introduction

this script may be used to graph signals in a report template. instructions to use this script may be found here.

sample script

```
! ========================================================
! set parameters
! ========================================================

prompt_for_multiple_pipeline_parameter_values
/pipeline_parameter_name=page_count+column+row+event_1+event_2
 +signal_type+signal_folder+color_r+color_l
 +graph_trial_style+graph_mean+graph_stddev
/datatype=s+s+s+s+s+s+s+s+s+s+s+s+s
/default_value=1+1+1+hs+hs+link_model_based+original+990000+109900+null+true
+true
! paremeter list explained:
! page_count
!  starting page
! column
!  starting column
! row
!  starting row
! event
!  if graphing kinematics use "hs", if graphing kinetics use "on"
!  there is a loop later in the script (::side) which will loop through l &
r
! the colors are specified in hex however only numbers (no letters) can be
specified in the expression
!  to pick colors: http://www.w3schools.com/tags/ref_colorpicker.asp
!  the first value in the color must be 1 or greater (it cannot be 0)
!  color information
!  990000 = red
!  109900 = green
!   100099 = blue/purple
!  if not using the "set pipeline parameter from expression" command to set
!   the line color, any value can be typed in the "line_color" parameter of
the
!   "make line graph" command
;
```

```
 set_pipeline_parameter
! list of components that will be graphed
/parameter_name=component_list
```

```
/parameter_value=x+y+z
;

! ========================================================
! indicate the signals being graphed using prompt
!  (1) kinematic (2) kinetic or (3) all
! ========================================================

set_pipeline_parameter
! list of signals that will be graphed
!  the side parameter will specify whether the signal is l or r
/parameter_name=kinematic_segments
/parameter_value=ankleangle+kneeangle+hipangle
;

set_pipeline_parameter
! list of signals that will be graphed
!  the side parameter will specify whether the signal is l or r
/parameter_name=kinetic_segments
/parameter_value=anklemoment + kneemoment + hipmoment + anklepower +
kneepower + hippower
;

set_pipeline_parameter
! list of signals that will be graphed
!  the side parameter will specify whether the signal is l or r
/parameter_name=all_segments
/parameter_value=::kinematic_segments& + &::kinetic_segments&
;

prompt_for_pipeline_parameter_value
/pipeline_parameter_name=segment_type
/prompt=select data type:
/data_type=string+string+string
/default_value=kinematic_segments + kinetic_segments + all_segments
;

set_pipeline_parameter
! list of signals that will be graphed
!  the side parameter will specify whether the signal is l or r
/parameter_name=data_name_list
/parameter_value=&:&:&::segment_type
/multi_pass=true
;

! ========================================================
! set counters
! ========================================================

set_pipeline_parameter
```

```
! set the original column number (this will be incremented at the end of
each side loop)
! at the end of each side loop, the column will be set back to this value
/parameter_name=column_initial
/parameter_value=::column
;

set_pipeline_parameter
! set the original row number (this will be incremented at the end of each
component loop)
! at the end of each component loop, the row will be set back to this value
/parameter_name=row_initial
/parameter_value=::row
;

! ========================================================
! begin graphing
! ========================================================

for_each
! a new page will be created for each signal in the data_name_list
/iteration_parameter_name=data_name
/items=::data_name_list
;

set_report_page_title
! the title of the current page will be set to the current signal name
/page_number=::page_count
/page_title=::data_name
;

for_each
! for each side (left/right) a column of graphs - the column counter will be
incremented at the end of this loop
/iteration_parameter_name=side
/items=l+r
;

set_pipeline_parameter_from_expression
! this command is comparing the current side (::side) to determine the color
of the graph
!  if the items in the side list are changed the comparisons will need to be
changed as well ("r" and "l")
! the colors are specified in hex however only numbers (no letters) can be
specified in the expression
!  to pick colors: http://www.w3schools.com/tags/ref_colorpicker.asp
/parameter_name=color
/expression=( "&::side&" == "r" ) * &::color_r& + ( "&::side&" == "l" ) *
&::color_l&
/as_integer=true
;
```

```
for_each
! each component (x,y,z) will be added to a new graph since the row number
will be incremented at the end of each loop
/iteration_parameter_name=component
/items=::component_list
;

make_line_graph
/page_number=::page_count
/column_number=::column
/row_number=::row
! /column_span=1
! /row_span=1
/graph_title=::side&::data_name& - &::component
! /files_to_graph=all_files
! /display_legend=false
! /legend_position=ur
! /legend_text=
! /display_legend_tag=false
! /display_legend_folder=false
! /display_legend_signal=false
/line_style=::graph_trial_style
/line_color=#&::color
! /line_bold=false
! /show_points=false
/graph_mean=::graph_mean
/graph_stddev=::graph_stddev
/mean_stddev_line_style=solid
/mean_stddev_line_color=#&::color
! /mean_stddev_line_bold=false
/use_event_range=true
/start_event_label=::side&::event_1
/end_event_label=::side&::event_2
! /intermediate_events=
! /exclude_events=
/x_axis_label=% gait cycle
/x_data_type=frame_numbers
/x_data_name=frames
/x_data_component=0
/x_data_processed=original
! /x_axis_scale=normalize 0% to 100%
! /x_axis_min=
! /x_axis_max=
! /y_axis_label=
/y_data_type=::signal_type
/y_data_name=::side&::data_name
/y_data_component=::component
/y_data_processed=::signal_folder
! /y_axis_scale=global min to max
! /y_axis_min=
```

```
! /y_axis_max=
! /y_use_scientific_notation=false
! /use_p2d_stddev=false
! /stddev_component=3
! /use_p2d_bargraph=false
! /p2d_bargraph_position=
! /p2d_bargraph_width=
;

set_pipeline_parameter_from_expression
! at the end of each component loop, a new row is created
! allows each component to have it's own row
/parameter_name=row
/expression=&::row& + 1
! /as_integer=true
;

end_for_each
! end component loop
/iteration_parameter_name=component
;

set_pipeline_parameter
! at the end of each side loop, row counter is set back to original value
! allows each side to start with a graph at the original row specified
/parameter_name=row
/parameter_value=::row_initial
;

set_pipeline_parameter_from_expression
! at the end of each side loop, a new column is created
! allows each side to have it's own column
/parameter_name=column
/expression=&::column& + 1
! /as_integer=true
;

end_for_each
! end side loop
/iteration_parameter_name=side
;

set_pipeline_parameter
! at the end of each data_name loop, column counter is set back to original
value
! allows each page to start with a graph at the original column specified
/parameter_name=column
/parameter_value=::column_initial
;

set_pipeline_parameter_from_expression
```

```
! at the end of each data_name loop, a new page is created
! allows each signal to have it's own page
/parameter_name=page_count
/expression=&::page_count& + 1
! /as_integer=true
;

end_for_each
! end data_name loop
/iteration_parameter_name=data_name
;
```