Tutorial 7 – Torque Driven Treatment Optimization

The Treatment Optimization toolset generates predictive simulations of a patient’s post-treatment movement function by optimizing specified treatment design parameters. The toolset consists of three tools designed using a “theme and variation” approach, where the tools are intended to be used in a specific order, with each tool serving a distinct purpose. Each tool uses the GPOPS-II direct collocation optimal control software for MATLAB and maintains a consistent structure for data inputs, problem design, cost function terms, constraint terms, and outputs, with variations.

This first section of the treatment optimization tutorial will cover the process of running Tracking Optimization (TO), Verification Optimization (VO), and Design Optimization (DO) in which joints are controlled by individual torque actuators. It will work through the design of settings files for all tools, along with analyzing the outputs of each tool with iterative problem-solving methods to generate good solutions.

**Section 1: Tracking Optimization**

The Tracking Optimization (TO) tool uses a personalized model to produce a dynamically consistent movement simulation that closely reproduces all available experimental motion data, including joint motions, joint moments, ground reaction forces and moments, and muscle activations. To achieve a dynamically consistent motion, the tool spreads out matching errors between the different experimental quantities based on user-specified maximum allowable errors.

The tool accepts a post-JMP OpenSim model (.osim fle) and personalized NMSM Pipeline model (.osimx fle) along with experimental IK motions, ID loads, ground reactions, muscle–tendon lengths and velocities, muscle moment arms, and, if using synergy controls, NCP results for the trial of interest.

This section of the tutorial will only be using torque controls with no external forces, so the only inputs we will use are a post-JMP OpenSim model, experimental IK motions, and ID loads.

**Before running Tracking Optimization:**

1. Open the OpenSim model KickingModel.osim in the OpenSim GUI.
2. This model is a full three-dimensional gait model, but similar to the model personalization tutorials, this tutorial will only be using the right hip flexion, knee angle, and ankle angle coordinates.
3. This model uses a ground-pelvis joint, so inverse dynamics will yield unrealistic “residual” loads applied to the pelvis body.
4. With the model open in OpenSim, load the IK results in preprocessed (preprocessed\IKData\drive\_kick1.sto) to visualize the motion.

**Creating a Tracking Optimization settings file:**

1. Activate the NMSM GUI in OpenSim by navigating to “Tools>User Plugins”, and click “rcnlPlugin.dll”
2. With “KickingModel.osim” selected in the OpenSim GUI, “Tools>Treatment Optimization >Tracking Optimization”
   1. The following window should be opened:

A screenshot of a computer

AI-generated content may be incorrect.

1. Leave the Osimx file field empty. An input OsimX file would be used if we wanted external loads or muscles to be used in the TO, but this run will not use either.
2. Set the initial guess directory to be “preprocessing”
   1. The initial guess directory specifies where the initial guess for the optimizer should be parsed from.
3. Set the tracked quantities directory to be "preprocessing”
   1. For an initial torque driven TO, this is often the same as the initial guess directory.
4. Ser the results directory to be “TorqueTOResults”
5. Under “Optimal Control Solver Settings File”, click the button “Generate default settings file”. Save this default settings file as “GpopsSettings.xml”. Next, click the browse button, and select the newly created “GpopsSettings.xml”
6. Under states coordinate list, select (hip\_flexion\_r, knee\_angle\_r, ankle\_angle\_r)
   1. The states coordinate list specifies which coordinates the optimizer is allowed to change. If a coordinate is not included in the states coordinate list, it is automatically prescribed. Therefore, most of the coordinates in this tutorial are prescribed.
7. Click to the RCNL Controllers tab at the top.
8. Under RCNL Torque Controller, add (hip\_flexion\_r, knee\_angle\_r, ankle\_angle\_r) to the coordinate list.
   1. This creates individual torque actuators for each of the coordinates that we include in the states.
9. Click to the Cost/Constraints tab at the top.
   1. Hint for the upcoming section: When you click on the GUI drop down arrows to select cost term type, you can type the name of the cost term and the GUI will auto-navigate to the correct cost term.
10. Add a new cost term:
    1. Name: Coordinate tracking
    2. Cost term type: generalized\_coordinate\_tracking
    3. Component list: (hip\_flexion\_r knee\_angle\_r ankle\_angle\_r)
    4. Max allowable error: 0.0873
11. Add a new cost term:
    1. Name: Load tracking
    2. Cost term type: inverse\_dynamics\_load\_tracking
    3. Component list: (hip\_flexion\_r\_moment knee\_angle\_r\_moment ankle\_angle\_r\_moment)
    4. Max allowable error: 50
12. Add a new constraint term:
    1. Name: Kinetic consistency
    2. Constraint term type: kinetic\_consistency
    3. Component list: (hip\_flexion\_r\_moment knee\_angle\_r\_moment ankle\_angle\_r\_moment)
    4. Max error: 0.1
    5. Min error: -0.1
13. Save this settings file as “TorqueTOSettings.xml”
14. Open the settings file in a text editor of your choice and explore it.
15. Change the trial\_name field to be “drive\_kick1”

**Running Tracking Optimization:**

1. Open MATLAB and open “runTO.m” in your tutorial directory.
2. Open the project file (Project.prj inside your installation of nmsm-core.)
3. Run the MATLAB script.

**Post TO Analysis:**

1. The script will create 4 plots for you:
   1. Joint Angles: Joint angles for all model coordinates (including prescribed) as output by the TO run.
   2. Joint Velocities: Joint velocities for all states coordinates.
   3. Joint Loads: Joint loads for all model coordinates.
   4. Torque Controls: Torque controls for all coordinates actuated by a torque controller.
2. These plots are a valuable way to analyze the results of the TO run. RMSE values between the tracked data and the TO results are reported for every plot where applicable.
3. It is also valuable to visualize the motion in the OpenSim GUI
   1. With the model selected in OpenSim, load the newly created IK motion in your TO results directory. Ensure the motion looks as close to the experimental motion as possible.
4. Getting a good TO run is very hard and often requires additional iteration after the first attempt. It is recommended to add/remove cost terms if you believe the problem would benefit, or change max allowable errors for cost terms to “nudge” the solution in a desired direction.

**Section 2: Verification Optimization**

The Verification Optimization tool is used to perform a “dry run” Design Optimization without including any “design elements”, the goal being to ensure a good initial guess and to verify the appropriateness of the optimal control problem formulation. The tool accepts the same inputs as the TO tool, but in general, the results directory of a TO run is used for both the initial guess and tracked quantities.

**Creating a Verification Optimization settings file:**

1. Activate the NMSM GUI in OpenSim by navigating to “Tools>User Plugins”, and click “rcnlPlugin.dll”
2. With “KickingModel.osim” selected in the OpenSim GUI, “Tools>Treatment Optimization >Verification Optimization”
   1. A window should open that looks very similar to the Tracking Optimization window shown in section 1.
3. Leave the Osimx file field empty.
4. Set the initial guess directory to be “TorqueTOResults”
5. Set the tracked quantities directory to be “TorqueTOResults”
6. Set the results directory to be TorqueVOResults
7. Set the optimal control solver settings file to be GpopsSettings.xml (The same file created in section 1).
8. Under states coordinate list, select (hip\_flexion\_r, knee\_angle\_r, ankle\_angle\_r)
9. Click to the RCNL Controllers tab at the top.
10. Under RCNL Torque Controller, add (hip\_flexion\_r, knee\_angle\_r, ankle\_angle\_r) to the coordinate list.
11. Click to the Cost/Constraints tab at the top.
12. Add a new cost term:
    1. Name: Controller tracking
    2. Cost term type: controller \_tracking
    3. Component list: (hip\_flexion\_r knee\_angle\_r ankle\_angle\_r)
    4. Max allowable error: 50
13. Add a new constraint term:
    1. Name: Kinetic consistency
    2. Constraint term type: kinetic\_consistency
    3. Component list: (hip\_flexion\_r\_moment knee\_angle\_r\_moment ankle\_angle\_r\_moment)
    4. Max error: 0.1
    5. Min error: -0.1
14. Save this settings file as “TorqueVOSettings.xml”
15. Open the settings file in a text editor of your choice and explore it. How does this settings file compare to your TO settings file?
16. Change the trial\_name field to be “drive\_kick1”

**Running Verification Optimization:**

1. Open MATLAB and open “runVO.m” in your tutorial directory.
2. Open the project file (Project.prj inside your installation of nmsm-core.)
3. Run the MATLAB script.

**Post VO Analysis:**

1. The script will create the same 4 plots as the TO run.
2. Ideally, the VO solution should be nearly identical to the TO solution. If there are large differences, there is likely a problem with your VO problem formulation.

**Section 3: Design Optimization**

The Design Optimization tool predicts or optimizes how a planned treatment will affect a patient’s post-treatment movement function. The tool accepts the same inputs as the other Treatment Optimization tools, but typically VO results are used as the initial guess. Similar to VO, DO allows for controller tracking and generalized coordinate tracking alongside other DO-specific cost function terms.

Unlike the other Treatment Optimization tools, the DO tool allows for both fixed and free final time problem formulations. For free final time problems, the time vector can be constrained to a user defined range and tracked control quantities are stretched or compressed in time to match the current time vector. In addition to all cost function and constraint terms available in the TO and VO tools, the DO tool includes a number of additional built-in cost function terms.

Three key features of the DO tool are support for model modification functions, user-defined cost function terms, and user-defined constraint terms. By employing model modification functions alongside GPOPS-II’s static parameters feature, users can change the OpenSim musculoskeletal model, modify the neural control model, or adjust parameter values in an assistive device.

**Creating a Design Optimization settings file:**

1. Activate the NMSM GUI in OpenSim by navigating to “Tools>User Plugins”, and click “rcnlPlugin.dll”
2. With “KickingModel.osim” selected in the OpenSim GUI, “Tools>Treatment Optimization >Verification Optimization”
   1. A window should open that looks very similar to the Tracking Optimization window shown in section 1.
3. Leave the Osimx file field empty.
4. Set the initial guess directory to be “TorqueVOResults”
5. Set the tracked quantities directory to be “TorqueVOResults”
6. Set the results directory to be TorqueDOResultsBase
7. Set the optimal control solver settings file to be GpopsSettings.xml (The same file created in section 1).
8. Under states coordinate list, select (hip\_flexion\_r, knee\_angle\_r, ankle\_angle\_r)
9. Click to the RCNL Controllers tab at the top.
10. Under RCNL Torque Controller, add (hip\_flexion\_r, knee\_angle\_r, ankle\_angle\_r) to the coordinate list.
11. Add a new cost term:
    1. Name: Controller tracking
    2. Cost term type: controller \_tracking
    3. Component list: (hip\_flexion\_r knee\_angle\_r ankle\_angle\_r)
    4. Max allowable error: 50
12. Add a new constraint term:
    1. Name: Kinetic consistency
    2. Constraint term type: kinetic\_consistency
    3. Component list: (hip\_flexion\_r\_moment knee\_angle\_r\_moment ankle\_angle\_r\_moment)
    4. Max error: 0.1
    5. Min error: -0.1
13. Save this settings file as “TorqueDOSettingsBase.xml”
    1. This will be the base settings file that we iterate out of later.
14. Open the settings file in a text editor of your choice and explore it. How does this settings file compare to your TO and VO settings files?
15. Change the trial\_name field to be “drive\_kick1”

**Creating user defined cost terms:**

1. At this point, we want to try to implement a cost term that will act as the “design” in “Design Optimization”. Our goal with this kicking motion can be to increase the power of the kick. That is, we want the final velocity of the foot to be higher.
2. Increasing the final foot velocity is a great use of **user defined cost terms**.
3. A user defined cost term is premade in the tutorial directory named footSpeedCost.m
   1. This cost term minimizes deviations away from a desired final velocity.
4. To include this cost term in the settings file, copy the following lines into your RCNLCostTermSet:

|  |
| --- |
| <RCNLCostTerm>  <type>user\_defined</type>  <function\_name>footSpeedCost</function\_name>  <cost\_term\_type>discrete</cost\_term\_type>  <is\_enabled>true</is\_enabled>  <marker\_name>R\_Midfoot\_Superior</marker\_name>  <final\_speed>20</final\_speed>  </RCNLCostTerm> |

**Running Design Optimization:**

1. Open MATLAB and open “runDO.m” in your tutorial directory.
2. Open the project file (Project.prj inside your installation of nmsm-core.)
3. Run the MATLAB section labeled Run DO V1.

**Iterating on DO solutions:**