```
function [ ] = setupRun(RunName, loadControlName, analysisName, modelName, axialLoad, curvatures, stepScale)
% Creates the TCL files parametrically for the moment curvature analysis
arguments
    RunName = 'run.tcl';
    loadControlName = 'loadControlStaticAnalysis.tcl';
    analysisName = 'dispControlAnalysis.tcl'
    modelName = 'model01.tcl';
    axialLoad = 0.0;
    curvatures = 0.03;
     stepScale = 0.0001;
end

    FID = fopen(RunName, 'w');
    fprintf(FID,[...
        'set analysisResultsDirectory "AnalysisResults";\n',...
        'file mkdir $analysisResultsDirectory;\n',...
        'set modelDirectory "Model";\n',...
        'file mkdir $modelDirectory;\n',...
        'set modelExportFileID [open "$modelDirectory/modelData.txt" "w"];\n',...
        'source units.tcl;\n',...
        'source ' modelName ';\n',...
        'source ' loadControlName ';\n',...
        'source ' analysisName ';\n',...
        'remove recorders\n',...
        'wipe;\n']);
    fclose(FID);

    FID = fopen(loadControlName, 'w');
        fprintf(FID,...
        ['set axialLoadTag 1;\n',...
        'set axialLoadRatio ' num2str(axialLoad,'%1.2f') ';\n',...
        'set P [expr abs($fpc)*$colArea*$axialLoadRatio];\n',...
        'pattern Plain $axialLoadTag "Linear" { load $controlNode -$P 0.0 0.0;};\n',...
        'set numAnalysisSteps 1 ;\n',...
        'integrator LoadControl [expr 1./$numAnalysisSteps];\n',...
        'system BandGeneral ;\n',...
        'test NormUnbalance 1e-6 100;\n',...
        'numberer Plain ;\n',...
        'constraints Plain ;\n',...
        'algorithm KrylovNewton ;\n',...
        'analysis Static;\n',...
        'set ok [analyze $numAnalysisSteps];\n',...
        'if {$ok == 0} {puts "Axial load applied and analyzed"};\n',...
        'loadConst -time 0.0']);
        fclose(FID);


        appendage = [];
        for k = curvatures
            appendage = [appendage, 'lappend peakDisp [expr ', num2str(k,"%1.5f"), '/$colDepth]' , newline];
        end

        FID = fopen(analysisName, 'w');
        fprintf(FID,[...
        'set controlDOF 3',newline,...
        'set dispControlLoadTag 2 ', newline,...
        '# Define reference moment', newline,...
```

```
58        'pattern Plain $dispControlLoadTag "Linear" {load $controlNode 0.0 0.0 1.0}', newline,...
59        'set peakDisp {};', newline,...
60        appendage,...
61        'set numCycles [llength $peakDisp]', newline,...
62        'set cyclelabel {};', newline,...
63        'for {set i 1} {$i <= $numCycles} {incr i 1} {', newline,...
64        '    lappend cyclelabel $i', newline...
65        '}', newline,...
66        'set maxDisp [expr 0.03/$colDepth];', newline,...
67        'set du [expr ' num2str(stepScale) '*$maxDisp];', newline,...
68        'set ok 0; ', newline,...
69        'set currentDisp 0.; # This is the current value of the displacement at the control DOF.', newline,...
70        'set tol 1e-6;', newline,...
71        'set iter 250', newline,...
72        'recorder Node -file $analysisResultsDirectory/MK.txt -time -node $controlNode -dof 1 $controlDOF disp', newline,...
73        'recorder Element -file $analysisResultsDirectory/ConcFib1_SS.txt  -time -ele 1 section fiber -$y1  0. $matTagConcCover stressStrain', ↙
newline,...
74        'recorder Element -file $analysisResultsDirectory/ConcFib2_SS.txt  -time -ele 1 section fiber  $y1  0. $matTagConcCover stressStrain', ↙
newline,...
75        'recorder Element -file $analysisResultsDirectory/SteelFib1_SS.txt -time -ele 1 section fiber -$y1  0. $matTagSteel     stressStrain',↙
newline,...
76        'recorder Element -file $analysisResultsDirectory/SteelFib2_SS.txt -time -ele 1 section fiber  $y1  0. $matTagSteel     stressStrain',↙
newline,...
77        'record; # This is to record the state before the analysis starts', newline,...
78        'for {set ii 1} {$ii<=[llength $peakDisp]} {incr ii} {', newline,...
79           '# Convergence check', newline,...
80           'if {$ok == 0} { ', newline,...
81              'set cycleDisp [expr [lindex $peakDisp [expr $ii-1]] - $currentDisp]; # the total deformation of the loading cycle', newline,...
82              '# determine the sign of loading:', newline,...
83              'if {$cycleDisp>0} {', newline,...
84              '   set sign 1.;', newline,...
85              '} else {', newline,...
86              '   set sign -1.;', newline,...
87              '};', newline,...
88              'set dU [expr $du*$sign];', newline,...
89              '# General analysis properties', newline,...
90              'constraints Transformation;        ', newline,...
91              'numberer Plain;            ', newline,...
92              'system BandGeneral;', newline,...
93              'integrator DisplacementControl $controlNode $controlDOF $dU;', newline,...
94              'test RelativeNormDispIncr $tol $iter;', newline,...
95              'algorithm KrylovNewton;', newline,...
96              'analysis Static;', newline,...
97              'set NSteps [expr int(abs($cycleDisp/$dU))];', newline,...
98              'puts "";', newline,...
99              'puts "Starting Cycle # [lindex $cyclelabel [expr $ii-1]] with target displacement of [expr [lindex $peakDisp [expr $ii-1]]]"', newline,...
100             'puts "======================================================"', newline,...
101             'puts "---> Running $NSteps steps with step size = $dU in. to go from displ. = $currentDisp to displ. = [expr [lindex $peakDisp ↙
[expr $ii-1]]]"', newline,...
102             'set ok1 [analyze $NSteps];', newline,...
103             'set currentDisp [nodeDisp $controlNode $controlDOF];', newline,...
104             '#If it does not converge, change strategies', newline,...
105             'if {$ok1 !=0 } {', newline,...
106                'set ok 0;', newline,...
107                'puts "     Try stuff, peak disp = [expr [lindex $peakDisp [expr $ii-1]]]";', newline,...
108                'puts "     Current disp = $currentDisp";', newline,...
109                'puts "     Cycle disp  = $cycleDisp";', newline,...
```

```
110             'set counter 1;', newline,...
111             'while {( ( ([expr $currentDisp] <= [expr [lindex $peakDisp [expr $ii-1]]]) && ($sign == 1)  ) || (  ([expr $currentDisp] >= [expr
[lindex $peakDisp [expr $ii-1]]]) && ($sign == -1)   ) )&&($ok==0)} {', newline,...
112             '   set ok 1;', newline,...
113             '   while {$ok!=0} {', newline,...
114             '      if {$counter == 0} {', newline,...
115             '         # return to initial conditions', newline,...
116             '         set dU [expr $du*$sign*1.00];', newline,...
117             '         test NormDispIncr $tol $iter 0;', newline,...
118             '         set counter 1;', newline,...
119             '      } elseif {$counter == 1} {', newline,...
120             '         # increase load stepsize', newline,...
121             '         set dU [expr $du*$sign*1.5];', newline,...
122             '         #puts "dU = $du*$sign*1.5 = $dU";', newline,...
123             '         set counter 2;', newline,...
124             '      } elseif {$counter == 2} {', newline,...
125             '         # increase load stepsize', newline,...
126             '         set dU [expr $du*$sign*2.00];', newline,...
127             '         #puts "dU = $du*$sign*2.0 = $dU";', newline,...
128             '         set counter 3;', newline,...
129             '      } elseif {$counter == 3} {', newline,...
130             '         # decrease load stepsize', newline,...
131             '         set dU [expr $du*$sign*0.5];', newline,...
132             '         #puts "dU = $du*$sign*0.5 = $dU";', newline,...
133             '         set counter 4;', newline,...
134             '      } elseif {$counter == 4} {', newline,...
135             '         # decrease load stepsize', newline,...
136             '         set dU [expr $du*$sign*0.1];', newline,...
137             '         #puts "dU = $du*$sign*0.1 = $dU";', newline,...
138             '         set counter 5;', newline,...
139             '      } elseif {$counter == 5} {', newline,...
140             '         # decrease load stepsize', newline,...
141             '         set dU [expr $du*$sign*0.05];', newline,...
142             '         #puts "dU = $du*$sign*0.05 = $dU";', newline,...
143             '         set counter 6;', newline,...
144             '      } elseif {$counter == 6} {', newline,...
145             '         # decrease load stepsize', newline,...
146             '         set dU [expr $du*$sign*0.01];', newline,...
147             '         #puts "dU = $du*$sign*0.01 = $dU";', newline,...
148             '         set counter 7;', newline,...
149             '      } elseif {$counter == 7} {', newline,...
150             '         # decrease load stepsize', newline,...
151             '         set dU [expr $du*$sign*0.001];', newline,...
152             '         #puts "dU = $du*$sign*0.001 = $dU";', newline,...
153             '         set counter 8;', newline,...
154             '      };', newline,...
155             '      integrator DisplacementControl $controlNode  $controlDOF  $dU;', newline,...
156             '      set ok [analyze 1]', newline,...
157             '      if {$ok != 0} {', newline,...
158             '         puts "Try Newton Initial"', newline,...
159             '         algorithm Newton -initial', newline,...
160             '         test NormDispIncr $tol $iter 0;', newline,...
161             '         set ok [analyze 1];', newline,...
162             '      };', newline,...
163             '      if {$ok != 0} {', newline,...
164             '         puts "Test Relative Displacement"', newline,...
165             '         test RelativeNormDispIncr $tol $iter 0;', newline,...
```

```
166             '          set ok [analyze 1];', newline,...
167             '      };', newline,...
168             '      if {$ok != 0} {', newline,...
169             '          puts "ModifiedNewton"', newline,...
170             '          algorithm ModifiedNewton', newline,...
171             '          set ok [analyze 1];', newline,...
172             '      };', newline,...
173             '      if {$ok != 0} {', newline,...
174             '          puts "Test Relative Energy"', newline,...
175             '          algorithm Newton -initial;', newline,...
176             '          test RelativeEnergyIncr $tol $iter 0;', newline,...
177             '          set ok [analyze 1];', newline,...
178             '      };', newline,...
179             '      if {$ok != 0} {', newline,...
180             '          puts "Newton Modified -initial"', newline,...
181             '          algorithm ModifiedNewton -initial', newline,...
182             '          set ok [analyze 1];', newline,...
183             '      };', newline,...
184             '      if {$ok != 0} {', newline,...
185             '          puts "Test Relative Force"', newline,...
186             '          algorithm Newton -initial;', newline,...
187             '          test RelativeNormUnbalance $tol $iter 0', newline,...
188             '          set ok [analyze 1];', newline,...
189             '      };', newline,...
190             '      if {$ok != 0} {', newline,...
191             '          puts "Test Relative Displ"', newline,...
192             '          test RelativeNormDisplncr $tol $iter 0', newline,...
193             '          set ok [analyze 1];', newline,...
194             '      };', newline,...
195             '      if {$ok != 0} {', newline,...
196             '          puts "Broyden"', newline,...
197             '          algorithm Broyden 8', newline,...
198             '          set ok [analyze 1];', newline,...
199             '      };', newline,...
200             '      if {$ok != 0} {', newline,...
201             '          puts "Newton Line Search"', newline,...
202             '          algorithm NewtonLineSearch .8', newline,...
203             '          set ok [analyze 1];', newline,...
204             '      };', newline,...
205             '      if {$ok != 0} {', newline,...
206             '          puts "BFGS"', newline,...
207             '          algorithm BFGS', newline,...
208             '          set ok [analyze 1];', newline,...
209             '      };', newline,...
210             '  };', newline,...
211             '  set counter 0;', newline,...
212             '  set currentDisp [nodeDisp $controlNode $controlDOF];', newline,...
213             '   #puts $currentDisp', newline,...
214             '};', newline,...
215             'puts "Cycle # [lindex $cyclelabel [expr $ii-1]] successfully finished!"', newline,...
216             'puts "target displ.  = [expr [lindex $peakDisp [expr $ii-1]]]"', newline,...
217             'puts "current displ. = [nodeDisp $controlNode $controlDOF]"', newline,...
218             'puts "--------------------------------x----------------------------------------";', newline,...
219         '} else {', newline,...
220             'puts "Cycle # [lindex $cyclelabel [expr $ii-1]] successfully finished!"', newline,...
221             'puts "target displ.  = [expr [lindex $peakDisp [expr $ii-1]]]"', newline,...
222             'puts "current displ. = [nodeDisp $controlNode $controlDOF]"', newline,...
```

```
223                 'puts "--------------------------------x---------------------------------------";', newline,...
224             '};', newline,...
225         '};', newline,...
226     '};', newline,...
227     'if { $ok<0 } {', newline,...
228     '   puts "FAILED TO CONVERGE!!" ', newline,...
229     '} else {', newline,...
230     '   puts "";', newline,...
231     '   puts "ALL SUCCESSFUL!!"', newline,...
232     '   puts $modelnum', newline,...
233     '}', newline]);
234     fclose(FID);
235
236 end
237
238
```