each line segment joining consecutive points on the curve in the discretization of the curve and adding these lengths, and then scales the two curves so that each curve has approximate length 1 (each point in the discretization of each curve is divided by the approximate length of the curve).

The routine then scales, if any, the two partitions of [0,T1] and [0,T2] that discretize the curves so that they become partitions of [0,1], or if no partitions are given, creates two partitions of [0,1], one for each curve, according to the number of points in the discretization of each curve, the discretization of each curve then assumed to be the result of discretizing the curve by the corresponding partition, each partition uniform if at least one curve is closed, each partition parametrizing the corresponding curve by arc length otherwise.

Utilizing the given or created partitions and the discretizations of the curves, with the exception of the case in which d equals 1 and both curves are open, the routine then creates a common partition of [0,1] for the two curves and discretizes each curve by this common partition using cubic splines.

If at least one curve is closed, and the numbers of points in the first curve and second curve are N and M, respectively, letting L equal the larger of N and M, the routine then takes the common partition to be the uniform partition of [0,1] of size equal to L. This is in accordance with the requirement established in Section 4 of the manuscript mentioned above that if at least one curve is closed (the set of starting points of one of the curves will have more than one point), in order to compute the elastic shape distance and registration in the appropriate manner, the curves should be discretized by the same uniform partition. Note that the routine then identifies a set of starting points of one of the curves having more than one point, satisfying that it is the discretization by the uniform partition of one of the curves (a closed curve), or a subset of it.

On the other hand, if both curves are open, d not equal to 1, the routine takes the common partition to be the union of the two partitions discretizing the curves minus certain points in this union that are eliminated systematically so that the distance between any two consecutive points in the common partition does not exceed some tolerance. This is in accordance with the

requirement also established in Section 4 of the manuscript that if both curves are open (each curve has exactly one starting point), d is not equal to 1, in order to compute the elastic shape distance and registration in the appropriate manner, the curves should be discretized by the same partition, a partition not necessarily uniform.

Finally, if both curves are open and d equals 1, the routine does not create a common partition and the curves continue to be discretized by the same given or created partitions.

Routine ESD_comp_alldim.m does require that values be assigned to a couple of variables in the driver Matlab routine ESD_driv_3dim.m

These variables go by the names: dmn_t_is_read, both_drct:

dmn_t_is_read - for determining whether the partitions of the curves are read from the same files that contain the discretizations of the curves or are computed by the program (1 if read, 0 if computed)

both_drct - for determining whether the first curve is to be considered in the reversed direction as well; as described in the Introduction section of the manuscript sometimes the curves are not defined in the proper directions so that everything must be done twice; first with the curves as they are and then with one curve reversed (0 if no, 1 if yes)

Once Matlab routine ESD_comp_alldim.m is done, the actual computations of the elastic shape distance and registration are carried out by Matlab routine ESD_core_alldim.m in which all of the procedures presented in Section 4 and Section 5 of the manuscript have been implemented.

The way in which the procedures just mentioned are executed in routine ESD_core_alldim.m, depends on the values assigned to certain variables in the driver Matlab routine ESD_driv_3dim.m

These variables go by the names: stage, iten, space, itop:

stage - level of methodology used; if set to 1, an L^2 type distance is minimized with an iterative procedure that alternates computations of optimal diffeomorphisms with computations of optimal rotation matrices, one starting point of one of the curves at time; this is the original (slow) way; if set to 2, the L^2 type distance is minimized with an iterative procedure that alternates computations of optimal diffeomorphisms (a constant number of them per iteration) with the successive computation of optimal rotation matrices for all starting points of one of the curves; this is the new (fast) way but without the FFT; if set to 3 and both curves are closed, the L^2 type distance is minimized with an iterative procedure that alternates computations of optimal diffeomorphisms (a constant number of them per iteration) with the successive computation of optimal rotation matrices using the FFT for all starting points of one of the curves; this way is a lot faster than without the FFT; if set to 3 and at least one of the curves is not closed, it is modified to equal 2

iten - maximum number of while loop iterations allowed in the procedure used in the routine if stage is set to 2 or 3, the choice of the procedure depending on the value of stage

space - spacing plus 1 between starting points on a closed curve, if any, a curve that is rotated; if spacing is 0 (space - 1), then the set of starting points on the curve is exactly the discretization of the curve; otherwise using the spacing it is a subset of this set, i.e., a subset of the discretization of the curve

itop - constant number of times Dynamic Programming algorithm is executed per iteration of while loop in the procedure used in the routine if variable stage is set to 2 or 3; for curves of simple curvature this variable set to 1 usually suffices

On output, in the driver routine ESD_driv_3dim.m, certain variables will have values that can be used to identify the elastic shape distance between the two curves under consideration, their elastic registration, and some other information. They are

clsd_crve - number of closed curves identified (0, 1 or 2)

whch_crve - if only one curve is closed, identifies the curve (1 or 2, i.e., first or second curve); if both are closed it is set to 1 ,i.e., identifies the first curve; if neither one is closed it is set to 0; note that the curve identified by whch_crve, if any, is the one that is rotated while the other curve is reparametrized

drct_crve - if equal to 1, even if the first curve was considered in the reverse direction as well, it is the direction given on input that gives the optimal solutions; otherwise the first curve was indeed considered in the reverse direction and it is in this direction that the optimal solutions were obtained

t1, t2 - given that the discretizations of the original first and second curves have N and M points, respectively, after being processed (scaled, etc.) in Matlab routine ESD_comp_alldim.m, these are 1xN and 1xM arrays, respectively, that contain the partitions of [0,1] that discretize the first and second curves, respectively.

f1, f2 - 3xN and 3xM arrays containing the discretizations of the first and second curves, respectively, after being processed (scaled, etc.) in Matlab routine ESD_comp_alldim.m, discretized by partitions t1, t2, respectively, neither one rotated or parametrized

t - given that the discretized rotated curve has N points, this is a 1xN array that contains the partition of [0,1] that discretizes the rotated curve

f1best, f2best - 3xN arrays containing discretizations of the curves that are interpreted to achieve the elastic registration of the two curves, both discretized by the partition t; depending on variable whch_crve, they are the result of rotating and reparametrizing

the two curves; one is rotated while the other one is reparametrized; f1best corresponds to the first curve while f2best corresponds to the second one

iter - number of while loop iterations that took place in the procedure used in Matlab routine ESD_core_alldim.m if stage is set to 2 or 3

strt_crve - index of optimal starting point in curve that is rotated

R - optimal rotation matrix; a 3x3 array

gammabest - optimal diffeomorphism; with N as for t above, a 1xN array

minE - the actual computed elastic shape distance between the two curves under consideration

Besides the values of these variables, on output, driver routine ESD_driv_3dim.m will also produce some Matlab files for plotting.

The zip file also contains a few other files that are not part of the software package described above. Some of these files are Fortran and Matlab programs for creating input data files for the software package and for creating some particular plots using output data from the package. One Fortran program (helix_gen.f) is for generating discretized helices, and another one (sph_ellips.f) is for generating dicretized spherical ellipsoids. All of these additional programs are short and simple and can be modified without much effort. Actually all of the Fortran and Matlab routines and programs in the zip file are essentially short and simple and can be easily read and modified. The exception is the Dynamic Programming Fortran routine which should only be modified when setting dimx as described above, and when setting parameter nmax, the maximum number of points in the discretization of a curve. Note that because there is a limit to the maximum size an integer variable can have and there are some calculations in the routine that can exceed this limit, it has been established that nmax should not exceed 46340.

As mentioned above, the software package is executed for the 3-dimensional case by typing ESD_driv_3dim in the Matlab window. Currently the driver routine is set up to read files ellip1.txt and ellip2.txt which contain the discretizations of two spherical ellipsoids, which we call the first and the second ellipsoid, respectively. These files were created with program sph_ellips.f, the first file containing 46001 points, the seconnd file 45001. The first ellipsoid has the positive z-axis as its axis of rotation while the second one has the positive x-axis. The shapes of the two ellipsoids are essentially the same, so that the elastic shape distance between them should be essentially zero with the first ellipsoid rotated in their elastic registration so that its axis of rotation is very close to the positive x-axis. See Section 6 of the manuscript for more details about this particular situation and the definition of a spherical ellipsoid. Also the way in which variables in the driver routine are set to values can be easily read in the routine itself. Two other files in the zip file are helix1.txt and helix2.txt that were generated with program helix_gen.f and contain discretizations of two helices. Again see Section 6 for more details about using these files, the definition of a helix, and the results obtained.

Again it is highly recommended, with the exception of the Dynamic Programming Fortran routine which happens to be extremely intricate, to open any file and look at its contents for gaining understanding and clarification of what the file is about. Also it is recommended to study Section 6 of the manuscript.