# Software Implementation for the recognition of the Sit-To-Stand (STS) events.

The categorization software was implemented in MATLAB [MATLAB. *(R2020a)*. Natick, Massachusetts: The MathWorks Inc.] and includes three main sub-functions to identify the significant events in the STS motion pattern.

## Initiation event

### METHODS

The *TrunkMOV.m* function recognises the Initiation event, as the beginning of the Trunk Leaning phase. As a first step, the Ground Reaction Force (GRF) profile is segmented into *n* time epochs of $N_e$ samples. Epochs are temporally defined by their middle sample $t_i$, with $i$ =1,…$n$. The software calculates the standard deviation of the force across each epoch as:

$$(1)\ \sigma_{ep}(t_i) = \sqrt{\frac{1}{N_e-1} \cdot \sum_{j=1}^{N_e} (F_{j,i} - \bar{F}_i)^2}$$

where $F_{j,i}$ is the $j$-th force sample of the $i$-th epoch and $\bar{F}_i$ is the respective mean force. The resulting $\sigma_{ep}(t_i)$ values are then averaged using a moving mean of 10 epochs and normalized over the sequence baseline, $B_{TL}$:

$$(2)\ \bar{\sigma}_{ep}(t_i) = E\{\sigma_{ep}(t_{i-5}), \dots, \sigma_{ep}(t_{i+4})\} \cdot \frac{1}{B_{TL}}$$

where:

$$(3)\ B_{TL} = \frac{1}{10} \cdot \sum_{i=1}^{10} \sigma_{ep}(t_i)$$

Hence, the initiation event is defined as the first instant $t_i$ that satisfies the condition:

$$(4)\ \bar{\sigma}_{ep}(t_i) > T_{TL}$$

where $T_{TL}$ is a user-defined threshold.

### CODE

```
function IndexMov=TrunkMOV(KS,Ttl)
      % INPUT
      % KS: GRF profile divided in epochs
      % T: User defined threshold

      % OUPUT
      % IndexMov: Initiation event

      % ROUTINE START
      % 1-For each epoch calculate the standard deviation.
      std_arr=stdonepoch(KS);

      % 2-Moving average of the resulting epochs sequence
      std_arr(2,:)=movmean(std_arr(2,:),10);

      % 3-Define the Baseline reference as average standard deviation on the
      first ten epochs
```

```
    BsLn=abs(mean(std_arr(2,1:10)));

    % 4-Normalize the epochs sequence
    std_arr(2,:)=std_arr(2,:)./BsLn;

    % 5-The first index surpassing the threshold identify the beginning of
    the movement
    IndexMov=std_arr(1,find(std_arr(2,:)>=Ttl,1));

function std_arr=stdonepoch(Signal)
    % Function to calculate the standard deviation across each epochs.
    % Save the values of devstd and the central indexes of the epochs.
    % INPUT
    % Signal: matrix MxN, where N is the epochs number and M is the number
    % of samples per epochs.
    % OUTPUT
    % std_arr: is a 2xM matrix with M number of epochs, in the first row
    % the epochs indexes are saved, in the second the standard deviation
    % are stored.

    % ------------> epochs
    % |__|__|__|... references indexes
    % |__|__|__|... standard deviations

    % 1-Declaration and pre-allocation of the variable
    std_arr=NaN(2,size(Signal,2));
    % 2-Number of samples per epochs
    nsamples=size(Signal,1);
    % 3-Standard deviations
    std_arr(2,:)=std(Signal,0,1);
    % 4-Middle samples
    std_arr(1,:)=(1:length(Signal))*nsamples+ceil(nsamples/2);
```

# Seat Off and Seat On events

## METHODS

The *Bottom_Transition.m* identifies voltage transitions of the electronic switch, from 0V to -5V and *vice versa* using the differential mask procedure.

## CODE

```
function [SeatOff,SeatOn]=Bottom_Transition(Switch,V)
    % INPUT
    % Switch: Electronic switch signal
    % V: Threshold voltage (usually -4)

    % OUTPUT
    % SeatOff / SeatOn

    % ROUTINE START
    % 1-Signal over threshold voltage is setted at 0
    Switch(Switch>V)=0;
    % 2-Signal under threshold voltage is setted at -5
    Switch(Switch<V)=-5;
    % 3-Differential mask
```

```
DerSwitch=diff(Switch);
SeatOff=find(DerSwitch<=-4,1,'first')+1;
SeatOn=find(DerSwitch>-(-4),1,'last');
```

# Standing and Sitting events

## METHODS

The Standing phase is recognised by retrieving the information from the electronic switch and by identifying the stable stance between the Seat Off and the Seat On events. The *SteadyStandingPoints.m* function identifies the balance stability in the upright stance. Firstly, the force signal is trimmed and divided in *n* time epochs of different $N_e$ samples between the Seat Off and the Seat On instants. Subsequently, the standard deviation across all the epochs is calculated using (1). Relying on the symmetrical shape of the GRF profile, the middle epoch $t_{mid}$ of the resulting signal is considered as the centre of the stable stance. Hence the stability baseline $B_{ST}$ is calculated around $t_{mid}$ as:

$$(5)\ B_{ST} = E\{\sigma_{ep}(t_{mid-10}), \dots, \sigma_{ep}(t_{mid+10})\}$$

In a similar way to that described in equation (3) the $B_{ST}$ value is used to normalise the moving average of $\sigma_{ep}(t)$. The resulting $\bar{\sigma}_{ep}(t)$ is used to identify the beginning and the ending instants of the stable stance as respectively the first and the last epochs that satisfy the condition:

$$(6)\ \bar{\sigma}_{ep}(t_i) < T_{ST}$$

Where $T_{ST}$ is a user-defined threshold.

## CODE

```
function [Standing,Sitting]=SteadyStandingPoints(F,fs,ep,SitOff,SitOn,Tst)
      % INPUT
      % F: Force signal
      % fs: Sample frequency (50 Hz)
      % ep: time epoch (0,1 s)
      % SeatOff: Seat Off event
      % SeatOn: Seat On event
      % T: Threshold

      % OUTPUT
      % SeatOff / SeatOn

      % ROUTINE START
      % 1-Calculate number of samples per epoch
      samples=fix(fs*ep);
      % 2-Time array from Seat Off to Sit On (in samples)
      t=SitOff:SitOn;
      % 3-Trimming force from stand to sit
      stand2sit=F(SitOff:SitOn);
      % 4-Division in time epochs
      bstdF=std(buffer(stand2sit,samples,0,'nodelay'));
      bavgT=mean(buffer(t,samples,0,'nodelay'));
      % 5-Middle point of the stable stance between Sit Off and Sit On
      midpoint=round(length(bstdF)/2);
      % 6-Calculating the baseline
      baseline=mean(bstdF(midpoint-10:midpoint+10));
      bstdF=bstdF/baseline;
      mask=movmean(bstdF,10)>Tst;
      Standing=bavgT(find(mask==0,1,'first'));
```

```
Sitting=bavgT(find(mask==0,1,'last'));
```

The default values used in this work for the described variables are reported in Table 1.

| | ***TrunkMOV.m*** | ***SteadyStandingPoints.m*** |
|---|---|---|
| **GRF sample frequency** | 50 Hz | 50 Hz |
| $\boldsymbol{Ne}$ | 5 epochs (0,1 s) | 5 epochs (0,1 s) |
| $\boldsymbol{T_{TL}}$ | 2 | N.A. |
| $\boldsymbol{T_{ST}}$ | N.A: | 1,5 |

*Table 1: Default values of the categorisation software used for this study.*