Sandia National Laboratories

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

Albuquerque, New Mexico 0834
87185-

date: April 1,

2006

to:

Distribution

from: E. D. Wilkes, MS-0834 (GRAM, Inc.), P. R. Schunk,
9114

subject: Structural shell application example: tensioned-web slot coater (GT-033.0)

keywords: shell elements, coating flows, continuation, linear stability analysis

input records:

Introduction

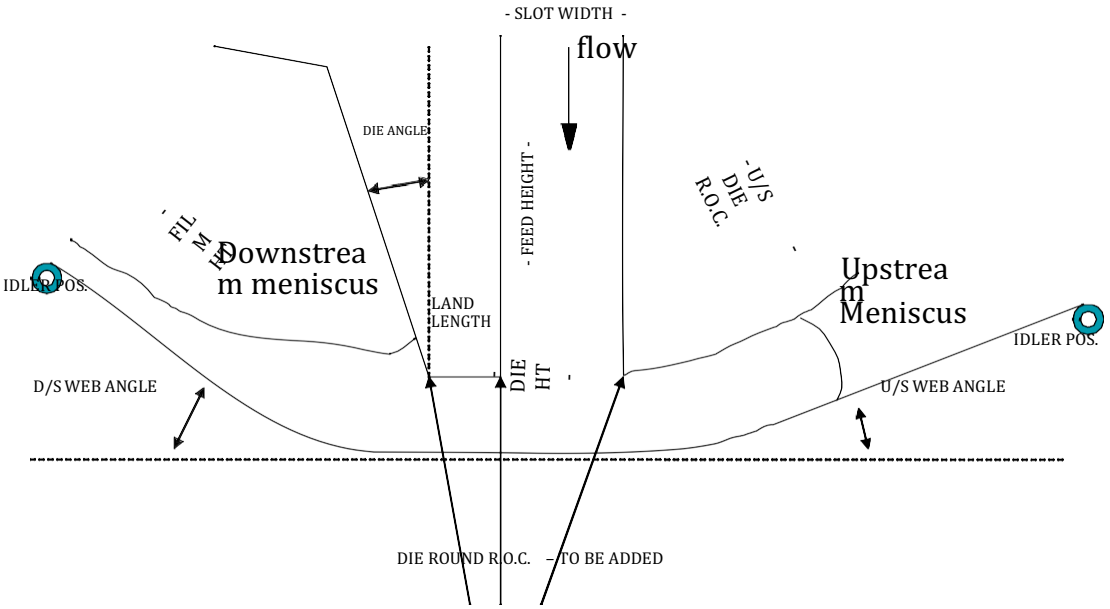
The new structural shell capability that has recently been implemented in *Goma* allows solution of a new class of physical problems which involve both 1D and 2D physics. General examples of this class of problems include reacting flows with surface chemistry, electrostatics, thin films, and problems with both fluid and thin solid mechanics. An example of the latter problem type is a slot coating process in which the substrate is a flexible (but inextensible) solid belt of web, which is the subject of this memorandum. The reader is encouraged to consult a companion tutorial addressing the theory of structural shell elements before attempting this tutorial (GT-027-1).

A typical slot coating process is sketched in Figure 1. Here, a coating material (initially liquid) is fed downward through a narrow slot bounded on either side by solid die "lips". The fluid then deposits on the moving substrate, forming a film with an initial thickness equal to the vertical gap between the bottom of the die and the web underneath it. This thickness then tapers down and reaches a level value, which will be the thickness of the exiting film product and simply determined by the incoming flow rate and the web speed.. Maintaining a constant value of this thickness is a primary objective in the design of such a process.

This slot coater process is similar to the one studied by Gates (1999), the main difference being that the substrate is no longer a flat surface and is undergoing mechanical deformation that affects the process. Here, the web substrate has a bending stiffness determined by its Young's modulus, Poisson's ratio, and a nominal thickness:

$$D=Et^3/12(1-\nu^2),$$

where E is the elastic modulus, ν Poison ratio, and t the shell thickness.



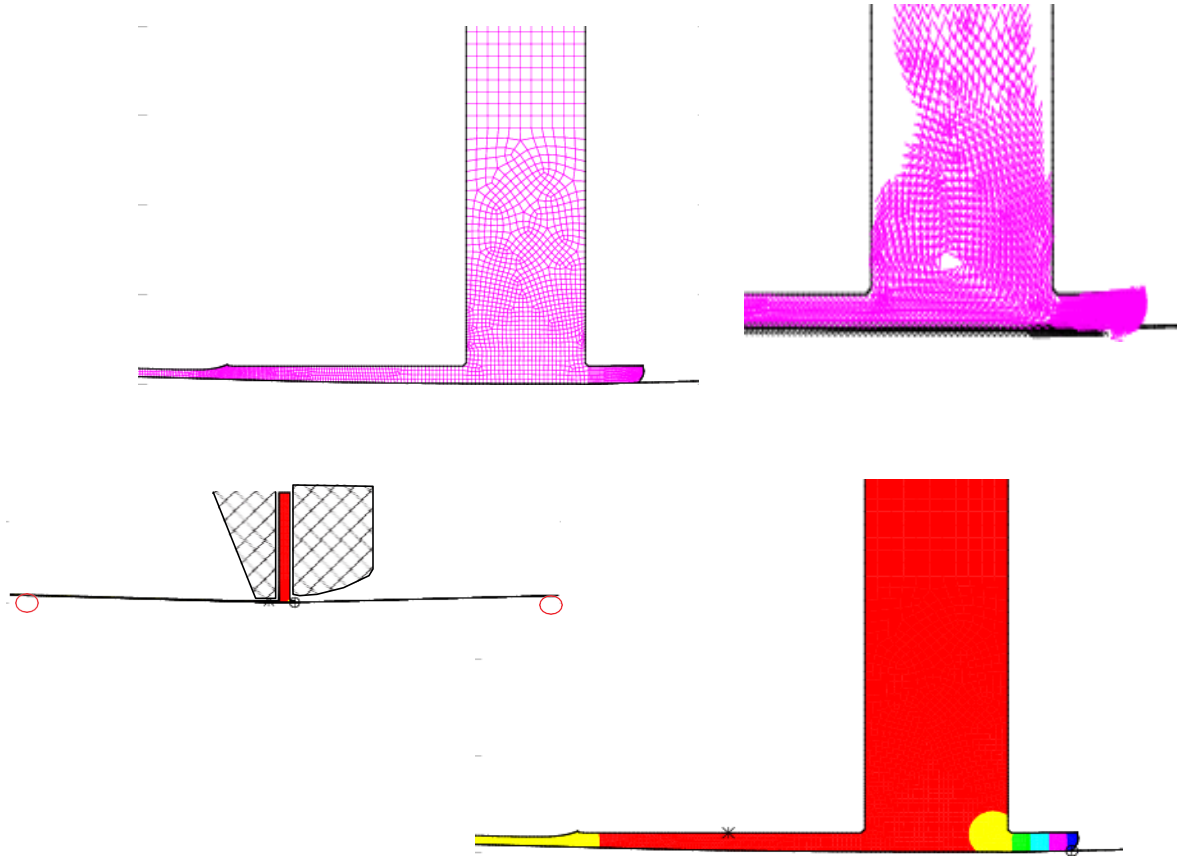
Using a flexible web substrate has some advantages over a flat substrate. For some applications, a flexible web may allow the computational model to more closely represent the process being modeled. Also, the upstream bead or meniscus can be stabilized against a large horizontal pressure gradient in the fluid by placing the upstream tensioner (idler) so that the incoming part of the web is at a small angle from the horizontal - this can eliminate the need to apply a vacuum to the upstream side, as is common practice for fixed nip/rigid backing roll applications.

Below we illustrate a preliminary solution to a specific tension-web slot coater configuration. In the parameter range which corresponds to this solution we see a coating bead which is stabilized by an upstream backpressure: applying a vacuum pressure is a tactic used in most fixed-web configurations to stabilize the meniscus at high coating speeds. The goal here is to continue in web tension and take- off and incoming web angles such that the vacuum back-pressure stabilization can be eliminated.
Achieving this state requires some knowledge of the forces at work. We discuss these where appropriate below

.

Problem Setup

The process is modeled for Goma through the use of a mesh with both 2D (bulk) and 1D (shell) element blocks. There is one bulk element block for the entire fluid domain, on which the governing equations include the Navier-Stokes system (momentum, continuity) and mesh displacement. The web is divided into two shell element blocks, one for the upstream part (which is dry and has no contact with fluid), and the other for the downstream part (which is bordered or “wetted” by the fluid. The equations solved on both of these shell blocks govern the shell tension, the shell curvature, and the mesh position(see GT-027.1 for details). The Goma formulation currently requires that such 1D domains be divided into “wet” and “dry” element blocks; this can handled by CUBIT.



Preliminary solution with an applied backpressure.

The files provided with the tutorial distribution (directory **`Shell_web_tension`**) are as follows:

- **`fluid.mat`** - fluid property inputs
- **`shell.mat`** - web property inputs
- **`slot.geom`** - contains problem dimension data and thermophysical property data for Goma and Cubit
- **`slot.jou`** - Cubit journal file which reads inputs and generates initial mesh
- **`upstream_land.pl`** - Perl script which generates upstream_land.dat
- **`upstream_land.dat`** - contains problem dimension data
- **`downstream_land.pl`** - Perl script which generates downstream_land.dat
- **`downstream_land.dat`** - contains problem dimension data
- **`step[1..8].input`** - Goma input files for each continuation segment

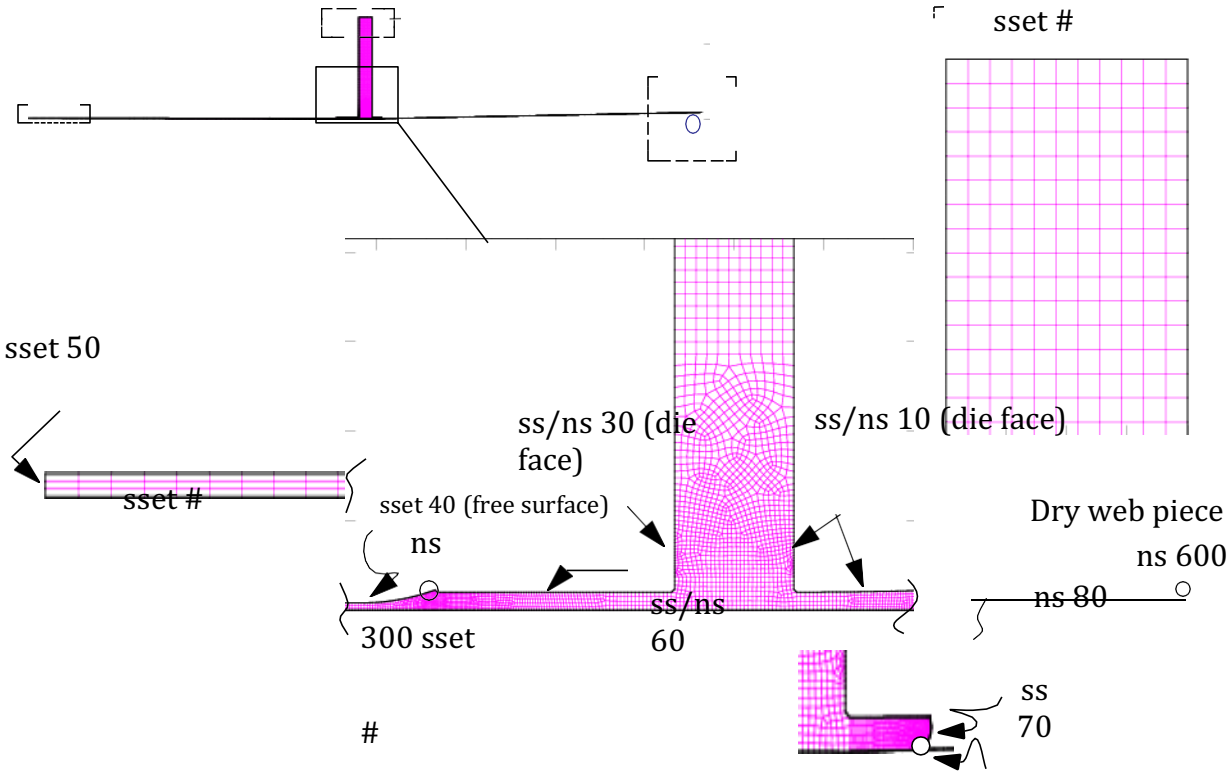
Here we discuss the most important aspects of each of these input files. Before exercising the model template you should understand how this problem is set up and parameterized.

We begin with the CUBIT journal file, **`slot.jou`**.

Most important to recognize in this file are that elements are linear **quad4** and **bar2** type. Curve 374 in the final geometry represents the unwetted web. Note that we mesh it up as a meshed curve, viz.

```
curve 374 scheme bias fine size {el_tny} coarse size {el_hug} start vertex
133
mesh curve 374
```

The best way to familiarize yourself with the CUBIT journal file is to start up CUBIT in the interactive mode and cut-and-paste section-by-section into the command-line. By doing this you can observe how each of the geometric features are defined and built. In any case, the picture below can be used as a general reference to all of the important side-sets and node-sets that a user of this template may want to reference.



Key geometric features and ExodusII side sets and node sets for the tensioned-web slot coating problem.

All parameters for the geometric features are contained in the **slot.geom** definition file. Here we discuss some important aspects of that file:

First, the geometric parameters.

```
# Radius of die rounds/fillets:{round_radius = 0.005}
# DOWNSTREAM (left) die:
# Land height above web:    {land_height_D = land_height_U}
# Slot height above land:   {slot_height_D = slot_height_U}
```

```
# Land length (along bottom):{land_length = 0.4}
# Angle of back side (from vertical){alpha_1 = 20}
# Die contact line distance from lip:   {clpos_D = 0.004201037}

# WEB (moving substrate):
# Initial y-position{web_init_y = 0.0}
# Thickness{web_thk = 0.1}
# Upstream length (to upstream die lip) {web_length_U = 5.0}
# Length of web CL slip region{slip_length = 0.004}
# Downstream length (from downstream die lip)
#   {web_length_D = 5.0}
# Downstream film thickness{film_height = 0.012}
```

Note that these are initial parameters that go into the original mesh. Typically these will not change during the simulation, or should not be changed as they just are used to specify the original mesh state (e.g. the Downstream film thickness, or film_height variable, will not change but is determined as part of the solution). The geometric quantities that have been identified as operating parameters are described below.

The mesh properties section is used to specify the level of refinement. For your convenience you can substitute any of these **Aprepro** variables in the **slot.jou** file to change the initial mesh. We suggest you stay with these settings until you are familiar with the problem.

```
# MESH PROPERTIES:
# Nominal element sizes:
# Teeny (near die lips & contact lines){el_tny = 0.0025}
# Teeny2 (for upstream meniscus){el_tny2 = 0.0021}
# Small (on downstream film thickness){el_sml = 0.003}
# Small2 (on downstream film thickness){el_sml2 = 0.004}
# Medium (at bottom of slot){el_med = 0.005}
# Big (start of flat film length){el_big = 0.007}
# Large (slot inlet & upstream web){el_lrg = 0.020}
# Huge (far end of flat film length){el_hug = 0.030}
# On slip region only:{el_slp = 0.002}
```

Next we have the material property Aprepro variables, which are mostly used in **fluid.mat** and **shell.mat**. The majority of these properties where chosen to correspond to real coating operation. Note the units are in g-mm-s.

```
# MATERIAL PROPERTIES:
#--NOTE TO SELF:  Check these units!!!!
# Density [g/mm^3]{density = 9.3e-4}
# Viscosity [g/mm-s]{viscosity = 5.2}
# Surface tension [dyn/mm]{surface_tension = 60.0}
# Navier slip coefficient{beta = 0.001}
# Web bending stiffness{web_E = 160.0}
```

The operating parameters include the web speed, the upstream and downstream idler position, and the web tension. Note that the idler positions are described by the angle from the horizontal and the web length. The length parameters are set up above.

```
# OPERATING PARAMETERS:
# Web speed [mm/s]{webspeed = -1000.0}
# Upstream idler tension [Pa]{web_tension = 1.0e+8}
# Upstream angle (from horizontal){web_angle_U = 0.0}
# Upstream angle at remesh step {web_angle_U_at_remesh = 0}
# Downstream angle (from horizontal){web_angle_D = 0.0}
# Downstream angle at remesh{web_angle_D_at_remesh = 0}
```

Note that the downstream and upstream idler angles are set and lead to a recalculation of the coefficients of the pinned positions and downstream out flow plane. The remainder of the **`slot.geom`** file addresses those calculations.

The material parameters and the operating parameters set here are targets. It is very difficult to go right to the solution from a zero initial guess using these parameters. Below we discuss those anticipated changes.

Requirements

A recent version of *Goma* (cvs-repository version dated later than 1 May 2004) is required for this tutorial, along with APREPRO and CUBIT (v9.1 or higher or 8.0-beta). MAPVAR may also be needed later if remeshing is found to be necessary. The linear solver used is UMFPACK, which must be linked in to *Goma*. In order to run the example with user-defined continuation conditions (input file **`tpuf.input`**), it will also be necessary to recompile *Goma* with the included version of **`user_continuation.c`**, but this does not require **`makefile`** changes. *Goma*'s memory requirement is about 1500MB for this problem. The input and output solution files can be viewed using BLOT, MUSTAFA, or ENSIGHT. Below we present one way to attain a steady state solution at realistic parameter values (e.g. webspeed 1 m/s, viscosity 1 P, density 1 g/cm^3, gap between die face and undeformed shell surface 200 microns, upstream wrap angle 20 degrees, downstream wrap angle10 degrees). We realize that there are probably several ways to achieve a solution at these parameter values, and even that there might be a better way than the one proposed. If you discover any procedure that expedites this process, please pass the information back to us.

User Inputs and Preliminary Steps

The template is set up such that all user inputs are provided in the input table near the top of file "**`slot.geom`**," which was discussed above. These data will be transferred to the Goma input and material files via Aprepro by using the -a flag on the *Goma* command line. Cubit also reads this file through Aprepro for use in generating the mesh geometry. However, there is an additional step which is not automated yet and presently requires manual handling:

Goma will use two external data files "**`upstream_land.dat`**" and "**`downstream_land.dat`**" to locate the solid boundaries of the two dies with **`GD_TABLE`** boundary conditions. These data files must be generated beforehand. Two Perl scripts are provided for generating these files. However, Perl cannot

use Aprepro, and the user must change the scripts manually to correspond to the desired geometry. Further instructions on doing this will be provided soon.

Once the Perl scripts have been edited, they are run as follows:

```
perl upstream_land.pl > upstream_land.dat
perl downstream_land.pl > downstream_land.dat
```

Then, the necessary die geometry data will be available to *Goma*. It is worth mentioning that the die lands and faces are connected by circular arcs, and the upstream land is not flat, hence the need for a more sophisticated approach in generating geometry boundary conditions for *Goma*.

Solution procedures

Several features of this problem make it far more difficult to obtain a steady state solution at realistic parameter values than the slot coating example problem discussed in GT-002.1, GT-10.1, and GT-
015.0. First and foremost is the difficulty caused by the flexible web that is simply supported at distant upstream and downstream points. Slot coating test problems heretofore incorporated rigid substrates, or deformable substrates on a rigid base (cf. GT-002.1). In any case, the freely suspended web makes the problem far more nonlinear. Also complicating this problem are the large aspect ratios that result from the operating space and the die dimensions used. To be faithful to the true web mechanics and to account accurately for the web-hydrodynamic interactions we need to model the effect of the idler position (cf. figure above) at their true placements upstream and downstream. Moreover, on this particular design, the feed slot is far wider than the coating gap during operation (about 10:1). These geometric features require substantial grid resolution. Moreover, in this model problem we allow the menisci to locate anywhere on the die faces. To accommodate this movement, we have rounded the corners between die lands, feed slot walls, and die lips. Althought these features allow for contact line migration around these corners under specific operating conditions, we find that the Goma's Newton-based solver needs to be highly relaxed when such an event occurs. Where appropriate, we point this out in the startup procedure below.

First, edit the input table as necessary for the desired parameters as indicated above. It is HIGHLY recommended that you start with the parameters in **`slot.geom.base`**, as those parameters are the ones from which the following start-up procedure used. Perform the preliminary steps that generate the die-lip geometry (described in the previous section), then generate the initial mesh:

```
cubit -batch -nojournal slot.jou
```

This will create the file "**`slot.exoII`**", which will be used for the first *Goma* step; thereafter, the output neutral file from each step, viz. **`soln.dat`**, can be used as the input for the next step by copying **`soln.dat`** into **`contin.dat`**.

The mesh can be viewed if desired as follows:

```
blot slot.exoII
```

The first computational step is to get a solution on the fixed mesh. The command for this step is:

```
goma -a -i step1.input.
cp soln.dat contin.dat
cp soln.dat contin.dat.1
```

(this last step is just good practice so as to save the fixed grid solution for future use if you need to revert).

Note that we are assuming this step will converge with full Newton iteration. However, if you change the parameters in **slot.geom**, you may not be able to get a solution for the fixed mesh case. It is also noteworthy that the following operating conditions and thermophysical properties were used in order to facilitate convergence of step 2, which involves the release of the downstream meniscus (sset 40):

```
# MATERIAL PROPERTIES:
#--NOTE TO SELF:  Check these units!!!!
# Density [g/mm^3]{density = 9.3e-4}
# Viscosity [g/mm-s]{viscosity = 0.6}
# Surface tension [dyn/mm]{surface_tension = 125.0}
# Navier slip coefficient{beta = 0.001}
# Web bending stiffness{web_E = 160.0}

# OPERATING PARAMETERS:
# Web speed [mm/s]{webspeed = -1000.0}
# Upstream idler tension [Pa]{web_tension = 1.0e+10}
# Upstream angle (from horizontal){web_angle_U = 0.0}
# Upstream angle at remesh step {web_angle_U_at_remesh = 0}
# Downstream angle (from horizontal){web_angle_D = 0.0}
# Downstream angle at remesh{web_angle_D_at_remesh = 0}
```

It is noteworthy that we desire to start this problem up at considerably larger webspeeds and viscosiy, e.g. 2500 mm/s and 60 P (or 6 g/mm-s), and even web tension (target is 1e+08), but the downstream meniscus in this case wants to receed far into the downstream lip region due to the drastic pressure drop under the downstream land. We will continue in these parameters later to achieve the desired operating state.

Next, the downstream meniscus is released. Execute:

```
goma -a -i step2.input -r 0.1 -n 20
cp soln.dat contin.dat
goma -a -i step2.input -r 0.1 -n 10
cp soln.dat contin.dat
goma -a -i step2.input -n 7
cp soln.dat contin.dat
cp contin.dat contin.dat.2
```

It is disconcerting that it takes so many iterations to achieve this first step. As of the time of this writing we find that the downstream static contact line has trouble going around the curved corner. We are actively trying to resolve this problem.

In any case, the next step is to release the upstream menisucs and the dynamic (fluid/web) contact line. To facilitate this step and the steps which follow it we apply a backpressure to that mensicus such that when it is released it will stay in roughly the same spot. That backpressure happens to be **{- 1.365e6}**, which was determined by the pressure at the upstream meniscus from step 2 (using **blot**). We will later reduce that backpressure to zero, which is the true operating state. This trick is detailed

in GT-002.1 and can also be accomplished with an augmenting condition (more later). In any case the necessary changes for releasing this meniscus are contained in **step3.input**, viz. :

```
goma -a -i step3.input -n 10 -r 0.1
cp soln.dat contin.dat
goma -a -i step3.input -n 7
cp soln.dat contin.dat
cp contin.dat contin.dat.3
```

This step may require significant relaxation depending on the problem parameters.

Next, the web is released, viz. **sset 60**. If you use the unix "**diff**'" utility to examine the differences between **step3.input** and **step4.input**, you will see what was changed. Run:

```
goma -a -i step4.input -n 20 -r 0.1
cp soln.dat contin.dat
goma -a -i step4.input -n 7
cp soln.dat contin.dat
cp contin.dat contin.dat.4
```

A key part to the web release is substantial web tension (set at 1.e10). We will take a continuation step to reduce this tension before we start changing some of the other operating parameters. In slot.geom change the web tension to 1.e9, from 1.e10, e.g.

```
# Upstream idler tension [Pa]{web_tension = 1.0e+9}
```

Run goma with full Newton:

```
goma -a -i step4.input -n 7
cp soln.dat contin.dat
cp soln.dat contin.dat.4b
```

Again, lets reduce the tension to 5.e8:

```
# Upstream idler tension [Pa]{web_tension = 5.e8}
goma -a -i step4.input -n 7
cp soln.dat contin.dat
cp soln.dat contin.dat.4c
```

Note that if you reduce the web tension to 1.e8 the downstream separation line wants to recede into the gap suddenly and you loose convergence. So we will keep the tension here right now and start to increase the web angles to stabilize the meniscus.

```
cp contin.dat.4c contin.dat.5
```

Step 5 deploys basically the same input deck as step 4, (in fact **contin.dat.4c** is the same as **contin.dat.5**) but contains some additional BCs that allow for downstream and upstream idler angle changes. Step 5 will involve manual continuation towards the target set up operating conditions. We will work on lowering the web tension (which forces both menisci inward making the bead extent smaller), and raising the web speed (which has the same effect), and increasing the web angle, which has the opposite effect. We will play these together. Remember our target is about

3000 mm/s webspeed and a web tension of about 1.e8. To achieve the other target parameters we still have web idler angles, etc., to adjust. Before we reduce the idler tension more and raise web speed, lets try to control the menisci by moving the idlers up. We will start with the downstream idler, changing the angle to 0.1 degree:

```
# Downstream angle (from horizontal){web_angle_D = 0.1}
```

Run Goma with the following-Newton sequence, viz.

```
goma -a -i step5.input -r 0.1 -n 20
cp soln.dat contin.dat
goma -a -i step5.input -n 7
cp soln.dat contin.dat.5b
```

This step results in pulling the upstream meniscus inward and pushing the downstream meniscus a little higher on the die face. We do know that raising the upstream angle will squeeze the upstream meniscus out drastically, so we will do some more continuation on the downstream angle. Try

```
# Downstream angle (from horizontal){web_angle_D = 0.2}

goma -a -i step5.input -r 0.1 -n 15
cp soln.dat contin.dat
goma -a -i step5.input -n 7
cp soln.dat contin.dat
cp soln.dat contin.dat.5c
```

Now we will counterbalance the inward movement of the upstream meniscus with the upstream web angle, which should push it out. Make the following *double* change step:

```
# Downstream angle (from horizontal){web_angle_D = 0.3}
# Upstream angle (from horizontal){web_angle_U = 0.1}

goma -a -i step5.input -r 0.1 -n 15
cp soln.dat contin.dat
goma -a -i step5.input -n 7
cp soln.dat contin.dat
cp soln.dat contin.dat.5d
```

Notice that even a slight change in upstream angle, together with the same 0.1 deg increment in downstream angle leads to a drastic relocation of the upstream meniscus. To counter this lets increase the webspeed. Here we will be a little bold:

```
# Web speed [mm/s]{webspeed = -2000.0}
goma -a -i step5.input -r 0.1 -n 20
cp soln.dat contin.dat
goma -a -i step5.input -n 7
cp soln.dat contin.dat
cp soln.dat contin.dat.5e
```

While we are at it, lets crank it up a little more, this time to -2500:

```
# Web speed [mm/s]{webspeed = -2500.0}
goma -a -i step5.input -r 0.1 -n 20
cp soln.dat contin.dat
goma -a -i step5.input -n 7
```

```
cp soln.dat contin.dat
cp soln.dat contin.dat.5f
```

With this higher webspeed the upstream meniscus position is not as sensitive to web angle. Let's work on the web angle again: Upstream angle from 0.1 to 0.2 deg, and then from 0.2 to 0.4 deg., and again to 0.5 deg. Notice that the upstream meniscus position is not as sensitive to these changes.
Raising the downstream angle from 0.3 deg. to 0.4 deg. causes the meniscus on the downstream end to wet further up the die face. In any case, all of these changes were performed with full Newton iteration and the steps which take us to these upstream and downstream angles of 0.5 deg. and 0.4 deg respectively. This solution is saved in **contin.dat.5j**.

With these changes it is time to try to bring the menisci in closer to the feed slot. I think we can do this with the viscosity, since we have to raise it substantially. First we raise by a small amount (from
0.6 g/mm-s to 0.8 g/mm-s to test the sensitivity. This solution was obtained with full Newton iteration and is saved in **contin.dat.5k**. We see that indeed the menisci are drawn inward with this change. We now change the viscosity more substantially, viz. from 0.8 g/mm-s to 1.8 g/mm-s and use some relaxation:

```
#Viscosity[g/mm-s]  {viscosity = 1.8}
goma -a -i step5.input -r 0.1 -n 20
cp soln.dat contin.dat
goma -a -i step5.input -n 7
cp soln.dat contin.dat
cp soln.dat contin.dat.5l
```

Again, because the effect of viscosity is to draw the menisci (static separation lines inward), we counter with increasing web angle. You can increase the upstream angle to 1 deg. with full Newton, and then the upstream angle again to 1.5 deg with the following sequence:

```
# Upstream angle (from horizontal){web_angle_U = 1.5}
goma -a -i step5.input -r 0.1 -n 30
cp soln.dat contin.dat
goma -a -i step5.input -n 7
cp soln.dat contin.dat
cp soln.dat contin.dat.5n
```

With one step you can reduce the ***surface tension*** to the desired value of 60 dyn/cm (from 125 dyn/ cm). You can use full Newton iteration to do this. We saved this solution in **contin.dat.5o**.

Next we take a big step in viscosity (from 1.8 to 3.8:

```
#Viscosity[g/mm-s]  {viscosity = 3.8}
goma -a -i step5.input -r 0.1 -n 20
cp soln.dat contin.dat
goma -a -i step5.input -n 7
cp soln.dat contin.dat
cp soln.dat contin.dat.5p
```

Note that the menisci are drawn in again. To counter this we increase downstream web angle from 0.4 deg to 0.8 deg:

```
# Downstream angle (from horizontal){web_angle_D = 0.8}
goma -a -i step5.input -r 0.1 -n 20
cp soln.dat contin.dat
goma -a -i step5.input -n 7
cp soln.dat contin.dat
cp soln.dat contin.dat.5q
```

Note *that as you increase the downstream web angle from horizontal, it pinches the flow on the downstream land and decreases the pressure drop, thereby pushing out the upstream meniscus. You should monitor the mesh quality on the outflow plane (way over to the left), as for some reason it gets real distorted with downstream angle changes. In fact, this distortion is what leads to the need to remesh (more below).* Again increase the downstream angle a bit more, from 0.8 deg to 1.2 degrees:

```
# Downstream angle (from horizontal){web_angle_D = 1.2}
goma -a -i step5.input -r 0.1 -n 20
cp soln.dat contin.dat
goma -a -i step5.input -n 7
cp soln.dat contin.dat
cp soln.dat contin.dat.5r
```

Now we will reduce the web tension, as you notice that the web is "pinching off" the flow. The purpose here is to try to create a more "remeshable" region and to achieve higher wrap angles. First reduce the tension to 4e+8:

```
# Upstream idler tension [Pa]{web_tension = 4.e+08}
goma -a -i step5.input -r 0.1 -n 20
cp soln.dat contin.dat
goma -a -i step5.input -n 7
cp soln.dat contin.dat
cp soln.dat contin.dat.5s
```

Notice that the flow domain and menisci are very sensitive to this parameter. Take another small step in web tension, to 3.8e+08 Pa, using the same relaxation scheme, and save the solution in **`contin.dat.5t`**. Now we can further increase the web angles. Using 20 or so Newton iterations relaxed at -r 0.1, followed by full Newton, take the following steps:

```
# Downstream angle (from horizontal){web_angle_D = 0.8}
```

to

```
# Downstream angle (from horizontal){web_angle_D = 2.2}
```

and then another step to

```
# Downstream angle (from horizontal){web_angle_D = 2.5}
```

We saved these solutions in **`contin.dat.5u`** and **`contin.dat.5v`**, respectively. Now the upstream angle:

```
# Upstream angle (from horizontal){web_angle_U = 0.5}
```

to

```
# Upstream angle (from horizontal){web_angle_U = 2.0}
```

and then an additional step to

```
# Upstream angle (from horizontal){web_angle_U = 3.0}
```

These solutions are saved in **contin.dat.5w** and **contin.dat.5x**, respectively.

At this point you should **blot** the output file **step5_out.exoII** and zoom in on the outflow plane (far left). Note that the mesh is getting quite distorted there. We need to remesh. Details of this procedure are given in GT-006.3. Here we give a brief review. In your distribution you have a **remesh.jou** file. This file begins by importing the mesh from **step5_out.exoII** and refitting the new geometry, cf. the first few lines of this file:

```
{include(slot.geom)}
nodeset associativity on
set associativity complete on
import free mesh "step5_out.exoII" step last
label surface on
```

This journal file remeshes based on the new geometry, and in fact the rest of the file contains the same meshing commands as **slot.jou**. In any case, run

```
cubit remesh.jou
```

and the new mesh will be built and put into a file called **rem.gen**. Now you need to map your solution that is currently held in **step5_out.exoII** onto this new mesh. To do this we use the SEACAS **MAPVAR** tool. To run **MAPVAR** copy your recipient mesh into **tmp.g**, and your donor mesh into **tmp.e** and run the tool as follows:

```
cp rem.gen tmp.g
cp step5_out.exoII tmp.e
mapvar tmp
mapvar>def 2
mapvar>search 0.1
mapvar>exit
```

A file called **tmp.int** (**int** stands for interpolate) is written from this sequence. You should **blot** this file and contour both components of velocity to make sure all is smooth around corners and menisci. You will notice that **mapvar** does not map the shell variables (viz. Tension **tens** and curvature **k**) on the unwetted portion of the web. This makes it tricky to get a restart.

Now we need to obtain a solution on the new mesh. We have put the necessary changes to the Goma input deck in **step6.input** (see tutorial GT006.3 or use the Unix "**diff**" utility to look at the modifications).

One real important note:

Update the following two aprepro variables in slot.geom to reflect the correct remesh angles:

```
# Upstream angle (from horizontal){web_angle_U = 3.0}
# Upstream angle at remesh step {web_angle_U_at_remesh = 3.0}
# Downstream angle (from horizontal){web_angle_D = 2.5}
# Downstream angle at remesh{web_angle_D_at_remesh = 2.5}
```

These updates are necessary to reset all dirichlet conditions on displacement. Recall that upon a remesh the mesh is "annealed" and the displacements are set back to zero. If you look in slot.geom how these parameters are used, this will all make more sense.

Run **Goma** to get a new solution on the new mesh (but first change the Initial guess card so that it reads the initial guess from the file **tmp.int**, viz.

```
Initial Guess= read_exoII_file tmp.int:
```

and run goma:

```
goma -a -i step6.input -n 30 -r 0.0.06
cp soln.dat contin.dat
```

Change the input deck **step6.input** to take the continuation step off of the neutral file **contin.dat**, viz.

```
Initial Guess= read
```

and rerun Goma

```
goma -a -i step6.input -n 20 -r 0.1
cp soln.dat contin.dat
goma -a -i step6.input -n 7
cp soln.dat contin.dat
cp contin.dat contin.dat.6
```

You've now a solution on the new mesh and you can continue in downstream web angle and other parameters. At this point let's review where we are in **slot.geom**:

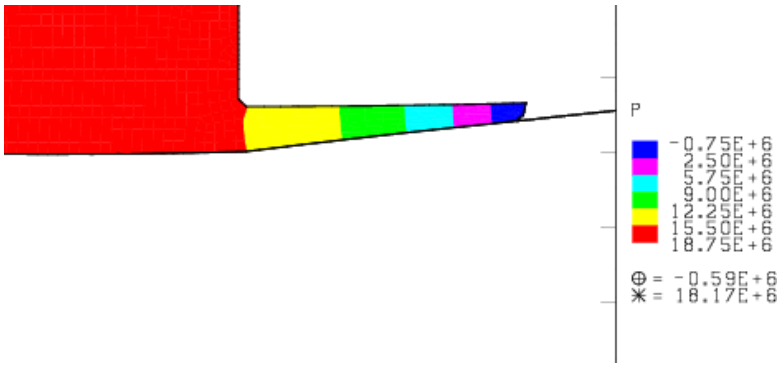
```
# MATERIAL PROPERTIES:
#--NOTE TO SELF:  Check these units!!!!
# Density [g/mm^3]{density = 9.3e-4}
# Viscosity [g/mm-s]{viscosity = 3.8}
# Surface tension [dyn/mm]{surface_tension = 60.0}
# Navier slip coefficient{beta = 0.001}
# Web bending stiffness{web_E = 160.0}

# OPERATING PARAMETERS:
# Web speed [mm/s]{webspeed = -2500.0}
# Upstream idler tension [Pa]{web_tension = 3.8e+8}
# Upstream angle (from horizontal){web_angle_U = 3.0}
# Upstream angle at remesh step {web_angle_U_at_remesh = 3.0}
# Downstream angle (from horizontal){web_angle_D = 2.7}
# Downstream angle at remesh{web_angle_D_at_remesh = 2.7}
```

Recall also that we still have a vacuum pressure applied to the upstream meniscus of **-1.365e6**, which needs to be reduced to zero. The viscosity is within a desired range, as is the surface tension, webspeed, and web tension. To reduce the vacuum to zero, let us increase the downstream wrap

angle a little to pinch the upstream meniscus out. Take three continuation steps at full Newton in the downstream angle, from 3 deg. to 4 deg., and then from 4 deg to 5 deg., and finally to 5.5 deg. These solutions are saved in **`contin.dat.6c`**, **`contin.dat.6d`** and **`contin.dat.6e`**, respectively.

These web-angle changes will help stabilize the upstream meniscus; in fact, so much so that one simple step to reduce the vacuum to zero is all that is required. This step is covered in **`step7.input`**. Differencing **`step7.input`** with **`step6.input`** reveals that only the vacuum pressure on the capillary boundary condition applied to ss70 has been changed. Specifically, the backpressure has been reduced to zero. Before attempting a solution, we suspected that now the meniscus would be less sensitive to backpressure due to the tensioning and idler angles that stabilize the meniscus.
Contouring the pressure reveals that this is the case



The pressure field reveals that a pressure change of 1.36e6 units (g/mm-s^2) will not lead to a major relocation of the upstream meniscus, as the total pressure change along the upstream land is of order 18e6 g/mm-s^2. Based on this we reduce the backpressure to zero in one step, viz.

BC = CAPILLARY SS 70 {surface_tension} {-1.365e6*0.} 0.0

Although we relaxed this solution a little, we didn't need to. The final sequence is

```
goma -a -i step7.input
cp soln.dat contin.dat
cp contin.dat contin.dat.7
```

To quote a now infamous goma phrasology: Use and enjoy your new solution!

The real goal is to determine the bounds of stability of all operating states that shed light on a range of operability. This is described next.

Coating Bead Stability

The linear stability analysis capability in Goma is brought to bear on this aspect of the problem. The user should consult SAND2000-2465 (Advanced Capability Manual) and GT-023.0 (LOCA/Goma tutorial) for further information.

The changes required to the Goma input files to access the linear stability capability are given in **`step8.input`**. Before we test the final steady state for stability, it might be instructive to test an

earlier steady state that corresponds to a flat web and a back-pressure stabilized meniscus. Actually, if we return to case corresponding to **step3.input** the web has not even been released. In that file you will find the following cards associated with Linear Stability Analysis:

```
Linear Stability                        = no

------------------------------------------------------------
               Eigensolver Specifications
------------------------------------------------------------
#Eigen Algorithm                    = si
Eigen Number of modes               = 10
Eigen Record modes                  = 5
Eigen Size of Krylov subspace       = 50
Eigen Initial Shifts                = 100.0 1000.0 0.0 0.0
```

In this case Goma's native eigenvalue solver is invoked and the 10 leading modes are pursued. To run step 3 with stability analysis, change the **Linear Stability = no** card to "**yes**" in **step3.input** and perform the following:

```
cp slot.geom.base slot.geom (Caution: you might want to save your slot.geom
file under another name).
cp contin.dat.3 contin.dat
goma -a -i step3.input
```

The eigenvalues corresponding to the 10 leading modes ("leading" defined by those with the largest real part) are

```
Eigensolver required 80 iterations.
 Found 10 converged eigenvalues.
 Leading Eigenvalue  = -9.091509e+02-0.000000e+00 i RES =  8.774236e-46
    Real            Imag            RES
 -9.091509e+02 -0.000000e+00 i   8.774236e-46
 -1.096160e+03 +3.824105e+02 i   7.136999e-41
 -1.096160e+03 -3.824105e+02 i   7.136999e-41
 -1.064780e+03 +1.128862e+03 i   1.117536e-35
 -1.064780e+03 -1.128862e+03 i   1.117536e-35
 -1.035557e+03 +1.848226e+03 i   1.001471e-29
 -1.035557e+03 -1.848226e+03 i   1.001471e-29
 -2.288632e+03 -0.000000e+00 i   1.351499e-28
 -1.012011e+03 +2.558144e+03 i   3.102089e-24
 -1.012011e+03 -2.558144e+03 i   3.102089e-24
```

We can see that this state is stable as all real-parts are negative. Proceeding with step8.input (viz. copy **contin.dat.6e** to **contin.dat** and run **step8.input** with the **Linear Stability** card set to "yes" and **slot.geom** params restored to the final state above), we see the following set of leading modes:

```
Assembling LSA Jacobian ...
Assembling LSA Mass matrix ...
    CAYLEY: sigma, mu, ncv = -100 -700 480

        Eigensolver Initial Guess Generation
        Requiring 2 extra orders resid reduction: 7.10156e-13

    Before poleze: sigma and mu = -100  -700
```

```
    After  poleze: sigma and mu = -100  -700

    360 converged of 470 candidate eigenvalues found
Printing eigenvector pair for complex eigenvalues:  -1469.91 +- 80488.1 i

 Ritz values (Real,Imag) and direct residuals
 --------------------------------------------
              Col   1       Col   2       Col   3
  Row   1:   -1.46991E+03  -8.04881E+04   1.90038E+01
  Row   2:   -1.46991E+03   Assembling LSA Jacobian ...
Assembling LSA Mass matrix ...
    CAYLEY: sigma, mu, ncv = -100 -700 480

        Eigensolver Initial Guess Generation
        Requiring 2 extra orders resid reduction: 7.10156e-13

    Before poleze: sigma and mu = -100  -700
    After  poleze: sigma and mu = -100  -700

    360 converged of 470 candidate eigenvalues found
Printing eigenvector pair for complex eigenvalues:  -1469.91 +- 80488.1 i

 Ritz values (Real,Imag) and direct residuals
 --------------------------------------------
              Col   1       Col   2       Col   3
  Row   1:   -1.46991E+03  -8.04881E+04   1.90038E+01
  Row   2:   -1.46991E+03   Assembling LSA Jacobian ...
Assembling LSA Mass matrix ...
    CAYLEY: sigma, mu, ncv = -100 -700 480

        Eigensolver Initial Guess Generation
        Requiring 2 extra orders resid reduction: 7.10156e-13

    Before poleze: sigma and mu = -100  -700
    After  poleze: sigma and mu = -100  -700

    360 converged of 470 candidate eigenvalues found
Printing eigenvector pair for complex eigenvalues:  -1469.91 +- 80488.1 i

 Ritz values (Real,Imag) and direct residuals
 --------------------------------------------
              Col   1       Col   2       Col   3
  Row   1:   -1.46991E+03   -8.04881E+04    1.90038E+01
  Row   2:   -1.46991E+03    8.04881E+04    1.90038E+01
  Row   3:   -1.47672E+03   -7.91721E+04    1.76170E+01
  Row   4:   -1.47672E+03    7.91721E+04    1.76170E+01
  Row   5:   -1.48207E+03   -8.17949E+04    1.93276E+01
  Row   6:   -1.48207E+03    8.17949E+04    1.93276E+01
8.04881E+04    1.90038E+01
  Row   3:   -1.47672E+03   -7.91721E+04    1.76170E+01
  Row   4:   -1.47672E+03    7.91721E+04    1.76170E+01
  Row   5:   -1.48207E+03   -8.17949E+04    1.93276E+01
  Row   6:   -1.48207E+03    8.17949E+04    1.93276E+01
8.04881E+04    1.90038E+01
```

```
Row    3:   -1.47672E+03   -7.91721E+04    1.76170E+01
Row    4:   -1.47672E+03    7.91721E+04    1.76170E+01
Row    5:   -1.48207E+03   -8.17949E+04    1.93276E+01
Row    6:   -1.48207E+03    8.17949E+04    1.93276E+01
```

Again, none of the leading eigenvalue in this case has a positive real part. It is important to note that to run the linear stability capability in Goma for this final case required substantially different options:

```
------------------------------------------------------------
               Eigensolver Specifications
------------------------------------------------------------
Eigen Algorithm                  = cayley
Eigen Cayley Sigma               = -100.0
Eigen Cayley Mu                  = -700.0
Eigen Number of modes            = 6
Eigen Record modes               = 2
Eigen Size of Krylov subspace    = 480
Eigen Matrix Output              = no
Eigen Relative tolerance         = 1.0e-12
Eigen Linear Solver tolerance    = 1.0e-6
Eigenvalue output frequency      = 1
Eigenvector output frequency     = 1
```

The main difference is that we are using ARPACK with these choices, and not Eggroll (Goma's native eigenvalue solver). Interpreting the results and deploying automated continuation to flush out operability windows will be the subject of a future effort.

Final Result

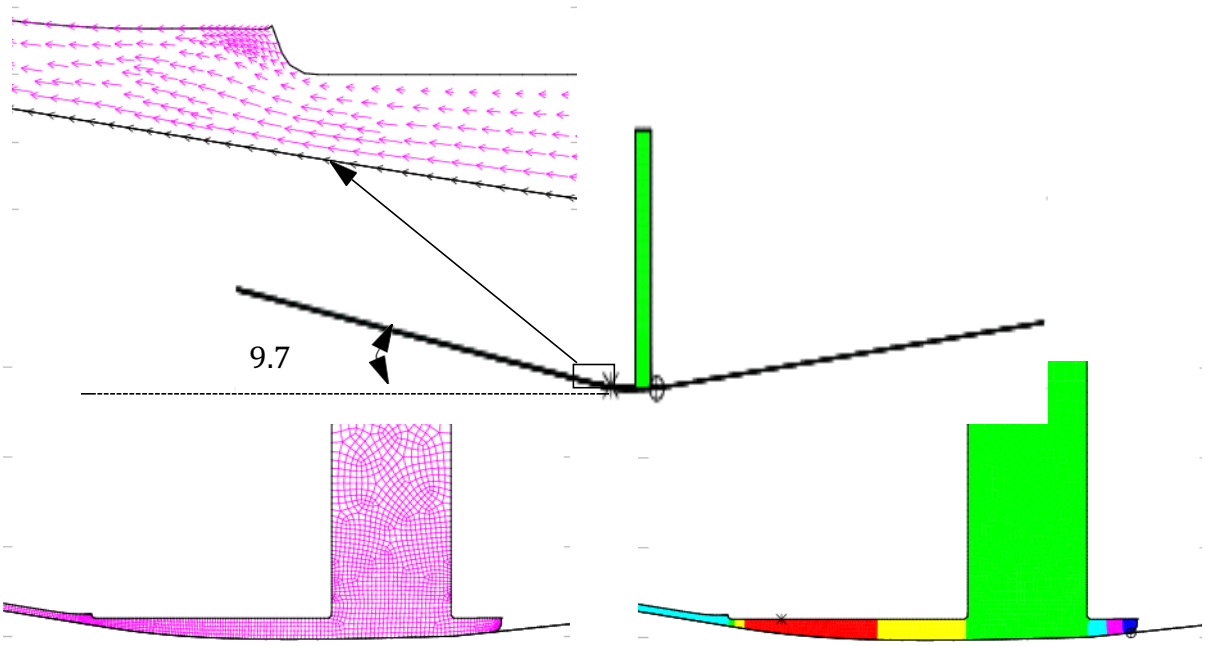
Shown in this picture is a sample solution at large web angles.

Know Problems needing Resolution:

-Remeshing/Remapping is functional with CUBIT 9.0 or later release, but the mapvar mapping of the shell variables (viz. tension and curvature) on the unwetted portion of the web are not mapped. This is an issue on restart, but with relaxation a remapped solution can be recovered.

-Mesh distortion on outflow film down web at high wrap angles. Zoom on on the outflow plane often during a continuation process. The mesh at the outflow plane gets quite distorted at high wrap angles, necessitating a remesh.

-Kinematic_Colloc smooths upstream free surface, but enough?


9.7