Sandia National Laboratories

Operated for the U.S. Department of Energy by
Sandia Corporation

Albuquerque, New Mexico 87185- 0826

date: February 21, 2000

to: Distribution

from: P. R. Schunk, MS0826

subject: 3D Roll coating template and tutorial for GOMA (GT-012.1)

keywords: three dimensional, 3D, roll coating, forward roll coating, MAPVAR, exosweep, volume constraint, multiparameter continuation, transient, augmenting conditions, gen3d, Algebra, rotation conditions, troubleshooting

input records: GD_LINEAR, GD_PARAB, SURFTANG, KINEMATIC, VELO_TANGENT, VELO_NORMAL, AC=VC, FLOW_PRESSURE, Continuation=hzero, HC=BC, ROT, VELO_TANGENT_3D, SURFTANG_EDGE, R_MESH_NORMAL, SEED, BASIS_RESEED

Introduction

This tutorial assumes the user has gone through the beginner's training tutorial on GOMA and SEAMS and the follow-on roll coating tutorial (GT-003.1). Specifically, it assumes the user has a strong familiarity with GOMA itself and the tools **`fastq`**, **`ex1ex2v2`**, **`aprepro`** and **`blot`**. In addition, this work requires working knowledge of **`mapvar`** and **`gen3d`**. Documents GT-001.4 and GT-006.3 provide some introduction and tutorial learning directly related to these tools. Finally, Document GT-013.1 provides some companion information on 3D free surface flow analysis with GOMA, but in the context of slot coating with a dynamic wetting line.

We will pursue the 3D roll coating model in several stages, including: (1) obtaining a base-case solution to a 2D forward roll film split problem with a flooded upstream nip (on rigid rolls), (2) creation of a 3D base solution from the 2D solution using **`exosweep`**, a tool for extruding/rotating EXODUS II databases into 3D, from 2D, and (3) obtaining a full 3D solution of a center section of a flooded forward roll coater with film split. With the full 3D solution we then discuss how such a solution can be used to study the ribbing instability via transient analysis.

Advanced Techniques/Approaches Exemplified in this tutorial are:

1.) Multiparameter continuation using hunting (see SAND2000-2465 on Advanced Capabilities usage)

2.) Volume constraint augmenting condition (see SAND2000-2465 on Advanced Capabilities usage)

3.) Solution mapping via **mapvar**. (cf. GT-006.3 on CUBIT/MAPVAR)

4.) Transient analysis.

The name of the directory to which this portion of the tutorial refers is **3D_film_split**. Contact Duane Labreche (dalabre@sandia.gov) or Randy Schunk (prschun@sandia.gov) for a copy.

3D Film Split Roll Coating Model

In the "roll" subdirectory (viz. **$(INSTALL_DIR)/Distribution/example_problems/roll**) there is a **3D_film_split** subdirectory. In there you will find several input decks and files, some of which are listed here

0.18-0.3_trans.exoII	0.28_base.exoII	restart.exoII
0.28_base_with_bump.exoII	1.gen	restart_2of3.exoII
2D.exoII	3d.exo	restart_base.exoII
3d.exoII	3d.gen	3d_rib.exo
roll.exoII	BCdup.txt	roll.fas
Gomau.mk	roll.fas.orig	roll.fas.save
roll.gen	roll_input	README
roll_input.save	README.backup	roll_input_2D
roll_input_2D.1	roll_input_2D.2	roll_input_2D.3
roll_input_2D.hunt	roll_input_3D	roll_input_3D.fixed
roll_input_3D_trans	soln.dat	
brk_inp	soln.dat_trans	
contin.dat	contin.dat.fixed	tmp.e
goma	tmp.exo	
liquid.mat	tmp.g	tmp.int
out.exoII	tmp.o	
out.exoII.2D		

Of these files, we will discuss

- README
- roll_input_2D*
- roll_input_3D
- roll_input_3D_trans
- brk_inp
- roll.fas

The README file actually encapsulates the steps taken in this memo, to guide you through the steps of obtaining a solution to the 3D roll coating flow, center section.

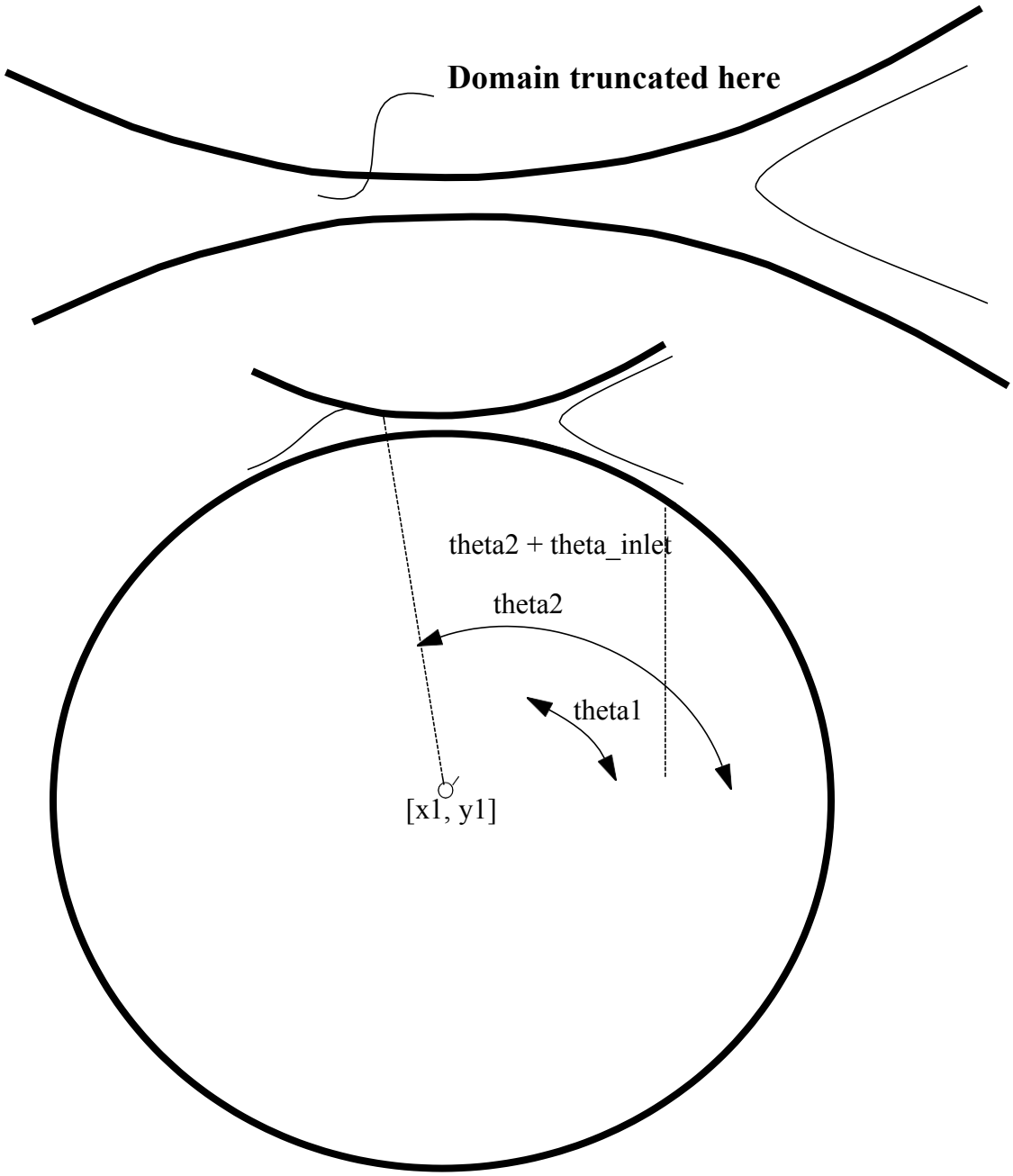
The first input file we will discuss is **roll.fas**.

The top of this file looks like this:

```
$ GEOMETRY AND OPERATING CONDITIONS (MKS)

$   Note here roll speed for the rolls in is radians/s because
$   of the type of bc that we use.

$ gap (leading edge to substrate)       {Gap = 0.001}    {Gap_new = 0.001}
$ roll cover thickness (1" = 0.0254)  {d = 0.0254/2.}
$$$$Set both roll speeds to 0.18 for original case
$ bottom roll speed       {rollsp_b = 0.18}
$ top roll speed{rollsp_t = 0.18}
$ initial film thickness for v2/v1=1    {h_t = .002}

$ Roll Radius                          {R_I = 7.*0.0254} {R_I_new = 7.0*0.0254}
$ Outer Roll Radius                     {R_O = R_I + d}   {R_O_new = R_I_new+d}
$ Outer Roll Radius + film thickness   {R_O_h = R_O + h_t} {R_O_h_new = R_O_new
+ h_t}
$ Outer Roll Radius + 2.5*film thickness  {R_O_2h = R_O + 2.5*h_t}

$ Roll extent   {theta1 = 60}   {theta2 = 100}

$
$  MESH PARAMETERS
$
${no_elem_along_roll = 30}
${no_elem_across_roll_cover = 3}
${no_elem_outflow = 5}
${no_elm_along_roll_for_film_split = 20}
${no_elm_between_films = 10}
${no_elm_across_film = 3}
```

Here we see the nearly identical setup to the 2D roll coating templates for the forward roll film split problem (viz. Document GT-003.1). The main reason for this is that we are pursuing here the solution to a center section of a roller nip, well away from the edges. In fact, the procedure outlined herein will use the 2D solution to get an initial guess for the 3D solution. The most important parameters here are the **gap** and the roll speeds (**rollsp_b** for the bottom roll and **rollsp_t** for the top roll). The rolls are taken as rigid so the roll speed cover thickness etc. are not used. The definitions of the roll extent angles **theta1** and **theta2** are discussed in the previous memo, but are illustrated below for convenience

Note here that this part of the file simply describes the geometry of the rolls, the operating conditions, and sets the density of the finite element discretization. In other words, the whole problem is parameterized in this section. Noteworthy are the units, which are all in MKS, and the **APREPRO** variables **Gap_new, R_I_new**, and the other ***_new** variables. These represent the geometrical changes that can be made in the file during a run, i.e, after an initial solution has been obtained. It is important to understand here that GOMA solves for displacements of the nodes and not the actual coordinates as they evolve, and so coordinate DIFFERENCES must be tracked instead of absolute values.



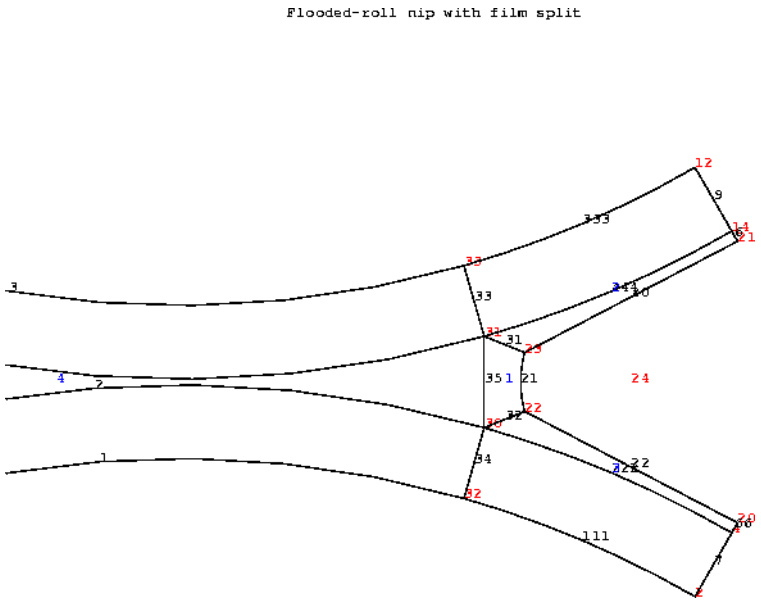
Hence, for the boundary condition changes for Gap and roll radius alterations must have both the initial value and the desired value, the initial value being that at which the mesh was originally generated.

The rest of the file describes a geometry with a series of **fastq** commands to set up the 2D problem. Note the extensive use of **APREPRO**. Also noteworthy is the definition of **y11_new** on line 44. This variable is basically used to track the requested change in the centroid of upper roll, as [x11, y11] are the coordinates of the centroid of that roll, which changes for gap and roll radius changes. **y11_new** is used in **roll_input_2D**, the 2D

GOMA input file. Also noteworthy are the following lines which define points 22, 30, 32, and 24:

```
point 22 {x22= R_O_h*cosd(theta1+14)} {y22 = R_O_2h*sind(theta1+14)}
point 30 {x30= R_O*cosd(theta1+16)} {y30 = R_O*sind(theta1+16)}
point 32 {x32= R_I*cosd(theta1+16)} {y32 = R_I*sind(theta1+16)}
```

These points are best explained in the following figure:



Point 22 is a point along the circular arc used to estimate the film split. Point 24 is the center of that circular arc as you can see it is used in a CIRC line type in **`roll.fas`**. Points 30 and 32 are a related position of this region of the mesh which describes the film split. To move the meniscus in or out of the nip, you change the phase angles on the points 22, 30 and 32 cards, keeping the phase angle of 30 and 32 greater than that of point 22, by about a couple of degrees. To position your meniscus further into the nip, increase all these phase angles by a few degrees, regenerate the mesh with **`fastq`**, and have a look at it. For splits that are WAY into the nip, you need to increase your curvature. You can accomplish this by moving point 24 to be weighted more heavily towards point 22 rather than 20, viz.

```
point 24 {x24 = (x22+x20)/2.} {y24 = (y23+y22)/2}
```

to

```
point 24 {x24 = (10.*x22+x20)/11.} {y24 = (y23+y22)/2}
```

First generate your mesh with the current conditions in the **`roll.fas`** file:

`fastq -a -mesh roll.gen roll.fas` (in batch mode, or you can do it interactively if you wish)

`ex1ex2v2 roll.gen roll.exoII`

The procedure to get an 2D solution at the operating conditions given (roll speeds 0.18 m/s, gap 0.001 m) is exactly the same as that in 2D roll coating memo (GT-003.1). Here that procedure will be repeated, but we will take advantage of some of the new features that did not exist at the time of the earlier memo to help obtain the solution. We will start with the base-case fixed grid solution. The input deck which is set up for this initial step is in **`roll_input_2D.1`**.

By way of review we will cover again how that initial guess is generated. Note that in **roll_input_2D.1** the "**`Initial Guess`**" card in the **`roll_input`** file is set to "**`zero`**", and the KINEMATIC, CAPILLARY, and SURFTANG cards are commented out, e.g.,

```
$
$ Film Split Meniscus
$
BC = VELO_NORMAL SS 100 0.
$$BC = KINEMATIC SS 100 0.
$$BC = CAPILLARY SS 100 0.01 0 0 0
$$BC = SURFTANG_SCALAR NS 200 0.01
$$BC = SURFTANG_SCALAR NS 210 0.01
```

Run GOMA: **`goma -a -i roll_input_2D.1`**

With the fixed grid solution you are now ready to release the film split. Your film split location is highly dependent on the operating conditions of the problem. Luckily, we can find an initial solution in which the location of the film split will be roughly the same as the initial mesh by finding the inflow pressure that keeps it there. Notice that in **`roll_input_2D.2`** you have the following boundary conditions:

```
BC = PLANE SS  5    1.                0.   0.  {-x5}
BC = PLANE SS  6   {-tand(-theta1)}   1.   0.  {-y11_new}
BC = PLANE SS  66   {-tand(theta1)}   1.   0.  {-y1}

$$ roll surfaces

BC = GD_PARAB SS 2 R_MESH_NORMAL 0 MESH_POSITION2 0 {x1*x1 + y1*y1 -
R_O_new*R_O_new} {-2.*y1} 1
BC = GD_PARAB SS 2 R_MESH_NORMAL 0 MESH_POSITION1 0 {0.} {-2.*x1} 1

BC = GD_PARAB SS 4 R_MESH_NORMAL 0 MESH_POSITION2 0 {x11*x11 +
y11_new*y11_new - R_O_new*R_O_new} {-2.*y11_new} 1
BC = GD_PARAB SS 4 R_MESH_NORMAL 0 MESH_POSITION1 0 {0.} {-2.*x11} 1

$$ Linear Velocity at roll surfaces
BC = VELO_TANGENT SS 2 0 {rollsp_b} 0. 0.
BC = VELO_NORMAL  SS 2 0
BC = VELO_TANGENT SS 4 0 {-rollsp_t} 0. 0.
BC = VELO_NORMAL  SS 4 0

$ Right now do nothing regarding flow
$ at inflow and outflow bndrys
```

```
$$ Inflow boundary  (specified pressure or velocity)
BC = FLOW_PRESSURE SS 5 3.265095e+03
$$BC = U NS 5  {h_t*rollsp_b*2/(y15-y5)}
$$BC = V NS 5 0.
$
$ Film Split Meniscus
$
$$BC = VELO_NORMAL SS 100 0.
BC = KINEMATIC SS 100 0.
BC = CAPILLARY SS 100 0.01 0 0 0
BC = SURFTANG NS 200 {sind(theta1)} {-cosd(theta1)} 0. -0.01
BC = SURFTANG NS 210 {sind(theta1)} {cosd(theta1)}  0. 0.01

BC = VELO_TANGENT SS 6 0 0. 0. 0. 0.
BC = VELO_TANGENT SS 66 0 0. 0. 0. 0.
```

A couple of things to note. First the **GD_*** boundary conditions are used to enforce the geometry along the roll surfaces. Next, **VELO_TANGENT** and **VELO_NORMAL** are used to enforce the tangential velocity of the liquid along those surfaces. Next, on the inflow plane denoted by side set 5, we apply a flow pressure. This condition is key and this tutorial will exploit the new advanced capability of solving for augmenting conditions to help determine what this pressure should be. Notice finally in these boundary conditions that we are releasing the free surface on this next step, as the **VELO_NORMAL** card has been commented out and the **KINEMATIC**, **CAPILLARY** and outflow **SURFTANG** cards are uncommented.

If we go for a solution with this set of boundary conditions, with an arbitrary inflow pressure, it is very unlikely we will converge. So we are going to float the pressure and apply a volume constraint to close mathematically the system. In **roll_input_2D.2** you will notice that we have an augmenting condition section of the form

```
-----------------------------------
Augmenting Conditions Specifications
-----------------------------------
Number of augmenting conditions = -1
AC = VC   {mat_id = 1} {volid = 1} {bcid = 11} {dfid=0} {compid = 0} {const
=5.710854e-04}
END OF AC
```

Actually, before we run this step we need to know the current volume of the liquid. This can be determined one of two ways. GOMA is equipped with a global variable capability which is used to record specific global features of a problem. One of those variables is the volume of a mesh. However, that variable is output to the EXODUSII database only if the volume constraint augmenting condition is active, via the **AC = VC** record above. Also, that volume is output to the standard output if this card is active. To get that volume, then, run one iteration with a newton update of zero, viz.

```
cp soln.dat contin.dat

goma -a -i roll_input_2D.2 -n 1 -r 0.
```

Output to the screen you will see

```
----------------------------
       Augmenting Conditions:       1
       Number of extra unknowns:    1

              MT[   1] VC[   1]=6.316069e-04 Param=3.265095e+03
```

Correspondingly, change the volume (last floating point entry) on the AC card to 6.316069e-04 (as this output suggests), as that is the volume that the augmenting condition will maintain. This has been done for you in **roll_input_2D.3**,

```
AC = VC   {mat_id = 1} {volid = 1} {bcid = 11} {dfid=0} {compid = 0} {const
=6.316070e-04}
```

The next step is to run GOMA and determine the pressure which maintains this volume:

```
goma -a -i roll_input_2D.3 -n 10 -r 0.1
cp soln.dat contin.dat
goma -a -i roll_input_2D.3
```

Notice how we relax for 10 Newton Iterations, then go for full Newton iteration to converge to the solution at this volume. The final few lines of that iteration process should be

```
10:35:41 [4] 1.9e-12 6.0e-11 6.0e-12 5.3e-07 4.1e-05 4.3e-06  1  3.2e+00/
1.4e-01
          AC 2.1e-16 2.1e-16 2.1e-16 5.3e-07 5.3e-07 5.3e-07     4.4e+00/
1.4e-01
L_oo cause:  dof=239    P_0 n=167    dof=339    P_0 n=237
10:36:21 [5] 3.8e-15 4.1e-14 7.9e-15 2.2e-10 9.8e-09 1.2e-09  1  3.2e+00/
1.4e-01
          AC 8.7e-19 8.7e-19 8.7e-19 1.9e-10 1.9e-10 1.9e-10     4.5e+00/
1.4e-01
L_oo cause:  dof=1735  u1_0 n=1247   dof=1624   P_0 n=1170

------------------------------
Augmenting Conditions:       1
Number of extra unknowns:    1

        MT[   1] VC[   1]=6.316070e-04 Param=9.169207e+03

-done
```

This output indicates that the pressure at the inflow side set 5 required to hold the coating bead in place and maintain the domain volume is 9,169.

CREATING A 3D Initial GUESS FROM A 2D SOLUTION:

First we need to extract the solution from the most recent **out.exoII** output file. It is that solution we will project into 3D. To extract that solution we use **algebra**, a SEAMS

tool that is used to manipulate **exoII** files. When we project to 3D we don't necessarily want all of the nodal variables associated with the 2D solution and in fact want only the primitive variables of velocity, displacements and pressure. We will run algebra on **out.exoII** which contains 6 "parameter" planes of solutions and save the last step with the nodal vars dmx, dmy, vx, vy, and pressure, IN THAT ORDER as a tool below depends on it.

```
algebra out.exoII tmp.exo
ALG> save dmx dmy vx vy pressure
ALG> step 6
ALG> exit
```

Now we have a 2D solution in the **tmp.exo** file with the appropriate variables. For the projection we will use the "not-yet-production" tool **exosweep**, which is not a part of your SEAMS distribution as of (2/21/00). Please contact Duane Labreche (dalabre@sandia.gov) or Randy Schunk (prschun@sandia.gov) for the source code. It is easily built if you have the EXODUS II and netcdf libraries that are shipped with SEAMS. **exosweep** enables us to sweep, or extrude, the 2D solution into a 3D section. It takes the following information as input

```
>exosweep -h
exosweep <n_intervals> <length> <in_file> <out_file>
```

We will extrude our 2D solution 0.05 length units (5 cm) and 6 elements, viz.

```
exosweep 6 .05 tmp.exo 3d.exo
```

If you **blot 3d.exo** you will notice now that it is 3D, but you have no sidesets and nodesets. Unfortunately **exosweep** has not yet been furbished with this capability. Luckily there is another tool that we can use to get this 3D solution, viz. velocity and pressure and displacement fields, onto a 3D mesh with all of the proper side set and node set information. That tool is **mapvar** (cf. Document GT-006.3 for basic instructions). First you need a mesh to map the solution onto, and we get that by using **gen3d** (A SEAMS tool) to extrude the original base 2D mesh. We also use that tool to place side sets and node sets on the front and back of the mesh; all other side sets and node sets are extruded with the mesh with gen3d. This procedure goes as follows:

```
gen3d roll.exoII tmp.g
  >translate 6 0.05
  >sset front 11
  >sset back 22
  >nset front 11
  >nset back 22
```

where **roll.exoII** was the original 2D mesh from the first **fastq** execution. Notice that we place the resultant mesh in **tmp.g**, "g" standing for "genesis" which is a naming convention for EXODUSII files that just have meshes and no solutions. We also extrude this mesh a distance that matches the **exosweep** step above. We also chose 6intervals,

although you could choose any number at this point because we will be mapping a solution onto this grid.

Now we have a mesh with the proper side set and node sets, viz. **tmp.g** from the **gen3d** step above, and a mesh of the same physical size with the extruded 2D solution, in **3d.exo** from the **exosweep** step above. We now simply map the solution onto the new grid with **mapvar** (see Remeshing Memo GT-006.3 for details):

```
cp 3d.exo tmp.e
mapvar tmp
MAP>deform 0
MAP>time 10
MAP>exit
```

After running **mapvar** your mapped 3D mesh and solution are in the **exodusII** file named **tmp.int**. You can **blot** this file to see if everything looks OK. In particular, make sure you have a displacement field and a velocity field. You can turn **deform off** in **blot** to see the displacements and contour the vx and vy velocities to make sure they are smooth. If variables are not smooth along the boundary, consult the remeshing and remapping tutorial (GT-006.3) for suggestions.

You now have an initial guess for a 3D solution. Before we discuss the boundary conditions in 3D, in a tutorial fashion, lets just obtain a solution in 3D for the base case. The input decks for the full 3D runs are in **roll_input_3D** and **roll_input_3D_trans**. We will first attempt to find a steady state solution with the former, and then use the latter to predict the onset of the ribbing instability.

Although we should be able to use the pressure on Side set 5 as predicted by the 2D base case to hold the meniscus in the nip at constant volume, we will re-invoke the volume constraint in 3D for the initial solution. Notice in the **roll_input_3D** file the volume constraint section:

```
-----------------------------------
Augmenting Conditions Specifications
-----------------------------------
Number of augmenting conditions = -1
AC = VC   {mat_id = 1} {volid = 1} {bcid = 6} {dfid=0} {compid = 0} {const
=3.158035e-05}
END OF AC
```

As we did in 2D, run GOMA one step to get the current volume of the mesh:

```
goma -a -i roll_input_3D -n 1 -r 0.
```

You will notice the following volume:

```
Augmenting Conditions:       1
Number of extra unknowns:    1

        MT[   1] VC[   1]=3.158035e-05 Param=-3.027660e+00
```

```
-done
```

Make sure that the last float on the AC record above corresponds to this volume, viz. 3.158035e-05. Also notice that we are floating the pressure and the inflow side set 5, which corresponds to the bcid=6 record in the AC condition (see Advanced Capabilities manual SAND2000-2465). Run GOMA to get a steady state solution:

```
goma -a -i roll_input_3D
```

After converging to a solution, which happens to be obtainable with full-Newton iteration, you will notice that the pressure required to keep the volume constant is 903 Pa, so make sure before going any further you change the flow pressure boundary condition accordingly so you can take off this augmenting constraint for further computations, viz.

```
BC = FLOW_PRESSURE SS 5 9.033860e+02
```

Correspondingly you can turn off the volume constraint condition by simply commenting it out. (You may want to keep it on, depending on what you are doing hereafter). Go ahead and save the solution file for further continuation either by saving **soln.dat** or by saving the **out.exoII** file.

USING AUTOMATIC HUNTING CONTINUATION TO INCREASE ROLL SPEED:

Before we discuss the boundary conditions and corresponding rotation conditions that were required to solve the 3D problem, we will do a couple more exercises to show the flexibility of the model. First we will continue in roll speed from 0.18 m/s to approx. 3 m/s using hunting continuation. In **roll_input_3D** you will notice the following section:

```
------------------------------------------------------------
                Hunting Specifications
------------------------------------------------------------
Number of hunting conditions  = -1
HC = BC 0 0    1 {rollsp_b} {3.0*rollsp_b} {0.1*rollsp_b} {0.1*rollsp_b}
{1.e7}
HC = BC 3 0   1 {-rollsp_t} {-3.0*rollsp_t} {0.1*rollsp_t}{0.1*rollsp_t}
{1.e7}
END OF HC
```

These two records, activated by changing the **Continuation** card to **hzero** or **hfirst**, guide the multiparameter continuation processes that results in both roll speeds to be increased by a factor of 3. We cannot in this case use **zero** or **first** order continuation because we have to continue in more than one parameter. Please read the Advanced Capabilities manual (SAND2000-2465) for details on these cards. At the time of this writing, **hfirst**, or first order continuation with hunting, has not been tested in 3D and using iterative solvers. So we will choose **hzero**, viz.

```
---
Continuation Specifications
---
```

```
Continuation                    = hzero
...
```

The Hunting specifications show that the boundary condition floats on boundary condition numbers 0 and 3, corresponding to **VELO_TANGENT_3D** conditions, will be increased. The functional form of **VELO_TANGENT_3D** is discussed below.

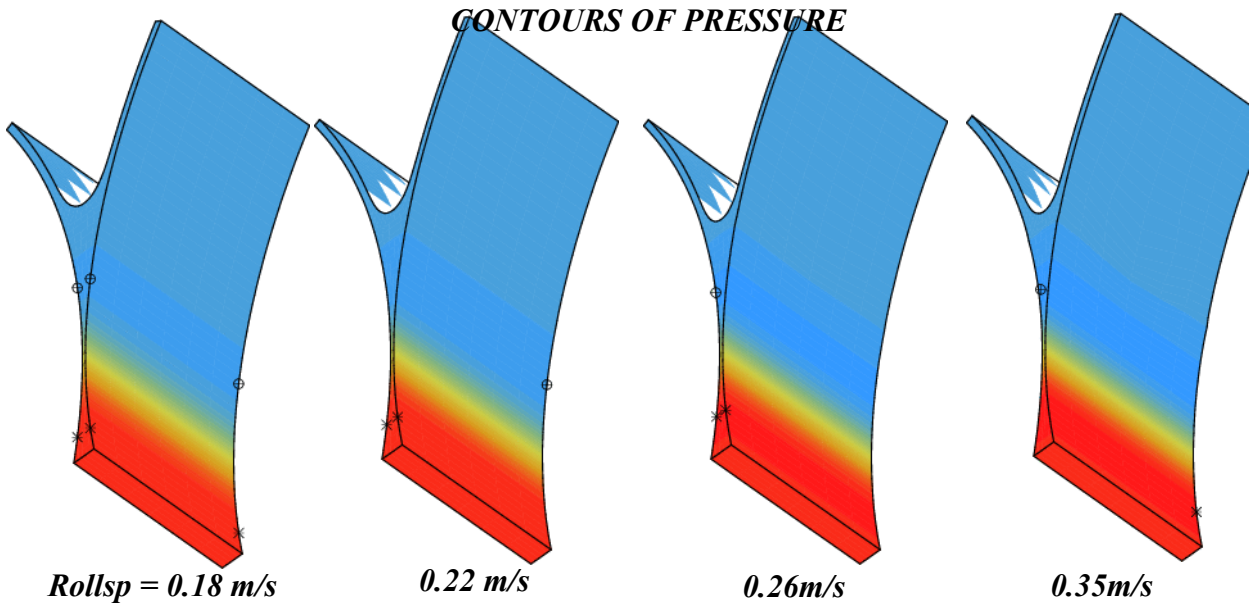
You can continue in roll speed simply by running GOMA (don't forget to change your initial guess to the appropriate source, like read for **contin.dat**, etc.):

```
goma -a -i roll_input_3D_trans,
```

The output will indicate that roll speed is be incremented:

```
---------------------------------
 Zero Order Hunting:
 Step number:    1 of   40 (max)
 Attempting solution at: theta = 0
BCID=  0 DFID=    0 Parameter=  1.800000e-01 delta_s= 9.230769e-03
BCID=  3 DFID=    0 Parameter= -1.800000e-01 delta_s= 9.230769e-03

          R e s i d u a l         C o r r e c t i o n

  ToD    itn   L_oo    L_1     L_2     L_oo    L_1     L_2   lis  asm/slv (sec)
-------- --- ------- ------- ------- ------- ------- ------- --- --------
-------
12:02:55 [0] 4.6e-09 8.0e-08 1.7e-08
```

Sample results from this continuation by "hunting" step are shown below.



CHANGING THE GAP SEPARATION:

For fun you can change the roll separation from 1 mm to 2 mm by changing **Gap_new** in **`roll.fas`** from 0.001 to 0.0012 with the following sequence (or just type "**`source run2`**"):

```
goma -a -i roll_input_3D
```

Notice when looking at this solution that you have widened the gap with a specified pressure, so the pressure drop changes significantly and the meniscus nearly gets blown out of the domain. In this case, unlike the deformable roll case above, you will notice that changing the gap we are actually changing the equation of the circular line on side set 2 through the following BCs:

```
BC = GD_PARAB SS 2 R_MESH_NORMAL 0 MESH_POSITION2 0 {x1*x1 + y1*y1 - R_O_new*R_O_new} {-2.*y1} 1
BC = GD_PARAB SS 2 R_MESH_NORMAL 0 MESH_POSITION1 0 {0.} {-2.*x1} 1
```

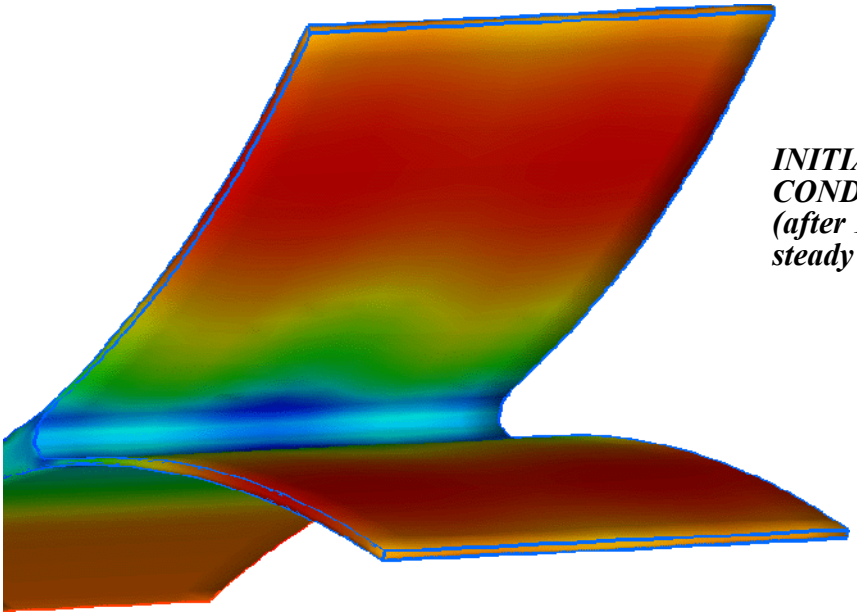
Please look at the manual to figure out what these mean. In a "nut-shell", you can see they specify an equation of a circle with radius **`R_O_new`** centered about the point **`[x1, y1]`**. The are accumulative and there are some nice examples in the manual of how to read them. The **`R_MESH_NORMAL`** entry means that the equation replaces the normal component of the mesh equations along that side set. This is very different from the deformable case in which the roll surface was "free".

RUNNING TRANSIENT, IN SEARCH OF RIBS

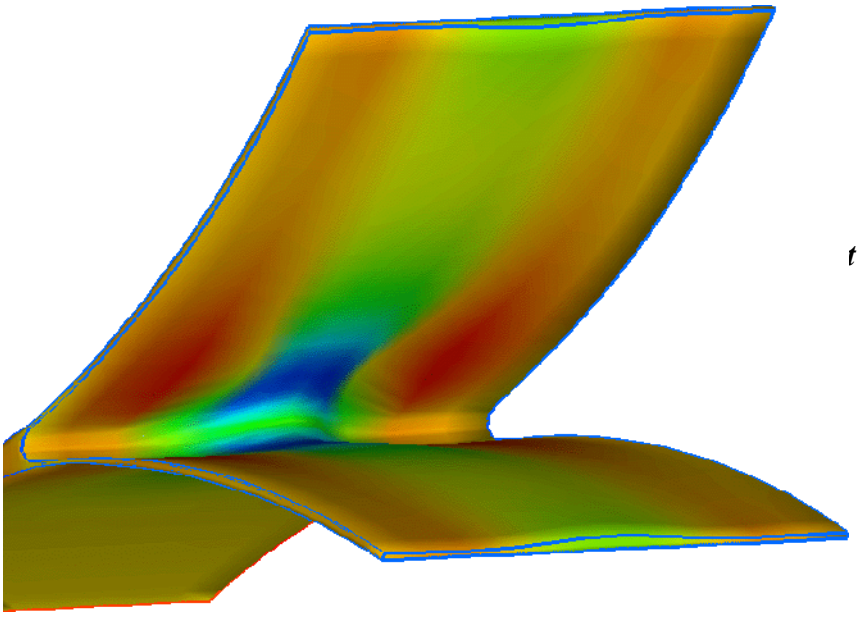
In the directory you will find an input file named **`roll_input_3D_trans`**. Use the UNIX diff command to differentiate it with **`roll_input_3D`**. You will notice that we have turned on the mass-matrix term multipliers on the fluid momentum equations and that we have changed **`steady`** to **`transient`** on the **`Time integration`** card. There are some other subtle differences in the matrix solver cards. You will also notice that we bumped the roll speeds by 3 percent over the base case to try to trigger rib formation. A few tests have indicated that you need to using hunting continuation to get up to about 0.3 m/s to see the formation of ribs. Below are some sample results. At the time of this memo (Feb. 24 2000) we have no deformed geometry remeshing capability in 3D and so we could not follow the formation of these ribs very far.

BOUNDARY CONDITIONS AND ROTATION CONDITIONS

The discussion here will focus on the figure showing the mesh and corresponding side sets below. Note that the z-direction is through the cross section, or across the flow. The inflow plane, i.e., sset 5, is oriented such that its outward facing normal is in the negative-x direction. The roll surfaces are denoted as sset 2 ("bottom roll") and sset 4 ("top roll"). The outflow planes are denoted as sset 6 and 66. The front and back end planes are denoted as sset 11 and 12. Finally the free surface corresponds to sset 100. The boundary conditions above correspond to the 2D case. Here we present them for



INITIAL STARTING CONDITIONS (after 1 time step from steady state)

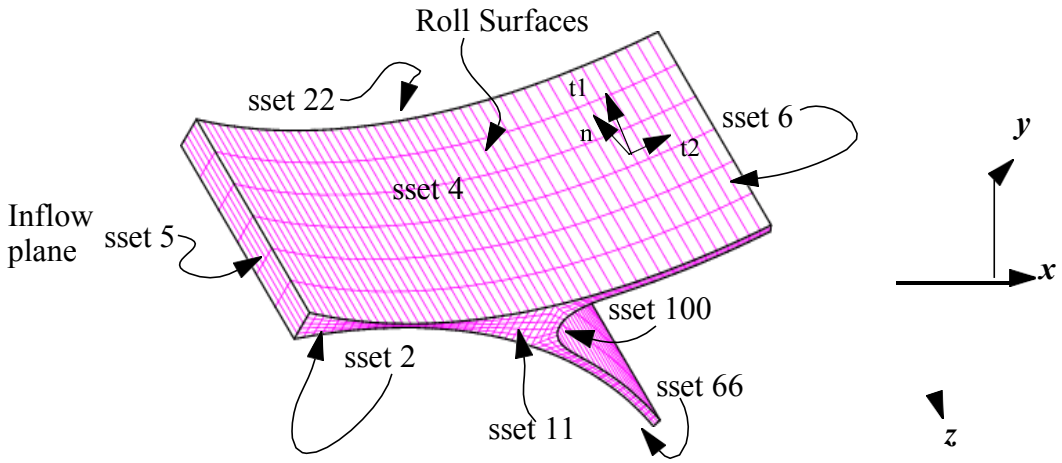


t = 0.1 s

Density = 0, Viscosity = 0.1, Uroll = 0.3, ST = 0.04
Ca = 0.875, Re = 0.

this 3D case. Although they are similar, there are subtle differences that deserve further discussion (viz. the ones underscored):

```
Number of BC = -1
```




```
$$ Linear Velocity at roll surfaces
BC = VELO TANGENT 3D SS 2 {rollsp b} 0. 0. -1.
BC = W NS 2 0.
BC = VELO_NORMAL SS 2 0
BC = VELO TANGENT 3D SS 4 {-rollsp t} 0. 0. -1.
BC = W NS 4 0.
BC = VELO_NORMAL  SS 4 0

$ Right now do nothing regarding flow
$ at inflow and outflow bndrys

$$ Inflow boundary  (specified pressure or velocity)
BC = FLOW_PRESSURE SS 5 9.033860e+02
BC = W NS 5 0.
$$BC = U NS 5  {h_t*rollsp_b*2/(y15-y5)}
BC = V NS 5 0.
$
$ Film Split Meniscus
$
$$BC = VELO_NORMAL SS 100 0.
```

```
BC = KINEMATIC SS 100 0.
BC = CAPILLARY SS 100 0.04 0 0 0
BC = SURFTANG_EDGE SS 6 100 {sind(theta1)} {cosd(theta1)} 0. 0.04
BC = SURFTANG_EDGE SS 66 100 {sind(theta1)} {-cosd(theta1)}  0. 0.04

$$Fully developed outflow cards
$$BC = VELO_TANGENT_3D SS 6 0. 0. 0. -1.
$$BC = VELO_TANGENT_3D SS 66 0. 0. 0. -1.
BC = W NS 6 0.
BC = W NS 66 0.

BC = W NS 11 0.
BC = W NS 22 0.

BC = PLANE SS  5    1.                0.   0.  {-x5}
BC = PLANE SS  6    {-tand(-theta1)}  1.   0.  {-y11_new}
BC = PLANE SS  66    {-tand(theta1)}  1.   0.  {-y1}

BC = PLANE SS 11 0. 0. 1. 0.
BC = PLANE SS 22 0. 0. 1. 0.05

$$ roll surfaces (the r_mesh_normal ones don't work because we cannot put
$$              two ROT conditions on the same side set and same condition
$$             So we won't rotate for now. Need new conditions here. GEOM

BC = GD_PARAB SS 2 R_MESH2 0 MESH_POSITION2 0 {x1*x1 + y1*y1 -
R_O_new*R_O_new} {-2.*y1} 1
BC = GD_PARAB SS 2 R_MESH2 0 MESH_POSITION1 0 {0.} {-2.*x1} 1

BC = GD_PARAB SS 4 R_MESH2 0 MESH_POSITION2 0 {x11*x11 + y11_new*y11_new
- R_O_new*R_O_new} {-2.*y11_new} 1
BC = GD_PARAB SS 4 R_MESH2 0 MESH_POSITION1 0 {0.} {-2.*x11} 1

######
END OF BC
######
```

We begin with the **VELO_TANGENT_3D** cards, which basically apply Dirichlet conditions to the fluid momentum equations on the linear roll speed at cylindrical roll surfaces. The form of this boundary condition is:

$$\underset{\sim}{n} \times \underset{\sim}{t}_1 \bullet \underset{\sim}{v} = \underset{\sim}{t}_2 \times \underset{\sim}{v} = v^0_t \qquad \text{(EQ 1)}$$

Here the normal vector-tangent surface basis $[n, t_1, t_2]$ is shown on the figure above, with t_1 being specified to be in the z-direction (last 3 floats on the **VELO_TANGENT_3D** card). The first float is the linear roll speed at the surface, which we desire to be in the direction t_2 as defined by the cross product. The key point of this boundary condition is that t_1 must be constant across the entire surface, which clearly limits this conditions applicability to surfaces with zero curvature in one direction (viz. extruded curves without warpage). In summary, these two cards are used to specify a portion of the no-

slip boundary condition tangential to the roll in the circumferential direction. Together with them we also specify the z-directed velocity (viz. the axial velocity relative to each cylindrical axis) is zero on these surfaces.

Next, the **SURFTANG_EDGE** conditions are used to specify the weak-form contribution to the surface curvature-dependent surface tension force, analogously to the SURFTANG conditions applied in 2D (see slot coating tutorial GT-002.1 and the Appendix of the User's guide, SAND97-2404). Notice the input on these cards are two side sets (the outflow planes 6 and 66 and the free surface 100) which define the seam. The remaining data define the outflow tangent vector and the surface tension.

Finally, the **GD_*** conditions used to define the geometry of the roll surfaces deserve some discussion. In the 2D case, cf. **roll_input_2D**, these conditions were applied to the **R_MESH_NORMAL** equation, which is appropriate and natural if you desire the mesh to slide along the surfaces tangentially and stress free (Sackinger, et al., 1996, Cairncross, et al., 2000). The problem in 3D is that the **ROTATION** conditions (see below) cannot accommodate more than one rotatable condition per surface, as two **GD_PARAB**/**GD_LINEAR** boundary conditions on mesh motion would require. Currently this "missing feature" is on a list of items to be fixed. The workaround here is to instead apply the geometry constraints on the **R_MESH2** equation (or y-directed mesh momentum equation) as the roll surfaces are predominantly oriented in that direction. Although this limits some of the range of motion of these points, it seems not to be too detrimental to the study here.

ROTATION SPECIFICATIONS FOR THIS PROBLEM

We will discuss the "all-important" rotation specifications for this problem section-by-section. You should consult Chapter 4 of the GOMA User's guide (SAND97-2404) first to familiarize yourself with the purpose and form of these conditions.

We start first with the surfaces of the problem. On each surface we apply both geometric and fluid momentum boundary conditions. Recall that each of these systems are really vector momentum equations, and so any boundary condition that is a scalar or strong condition on only one component, with the others being left as natural weak form conditions, necessitates rotation of those momentum equations at that surface (solid mesh and fluid). For this problem we force those rotations and associate the proper boundary condition defining the surface as follows:

```
Rotation Specifications =

ROT = MESH SURFACE 5  PLANE 5 T1  0  T2  0 SEED  -0. 0. 1.
ROT = MESH SURFACE 6  PLANE 6 T1  0  T2  0 SEED  -0. 0. 1.
ROT = MESH SURFACE 66  PLANE 66 T1  0  T2  0 SEED  -0. 0. 1.
ROT = MESH SURFACE 4  T1 0 GD_PARAB 4   T2  0 SEED  -0. 0. 1.
ROT = MESH SURFACE 2  T1 0 GD_PARAB 2   T2  0 SEED  -0. 0. 1.
ROT = MESH SURFACE 11  T1 0 T2 0 PLANE 11 BASIS_RESEED 0. -1. 0.
ROT = MESH SURFACE 22  T1 0 T2 0 PLANE 22 BASIS_RESEED 0. -1. 0.
ROT = MESH SURFACE 100  T1  0  KINEMATIC   100 T2  0 BASIS_RESEED  -0. 0. -1.
```

```
ROT = MOM SURFACE  4  VELO_TANGENT_3D  4 VELO_NORMAL  4   T2  0 SEED  -1. 0. 0.
ROT = MOM SURFACE  2  VELO_TANGENT_3D  2 VELO_NORMAL  2   T2  0 SEED  -1. 0. 0.
```

Notice that there are eight such conditions pertaining to mesh, and each dictates to which component of the mesh equations the geometric condition is applied. All **PLANE** conditions at the inflow and outflow planes are applied to the normal component of the momentum equation, which in this case supplants the x-component. The other two components denoted by **T1** and **T2** are simply left in their weak forms. The normal-tangential basis is generated in these surfaces by the **SEED** option. Note that the vector following the **SEED** option is always in the z-direction, which is appropriate for all surfaces except 11 and 12, which are oriented in the z-direction. For those surfaces and the free surface we use **BASIS_RESEED**, which is a more robust option. This we usually use on surfaces for which no single vector is guaranteed to never align with the surface normal. Also notice that we do apply a **ROT** condition for the **GD_PARAB** geometry conditions as well, even though these conditions are not rotated. It is a good practice to apply a ROT condition to all geometric surface, viz. of type **MESH**, whether they are rotated or not.

----------ASIDE--------

Trouble shooting tip: We have noticed through extensive use that sometimes SEED and sometimes BASIS_RESEED options are better. It depends on the problem. The **MESH SURFACE ROT** section you should scrutinize very carefully. It is important that all surfaces have appropriate **MESH ROT** conditions. You should try changing the seed vectors if appropriate as there is no unique right answer; you should also try changing the component position (first, second, or third) of the boundary condition, the other two being **T1** and **T2** options, if the problem isn't converging and if it seems to make sense. If you take a few iterations and the mesh diverges, turn **Write Intermediate Solutions** to "yes" and run a few steps. De-magnify or magnify the displacements, whichever is more appropriate, and try to ascertain on which surface the mesh is misbehaving. Before solving for any flow equations you should always solve the geometry first to make sure everything is OK, viz. only solve the **mesh1-mesh3** equations and on small representative meshes. You of course will have to turn off all **ROT** conditions associated with the fluid momentum equations.

The next set of conditions are those of type "**MOM SURFACE**". Here we only have two, corresponding to the roll surfaces. All other surfaces have either natural, do-nothing boundary conditions or hard-set dirichlet conditions, neither of which trigger rotation of the fluid-momentum equations. The **VELO_TANGENT_3D** BC does, however. You can see that we specify these **ROT** conditions in much the same way as those for geometry.

----------ASIDE-------

If you are confused as to which boundary conditions are of the **ROTATED** type, look at the source code file **mm_names.h**. In that header file is a complete list of all of the boundary conditions with entries that describe whether they are rotated or not.

The next set of **ROT** conditions are those of type **EDGE**, for both the mesh and the fluid momentum equations. We will begin with those applied to the mesh equations:

```
ROT = MESH EDGE 100 6 PLANE 6 KINEMATIC 100 T 0 NONE
ROT = MESH EDGE 100 66 PLANE 66 KINEMATIC 100 T 0 NONE
ROT = MESH EDGE 6 4 PLANE 6 GD_PARAB 4 T 0 NONE
ROT = MESH EDGE 66 2 PLANE 66 GD_PARAB 2 T 0 NONE
#ROT = MESH EDGE 2 5 PLANE 5 GD_PARAB 2 T 0 NONE
#ROT = MESH EDGE 4 5 PLANE 5 GD_PARAB 4 T 0 NONE
ROT = MESH EDGE 100 11 T 0 KINEMATIC 100 PLANE 11 NONE
ROT = MESH EDGE 100 22 T 0 KINEMATIC 100 PLANE 22 NONE
ROT = MESH EDGE 6 11 PLANE 6 T 0 PLANE 11 NONE
ROT = MESH EDGE 66 11 PLANE 66 T 0 PLANE 11 NONE
ROT = MESH EDGE 6 22 PLANE 6 T 0 PLANE 22 NONE
ROT = MESH EDGE 66 22 PLANE 66 T 0 PLANE 22 NONE
```

To put it bluntly, these edge cards can be tricky. Luckily experience has shown that the correct order and form do make perfect sense, although the "learning curve" is steep. The purpose of these conditions is to control the mesh motion along seams defined by the intersection of to geometric surfaces. In GOMA we defined these surfaces by the combination of a side set with a geometric boundary condition (like **PLANE**, **KINEMATIC**, etc.) for **MESH** type **ROT** conditions or boundary conditions on velocity components for **MOM** type **ROT** conditions. We begin here with mesh conditions. It is a good idea to add a **ROT = MESH EDGE** card for every seam in your domain, although it is not necessarily required. You can see from the figure above that there are 18 seams (count all the edges between surfaces). We have employed 10 mesh edge cards. Although we experimented with **MESH EDGE** cards on the remaining 8 seams, they didn't seem to help matters, although they should be employed.You will notice that they are "commented out" in the input deck. Notice that the first to integer entries of each "**EDGE**" card are the two side set IDs defining the edge. For example, the edge defined by side sets 100 and 66 is that which is the intersection of the free surface and the outflow plane 66. These side sets can be input in any order, although it does affect the way in which seam norma/tangent bases are computed. A few conditions however (e.g. CA_EDGE), require certain ordering (cf. GT-013.1 on 3D slot coating). Switching them if you notice poor mesh behavior is troubleshooting option. Notice that the next 6 entries then dictate to which vector component of the mesh stress equation each geometry condition will be applied. In the first case, with the edge between 100 and 66, notice that we put the **PLANE** command in the normal position and the **KINEMATIC** command in the first tangent condition, although the order shouldn't matter. We find that the order does affect the conditioning of the resulting matrix system. As a rule-of-thumb try to place these conditions in consistent positions with the **MESH SURFACE ROT** cards above. Notice that the **PLANE** condition on 66 is on the x-component on the **SURFACE** card and the **KINEMATIC** condition on 100 is on the y-component, just like on the mesh edge card for 100/66 (first **MESH** card).

Please keep in mind that you may have to make several trials to get this right. You can get clues as to which **ROT** condition may be problematic by magnifying

displacements in **blot** or **mustafa**. If you notice strange behavior in mesh motion along a seam, you should examine the appropriate card.

Next, in the **ROT** condition set, are the **ROT = MOM EDGE** cards which apply to the fluid momentum boundary conditions. These look like

```
ROT = MOM EDGE 6 4 VELO_TANGENT_3D 4 VELO_NORMAL 4 T 0 NONE
ROT = MOM EDGE 66 2 VELO_TANGENT_3D 2 VELO_NORMAL 2 T 0 NONE
$$These are going to get surftangs
$ROT = MOM EDGE 100 66 T 0  T 0  W 66 NONE
$ROT = MOM EDGE 100 6  T 0 T 0  W 6 NONE
ROT = MOM EDGE 100 11 T 0 T 0  W 11 NONE
ROT = MOM EDGE 100 22 T 0 T 0  W 22 NONE
$
ROT = MOM EDGE 2 11 VELO_TANGENT_3D 2 VELO_NORMAL 2  W 11 NONE
ROT = MOM EDGE 4 11 VELO_TANGENT_3D 4 VELO_NORMAL 4  W 11 NONE
ROT = MOM EDGE 2 22 VELO_TANGENT_3D 2 VELO_NORMAL 2  W 22 NONE
ROT = MOM EDGE 4 22 VELO_TANGENT_3D 4 VELO_NORMAL 4  W 22 NONE
```

Notice there that we again are trying to instruct GOMA how to apply boundary conditions on vector fluid momentum equations along the seams of the domain. It turns out to be important here to resolve mostly how **VELO_TANGENT_3D** interacts with **VELO_NORMAL** and the Dirichlet condition on the z-component of velocity, **W**, along the edges of the roll surfaces.

Next we will talk about the **ROT = MESH VERTEX** cards:

```
ROT = MESH VERTEX 6 11 4 PLANE 6 GD_PARAB 4 PLANE 11 NONE
ROT = MESH VERTEX 6 11 100 PLANE 6 KINEMATIC 100 PLANE 11 NONE
ROT = MESH VERTEX 6 22 4 PLANE 6 GD_PARAB 4 PLANE 22 NONE
ROT = MESH VERTEX 6 22 100 PLANE 6 KINEMATIC 100 PLANE 22 NONE
ROT = MESH VERTEX 66 11 2 PLANE 66 GD_PARAB 2 PLANE 11 NONE
ROT = MESH VERTEX 66 11 100 PLANE 66 KINEMATIC 100 PLANE 11 NONE
ROT = MESH VERTEX 66 22 2 PLANE 66 GD_PARAB 2 PLANE 22 NONE
ROT = MESH VERTEX 66 11 100 PLANE 66 KINEMATIC 100 PLANE 11 NONE
```

VERTEX rotation cards provide exact ordering of boundary conditions at vertices in the mesh, which can be real important. Where multiple mesh edges come together **GOMA** can get confused as to which component of the mesh motion equations should receive which boundary condition. This turned out to be important at the outflow planes, as was indicated by strange mesh behavior during early trial simulations. Notice how there are 3 integer floats following the **VERTEX** option, and each integer corresponds to a mesh side set. Following those hints is an ordering, in x-y-z form, of the 3 appropriate geometric boundary conditions. Notice that in all cases they are consistent with the **MESH SURFACE** rotation conditions above.

It turns out that no **ROT = MOM VERTEX** conditions were required for the fluid momentum equations in this problem, although applying them correctly should not affect matters.

Much remains to be done on this problem but this tutorial should get you started and help you become proficient in 3D capillary hydrodynamics flows.

DEBUGGING NOTES:

- If you ever get termination of GOMA with the following message, that means the mesh "blew up". You should relax more or take a smaller parameter step change.

```
Volume change  -0.126022
Deformation Gradient 4.865954  3.642131
Deformation Gradient 2.177972  -0.000544
```

- When you are unsure of what is going wrong when you repeatedly get this message or when the flow is not converging, it helps to turn the "**`Write Intermediate Solution`**" card to "**`yes`**" and take some relaxed (i.e., **`-r 0.1`** or **`-r 0.01`**) steps and view the results with **`blot`**. This option causes GOMA to dump all Newton iterations to the exoII file.