

# Manual SimTK optcntrlmuscle (v2.1)

Friedl De Groote, Maarten Afschrift, Tom Van Wouwe, Antoine Falisse

05/11/2018

## Contents

<b>1</b>	<b>Release notes</b>	<b>2</b>
1.1	Release 2.1 . . . . .	2
<b>2</b>	<b>Overview</b>	<b>2</b>
<b>3</b>	<b>Installation Instruction</b>	<b>2</b>
<b>4</b>	<b>Main Function</b>	<b>3</b>
4.1	Using GPOPS . . . . .	3
4.1.1	With explicit activation dynamics formulation (De Groote et al. (2016)) .	3
4.1.2	With implicit activation dynamics formulation (De Groote et al. (2009)) .	3
4.2	Using CasADi . . . . .	4
4.2.1	With explicit activation dynamics formulation (De Groote et al. (2016)) .	4
4.2.2	With implicit activation dynamics formulation (De Groote et al. (2009)) .	4
4.3	Input Arguments . . . . .	4
4.4	Output arguments . . . . .	5
4.4.1	Using GPOPS . . . . .	5
4.4.2	Using CasADi . . . . .	6
<b>5</b>	<b>GPOPS-II</b>	<b>7</b>
5.1	Setup . . . . .	7
5.2	Output . . . . .	7
<b>6</b>	<b>CasADi</b>	<b>8</b>
6.1	Setup . . . . .	8
6.2	Output . . . . .	8
<b>7</b>	<b>Muscle model</b>	<b>8</b>
<b>8</b>	<b>Examples</b>	<b>8</b>
8.1	Walking example De Groote et al. 2016 . . . . .	8
8.2	Running example De Groote et al. 2016 . . . . .	9
8.3	OpenSim installation example Gait10dof18m . . . . .	9
8.4	OpenSim installation example Gait23dof54m . . . . .	10

# 1 Release notes

## 1.1 Release 2.1

- CasADi was added as an alternative to GPOPS-II and ADiGator (see section 2 for details).
- The reserve actuators (RActivation) were unscaled in the output of the main functions.
- The time derivatives of the muscle contraction dynamics states, i.e. normalized muscle fiber velocities or derivatives of normalized tendon forces, were added to the cost function with a small weighting factor to prevent spiky outputs.
- The tendon stiffness was added as an optional user parameter (see section 8.3 for example).

## 2 Overview

The provided MATLAB code solves the muscle redundancy problem using direct collocation as described in *De Groote F, Kinney AL, Rao AV, Fregly BJ. Evaluation of direct collocation optimal control problem formulations for solving the muscle redundancy problem. Annals of Biomedical Engineering (2016).* <http://link.springer.com/article/10.1007%2Fs10439-016-1591-9>.

From v2.1, CasADi can be used as an alternative to GPOPS-II and ADiGator. CasADi is an open-source tool for nonlinear optimization and algorithmic differentiation (<https://web.casadi.org/>). Results using CasADi and GPOPS-II are very similar (differences can be attributed to the different direct collocation formulations and scaling). We used CasADi's Opti stack, which is a collection of CasADi helper classes that provides a close correspondence between mathematical NLP notation and computer code (<https://web.casadi.org/docs/#document-opti>). CasADi is actively maintained and developed, and has an active forum (<https://groups.google.com/forum/#!forum/casadi-users>).

From v1.1, an implicit formulation of activation dynamics can be used to solve the muscle redundancy problem. Additionally, by using the activation dynamics model proposed by Raasch et al. (1997), we could introduce a nonlinear change of variables to exactly impose activation dynamics in a continuously differentiable form, omitting the need for a smooth approximation such as described in De Groote et al. (2016). A result of this change of variables is that muscle excitations are not directly accessible during the optimization. Therefore, we replaced muscle excitations by muscle activations in the objective function. This implicit formulation is described in *De Groote F, Pipeleers G, Jonkers I, Demeulenaere B, Patten C, Swevers J, De Schutter J. A physiology based inverse dynamic analysis of human gait: potential and perspectives F. Computer Methods in Biomechanics and Biomedical Engineering (2009).* <http://www.tandfonline.com/doi/full/10.1080/10255840902788587>. Results from both formulations are very similar (differences can be attributed to the slightly different activation dynamics models and cost functions). However, the formulation with implicit activation dynamics (De Groote et al., (2009)) is computationally faster. This can mainly be explained by the omission of a tanh function in the constraint definition, whose evaluation is computationally expensive when solving the NLP.

## 3 Installation Instruction

Add the main folder and subfolder to your MATLAB path

```
1 addpath(genpath('C/...../SimTK-optcntrlmuscle')).
```

Several software packages are needed to run the program

- The OpenSim MATLAB interface is used to generate the inputs to the optimal control problem based on a scaled OpenSim model and the solution of inverse kinematics (providing the solution of inverse dynamics is optional). To this aim, install OpenSim and set up the OpenSim MATLAB interface (OpenSim: [https://simtk.org/frs/?group\\_id=91](https://simtk.org/frs/?group_id=91), OpenSim API: <http://simtk-confluence.stanford.edu:8080/display/OpenSim/Scripting+with+Matlab>).
- Using GPOPS
  - GPOPS-II is used to solve the optimal control problem using direct collocation (<http://www.gpops2.com/>). A one-time 30-day trial license is available for all users who register.
  - ADiGator is used for automatic differentiation (<https://sourceforge.net/projects/adigator/>).
- Using CasADi (from v2.1)
  - CasADi is used for nonlinear optimization and algorithmic differentiation (<https://web.casadi.org/>).

## 4 Main Function

SolveMuscleRedundancy is the main function of this program and is used to solve the muscle redundancy problem. There are eight variants of this function that differ based on the chosen optimization package (GPOPS or CasADi), the formulation of the contraction dynamics (normalized tendon force or normalized muscle fiber length as a state), and the formulation of the activation dynamics (explicit or implicit).

### 4.1 Using GPOPS

#### 4.1.1 With explicit activation dynamics formulation (De Groote et al. (2016))

- SolveMuscleRedundancy\_FtildeState\_GPOPS uses the normalized tendon force as a state
- SolveMuscleRedundancy\_lMtildeState\_GPOPS uses the normalized muscle fiber length as a state

#### 4.1.2 With implicit activation dynamics formulation (De Groote et al. (2009))

- SolveMuscleRedundancy\_FtildeState\_actdyn\_GPOPS uses the normalized tendon force as a state
- SolveMuscleRedundancy\_lMtildeState\_actdyn\_GPOPS uses the normalized muscle fiber length as a state

## 4.2 Using CasADi

### 4.2.1 With explicit activation dynamics formulation (De Groote et al. (2016))

- `SolveMuscleRedundancy_FtildeState_CasADi` uses the normalized tendon force as a state
- `SolveMuscleRedundancy_lMtildeState_CasADi` uses the normalized muscle fiber length as a state

### 4.2.2 With implicit activation dynamics formulation (De Groote et al. (2009))

- `SolveMuscleRedundancy_FtildeState_actdyn_CasADi` uses the normalized tendon force as a state
- `SolveMuscleRedundancy_lMtildeState_actdyn_CasADi` uses the normalized muscle fiber length as a state

## 4.3 Input Arguments

### Required input arguments for `SolveMuscleRedundancy`

1. **model\_path:** directory and filename of the scaled OpenSim model (.osim file). The code should work with any OpenSim model with valid muscle-tendon parameters for which OpenSim's Inverse Dynamics and Muscle Analysis Tools generate reliable results. Note that only the muscle-tendon parameters and not the muscle model specified in the osim-file are used (for details see Muscle model).
2. **IK\_path:** directory and filename of the inverse kinematics solution (.mot file).
3. **ID\_path:** directory and filename of the inverse dynamics solution (.sto file). If left empty, the inverse dynamics solution will be computed from the external loads (see Optional input arguments).
4. **time:** 1 x 2 MATLAB array with the initial and final time of the analysis in seconds. Initial and final states influence the optimal controls over a period of about 50 ms at the beginning and end of the time interval over which the optimal control problem is solved. Since in practice the initial and final states are generally unknown, problems should be solved for a time interval containing five additional data points (considering a 100Hz sampling frequency) at the beginning and end of the motion cycle. Those additional data points should not be considered in further analyses. The user should thus not be surprised to observe unrealistically high muscle activation at the beginning of the motion (more details in companion paper).
5. **Out\_path:** directory where you want to store the results from the muscle analysis.
6. **Misc:** miscellaneous input arguments
  - *DofNames\_Input* is a cell array specifying for which degrees of freedom you want to solve the muscle redundancy problem. Typically the muscle redundancy problem is solved for one leg at a time (there are no muscles spanning both legs).
  - *MuscleNames\_Input* is a cell array that specifies the muscles to be included when solving the muscle redundancy problem. All muscles that actuate (i.e. have a moment arm with respect to) the degrees of freedom specified in *DofNames\_Input* will be selected by default if this array is left empty.

## Optional input arguments for SolveMuscleRedundancy

1. **Misc.Loads\_path**: directory and filename of the external loads (.xml file). The program will use the OpenSim libraries to solve the inverse dynamics problem when the required input argument *ID\_path* is empty and *Misc.Loads\_path* points to an external loads file.
2. **Misc.ID\_ResultsPath**: directory where the inverse dynamics results will be saved when the input argument *ID\_path* is left empty.
3. **Misc.f\_cutoff\_ID**: cutoff frequency for the butterworth recursive low pass filter applied to the inverse dynamics data (default is 6 Hz).
4. **Misc.f\_order\_ID**: order of the butterworth recursive low pass filter applied to the inverse dynamics data (default is 6).
5. **Misc.f\_cutoff\_LMT**: cutoff frequency for the butterworth recursive low pass filter applied to the muscle tendon lengths from the muscle analysis (default is 6 Hz).
6. **Misc.f\_order\_LMT**: order of the butterworth recursive low pass filter applied to the muscle tendon lengths from the muscle analysis (default is 6).
7. **Misc.f\_cutoff\_dM**: cutoff frequency for the butterworth recursive low pass filter applied to the muscle moment arms from the muscle analysis (default is 6 Hz).
8. **Misc.f\_order\_dM**: order of the butterworth recursive low pass filter applied to the muscle moment arms from the muscle analysis (default is 6).
9. **Misc.f\_cutoff\_IK**: cutoff frequency for the butterworth recursive low pass filter applied to the inverse kinematics data (default is 6 Hz) when performing the muscle analysis to compute muscle-tendon lengths and moment arms.
10. **Misc.f\_order\_IK**: order of the butterworth recursive low pass filter applied to the inverse kinematics data (default is 6).
11. **Misc.Mesh\_Frequency**: number of mesh interval per second (default is 100, but a denser mesh might be required to obtain the desired accuracy especially for faster motions).
12. **Misc.Atendon**: vector with tendon stiffness for the selected muscles. The order should correspond to *MuscleNames\_Input*. The default value is 35 and a lower value corresponds to a more compliant tendon. The default value will be used when left empty. An example is provided in section 8.3 to set a different stiffness to the Achilles tendon.

## 4.4 Output arguments

### 4.4.1 Using GPOPS

1. Time: time vector.
2. MExcitation: optimal muscle excitation (matrix dimension: number of collocation points x number of muscles).
3. MActivation: optimal muscle activation (matrix dimension: number of collocation points x number of muscles).

4. RActivation: activation of the reserve actuators (matrix dimension: number of collocation points x number of degrees of freedom).
5. TForcetilde: normalized tendon force (matrix dimension: number of collocation points x number of muscles).
6. TForce: tendon force (matrix dimension: number of collocation points x number of muscles).
7. IMtilde: normalized muscle fiber length (matrix dimension: number of collocation points x number of muscles).
8. IM: muscle fiber length (matrix dimension: number of collocation points x number of muscles) .
9. MuscleNames: cell array that contains the names of the selected muscles (matrix dimension: number of muscles).
10. OptInfo: output structure created by GPOPS-II.
11. DatStore: data structure with input information for the optimal control problem.

#### 4.4.2 Using CasADi

CasADi uses piecewise-constant controls in the mesh intervals. We therefore distinguish between collocation points (for the states) and mesh points (for the states and controls) in the output arguments.

1. Time: time vector
  - (a) Time.meshPoints: time at mesh points
  - (b) Time.collocationPoints: time at collocation points
2. MExcitation.meshPoints: muscle excitation (matrix dimension: number of mesh points x number of muscles).
3. MActivation: muscle activation
  - (a) MActivation.meshPoints: muscle activation at mesh points (matrix dimension: number of mesh points x number of muscles).
  - (b) MActivation.collocationPoints: muscle activation at collocation points (matrix dimension: number of collocation points x number of muscles).
4. RActivation.meshPoints: activation of the reserve actuators (matrix dimension: number of mesh points x number of degrees of freedom).
5. TForcetilde: normalized tendon force
  - (a) TForce.meshPoints: normalized tendon force at mesh points (matrix dimension: number of mesh points x number of muscles).
  - (b) TForce.collocationPoints: normalized tendon force at collocation points (matrix dimension: number of collocation points x number of muscles).
6. TForce: tendon force

- (a) TForce.meshPoints: tendon force at mesh points (matrix dimension: number of mesh points x number of muscles).
  - (b) TForce.collocationPoints: tendon force at collocation points (matrix dimension: number of collocation points x number of muscles).
7. IMtilde: normalized muscle fiber length
- (a) TForce.meshPoints: normalized muscle fiber length at mesh points (matrix dimension: number of mesh points x number of muscles).
  - (b) TForce.collocationPoints: normalized muscle fiber length at collocation points (matrix dimension: number of collocation points x number of muscles).
8. IM: muscle fiber length
- (a) TForce.meshPoints: muscle fiber length at mesh points (matrix dimension: number of mesh points x number of muscles).
  - (b) TForce.collocationPoints: muscle fiber length at collocation points (matrix dimension: number of collocation points x number of muscles).
9. MuscleNames: cell array that contains the names of the selected muscles (matrix dimension: number of muscles).
10. OptInfo: output structure with settings used in CasADi.
11. DatStore: data structure with input information for the optimal control problem.

## 5 GPOPS-II

### 5.1 Setup

The GPOPS-II setup is accessible through the function `SolveMuscleRedundancy_< ... >_-GPOPS.m` under GPOPS setup. The user is referred to the GPOPS-II user guide for setup options. A higher accuracy can be reached by adjusting, for instance, the number of mesh intervals. This however comes at the expense of the computational time. 100 mesh intervals per second are used by default.

### 5.2 Output

The output variable `OptInfo` contains all information related to the optimal control problem solution. Convergence to an optimal solution is reached when `output.result.nlpinfo` is flagged 0 ("EXIT: Optimal solution found" in the command window of MATLAB). The mesh accuracy can be assessed with `output.result.maxerrors`. Cost functional, control, state (and costate) can be accessed in `output.result.solution.phase`.

To recall, the user should consider extending the time interval by 50-100 ms at the beginning and end of the motion to limit the influence of the unknown initial and final state on the solution. Results from those additional periods should not be considered realistic and will typically result in high muscle activation.

## 6 CasADi

### 6.1 Setup

The CasADi setup is accessible through the function `SolveMuscleRedundancy_< ... >_CasADi.m` under CasADi setup. The user is referred to the CasADi user guide for setup options.

### 6.2 Output

The output variable `OptInfo` contains information related to the optimal control problem settings. As with GPOPS, the user should consider extending the time interval by 50-100 ms at the beginning and end of the motion to limit the influence of the unknown initial and final state on the solution. Results from those additional periods should not be considered realistic and will typically result in high muscle activation.

## 7 Muscle model

The musculotendon properties are fully described in the supplementary materials of the aforementioned publication. Importantly, only the tendon slack length, optimal muscle fiber length, maximal isometric muscle force, optimal pennation angle and maximal muscle fiber contraction velocity are extracted from the referred OpenSim model. Other properties are defined in the code and can be changed if desired. By default, the activation and deactivation time constants are 15 and 60 ms respectively (see `tau_act` and `tau_deact` in `SolveMuscleRedundancy_< state >.m`).

## 8 Examples

Four examples are provided in the folder `examples`.

### 8.1 Walking example De Groote et al. 2016

```
1 clear all; close all; clc
2 %% Choose formulation
3 % formulation = 'lMtildeState';
4 formulation = 'FtildeState';
5 %% Choose activation dynamics
6 % actdyn = 'DeGroote2016';
7 actdyn = 'DeGroote2009';
8 %% Example
9 % add main folder and subfolder to matlab path (installation)
10 filepath=which('Walking_DeGrooteetal2016.m');
11 [DirExample_Walking,~,~]=fileparts(filepath); ...
    [DirExample,~]=fileparts(DirExample_Walking); [MainDir,~]=fileparts(DirExample);
12 addpath(genpath(MainDir));
13
14 % Needed Input Arguments
15 IK_path=fullfile(MainDir, 'Examples', 'Walking_DeGrooteetal2016', 'WalkingData', 'inverse_kinematics.mot');
16 ID_path=fullfile(MainDir, 'Examples', 'Walking_DeGrooteetal2016', 'WalkingData', 'inverse_dynamics.sto');
17 model_path=fullfile(MainDir, 'Examples', 'Walking_DeGrooteetal2016', 'WalkingData', 'subject01.osim');
18 time=[0.516 1.95]; % Right stance phase (+50ms beginning and end of time interval, more ...
    details see manual and publication)
19 OutPath=fullfile(MainDir, 'Examples', 'Walking_DeGrooteetal2016', 'Results');
20
21 Misc.MuscleNames_Input={}; % Selects all muscles for the Input DOFS when this is left empty.
22 Misc.DofNames_Input={'ankle_angle_r', 'knee_angle_r', 'hip_flexion_r', 'hip_rotation_r', 'hip_adduction_r'};
23
24 % Optional Input Arguments
25 Misc.f.cutoff_ID = 6; % cutoff frequency filtering ID
26 Misc.f.order_ID = 4; % order frequency filtering ID
27 Misc.f.cutoff_IMT = 6; % cutoff frequency filtering IMT
28 Misc.f.order_IMT = 4; % order frequency filtering IMT
29 Misc.f.cutoff_dM = 6; % cutoff frequency filtering MA
30 Misc.f.order_dM = 4; % order frequency filtering MA
31 Misc.f.cutoff_IK = 6; % cutoff frequency filtering IK
32 Misc.f.order_IK = 4; % order frequency filtering IK
33 Misc.Atendon = []; % Tendon Stiffness for the selected muscles
```



```

34 %% Solve the problem
35 switch actdyn
36     case 'DeGroote2016' % Activation dynamics from De Groote et al. (2016)
37         switch formulation
38             case 'lMtildeState'
39                 [Time, MExcitation, MActivation, RActivation, TForcetilde, TForce, lMtilde, lM, MuscleNames, OptInfo, Data] = ...
40             case 'FtildeState'
41                 [Time, MExcitation, MActivation, RActivation, TForcetilde, TForce, lMtilde, lM, MuscleNames, OptInfo, Data] = ...
42             end
43         end
44     case 'DeGroote2009' % Activation dynamics from De Groote et al. (2009)
45         switch formulation
46             case 'lMtildeState'
47                 [Time_actdyn, MExcitation_actdyn, MActivation_actdyn, RActivation_actdyn, TForcetilde_actdyn, TForce_actdyn, lMtilde_actdyn, lM_actdyn, MuscleNames_actdyn, OptInfo_actdyn, Data_actdyn] = ...
48             case 'FtildeState'
49                 [Time_actdyn, MExcitation_actdyn, MActivation_actdyn, RActivation_actdyn, TForcetilde_actdyn, TForce_actdyn, lMtilde_actdyn, lM_actdyn, MuscleNames_actdyn, OptInfo_actdyn, Data_actdyn] = ...
50             end
51 end

```

## 8.2 Running example De Groote et al. 2016

```

1 clear all; close all; clc
2
3 %% Choose optimization framework
4 % framework = 'GPOPS';
5 framework = 'CasADi';
6
7 %% Choose contraction dynamics formulation
8 % formulation_contdyn = 'lMtildeState';
9 formulation_contdyn = 'FtildeState';
10
11 %% Choose activation dynamics formulation
12 % formulation_actdyn = 'DeGroote2016';
13 formulation_actdyn = 'DeGroote2009';
14
15 %% Example
16 % Add main folder and subfolder to matlab path (installation)
17 filepath=which('Walking-DeGrooteetal2016.m');
18 [DirExample, Walking, ~]=fileparts(filepath);
19 [DirExample, ~]=fileparts(DirExample-Walking); [MainDir, ~]=fileparts(DirExample);
20 addpath(genpath(MainDir));
21
22 %% Needed Input Arguments
23 IK_path=fullfile(MainDir, 'Examples', 'Walking-DeGrooteetal2016', 'WalkingData', 'Walking_IK.mot');
24 ID_path=fullfile(MainDir, 'Examples', 'Walking-DeGrooteetal2016', 'WalkingData', 'Walking_ID.sto');
25 model_path=fullfile(MainDir, 'Examples', 'Walking-DeGrooteetal2016', 'WalkingData', 'subject1.osim');
26 time=[0.516 1.95]; % Right stance phase (+50ms beginning and end of time interval, more ...
    details see manual and publication)
27 OutPath=fullfile(MainDir, 'Examples', 'Walking-DeGrooteetal2016', 'Results');
28
29 Misc.DofNames_Input={ 'ankle_angle_r', 'knee_angle_r', 'hip_flexion_r', 'hip_rotation_r', 'hip_adduction_r' };
30 Misc.MuscleNames_Input={}; % Selects all muscles for input DOFs when empty
31
32 %% Optional Input Arguments
33 Misc.f_cutoff_ID = 6; % cutoff frequency filtering ID
34 Misc.f_order_ID = 4; % order frequency filtering ID
35 Misc.f_cutoff_IMT = 6; % cutoff frequency filtering IMT
36 Misc.f_order_IMT = 4; % order frequency filtering IMT
37 Misc.f_cutoff_dM = 6; % cutoff frequency filtering MA
38 Misc.f_order_dM = 4; % order frequency filtering MA
39 Misc.f_cutoff_IK = 6; % cutoff frequency filtering IK
40 Misc.f_order_IK = 4; % order frequency filtering IK
41
42 %% Solve the problem
43 switch framework
44     case 'GPOPS'
45         switch formulation_actdyn
46             case 'DeGroote2016'
47                 switch formulation_contdyn
48                     case 'lMtildeState'
49                         [Time, MExcitation, MActivation, RActivation, TForcetilde, TForce, lMtilde, lM, MuscleNames, OptInfo, Data] = ...
50                     case 'FtildeState'
51                         [Time, MExcitation, MActivation, RActivation, TForcetilde, TForce, lMtilde, lM, MuscleNames, OptInfo, Data] = ...
52                     end
53             end
54         case 'DeGroote2009'
55             switch formulation_contdyn
56                 case 'lMtildeState'
57                     [Time_actdyn, MExcitation_actdyn, MActivation_actdyn, RActivation_actdyn, TForcetilde_actdyn, TForce_actdyn, lMtilde_actdyn, lM_actdyn, MuscleNames_actdyn, OptInfo_actdyn, Data_actdyn] = ...
58                 case 'FtildeState'
59                     [Time_actdyn, MExcitation_actdyn, MActivation_actdyn, RActivation_actdyn, TForcetilde_actdyn, TForce_actdyn, lMtilde_actdyn, lM_actdyn, MuscleNames_actdyn, OptInfo_actdyn, Data_actdyn] = ...
60                 end
61             end
62         case 'CasADi'
63             switch formulation_actdyn
64                 case 'DeGroote2016'
65                     switch formulation_contdyn
66                         case 'lMtildeState'

```

```

67         [Time, MExcitation, MActivation, RActivation, TForcetilde, TForce, lMtilde, lM, MuscleNames, OptI
68         case 'FtildeState'
69         [Time, MExcitation, MActivation, RActivation, TForcetilde, TForce, lMtilde, lM, MuscleNames, OptI
70     end
71
72     case 'DeGroote2009'
73     switch formulation_contdyn
74     case 'lMtildeState'
75     [Time_actdyn, MExcitation_actdyn, MActivation_actdyn, RActivation_actdyn, TForcetilde_actdyn,
76     case 'FtildeState'
77     [Time_actdyn, MExcitation_actdyn, MActivation_actdyn, RActivation_actdyn, TForcetilde_actdyn,
78     end
79     end
80 end

```

### 8.3 OpenSim installation example Gait10dof18m

```

1  clear all; close all; clc
2  %% Choose formulation
3  % formulation = 'lMtildeState';
4  formulation = 'FtildeState';
5  %% Choose activation dynamics formulation
6  % actdyn = 'DeGroote2016';
7  actdyn = 'DeGroote2009';
8  %% Example
9  % add main folder and subfolder to matlab path (installation)
10 filepath=which('Example_Gait10dof18m.m'); [DirExample, ~, ~]=fileparts(filepath); ...
    [DirExample2, ~, ~]=fileparts(DirExample); [MainDir, ~]=fileparts(DirExample2);
11 addpath(genpath(MainDir));
12
13 % Needed Input Arguments
14 Datapath='C:\OpenSim 3.3\Models\Gait10dof18musc\OutputReference';
15 IK_path=fullfile(Datapath, 'IK', 'subject01_walk_IK.mot');
16 ID_path=[]; % compute ID from the external loads
17 model_path=fullfile(Datapath, 'subject01.osim');
18 time=[0.7 1.4]; % Part of the right stance phase
19 OutPath=fullfile(MainDir, 'Examples', 'OpenSimInstallation_Gait10dof18m', 'Results');
20
21 Misc.DofNames_Input={ 'ankle_angle_r', 'knee_angle_r', 'hip_flexion_r' };
22 Misc.Loads_path=fullfile(Datapath, 'ExperimentalData', 'subject01_walk_grf.xml');
23 Misc.ID_ResultsPath=fullfile(Datapath, 'ID', 'inversedynamics.sto');
24
25 %% Solve the problem
26 switch actdyn
27 case 'DeGroote2016' % Activation dynamics from De Groote et al. (2016)
28     switch formulation
29     case 'lMtildeState'
30     [Time, MExcitation, MActivation, RActivation, TForcetilde, TForce, lMtilde, lM, MuscleNames, OptInfo, DatS
31     case 'FtildeState'
32     [Time, MExcitation, MActivation, RActivation, TForcetilde, TForce, lMtilde, lM, MuscleNames, OptInfo, DatS
33     end
34
35 case 'DeGroote2009' % Activation dynamics from De Groote et al. (2009)
36     switch formulation
37     case 'lMtildeState'
38     [Time_actdyn, MExcitation_actdyn, MActivation_actdyn, RActivation_actdyn, TForcetilde_actdyn, TForce_
39     case 'FtildeState'
40     [Time_actdyn, MExcitation_actdyn, MActivation_actdyn, RActivation_actdyn, TForcetilde_actdyn, TForce_
41     end
42 end

```

### 8.4 OpenSim installation example Gait23dof54m

```

1  clear all; close all; clc
2  %% Choose formulation
3  % formulation = 'lMtildeState';
4  formulation = 'FtildeState';
5  %% Choose activation dynamics formulation
6  % actdyn = 'DeGroote2016';
7  actdyn = 'DeGroote2009';
8  %% Example
9  % add main folder and subfolder to matlab path (installation)
10 filepath=which('Example_Gait23dof54m.m'); [DirExample, ~, ~]=fileparts(filepath); ...
    [DirExample2, ~, ~]=fileparts(DirExample); [MainDir, ~]=fileparts(DirExample2);
11 addpath(genpath(MainDir));
12
13 % Needed Input Arguments
14 Datapath='C:\OpenSim 3.3\Models\Gait2354_Simbody\OutputReference';
15 IK_path=fullfile(Datapath, 'subject01_walk1_ik.mot');
16 ID_path=fullfile(Datapath, 'ResultsInverseDynamics', 'inverse_dynamics.sto');
17 model_path=fullfile(Datapath, 'subject01_scaledOnly.osim');
18 time=[0.7 1.4]; % Part of the right stance phase

```

```

19 OutPath=fullfile(MainDir,'Examples','OpenSimInstallation_Gait23dof54m','Results');
20
21 Misc.MuscleNames_Input={}; % Selects all muscles for the Input DOFS when this is left empty.
22 Misc.DofNames_Input={'ankle-angle-r','knee-angle-r','hip-flexion-r'};
23
24 %% Solve the problem
25 switch actdyn
26     case 'DeGroote2016' % Activation dynamics from De Groote et al. (2016)
27         switch formulation
28             case 'lMtildeState'
29                 [Time,MExcitation,MActivation,RActivation,TForcetilde,TForce,lMtilde,lM,MuscleNames,OptInfo,DatS
30             case 'FtildeState'
31                 [Time,MExcitation,MActivation,RActivation,TForcetilde,TForce,lMtilde,lM,MuscleNames,OptInfo,DatS
32         end
33
34     case 'DeGroote2009' % Activation dynamics from De Groote et al. (2009)
35         switch formulation
36             case 'lMtildeState'
37                 [Time_actdyn,MExcitation_actdyn,MActivation_actdyn,RActivation_actdyn,TForcetilde_actdyn,TForce_a
38             case 'FtildeState'
39                 [Time_actdyn,MExcitation_actdyn,MActivation_actdyn,RActivation_actdyn,TForcetilde_actdyn,TForce_a
40         end
41 end

```