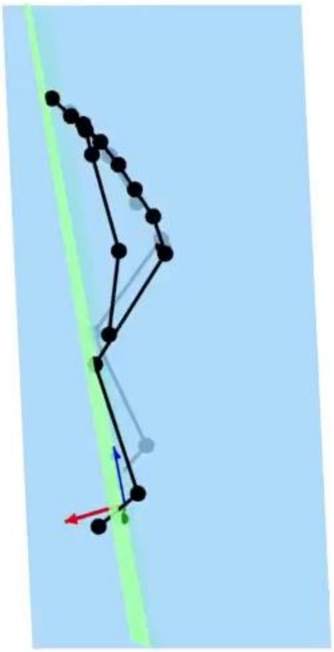
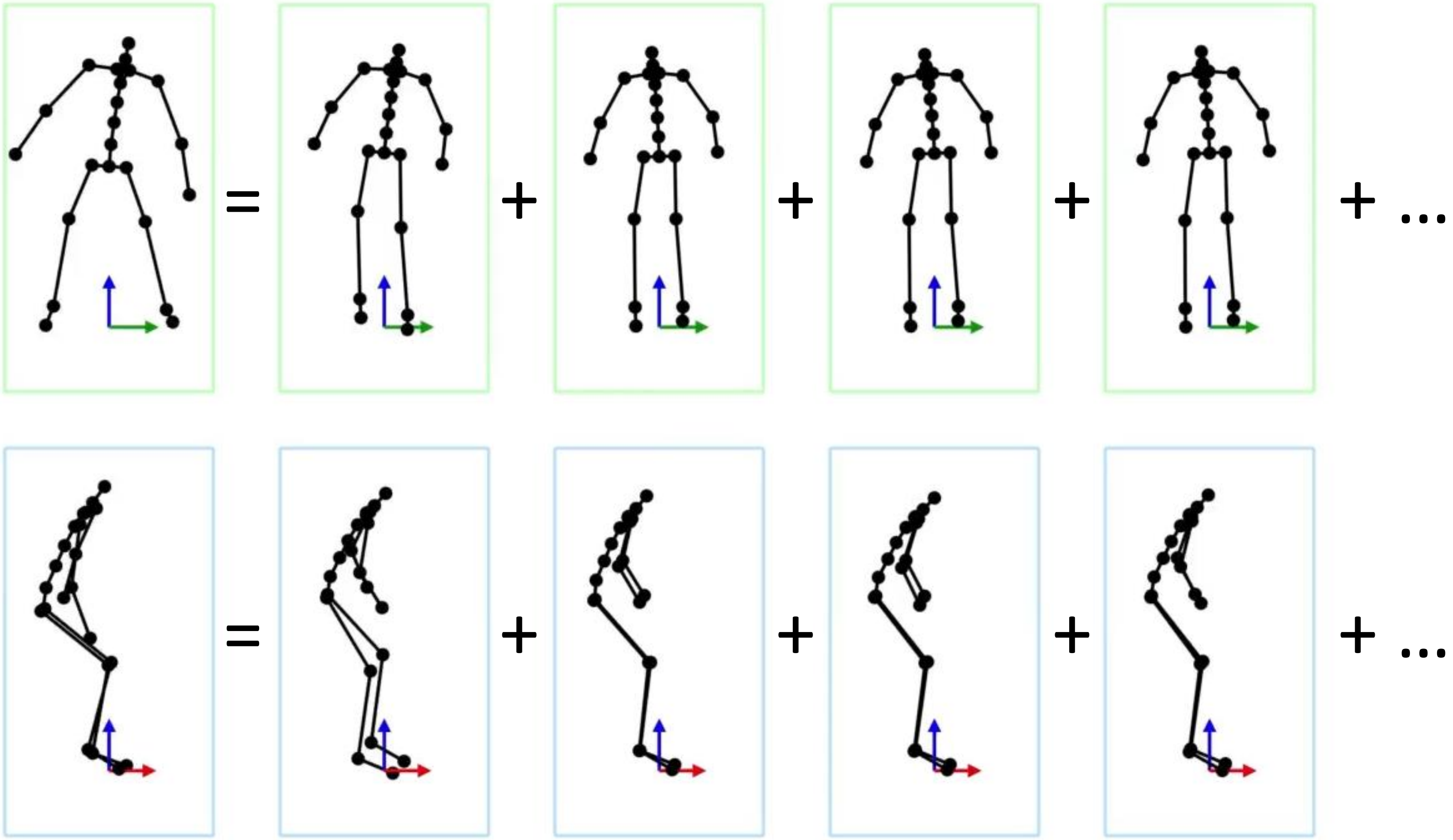
PManalyzer – How to

25/04/2025





Background – PCA for human movement analysis

» https://www.uibk.ac.at/de/isw/forschung/neurophysiology-of-exercise/pm_analyzer/

» Haid, T. H., Zago, M., Promsri, A., Doix, A.-C. M., & Federolf, P. A. (2019). PManalyzer: A Software Facilitating the Study of Sensorimotor Control of Whole-Body Movements. *Frontiers in Neuroinformatics*, *13*. https://doi.org/10.3389/fninf.2019.00024

Origins in human movement context

» Troje, N. F. (2002). Decomposing biological motion: A framework for analysis and synthesis of human gait patterns. *Journal of Vision*, *2*(5), 2. https://doi.org/10.1167/2.5.2

» Daffertshofer, A., Lamoth, C. J., Meijer, O. G., & Beek, P. J. (2004). PCA in studying coordination and variability: a tutorial. *Clinical biomechanics*, *19*(4), 415-428. https://doi.org/10.1016/j.clinbiomech.2004.01.005

Very good explanations and examples (technique analysis, postural control)

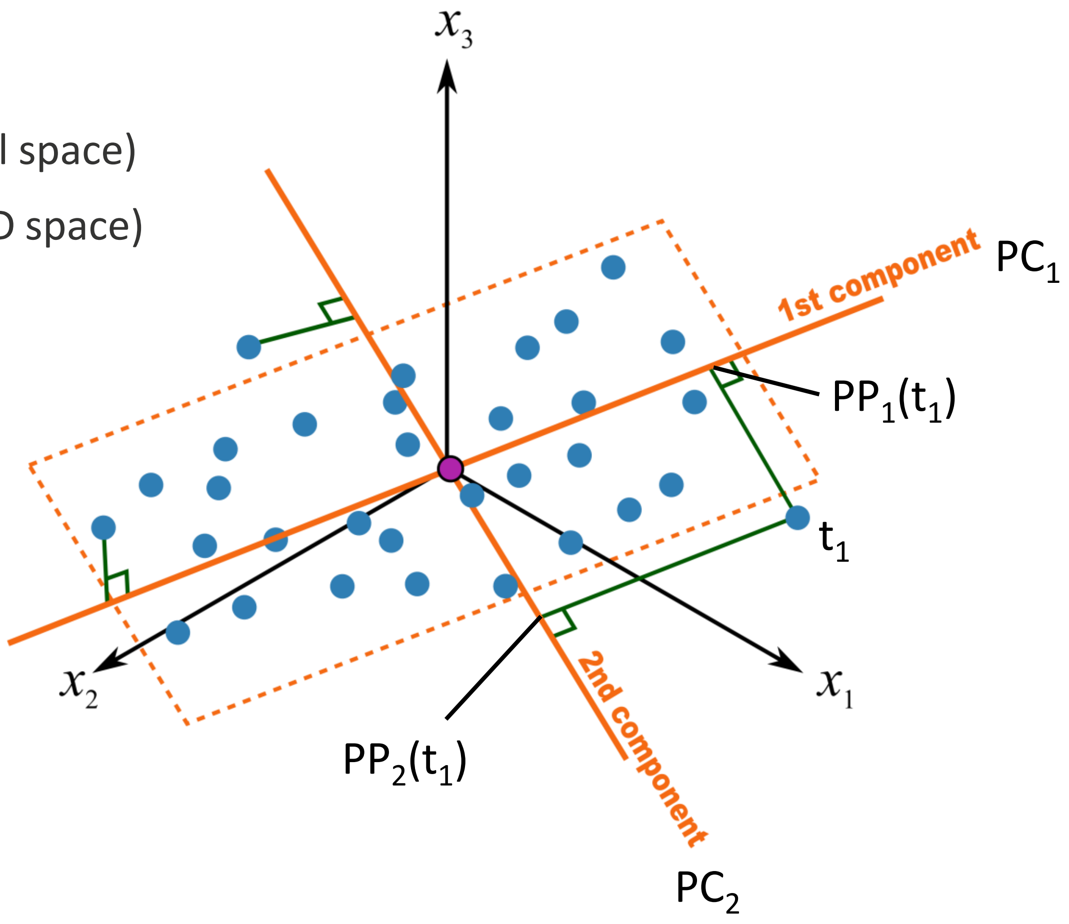
» Anwendungen der PCA in der Bewegungsanalyse – Keynote Präsentation für die dvs Section Biomechanik 2023 in Magdeburg (https://www.youtube.com/watch?v=x7DHdEOZOAQ)

» Federolf, P., Reid, R., Gilgien, M., Haugen, P., & Smith, G. (2014). The application of principal component analysis to quantify technique in sports. *Scandinavian Journal of Medicine & Science in Sports*, *24*(3), 491–499. https://doi.org/10.1111/j.1600-0838.2012.01455.x

» Federolf, P. (2016). A novel approach to study human posture control: “Principal movements” obtained from a principal component analysis of kinematic marker data. *Journal of Biomechanics*, *49*(3), 364–370. https://doi.org/10.1016/j.jbiomech.2015.12.030

Abbreviations

- PCA – principal component analysis
- PC – principal component (eigenvector in higher dimensional space)
- PM – principal movement (movement along PC in posture 3D space)
- PP – principal position (score along PC, time series)
- PV – principal velocity (1. derivative of PP)
- PA – principal acceleration (2. derivative of PP)
- PJ – principal jerk ;-)
- EV – eigenvalue (relative variance explained by the PC)
- rVAR – relative variance explained by the PP
- NoZC – number of zero crossings



Maths

Input
(per subject, per trial)

Posture vector $\boldsymbol{p}(t)$ $(t \times n)$

"Marker coordinates"

		x_1	y_1	z_1	x_2	y_2	z_2	… n
Time steps	t_1							
	t_2							
	t_3							
	…							
	t							

Mean posture vector $\overline{\boldsymbol{p}}$ $(1 \times n)$

Normalization
(per subject, per trial)

$(t \times n)$ $(1 \times n)$ *

$$\boldsymbol{p}_{\boldsymbol{norm}}(t) = \frac{\boldsymbol{p}(t) - \overline{\boldsymbol{p}}}{d_{norm}} \cdot \boldsymbol{\omega} \qquad (t \times n)$$

(1×1) $(n \times n)$

e.g. mass distribution

e.g. mean Euclidian distance

* $\overline{\boldsymbol{p}}$ is subtracted from every row of $\boldsymbol{p}(t)$ -> in Matlab it's okay to have $\overline{\boldsymbol{p}}$ with only one row

Maths

PCA matrix
(all subjects/trials together)

$\boldsymbol{p}_{norm,s_1 t_1}(t)$

$\boldsymbol{p}_{norm,s_1 t_2}(t)$

$\boldsymbol{p}_{norm,s_1 t_j}(t)$ $\quad (t \times n)$

$\boldsymbol{p}_{norm,s_2 t_1}(t)$

$\boldsymbol{p}_{norm,s i t_j}(t)$

PCA

Output

Principal components $\boldsymbol{PC}_k$ $(n \times 1)$

Eigenvectors

$$\begin{array}{c|l} & PC_1 \;\; PC_2 \;\; PC_3 \ldots PC_k \\ \hline x_1 & \\ y_1 & \\ z_1 & \\ \ldots & \\ n & \end{array}$$

"Marker coordinates"

Maths

Plots
(per subject, per trial)

Principal positions (PC scores) $\boldsymbol{PP_k}(t)$ $(t \times 1)$

$$
\begin{array}{ll|l}
 & & \text{Principal positions} \\
 & & PP_1 \;\; PP_2 \;\; PP_3 \ldots PP_k \\
\hline
 & t_1 & \\
\text{Time} & t_2 & \boldsymbol{PP_k}(t) = \boldsymbol{p_{norm}}(t) \cdot \boldsymbol{PC}_k \\
\text{steps} & t_3 & \qquad\qquad\;\; (t \times n) \quad (n \times 1) \\
 & \ldots & \\
 & t & \\
\end{array}
$$

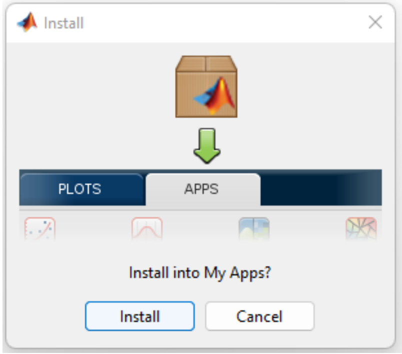
Videos
(per subject, per trial)

Principal movements $\boldsymbol{PM_k}(t)$ $(t \times n)$

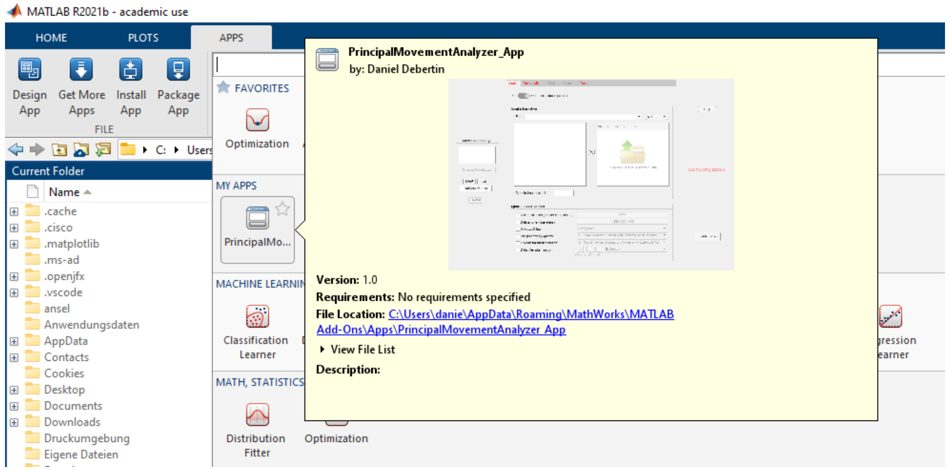
$$
\begin{array}{ll|l}
 & & \text{“Marker coordinates”} \\
 & PM_k & x_1 \;\; y_1 \;\; z_1 \;\; x_2 \;\; y_2 \;\; z_2 \;\; \ldots \; n \\
\hline
 & t_1 & \\
\text{Time} & t_2 & \boldsymbol{PM_k}(t) = \boldsymbol{PP_k}(t) \cdot \boldsymbol{PC}_k^T \cdot d_{norm} \cdot \boldsymbol{\omega}^{-1} + \overline{\boldsymbol{p}} \\
\text{steps} & t_3 & \qquad\qquad\;\; (t \times 1)\;(1 \times n)\;(1 \times 1)\;(n \times n)\;(1 \times n) \\
 & \ldots & \\
 & t & \\
\end{array}
$$

Download and installation

- Requirements: **Windows OS** + **MATLAB R2022b** or higher
- Download PManalyzer New version from https://www.uibk.ac.at/de/isw/forschung/neurophysiology-of-exercise/pm_analyzer/
- Double click on PrincipalMovementAnalyzer_App.mlappinstall
- Install



- Find it under APPS -> MY APPS in Matlab



Changes to the old version

Old version

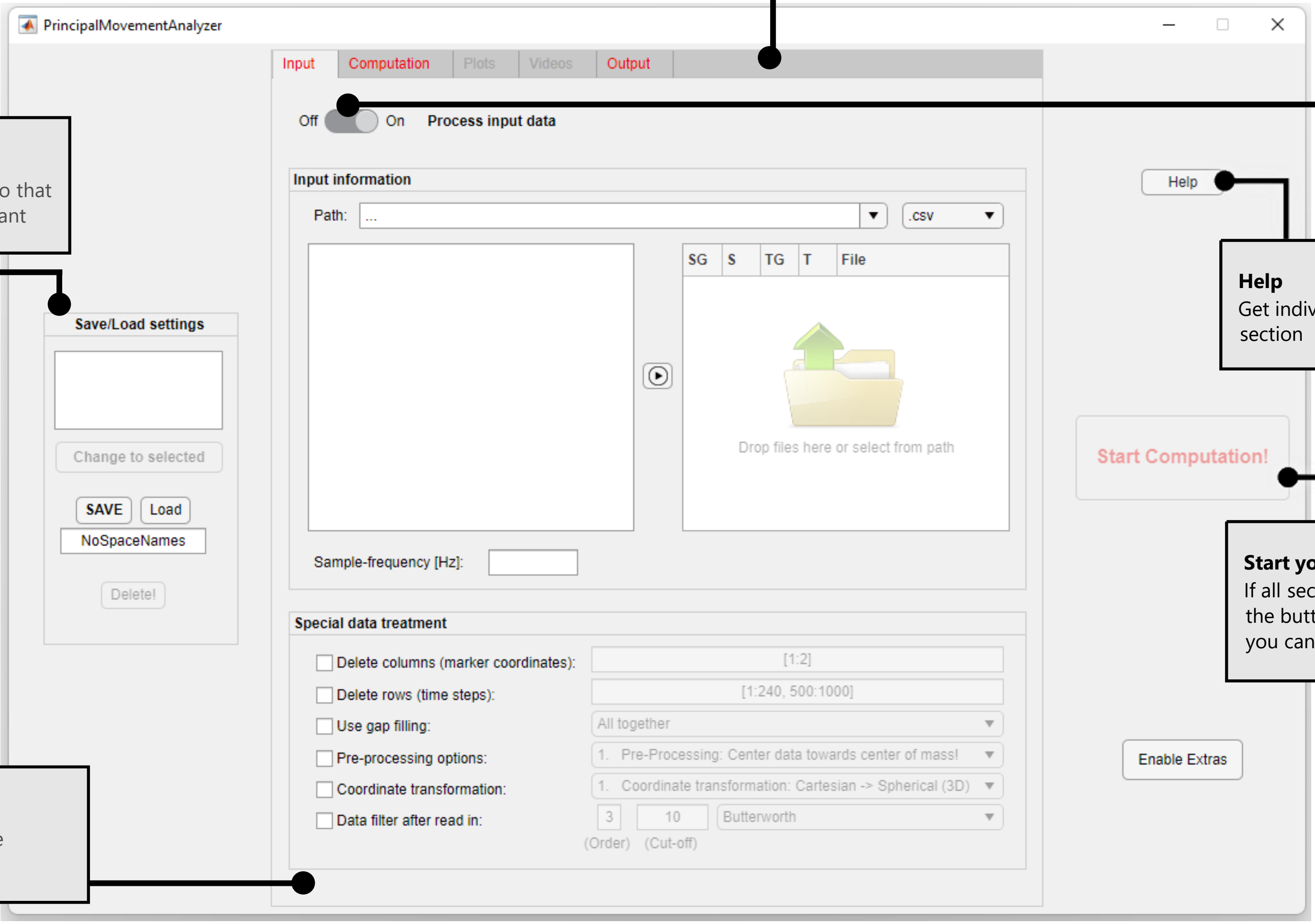
- Transition from the outdated Matlab GUIDE development environment to Matlab App Designer
- Simplified layout (devided into sections) and data read-in options
- Unbalanced input data is allowed, e.g. it is not required that all subjects have the same number of trials
- Videos of the overall movement without specific PMs can be generated

Overview

Section bar
Go from left to right to specify your different preferences and settings for Input, Computation, Plot, Video and Output

On/Off button for sections
For each section, you can turn the settings panel on and off. For "Input" and "Computation" this requires to load .mat data of a previous analysis

Select settings
Save or load your current settings, so that you can go back to any state you want

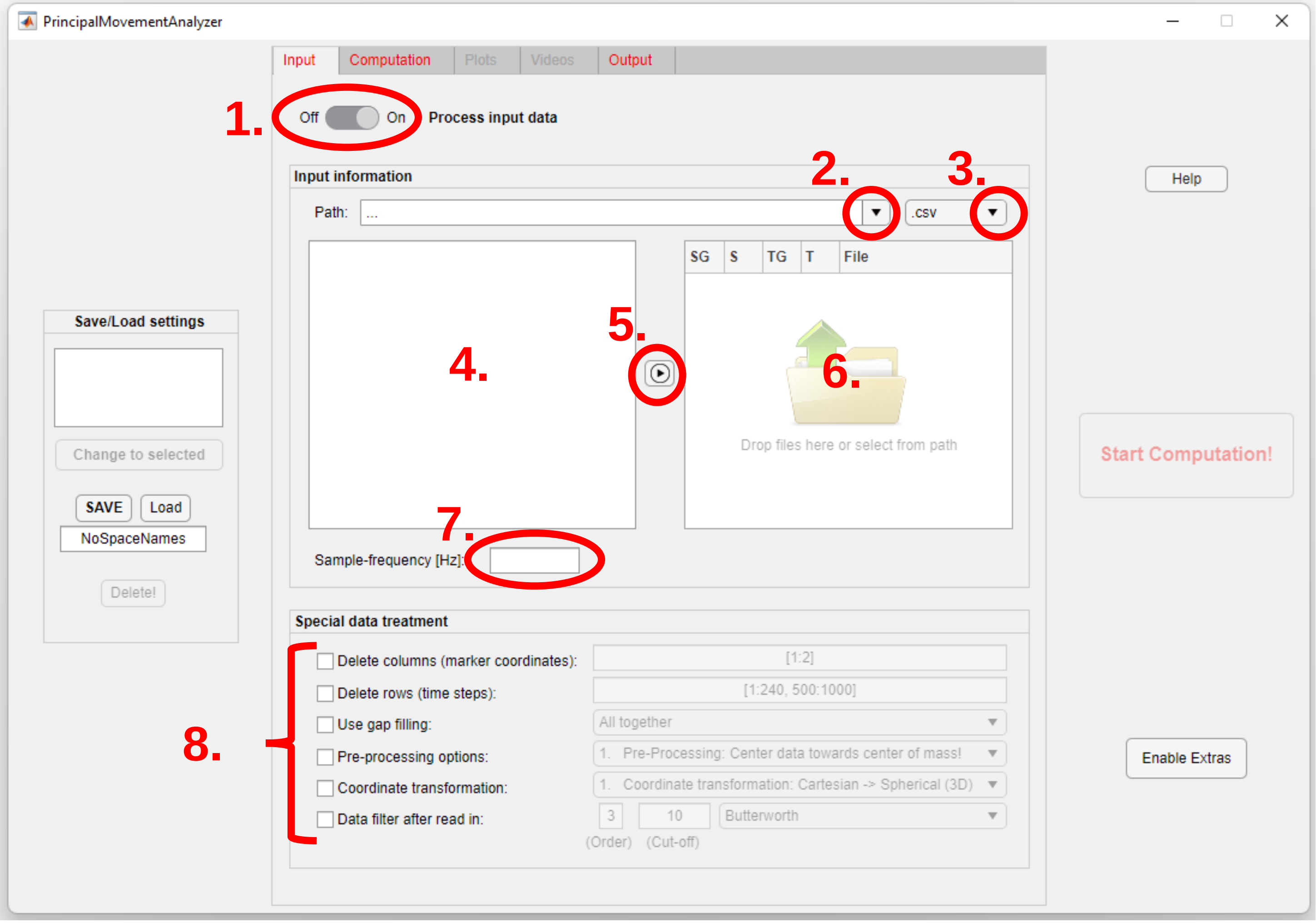


Help
Get individual help for each section

Start your computation
If all sections are set correctly the button will turn green and you can start the PCA

Settings panel
Specify different settings for the different sections

Workflow (Section “Input”)



(1) Turn “On” to specify the input data information and processing options. Turn “Off” to only specify a formerly saved input data file (.mat) of a previous analysis in the Path field (2).

(2) Specify path of the input files. Click ▼ and on “Select folder” or paste a file path directly. All files in this folder and in its subfolders with the specified file extension (3) will appear in a list in (4).

(3) Specify file extension (.csv, .txt., mat).

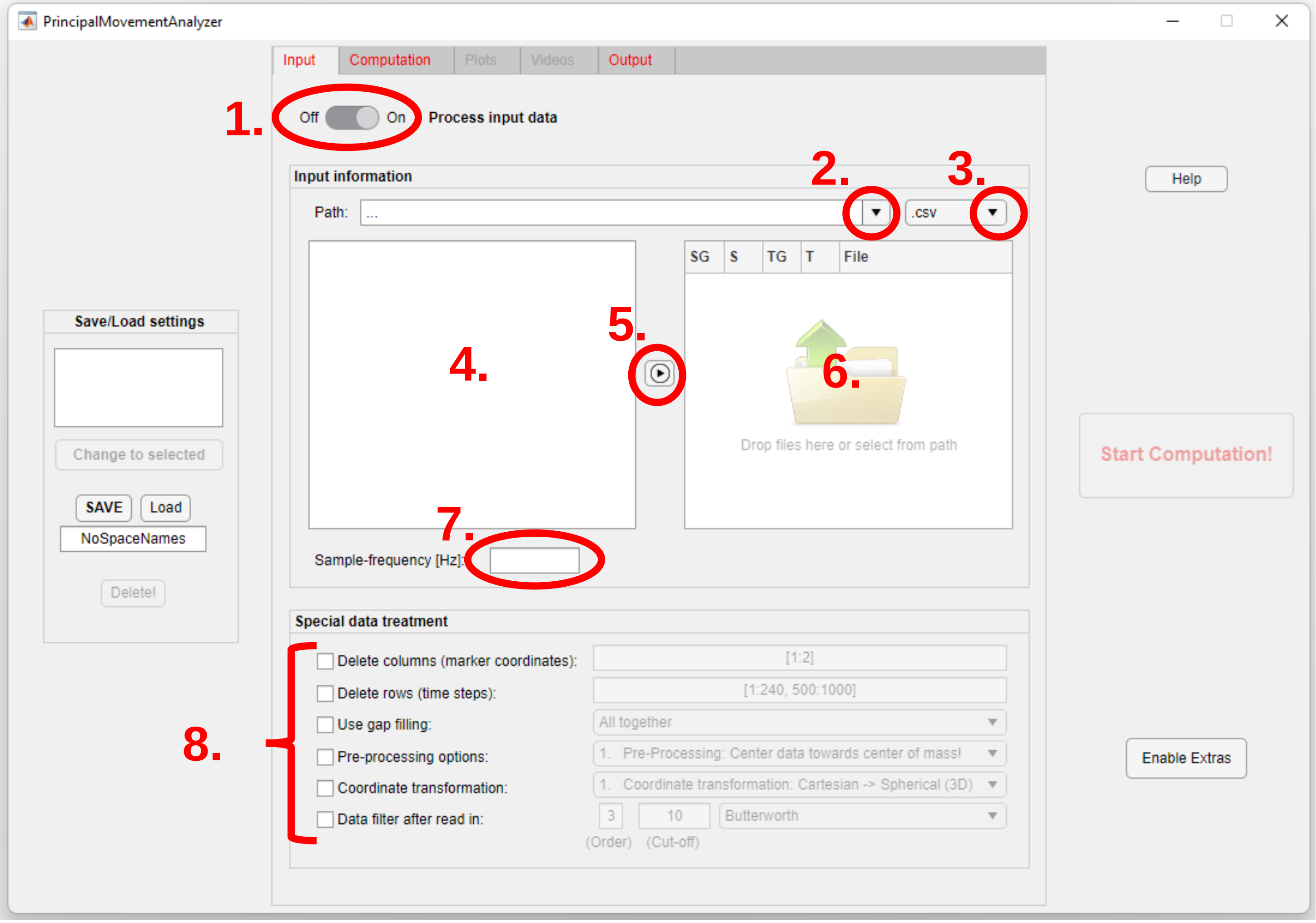
(4) Click all the files that should be used for the PCA.

(5) Click the button to load them into the overview table of the final input files (6).

(6) Input files can also be dropped here directly. Assign a number for the subject (S) and trial (T) of each file. If the filename already contains the string “s*(number)*t*(number)*” this is filled automatically. SG and TG is reserved for grouping but not yet implemented. Check that there is no duplicate.

(7) Specify sample frequency. It is set automatically for original Vicon and Theia .csv output files.

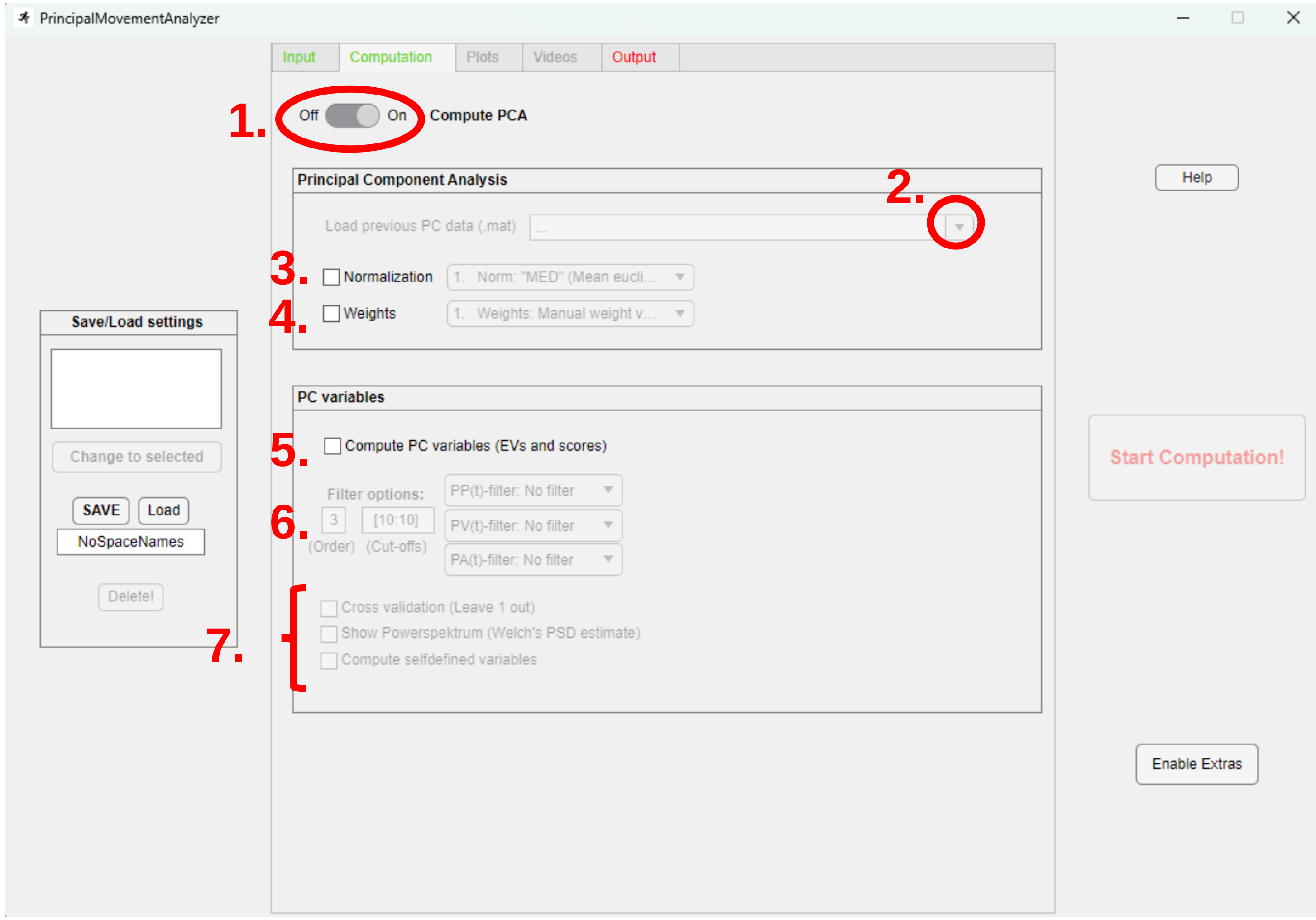
Workflow (Section “Input”)



(8) Selected optional data treatment functions:

- Delete columns: Specify column numbers in a row vector (only use “:” or “,”) that will be deleted from the data of all input files, e.g. [1, 5:7] to delete column 1 with frame numbers and the second marker (x,y,z coordinates).
- Delete rows: Specify row numbers in a row vector that will be deleted from the data of all input files, e.g. [1:100] to delete the first 100 frames.
- Use gap filling: If the input data contains gaps you can apply a gap filling algorithm to the input data. Only works if the gaps are not too big. Choose between all data together, all trials per subject together, every data set separately.
- Pre-Processing: Select a data processing option. Can also be self coded in optionsPreProcessing.m (see “Customization”).
- Coordinate transformation: Select a coordinate transformation. Can also be self-coded in optionsTransformData.m (see “Customization”).
- Data filter: Use a filter for the input data.

Workflow (Section “Computation”)



(1) Turn “On” to compute the PCA. Turn “Off” to only specify a formerly saved data file (.mat) with the PCs of a previous analysis in the Path field (2).

(2) Specify path of a file (.mat) with PC vectors corresponding to the input data (matrix with PCs as columns and coordinates as rows).

(3) Click to specify a normalization option. Can also be self coded in optionsNormalization.m (see “Customization”).

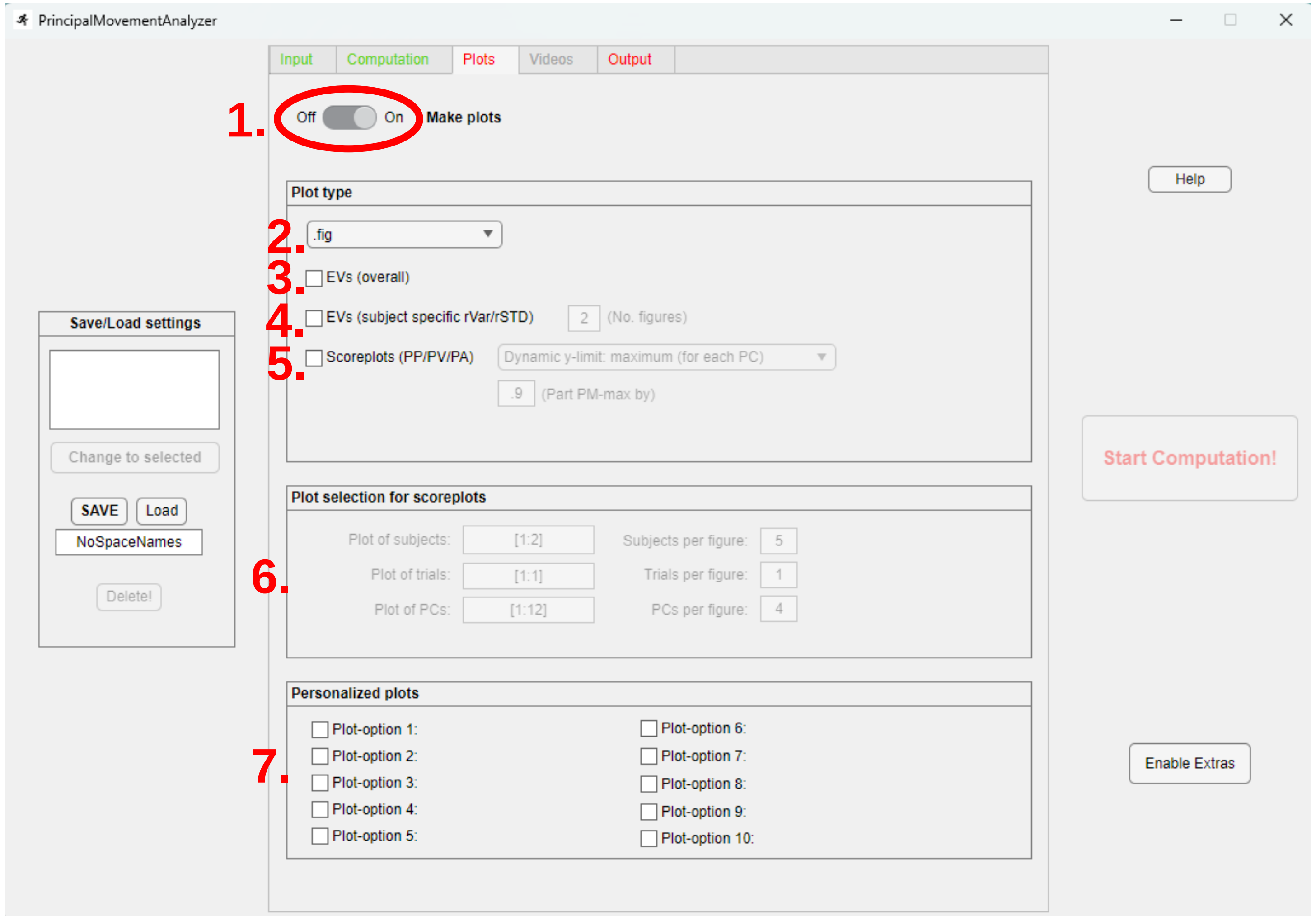
(4) Click to specify a weighting option. Can also be self coded in optionsWeights.m (see “Customization”).

(5) Click to compute the PC vectors and PP time series.

(6) Specify filters for PP, PV and PA time series. Different cut-off frequencies can be entered in a row vector.

(7) Specify further calculations, e.g. “Compute selfdefined variables” for NoZC. Can also be self-coded in optionsVariablesComp.m (see “Customization”).

Workflow (Section “Plots”)



(1) Turn “On” to create EV and PP/PV/PA plots. Turn “Off” to not create plots.

(2) Select the file type of the plots (.fig, .jpg)

(3) Click to create bar plots of relative variance expressed by the PCs for the overall data (equivalent to EV).

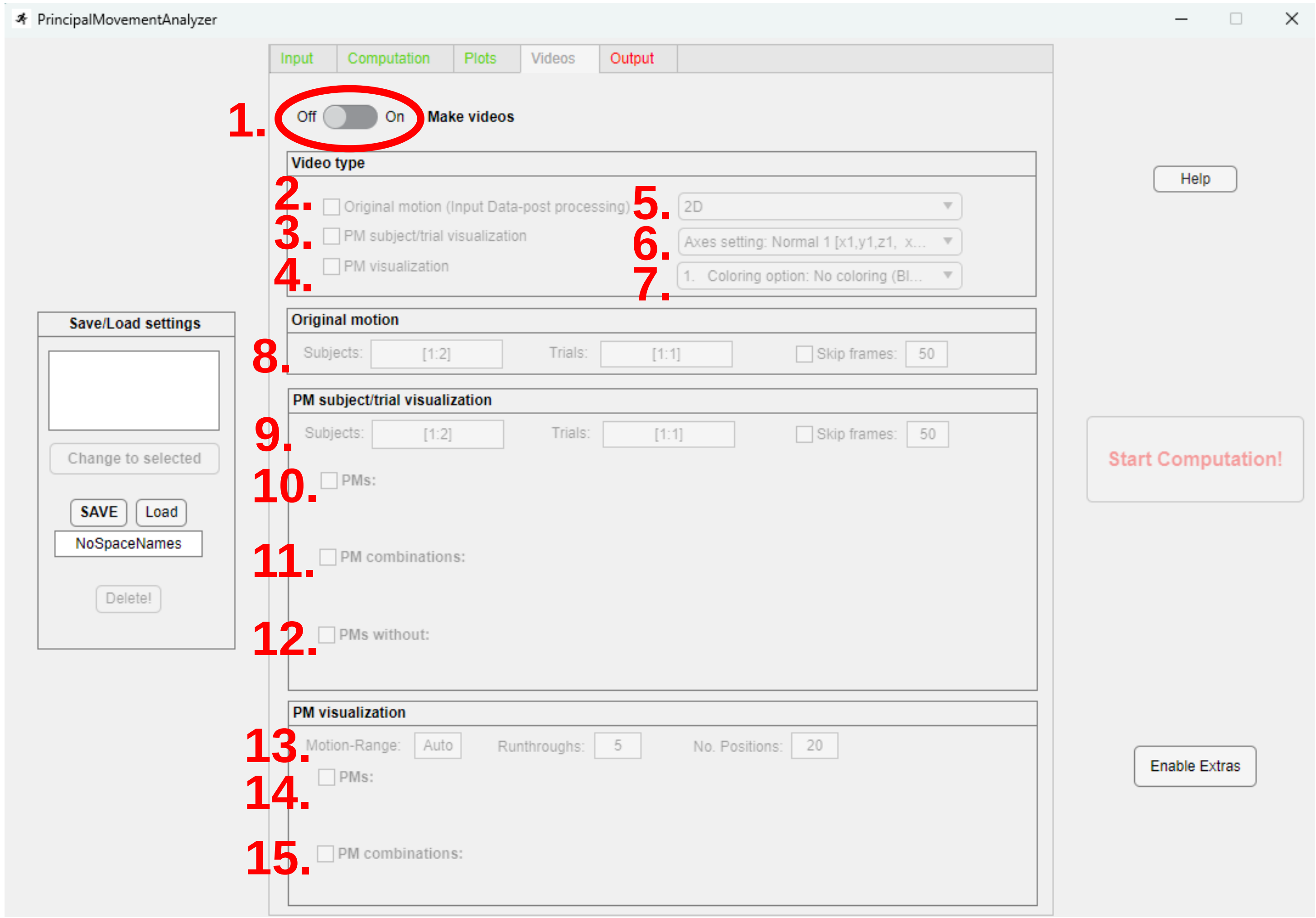
(4) Click to create bar plots of relative variance and standard deviation expressed by the PCs for individual data (subject and trial specific). The number of plots per Matlab figure (i.e. per saved file) can be set.

(5) Click to create line plots of PP/PV/PA. Different y-axis limits can be set.

(6) Specify the subjects, trials and PCs that should be plotted as row vectors. Numbers refer to the final input data table. It can be set, how many subjects, trials and PCs are plotted maximally per Matlab figure.

(7) Select further personalized plots. Has to be self-coded in personalizedPlots.m (see “Customization”).

Workflow (Section “Videos”)



(1) Turn “On” to create original motion and PM videos. Turn “Off” to not create videos.

(2) Click to create videos of the original motion (raw input data).

(3) Click to create videos of the PMs for specific data of subjects and trials.

(4) Click to create videos of the PMs for fictive data as visualizations.

(5) Specify 2D and/or 3D video format.

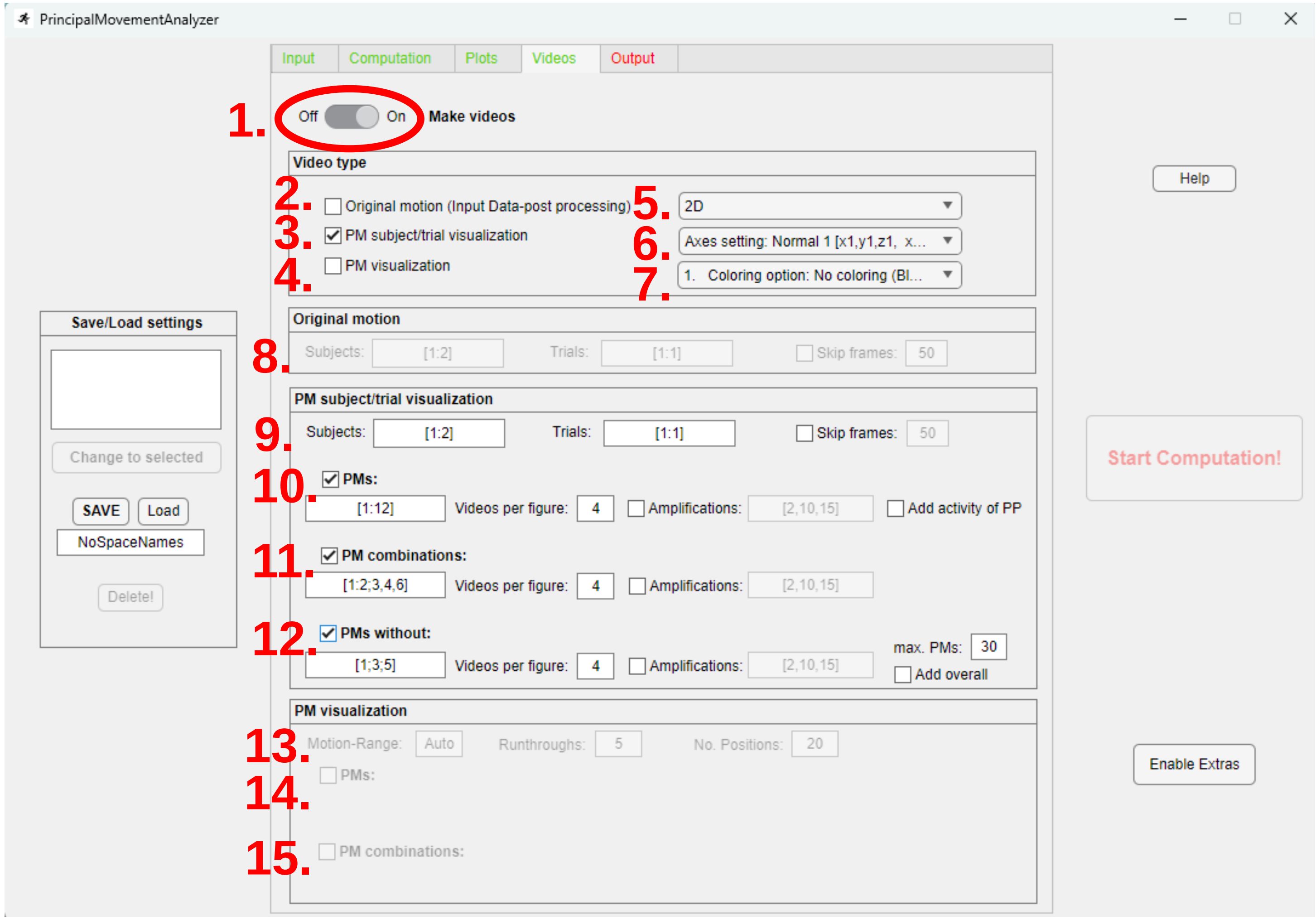
(6) Specify axes direction to align with frontal and sagittal video view.

(7) Specify color and stick-figure option. Can also be self-coded in optionsColoringVids.m (see “Customization”).

(8) Specify numbers of subjects and trials for original motion videos. Use “Skip frames” to reduce computation time.

(9) Specify numbers of subjects and trials for PM videos. Use “Skip frames” to reduce computation time.

Workflow (Section “Videos”)

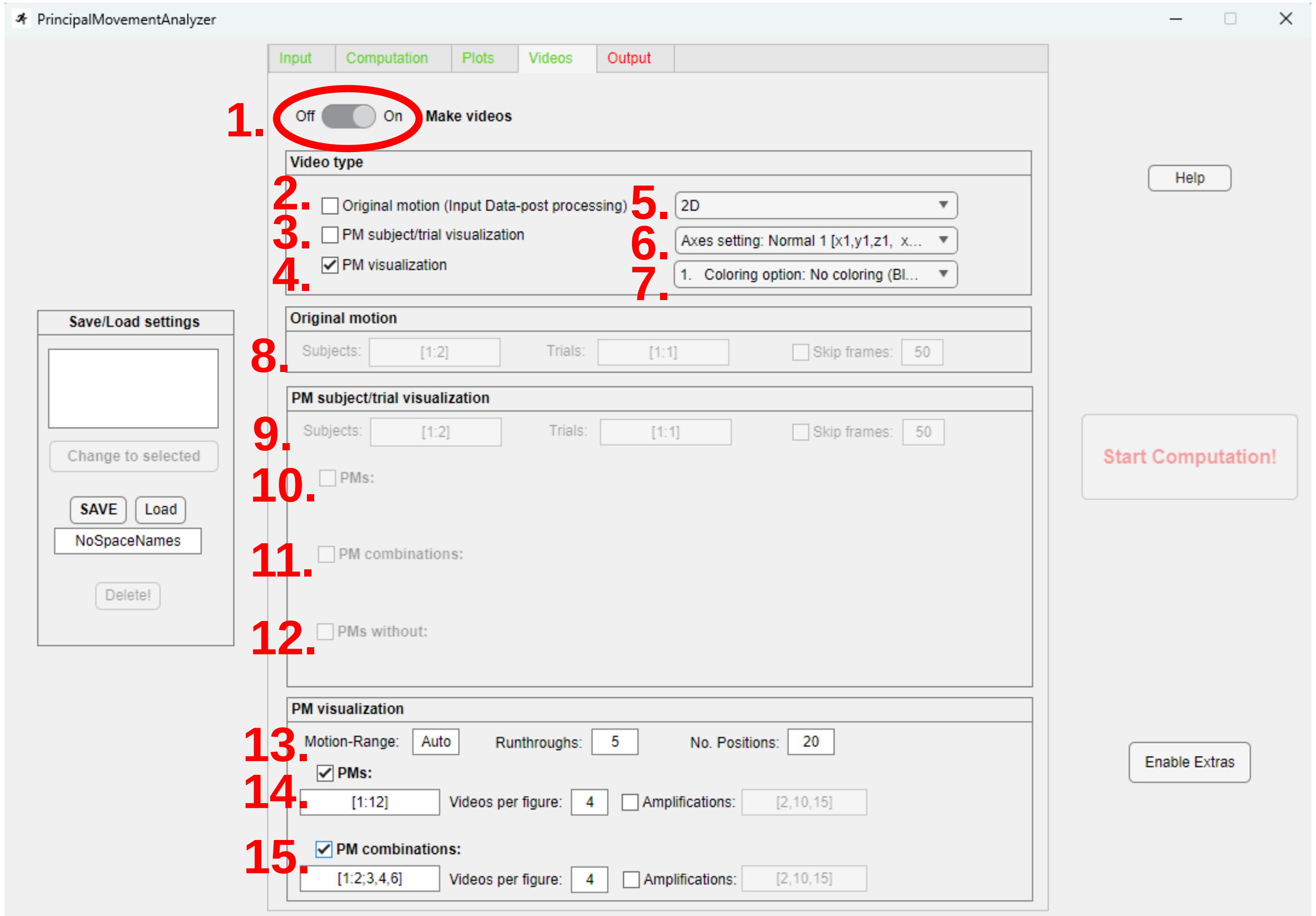


(10) Select to create videos of the PMs separately. Specify the order of the PMs as a Matlab row vector, the maximal number of PM videos per Matlab figure, possibly amplification factors and possibly an additional dynamic activity bar graph. Amplification factors can be set, where the factors are equally distributed on the PMs. E.g. when specifying two factors while having 12 PMs in total, the first 6 will be amplified with the first factor and the last 6 with the second.

(11) Select to create videos of PM combinations. Specify the order of the PMs, that should be combined using “:” and “,”. Separate the numbers by “;” to get a new video with a new combination.

(12) Select to create videos without specific PMs. Specify the order of the PMs, that should be left out. E.g. [1;2,3] will create two videos. In the first video the first PM is subtracted from the sum of the first 30 PMs. In the second video PM 2 and 3 are subtracted. The PMs that are considered for the sum can be changed under “max. PMs”. The sum can also be visualized in the videos (“Add overall”).

Workflow (Section “Videos”)

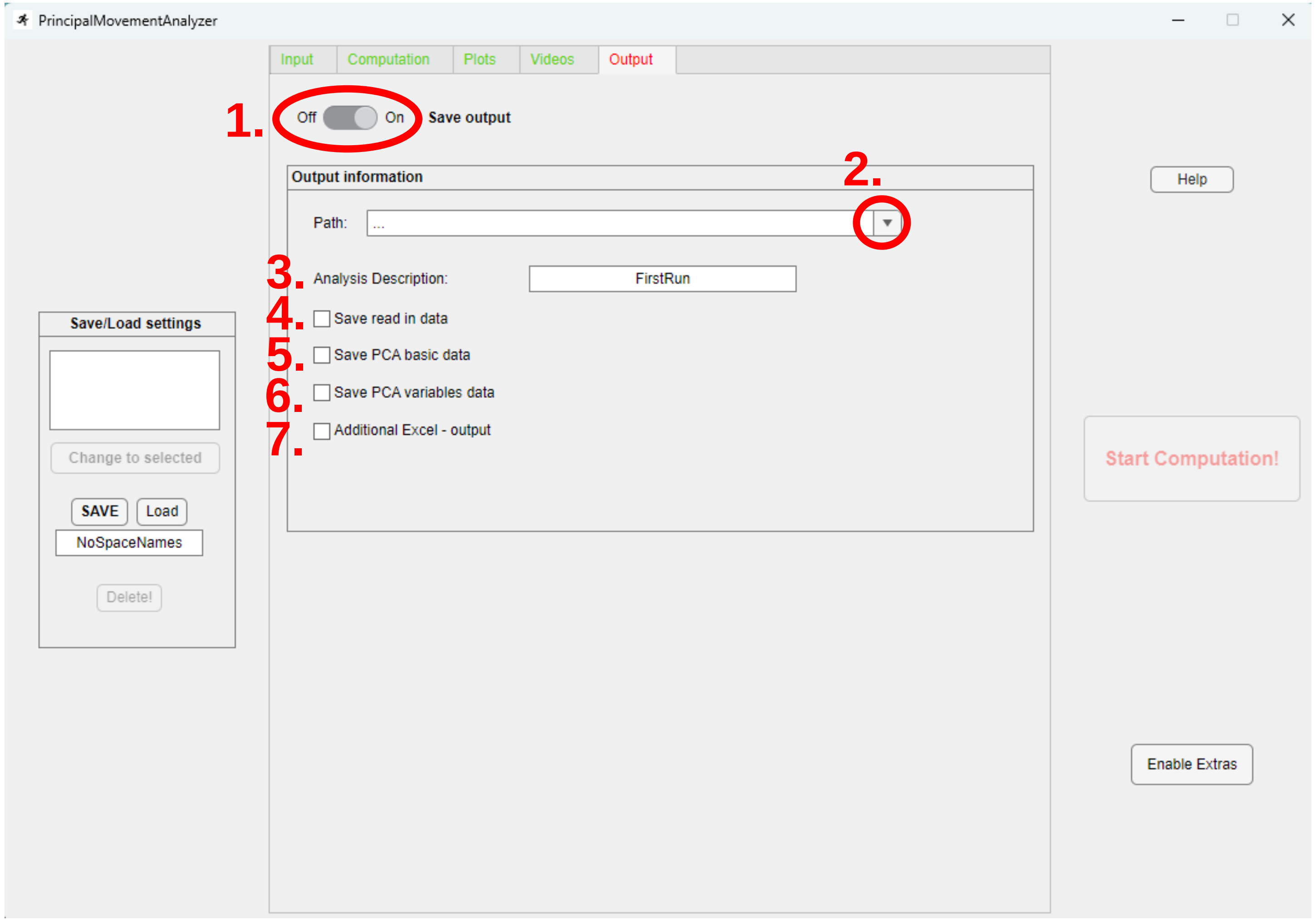


(13) Specify parameters for the visualization. A fictive PP is used to calculate the PMs, with specific range (extreme scores), run-throughs (repetitions of alternating between the extremes) and no. positions (time points between the extremes).

(14) Select to create videos of the PMs separately. Specify the order of the PMs in a row vector, the maximal number of PM videos per Matlab figure and possibly amplification factors. Amplification factors can be set for each Matlab figure or for each PM depending on the length of the specified Matlab row vector.

(15) Select to create videos of PM combinations. Specify the order of the PMs, that should be combined, as under (11).

Workflow (Section “Output”)



(1) Turn “On” save the output. Turn “Off” to not save the output.

(2) Specify path, where the output folder will be created. Click ▾ and on “Select folder”, select a folder from the list or paste a file path directly.

(3) Specify the name of the output folder. The folder will be named “PCA_*(name)*”

(4) Click to create an “Input” folder and to save input data there after possible gap filling and filtering, but before normalization.

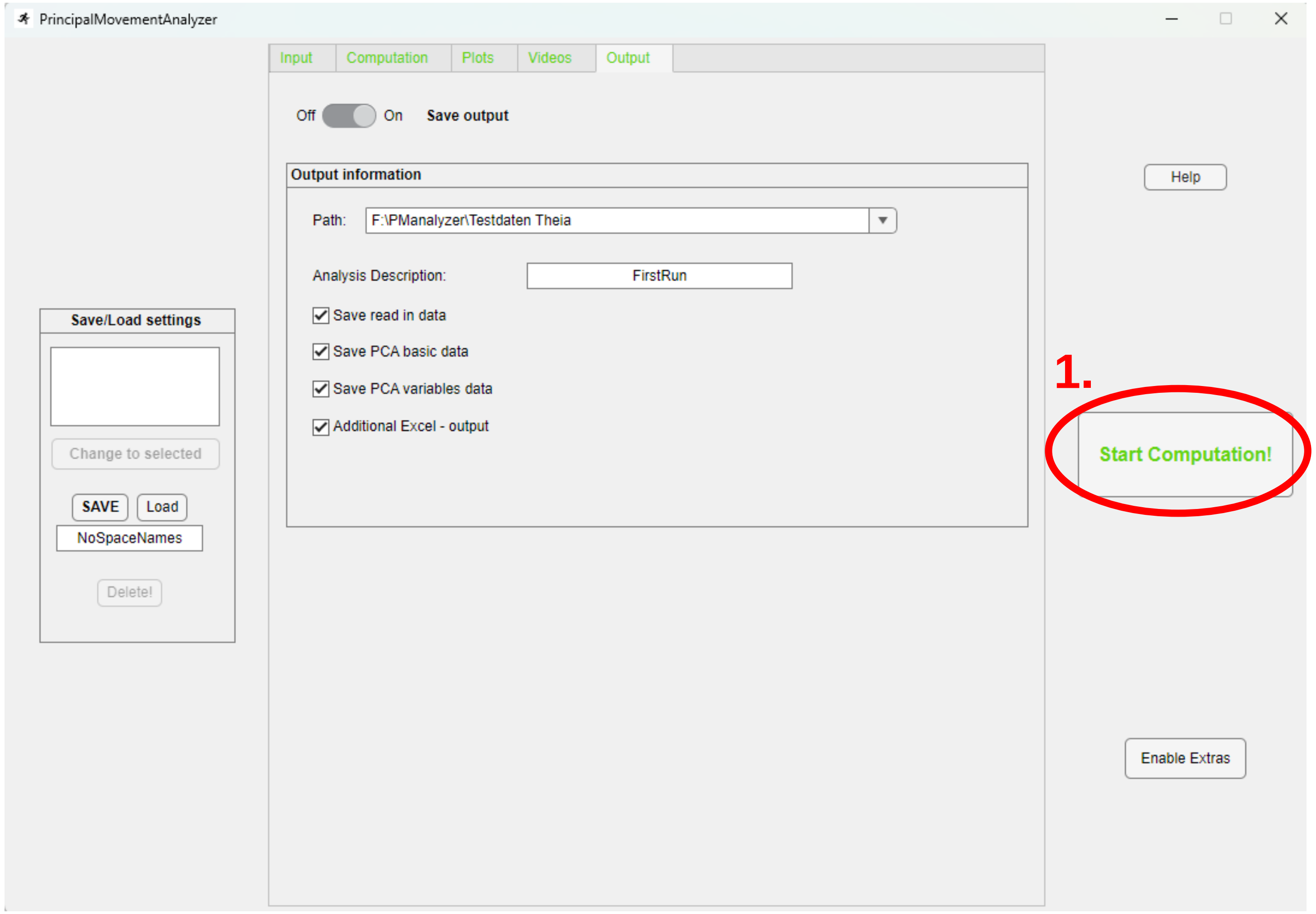
(5) Click to create a “Computation” folder and to save PCA basic output (PCs) there.

(6) Click to also save PCA variables (e.g. PP/PV/PA) to the “Computation” folder

(7) Click to save an Excel file with PCA variables (e.g. EV,rVar) to the “Computation” folder

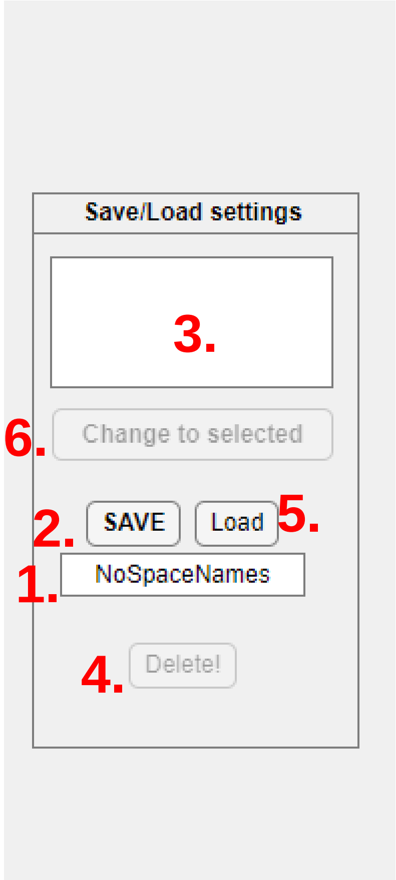
Note: If plots/videos are generated, they will be directly saved to respective folders.

Start computation



(1) Start computation and follow log output in Command Window of Matlab environment

Settings



(1) Specify a name for the current settings (all clicked buttons and filled fields of the app).

(2) Save the current settings. If not settings file (.mat) has been created before a file path and file name can be specified. The new settings name will appear in the list (3).

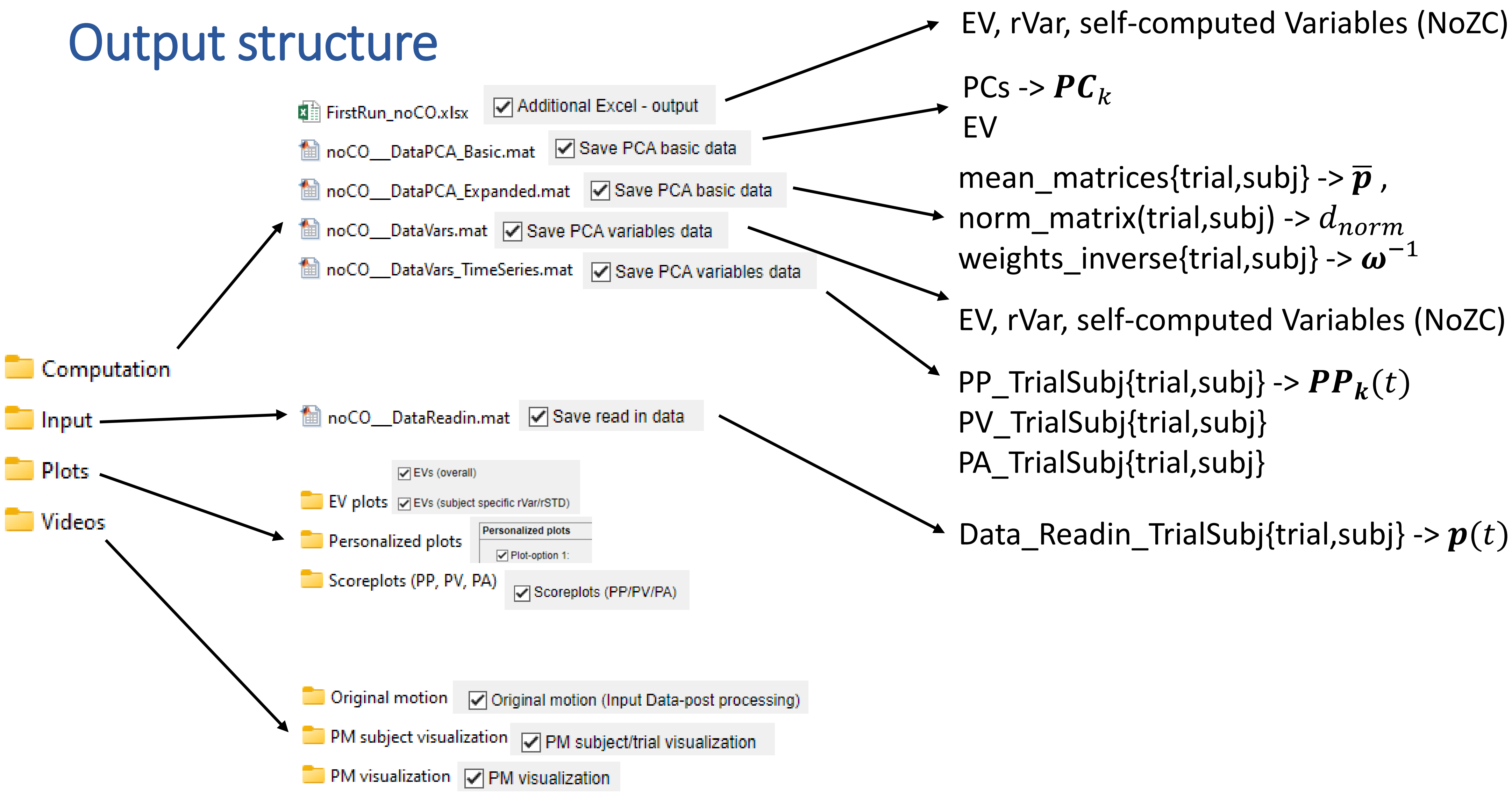
(3) List with settings (different names) in the currently loaded settings file (.mat).

(4) Delete a selected setting from the list.

(5) Load a previously saved settings file (.mat). The new available setting names will appear in (3).

(6) Change to a selected setting in the list.

Output structure



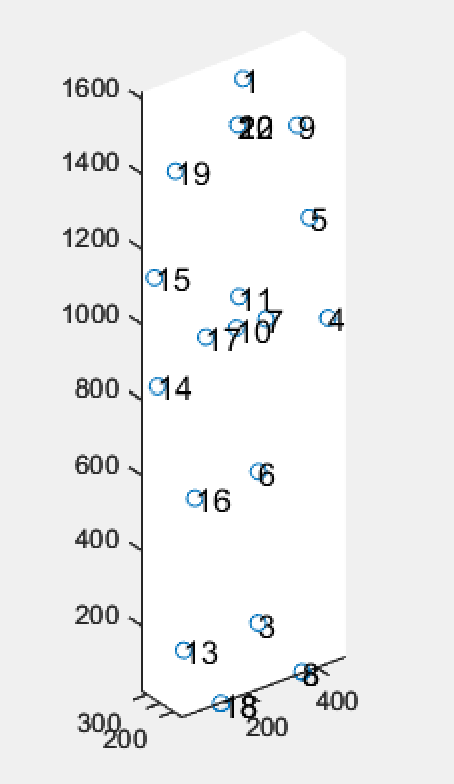
universität innsbruck

Customization

» Get an overview over the marker numbers first

» Choose a sample file and a sample frame and plot the markers together with the numbers from 1 to n

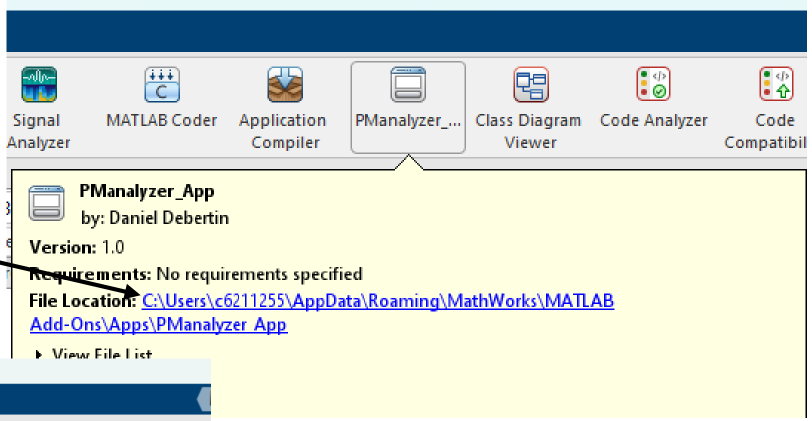
```
% load data
sample_data = readmatrix('s01_t1_theia_balance_bb_open_converted.csv');
% specify a frame, when to plot the data
sample_frame = 10;
% plot markers
plot3(sample_data(sample_frame,1:3:end),sample_data(sample_frame,2:3:end),sample_data(sample_frame,3:3:end),'o');
hold on
% plot marker numbers
text(sample_data(sample_frame,1:3:end),sample_data(sample_frame,2:3:end),sample_data(sample_frame,3:3:end), ...
    string(1:size(sample_data,2)/3), 'FontSize', 12)
% important to get the right proportions
axis('equal')
```

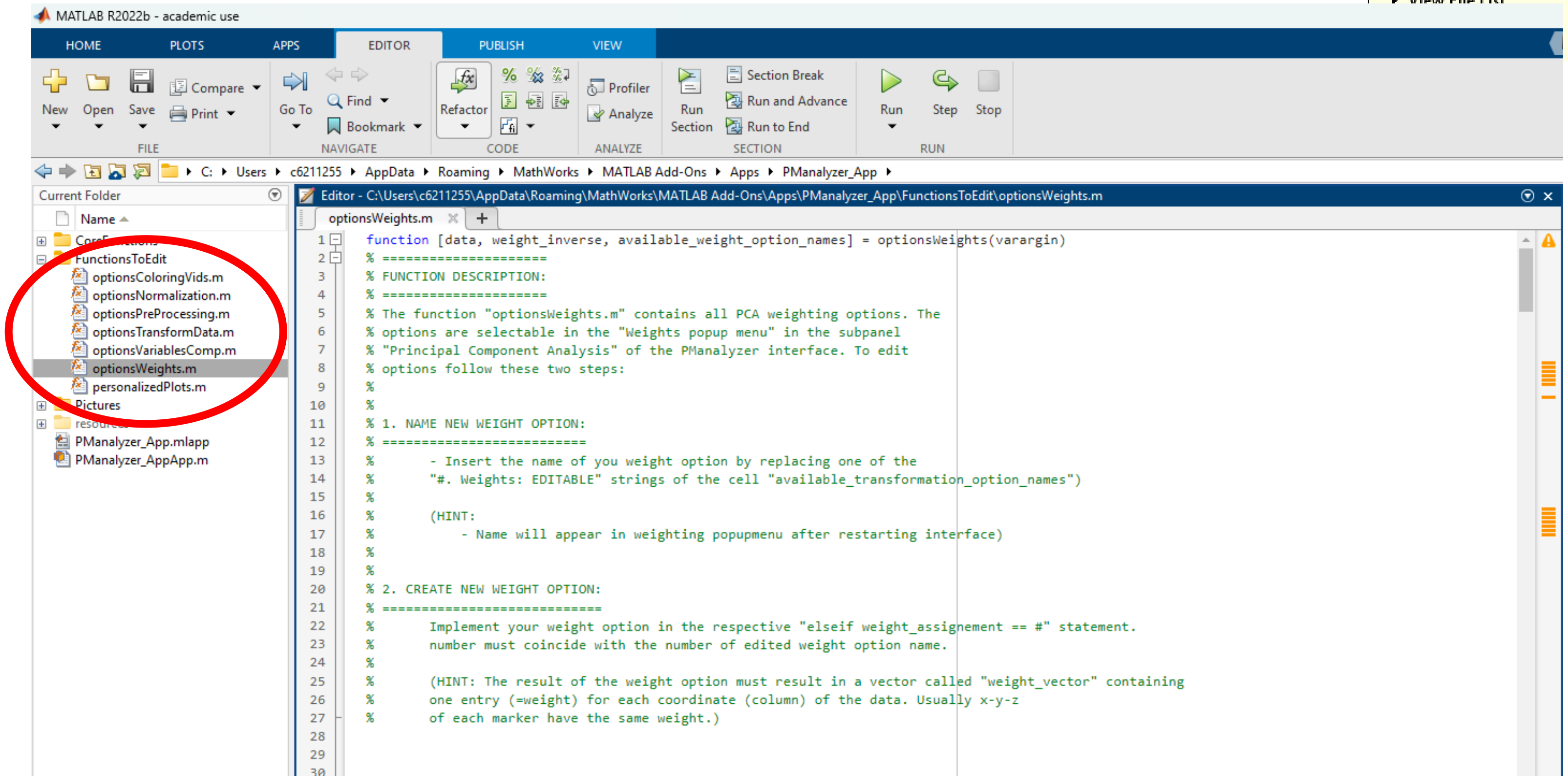


Customization

» Selected the folder of the app source files -> Click on

» Selected a Matlab script in the Folder „FunctionsToEdit“





Customization (Example “weights”)

» Open the script „optionsWeights.m“

» Add a name for the new weight option

optionsWeights.m

```
if nargin == 0
    available_weight_option_names = {'1.   Weights: Manual weight vector', ...
        '2.   Weights: Standard plug-in (39 markers)', ...
        '3.   Weights: Standard plug-in (37 Markers - no fingers)', ...
        '4.   Weights: Specialized - Tandem (unsymmetric markers deleted and mirrored)', ...
        '5. Weights: EDITABLE', ...   % EDIT NEW OPTION NAME FOR "weight_assignement == 6"
        '6. Weights: EDITABLE', ...   % EDIT NEW OPTION NAME FOR "weight_assignement == 7"
        '7. Weights: EDITABLE', ...   % EDIT NEW OPTION NAME FOR "weight_assignement == 8"
        '8. Weights: EDITABLE', ...   % EDIT NEW OPTION NAME FOR "weight_assignement == 9"
        '   ', ...
        '   *** Future options ***'};
    data = 0;
    weight_inverse = 0;
else
    data = varargin{1};
    available_weight_option_names = {};
```

» Implement the new weight_vector under the respective weight_assignment (+1 needed compared to the list name). Specify a weight for each marker in the vector and reshape it so that the weight is applied equally to the x,y,z coordinates. The resulting weight vector must be of the same length as the number of columns in the input matrix (i.e. number of marker coordinates). Use the other weight implementations as examples.

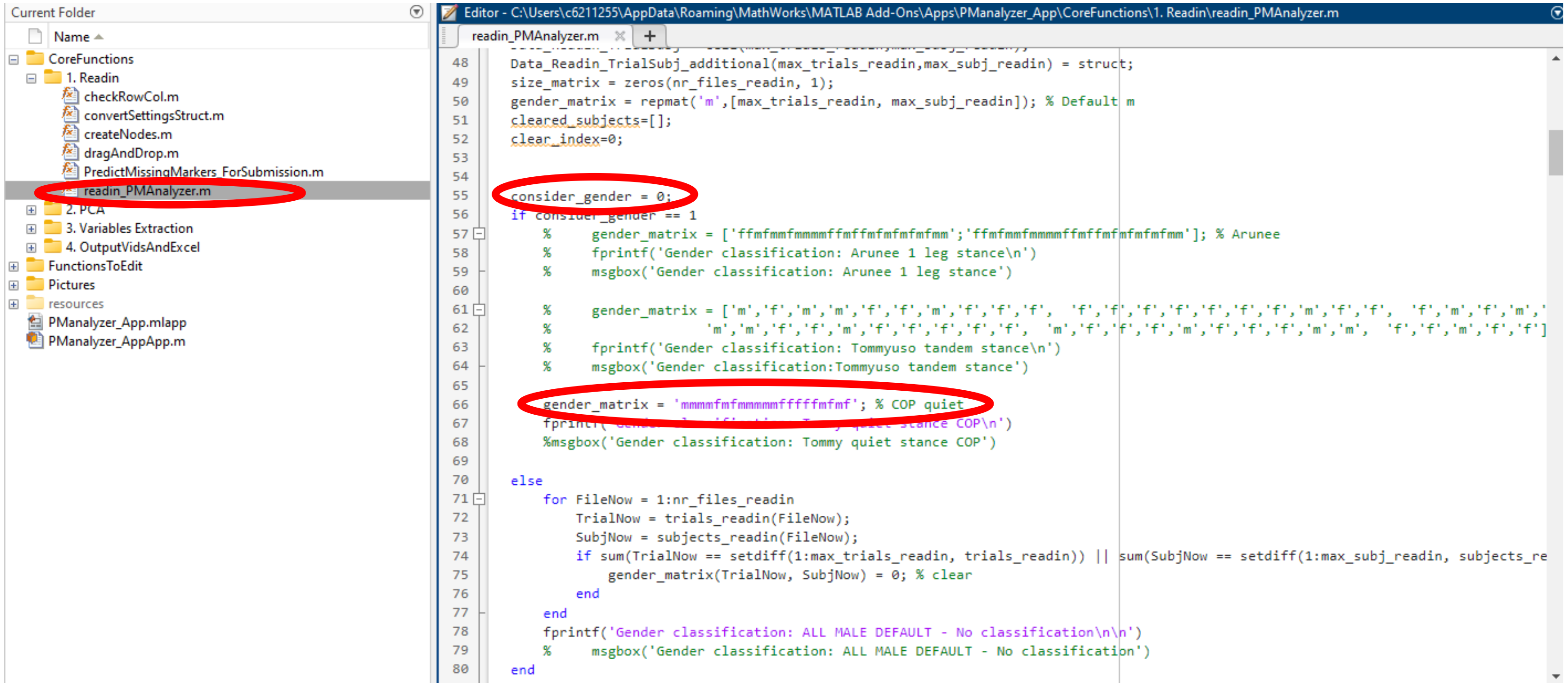
optionsWeights.m

```
elseif weight_assignement == 6 % EDIT weights for option number 12
%     ==================================================
%      Example template: one number for each marker (x-y-z coordinates will have same number)
%     ==================================================
%     weight_vector = ones(1,size(data,2)/3);
%     weight_vector = reshape([weight_vector; weight_vector; weight_vector], 1, []);

%     ==================================================
%      Alternative template: one number for each coordinate
%     ==================================================
%     weight_vector = ones(1,size(data,2));
```

Customization (Example “weights”)

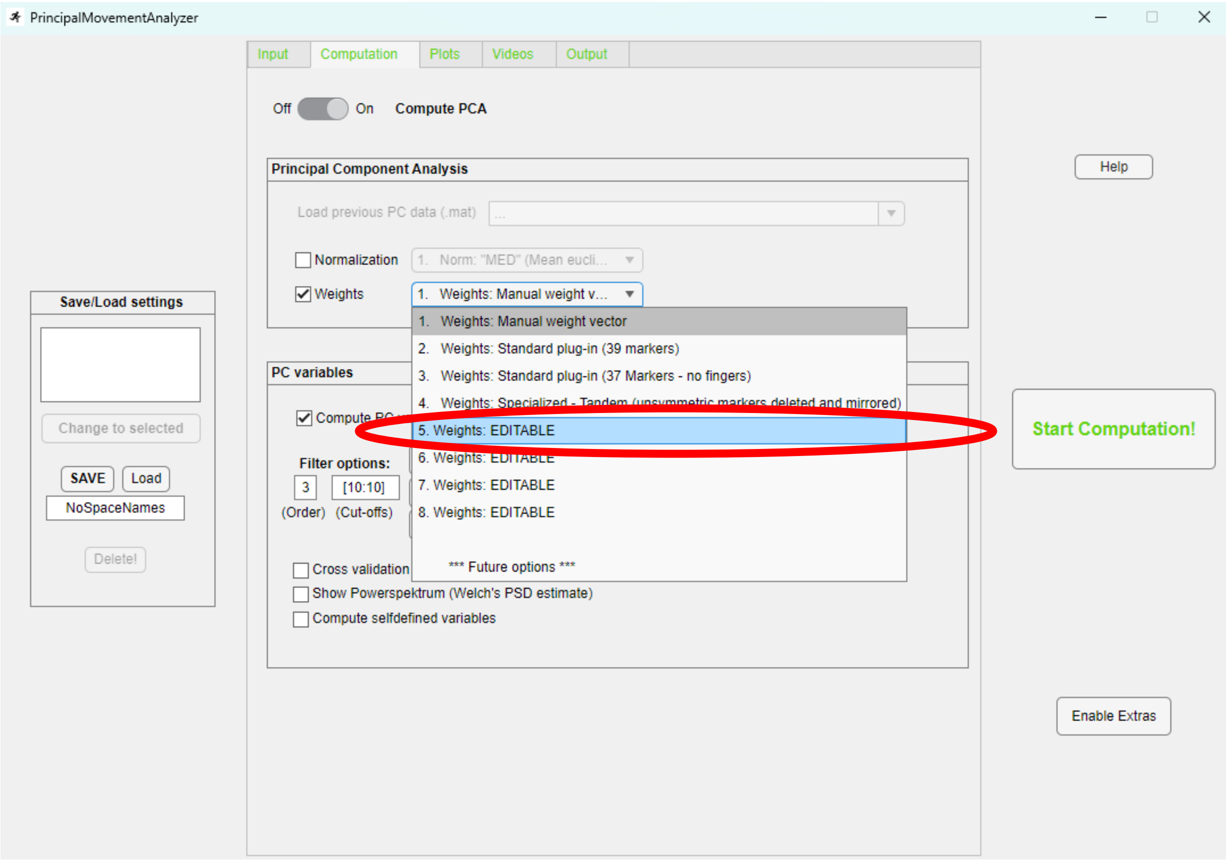
» Use for example: de Leva, P. (1996). Adjustments to Zatsiorsky-Seluyanov’s segment inertia parameters. *Journal of Biomechanics*, *29(9)*, 1223–1230. https://doi.org/10.1016/0021-9290(95)00178-6

» If using different weights for different gender, open the script „readin_PMAnalyzer.m“, change consider_gender to 1 and adapt the gender_matrix according to your subject order. Use the variable „sex_subject“ in „optionsWeights.m“ then.



Customization (Example “weights”)

» When restarting the app, the new weight option will appear in the list



Customization (Example “video stick-figure”)

» Open the script „optionsColoringVids.m“

» Add a name for the new video option

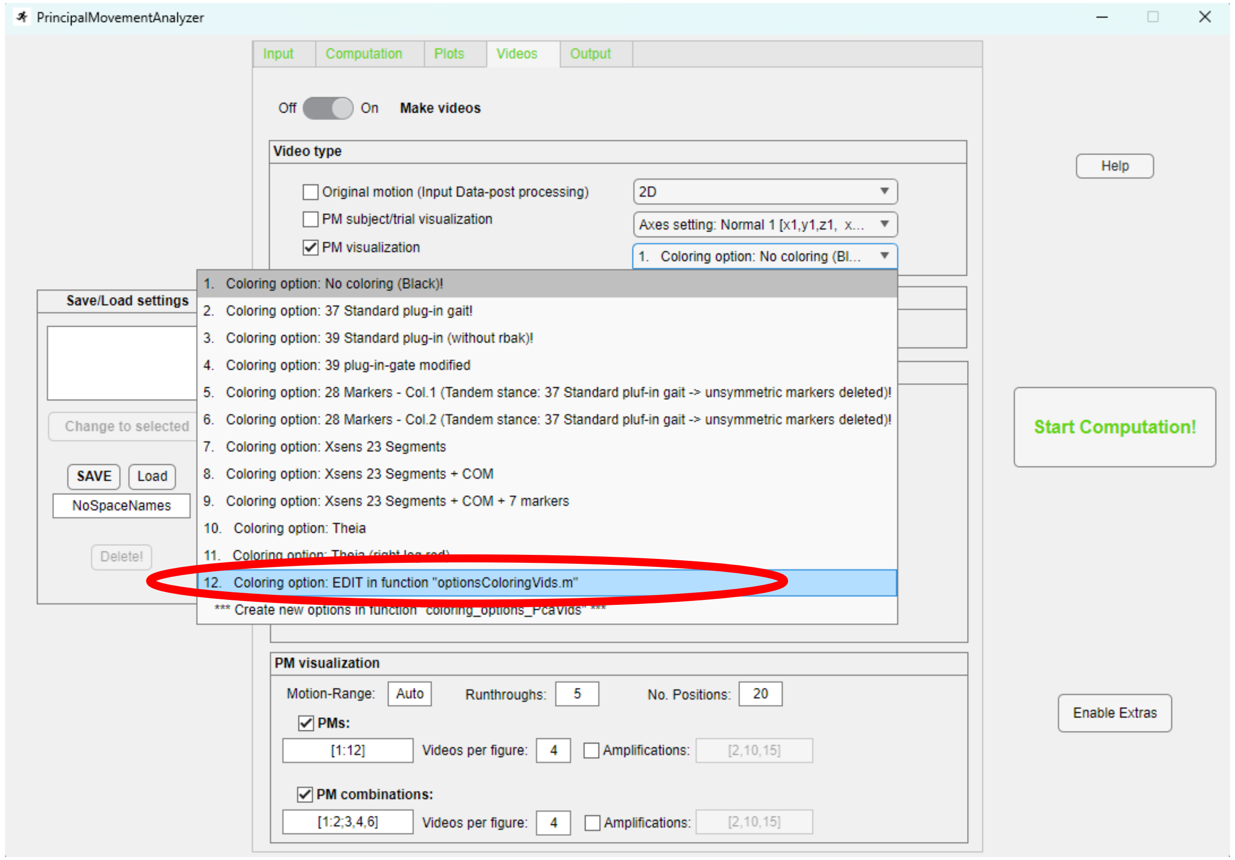
```
optionsColoringVids.m

if nargin == 0
    ColorOptionsCell = {};
    coloring_option_chosen = 0;
    available_coloring_option_names = {'1.   Coloring option: No coloring (Black)!', ...
                                       '2.   Coloring option: 37 Standard plug-in gait!', ...
                                       '3.   Coloring option: 39 Standard plug-in (without rbak)!', ...
                                       '4.   Coloring option: 39 plug-in-gate modified',...
                                       '5.   Coloring option: 28 Markers - Col.1 (Tandem stance: 37 Standard pluf-in gait -
                                       '6.   Coloring option: 28 Markers - Col.2 (Tandem stance: 37 Standard pluf-in gait -
                                       '7.   Coloring option: Xsens 23 Segments', ...
                                       '8.   Coloring option: Xsens 23 Segments + COM', ...
                                       '9.   Coloring option: Xsens 23 Segments + COM + 7 markers',...
                                       '10.  Coloring option: Theia', ...
                                       [illegible]
                                       '12.  Coloring option: EDIT in function "optionsColoringVids.m"', ...
                                       [illegible]
                                       };
else
    available_coloring_option_names = {};
    coloring_option_chosen = varargin{1};
```

» Implement the new video stick-figures under the respective coloring_option_chosen. Create a cell array with marker numbers (1st column), style (2nd column), color (3rd column) and size (4th column). Depending on the style, the specified markers will be plotted as points (e.g. ‘o’) or as points connected with lines (e.g. ‘o-’). Use different rows for different styles/marker connections. Use the other stick-figure implementations as an example. Note: Only the markers that are specified will be plotted.

```
optionsColoringVids.m

elseif coloring_option_chosen == 12

%     fprintf('\n        Coloring option: Julians 14 marker coloring video option is chosen!')
%     ColorOptionsCell = cell(4,4); % 4 lines
%
%     % Markers to plot per line
%     ColorOptionsCell{1,1} = [13, 14, 1, 2]; % Line 1 (Connection: Marker nr. 13-14-1-2)
%     ColorOptionsCell{1,2} = [11, 12]; % Line 2 (Connection: Marker nr. 11-12)
%     ColorOptionsCell{1,3} = [ 6,  1]; % Line 3 (Connection: Marker nr. 6-1)
%     ColorOptionsCell{1,4} = [ 2,  7]; % Line 4 (Connection: Marker nr. 1-7)
%     % Symbol and connection of each line      % Colors of each Markerface        % Markesizes for each line
%     ColorOptionsCell{2,1} = 'o-';             ColorOptionsCell{3,1} = 'r';       ColorOptionsCell{4,1} = 5;
%     ColorOptionsCell{2,2} = 'o-';             ColorOptionsCell{3,2} = 'g';       ColorOptionsCell{4,2} = 5;
%     ColorOptionsCell{2,3} = 'o-';             ColorOptionsCell{3,3} = 'b';       ColorOptionsCell{4,3} = 1;
%     ColorOptionsCell{2,4} = 'o-';             ColorOptionsCell{3,4} = 'b';       ColorOptionsCell{4,4} = 1;
```

Customization (Example “video stick-figure”)

» When restarting the app, the new video option will appear in the list



universität innsbruck

Evaluation of the results

- Look at PM videos in „Video“ folder and interpret them -> subjective visual assessment, but PMs are still objective

  (Especially PM subject/trial visualization is recommended for a first impression; PM visualization can be confusing if the extremes are too pronounced)

- Look at Excel output in „Computation“ folder, e.g. relative variances, self-computed variables

- Look at score waveforms in „Plot“ folder, e.g. mean or extremes of PP/PV/PA or their entire shape via waveform PCA or SPM.

Pataky, T. C., Robinson, M. A., & Vanrenterghem, J. (2013). Vector field statistical analysis of kinematic and force trajectories. *Journal of Biomechanics*, *46*(14), 2394–2401. https://doi.org/10.1016/j.jbiomech.2013.07.031

Mohr, M., Pieper, R., Löffler, S., Schmidt, A. R., & Federolf, P. A. (2021). Sex-Specific Hip Movement Is Correlated With Pelvis and Upper Body Rotation During Running. *Frontiers in Bioengineering and Biotechnology*, *9*. https://doi.org/10.3389/fbioe.2021.657357

Visualization of the results

» Visualize PMs with specific self-selected scores

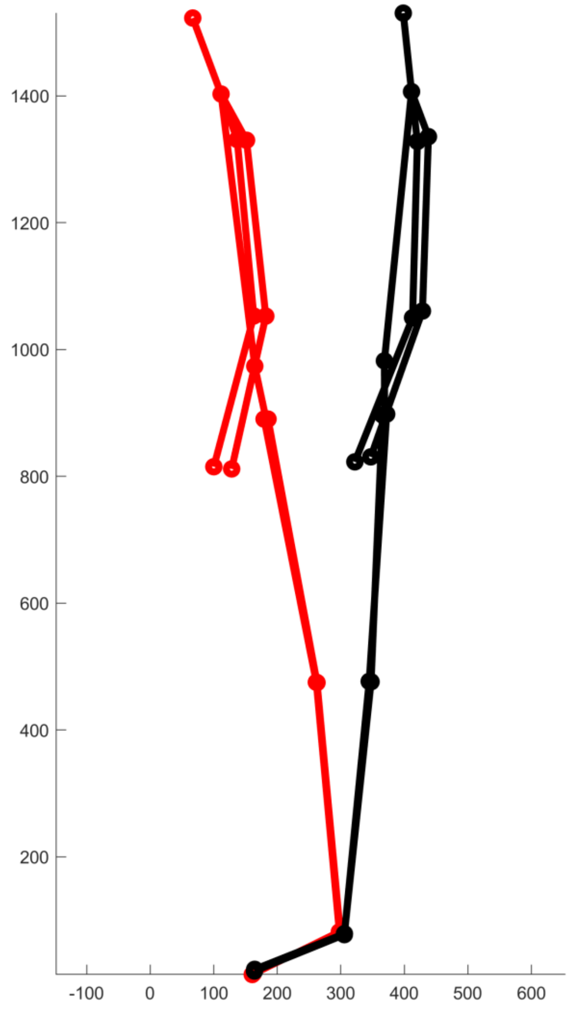
$$\boldsymbol{PM_k}(t) = \boldsymbol{PP_k}(t) \cdot \overbrace{\boldsymbol{PC_k^T}}^{\text{in Computation\\\_\_DataPCA\_Basic.mat -> PCs}} \cdot \underbrace{d_{norm} \cdot \boldsymbol{\omega}^{-1} + \overline{\boldsymbol{p}}}_{\text{in Computation\\\_\_DataPCA\_Expanded.mat}}$$

in Computation__DataPCA_Expanded.mat
-> norm_matrix()
-> weights_inverse{}
-> mean_matrices{}

```
% specify PCA data path:
data_pca = '.\Testdaten Theia\PCA_FirstRun3';

% load data
data_pca_basic = load([data_pca '\Computation\noCO___DataPCA_Basic.mat']);
data_pca_expanded = load([data_pca '\Computation\noCO___DataPCA_Expanded.mat']);

% specify subject and trial:
subject = 13;
trial = 1;

%% compute PMk with two different scores (PP values)
k = 2;
pp = 10;
pm_plus = pp*data_pca_basic.PCs(:,k)'*data_pca_expanded.norm_matrix(trial,subject)*data_pca_expanded.weights_inverse{trial,subject} + ...
    data_pca_expanded.mean_matrices{trial,subject}(1,:);
pm_minus = -pp*data_pca_basic.PCs(:,k)'*data_pca_expanded.norm_matrix(trial,subject)*data_pca_expanded.weights_inverse{trial,subject} + ...
    data_pca_expanded.mean_matrices{trial,subject}(1,:);
```

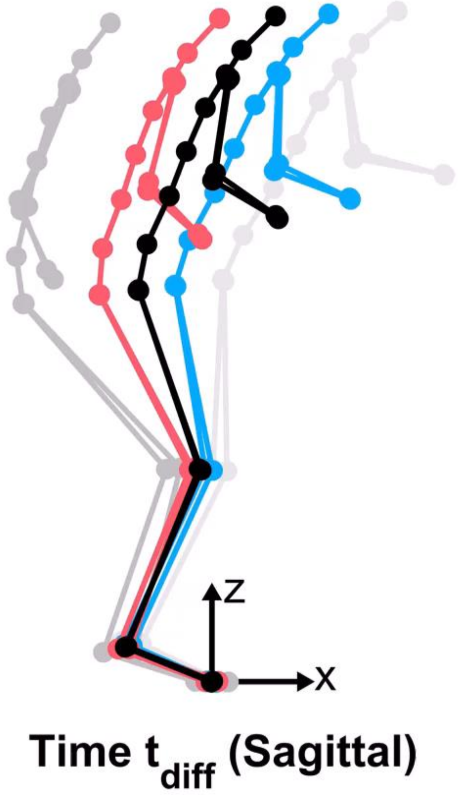
Visualization of the results

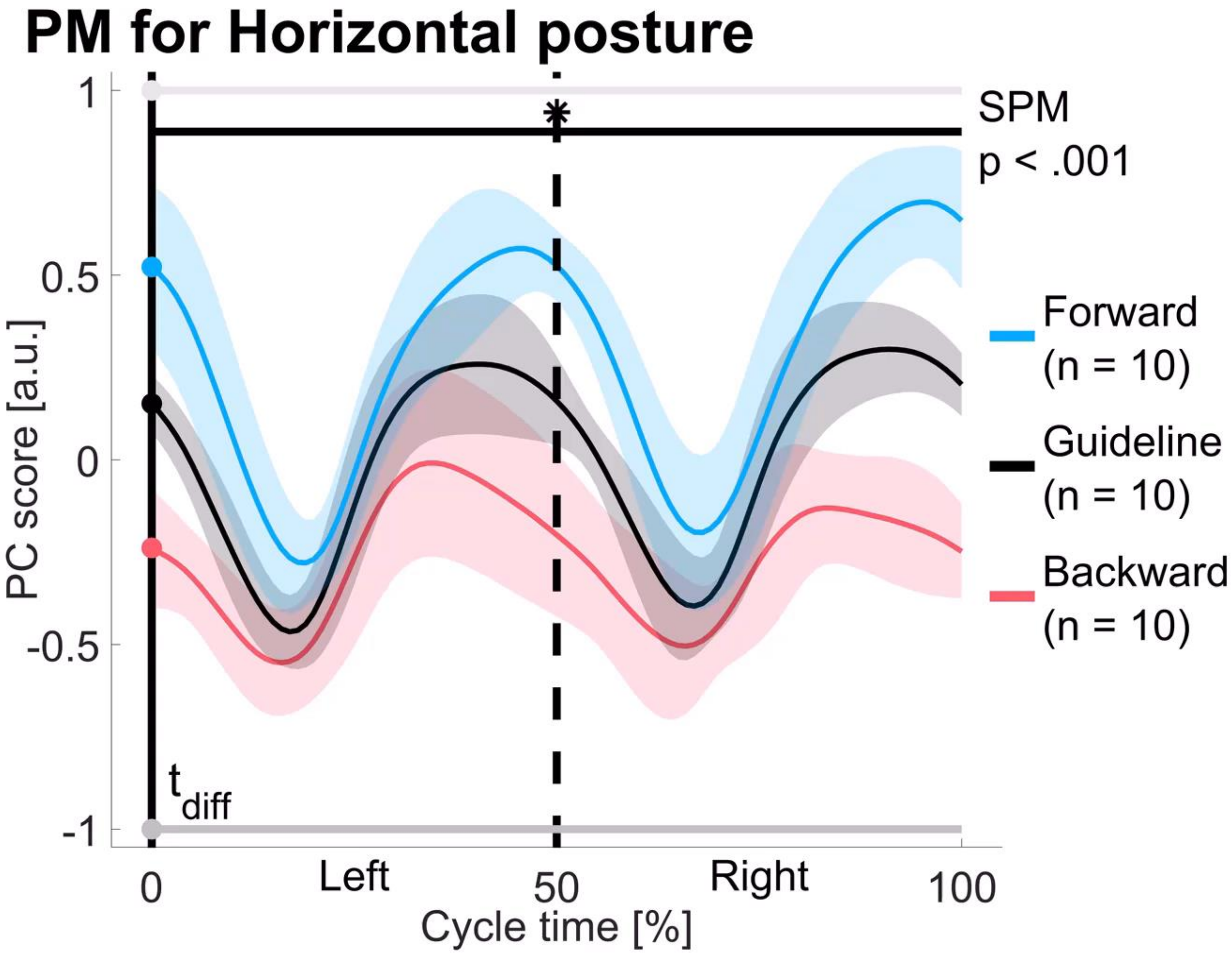
```
%% visualize it
% Markers to plot per line -> get it from PManaylzer "optionsColoringVids.m"
theia_ColorOptionsCell{1,1} = [1, 20, 11, 10]; % Line 1
theia_ColorOptionsCell{1,2} = [20, 2, 9, 5, 4]; % Line 2
theia_ColorOptionsCell{1,3} = [20, 12, 19, 15, 14]; % Line 3
theia_ColorOptionsCell{1,4} = [10, 7, 6, 3, 8]; % Line 4
theia_ColorOptionsCell{1,5} = [10, 17, 16, 13, 18]; % Line 5

sagittal = 1; % 2 = true

h1=figure('Position',[10,10,500,900]);
hold on

% PM plus (in red)
for PlotLine = 1:size(theia_ColorOptionsCell, 2)
     plot(pm_plus(end,theia_ColorOptionsCell{1,PlotLine}*3+sagittal-3),... % x or y
    pm_plus(end,theia_ColorOptionsCell{1,PlotLine}*3),... % z
    'o-','Color','r','LineWidth',4);
end

% PM minus (in black)
for PlotLine = 1:size(theia_ColorOptionsCell, 2)
    plot(pm_minus(end,theia_ColorOptionsCell{1,PlotLine}*3+sagittal-3),... % x or y
    pm_minus(end,theia_ColorOptionsCell{1,PlotLine}*3),... % z
    'o-','Color','k','LineWidth',4);
end

axis('equal')

exportgraphics(gcf, ['Visualization_PM' num2str(k) '.png'], 'Resolution', 300);

close(h1)
```



Visualization examples

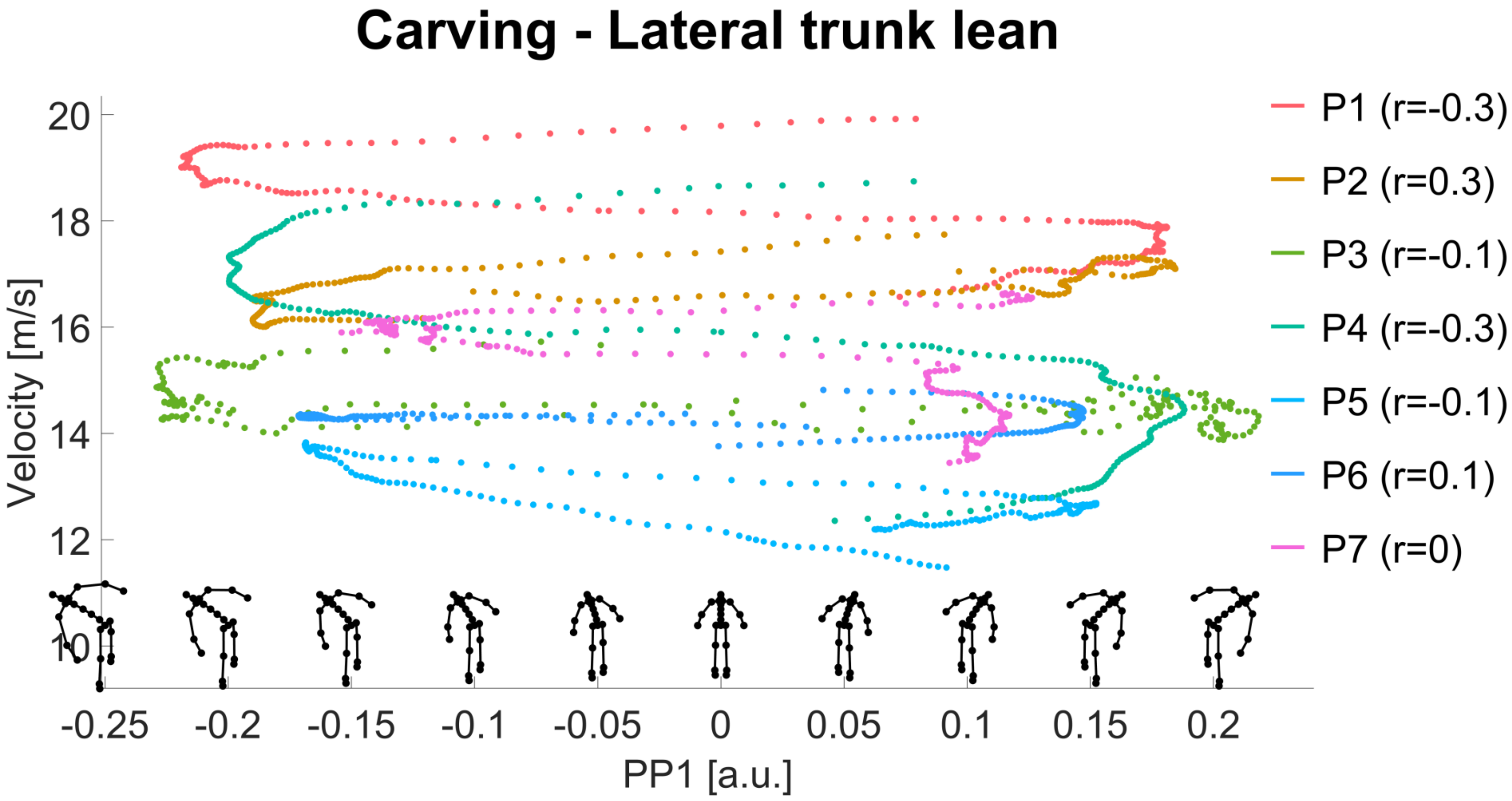
Cross-country skiing





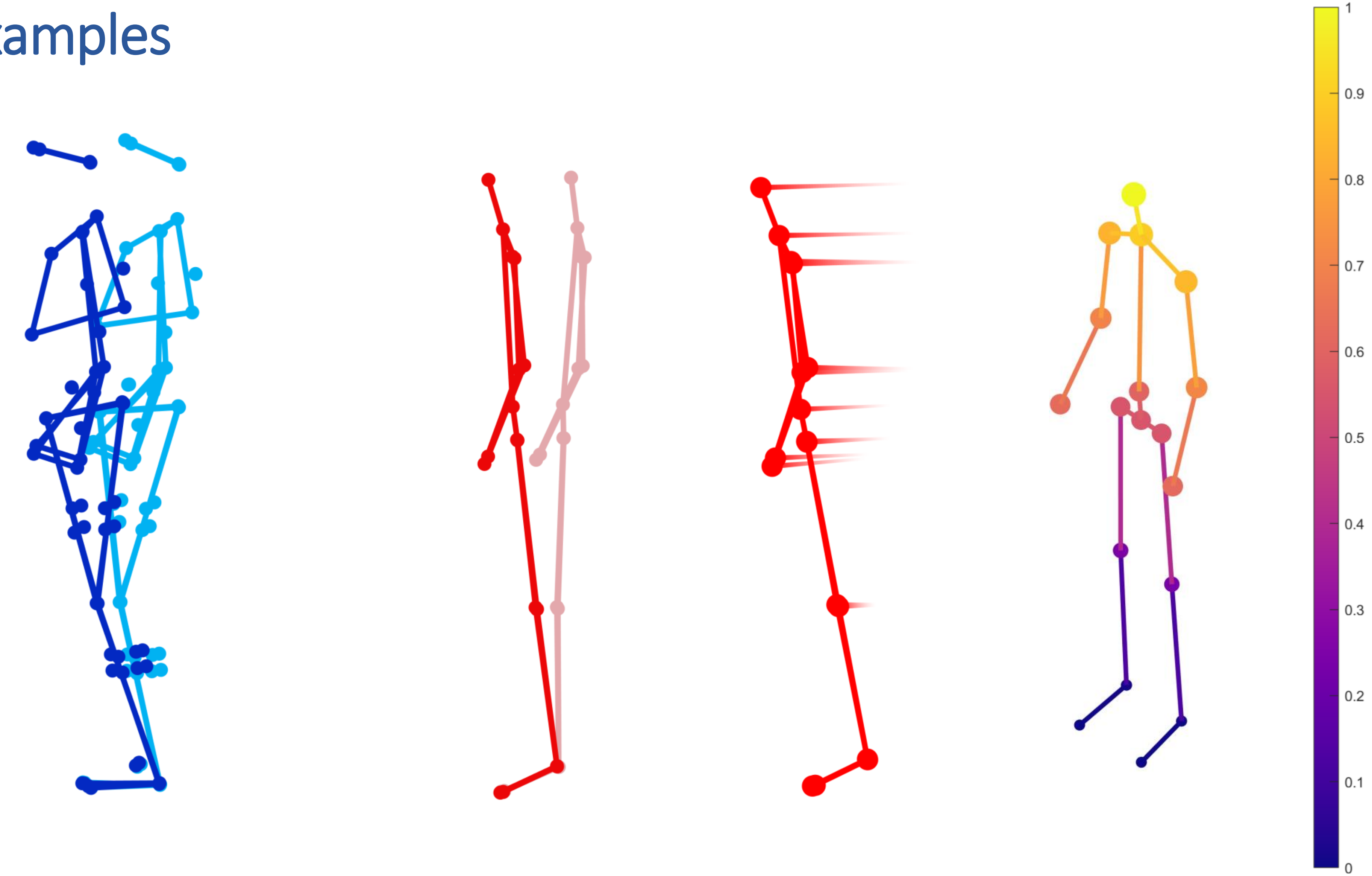
Visualization examples

Alpine skiing



Visualization examples

Postural control
(Theia validation)



Visualization examples

Postural control
(Theia validation)