

averdtw.dll Introduction

General

averdtw.dll calculates an average signal from several varying, time dependent signals, using a dynamic time warping procedure.

Simplified description: k signals are first time-normalized and then time-distorted (warped) so that the summed squared errors between all of them become a minimum. All distorted signals are treated statistically, obtaining different output signals: arithmetic mean, median, minimum, maximum, 25 percentiles, 75 percentiles.

For details see:

Bender A., Bergmann G., Determination of Typical Patterns from Strongly Varying Signals, [Comput Methods Biomech Biomed Engin.](#) 2011 Jul 4. [Epub ahead of print]

Data Structure

Example: The signal $A_k(i, j)$ consists of 4 sub-signals: The resultant force F and its components Fx, Fy, Fz in directions x, y, z. It is measured 5 times. Each of the $p = 5$ measurements is 1 second long and consists of 1001 data sets.

Each measurement $k = 1 \dots p$ ($p = 5$) is stored in a 2-dimensional, null-terminated array $A_k(i, j)$ with the indices

$i = 0 \dots m = 0 \dots 4$ = time, F, Fx, Fy, Fz = sub-signals
 $j = 0 \dots n = 0 \dots 1000$ = data sets 1 ... 1001 = data sets in each measurement

The time can be measured using any time base (e.g. milliseconds, hours ...). It must start with zero. The time intervals don't have to be equidistant. The number m of sub-signals is not limited but must be the same for each measurement. The number n of data sets can be different for each measurement.

Example:

$i \rightarrow$	<i>time (ms)</i>	<i>F (N)</i>	<i>Fx (N)</i>	<i>Fy (N)</i>	<i>Fz(N)</i>	
	0	3.74	1.00	2.00	3.00	
	1.040	10.77	4.00	6.00	8.00	
	1.095	21.95	7.00	12.00	17.00	
	3.093	32.31	12.00	18.00	24.00	
$A_k(i, j) =$						k = number of measurement
	1000	3.74	1.00	2.00	3.00	

Input for the dll:

Arrays $A_k(i,j)$ from p different measurements.

Output for the dll:

Array $A_{out}(i,j)$ with average (median/min etc) signal, having the same data structure as $A(i,j)$.

Sub-signal used for warping

Warping is done with one of the sub-signals i. All other sub-signals are then distorted identically before the average/ median etc. signals are calculated. That signal i must be taken for warping which is most representative for all sub-signals. In case of a spatial force this is the resultant force F.

System Requirements

DLL Type averdtw.dll

+ Windows XP

+ DLL registration:

averdtw.dll must be registered on your computer. Either you can do this in the MS-DOS command prompt or by pressing "start/execute...". You need the dll path for registration.

registration command: regsvr32 "path\averdtw.dll"

deregistration command: regsvr32 /u "path\averdtw.dll"

with "path" being the averdtw.dll path.

Example: path = "C:\average_dtw"

command for the registration: regsvr32 "C:\average_dtw\averdtw.dll"

command for the deregistration: regsvr32 /u "C:\average_dtw\averdtw.dll"

DLL Type averdtw8.dll

+ Windows XP with .NET 2.0, Vista, Windows 7

Calculation Parameters

The following parameters can be chosen:

DTWComp - number i of that sub-signal which is used for calculation of dynamic time warping path (integer). Standard value DTWComp = 1

NumOfIntervals - number j of data sets in the average signal (integer). Standard value = 600

WinConstraint - windows constraint for dynamic time warping matrix in percent (integer). Standard value = 20 (details see Bender and Bergmann 2011).

Filter

all sub-signals i in the output signal $A_{out}(i, j)$ can be smoothed by a single pole low-pass filter.

The filtered output array $A_{smooth}[i, j]$ is

$$A_{smooth}[i, 0] = A_{out}[i, 0]$$

$$A_{smooth}[i, j] = \text{smoothing factor}/100 * A_{smooth}[i, j - 1] + (100 - \text{smoothing factor})/100 * A_{out}[i, j - 1]$$

Recursive with $j = 1$ to n and smoothing factors between 0 and 100.

SmoothAver - smoothing factor for the average signal

SmoothMedian - smoothing factor for median signal

Smooth25Perc - smoothing factor for 25 percentile signal

Smooth75Perc - smoothing factor for 75 percentile signal

SmoothMax - smoothing factor for maximum signal

SmoothMin - smoothing factor for minimum signal

DLL functions

StartNewCalc()	- preparation for new calculation of average signal
AddOneSignal(dblNewSignal())	- add new signal k
CalculateDTWAver(dblAverSignal())	- start calculations
AverDTW()	- array with average signal
MedianDtw()	- array with median signal
MinDtw()	- array with min values signal
MaxDtw()	- array with max values signal
Perc25Dtw()	- array with 25 percentile signal
Perc75Dtw()	- array with 75 percentile signal
Comment	- comment to evaluation progress
CalcReady	- true = all signals are calculated
AverDtwError	- = 0 - ok; <> - error in the calculating process