Open-Source FISH: Principal Strains Calculation in a Shell-Type Structural Elements

By Roozbeh Geraili Mikola, PhD, PE

www.roozbehgm.com
www.geowizard.org

May 2023

FLAC3D does not provide any specific command for directly calculating and plotting principal strains in shell-type structural elements (Version 7 and below), but the majority of the information required for strain calculation is provided via the FISH programming language. This short tutorial teaches how to use FISH to determine the principal strains in shell-type structural elements. The FISH function itself is provided at the end. Although the author has not tested it, the offered open-source FISH may function in other 3D programs (such as 3DEC and PFC3D) as well.

How it works:

The following are the methodologies for calculating strain:

1. Go through all the shell elements and extract the structural node position and velocity in global coordination system.

```
; Get a pointer to the node at index 1,2,3 in the structural element
local snp1 = struct.node(sep,1)
local snp2 = struct.node(sep,2)
local snp3 = struct.node(sep,3)

; Get the current position vector for the structure node in global system
local glob_pos1 = struct.node.pos.reference(snp1)
local glob_pos2 = struct.node.pos.reference(snp2)
local glob_pos3 = struct.node.pos.reference(snp3)

; Get the velocity of the structure nodes expressed in global system
local gvu1 = struct.node.vel.global(snp1,1)
local gvv1 = struct.node.vel.global(snp1,2)
local gvw1 = struct.node.vel.global(snp1,3)
local gvu2 = struct.node.vel.global(snp2,1)
local gvv2 = struct.node.vel.global(snp2,2)
local gvw2 = struct.node.vel.global(snp2,3)
local gvu3 = struct.node.vel.global(snp3,1)
local gvv3 = struct.node.vel.global(snp3,2)
local gvw3 = struct.node.vel.global(snp3,3)

local glob_vel1 = (gvu1,gvv1,gvw1)
local glob_vel2 = (gvu2,gvv2,gvw2)
local glob_vel3 = (gvu3,gvv3,gvw3)
```

2. For each shell element, calculate the structural node position and velocity in local coordination system.

```
; Get the local coordinate system for the structural element
local loc_sys = struct.local.system(sep)

; Perform global to local transformation
```

```
    local loc_pos1 = loc_sys*glob_pos1
    local loc_pos2 = loc_sys*glob_pos2
    local loc_pos3 = loc_sys*glob_pos3

    local loc_vel1 = loc_sys*glob_vel1
    local loc_vel2 = loc_sys*glob_vel2
    local loc_vel3 = loc_sys*glob_vel3

    local loc_vel_x1 = loc_vel1(1)
    local loc_vel_y1 = loc_vel1(2)
    local loc_vel_x2 = loc_vel2(1)
    local loc_vel_y2 = loc_vel2(2)
    local loc_vel_x3 = loc_vel3(1)
    local loc_vel_y3 = loc_vel3(2)
```

3. Now the strains in the local coordination system can be calculated using triangular shell finite element (FE) formulation:

```
    ; Calculate strains in local coordination system using triangular shell FE formulation
    local x1 = loc_pos1(1)
    local x2 = loc_pos2(1)
    local x3 = loc_pos3(1)

    local y1 = loc_pos1(2)
    local y2 = loc_pos2(2)
    local y3 = loc_pos3(2)

    ; Get element's Area
    local sh_A = struct.shell.area(sep)

    ; Calculate incremental displacement in local coordination system
    local u1 = loc_vel_x1*dt
    local v1 = loc_vel_y1*dt
    local u2 = loc_vel_x2*dt
    local v2 = loc_vel_y2*dt
    local u3 = loc_vel_x3*dt
    local v3 = loc_vel_y3*dt

    ; Calculate incremental strains (Note: it's in local cood. System)
    local _desxx = (u1*(y2 - y3) - u2*(y1 - y3) + u3*(y1 - y2))/(2*sh_A)
    local _desyy = (v2*(x1 - x3) - v1*(x2 - x3) - v3*(x1 - x2))/(2*sh_A)
    local _desxy = (u2*(x1 - x3) - u1*(x2 - x3) - u3*(x1 - x2) + v1*(y2 - y3) - v2*(y1 - y3) + v3*(y1
- y2))/(2*sh_A)

    ; Accumulate total strains (Note: it's in local cood. System)
    local _esxx = struct.extra(sep,3) + _desxx
    local _esyy = struct.extra(sep,4) + _desyy
    local _esxy = struct.extra(sep,5) + _desxy

    ; Calculate principal strains (Note: it's in local cood. System)
    local prin_strain_max = 0.5*(_esxx+_esyy)+math.sqrt(((_esxx-_esyy)/2)^2+(0.5*_esxy)^2)
    local prin_strain_min = 0.5*(_esxx+_esyy)-math.sqrt(((_esxx-_esyy)/2)^2+(0.5*_esxy)^2)
```

4. Store the principal and total (in local system) strains in the extra array associated with the structural element.

```
    ; Store Data
    struct.extra(sep,1) = prin_strain_max ; Max principal strain
    struct.extra(sep,2) = prin_strain_min ; Min principal strain
    struct.extra(sep,3) = _esxx ; Strain xx at local coord. system
    struct.extra(sep,4) = _esyy ; Strain yy at local coord. system
    struct.extra(sep,5) = _esxy ; Strain xy at local coord. system
```

The steps 1-4 will be repeated at each cycle using "*while_stepping*" FISH command. The complete FISH function is presented in Appendix A.

How to use it:

The following example explains how to compute and store the principal strains in the shell elements using the FISH function from Appendix A. The FLAC3D model represents a 10m x 1m shell. A cross-diagonal mesh pattern and the DKT-CST Hybrid Shell Element are employed to generate symmetric response. Both the Young's modulus and the Poisson's ratio are set to 200 GPa and 0.25, respectively. The shell has a thickness of 1.0. Before the model is cycled 50000 times, the left end is completely fixed in the x-direction, and a velocity of 1E-6 m/s in the x-direction is imposed at the tip of the shell. The shell element's analytical strain should be 0.005 (50000 x 1.0E-6 / 10). Figure 1 shows the contour plot of the calculated principal strain in the shell elements using FISH function. The calculated principal strain in the middle of the shell is in good agreement with the analytical solution.

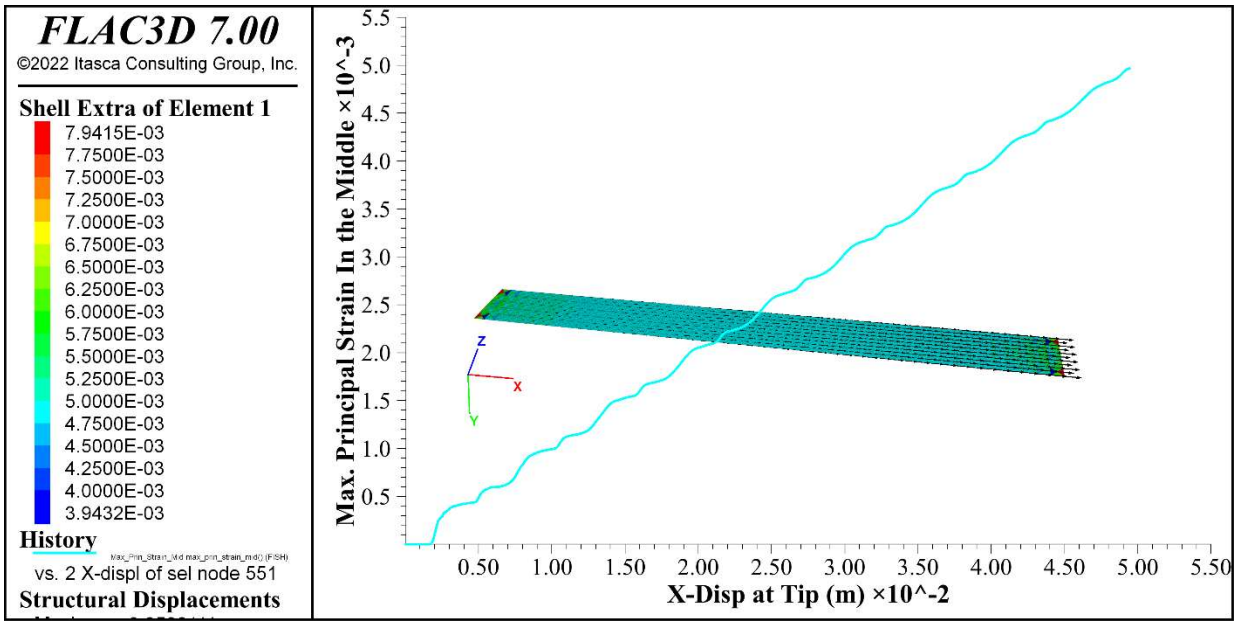

Figure 1: Final structural configuration.

Data File:

TestStoreShellStrains.dat

```
;-----------------------------------------------------------------------------------------------------------
; FLAC3D Data File
;-----------------------------------------------------------------------------------------------------------
; Name:         TestStoreShellStrains
; Author:       Roozbeh Geraili Mikola, PhD, PE
; Websites:     www.roozbehgm.com & www.geowizard.org
; Last Modified: 05-09-2023
; Description:  Simple model to test the 'store_shell_strains()' FISH function
; Notes:
;-----------------------------------------------------------------------------------------------------------

model new
model title ...
    "Calculate Principal Strains in Shell Structural Elements"
; Create shells and assign properties
```

```
struct shell create by-quad (0,0,0) (10,0,0) (10,0,1) (0,0,1) size (50,5) ...
                    cross-diagonal element-type=dkt-csth
struct shell property isotropic=(2e11,0.25) thick=1.0
; Boundary conditions
struct node fix velocity-x range position-x=-0.01 0.01
struct node initialize velocity=(1e-6;,0,0) range position-x=9.99 10.01
struct node fix velocity-x range position-x=9.99 10.01

program call 'StoreShellStrains.fis'

; Create FISH function to record max. principal strain in the middle of shell
define max_prin_strain_mid
    local sp = struct.near((5.0,0.0,0.5))
    max_prin_strain_mid = struct.extra(sp,1)
end

; Assign history point in the tip and middle of the shell
history interval 100
fish history name='Max_Prin_Strain_Mid' max_prin_strain_mid
structure node history name='Tip_X_Disp' displacement-x position (10.0,0.0,0.5)

; Solve model
model large-strain off
model cycle 50000
model save 'ShellStrain'
```

Appendix A:

StoreShellStrains.fis

```
;-----------------------------------------------------------------------------------------------------------
; FLAC3D FISH Function
;-----------------------------------------------------------------------------------------------------------
; Name:         store_shell_strains
; Author:       Roozbeh Geraili Mikola, PhD, PE
; Last Modified: 05-09-2023
; Description:  Open-Source FISH function to calculate the principal strains in a shell-type structural
;               element
; Notes:        This function does not require any input parameters. To use it simple call the '*.fis'
;               file using 'program call' command
;-----------------------------------------------------------------------------------------------------------

define store_shell_strains()
    while_stepping
    local dt = global.timestep
    local str_error = '"store_shell_strains" fish function only works with shell, liner and geogrid
structural elements'

    ; Loop through all the structural elements
    loop foreach sep struct.list

        local is_valid_struct = false

        if struct.type(sep) = 'shell' then
            is_valid_struct = true
        end_if

        if struct.type(sep) = 'liner' then
            is_valid_struct = true
        end_if

        if struct.type(sep) = 'geogrid' then
            is_valid_struct = true
        end_if

        if is_valid_struct = false
```

```
        system.error = str_error
    end_if

    ; Get a pointer to the node at index 1,2,3 in the structural element
    local snp1 = struct.node(sep,1)
    local snp2 = struct.node(sep,2)
    local snp3 = struct.node(sep,3)

    ; Get the current position vector for the structure node in global system
    local glob_pos1 = struct.node.pos.reference(snp1)
    local glob_pos2 = struct.node.pos.reference(snp2)
    local glob_pos3 = struct.node.pos.reference(snp3)

    ; Get the velocity of a structure node expressed in global system
    local gvu1 = struct.node.vel.global(snp1,1)
    local gvv1 = struct.node.vel.global(snp1,2)
    local gvw1 = struct.node.vel.global(snp1,3)
    local gvu2 = struct.node.vel.global(snp2,1)
    local gvv2 = struct.node.vel.global(snp2,2)
    local gvw2 = struct.node.vel.global(snp2,3)
    local gvu3 = struct.node.vel.global(snp3,1)
    local gvv3 = struct.node.vel.global(snp3,2)
    local gvw3 = struct.node.vel.global(snp3,3)

    local glob_vel1 = (gvu1,gvv1,gvw1)
    local glob_vel2 = (gvu2,gvv2,gvw2)
    local glob_vel3 = (gvu3,gvv3,gvw3)

    ; Get the local coordinate system for the structural element
    local loc_sys = struct.local.system(sep)

    ; Perform global to local transformation
    local loc_pos1 = loc_sys*glob_pos1
    local loc_pos2 = loc_sys*glob_pos2
    local loc_pos3 = loc_sys*glob_pos3

    local loc_vel1 = loc_sys*glob_vel1
    local loc_vel2 = loc_sys*glob_vel2
    local loc_vel3 = loc_sys*glob_vel3

    local loc_vel_x1 = loc_vel1(1)
    local loc_vel_y1 = loc_vel1(2)
    local loc_vel_x2 = loc_vel2(1)
    local loc_vel_y2 = loc_vel2(2)
    local loc_vel_x3 = loc_vel3(1)
    local loc_vel_y3 = loc_vel3(2)

    ; Calculate strains in local coordination system using triangular shell FE formulation
    local x1 = loc_pos1(1)
    local x2 = loc_pos2(1)
    local x3 = loc_pos3(1)

    local y1 = loc_pos1(2)
    local y2 = loc_pos2(2)
    local y3 = loc_pos3(2)

    ; Get element's Area
    local sh_A = struct.shell.area(sep)

    ; Calculate incremental displacement in local coordination system
    local u1 = loc_vel_x1*dt
    local v1 = loc_vel_y1*dt
    local u2 = loc_vel_x2*dt
    local v2 = loc_vel_y2*dt
    local u3 = loc_vel_x3*dt
    local v3 = loc_vel_y3*dt
```

```
        ; Calculate incremental strains (Note: it's in local cood. System)
        local _desxx = (u1*(y2 - y3) - u2*(y1 - y3) + u3*(y1 - y2))/(2*sh_A)
        local _desyy = (v2*(x1 - x3) - v1*(x2 - x3) - v3*(x1 - x2))/(2*sh_A)
        local _desxy = (u2*(x1 - x3) - u1*(x2 - x3) - u3*(x1 - x2) + v1*(y2 - y3) - v2*(y1 - y3) + v3*(y1
- y2))/(2*sh_A)

        ; Accumulate total strains (Note: it's in local cood. System)
        local _esxx = struct.extra(sep,3) + _desxx
        local _esyy = struct.extra(sep,4) + _desyy
        local _esxy = struct.extra(sep,5) + _desxy

        ; Calculate principal strains (Note: it's in local cood. System)
        local prin_strain_max = 0.5*(_esxx+_esyy)+math.sqrt(((_esxx-_esyy)/2)^2+(0.5*_esxy)^2)
        local prin_strain_min = 0.5*(_esxx+_esyy)-math.sqrt(((_esxx-_esyy)/2)^2+(0.5*_esxy)^2)

        ; Store data
        struct.extra(sep,1) = prin_strain_max ; Max principal strain in local coord. system
        struct.extra(sep,2) = prin_strain_min ; Min principal strain in local coord. system
        struct.extra(sep,3) = _esxx ; Strain xx in local coord. system
        struct.extra(sep,4) = _esyy ; Strain yy in local coord. system
        struct.extra(sep,5) = _esxy ; Strain xy in local coord. system

    endloop
end
```