**SANDIA REPORT**
SAND2015-0652
Unlimited Release
Printed Jan 2014

# Advanced Capabilities in GOMA 6.0 - Augmenting Conditions, Automatic Continuation and Linear Stability Analysis

P. Randall Schunk, Duane A. Labreche, Edward D. Wilkes, Matthew M. Hopkins and Amy C. Sun

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.



Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.



Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: http://www.osti.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd.
Springfield, VA 22161

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.fedworld.gov
Online order: http://www.ntis.gov/help/ordermethods.asp?loc=7-4-0#online

SAND2015-0652
Released February 2015

# Advanced Capabilities in GOMA 6.0 - Augmenting Conditions, Automatic Continuation and Linear Stabilty Analysis

P. Randall Schunk, Duane A. Labreche, Edward D. Wilkes, Matthew M. Hopkins and Amy C. Sun

**Engineering Sciences Center**
**Sandia National Laboratories**
**P. O. Box 5800**
**Albuquerque, New Mexico 87185-0827**

## Abstract

This document describes the form and use of three supplemental capabilities added to *Goma* during 1998 -- augmenting conditions, automatic continuation and linear stability analysis. Augmenting conditions allow the addition of constraints and auxiliary conditions which describe the relationship between unknowns, boundary conditions, material properties and post-processing extracted quantities. Automatic continuation refers to a family of algorithms (zeroth and first order here, single and multi-parameter) that allow tracking steady-state solution paths as material parameters or boundary conditions are varied. The stability analysis capability in *Goma* uses the method of small disturbances and superposition of normal modes to test the stability of a steady-state flow, i.e., it determines if the disturbance grows or decays in time.

**Acknowledgment**

Development of *Goma* was funded in part by the Engineering Science Research Foundation, Laboratory Directed Research and Development, and the Basic Energy Science Program of the DOE. Credit is due to Tom Baer of Sandia for re-packaging the augmenting conditions into standard and user routines. The authors would like to thank Robert Secor of 3M for testing and debugging the continuation and hunting routines, Allen Roach and Mike Kanouff for their review comments, and Randy Schunk, Sharon Healy, JoAnn Keicher and Mindy Morgan for their assistance in completing the initial version of this report.

# 1 Introduction

*Goma* is a two- or three-dimensional finite element program specialized for the analysis of manufacturing flows and related processes that involve one or more transport fields, i.e., any combination of heat, mass, momentum (solid and fluid) and species transport. *Goma* is especially suited to problems with free or moving boundaries between dissimilar materials or phases. This report is a supplement to the *Goma* Users Manual (Schunk, et. al., 2013) and is an update to the previous report (SAND2006-7304) documenting the advanced capabilities of *Goma* 5.0. That report described three capabilities - augmenting conditions, automated continuation and linear stability analysis - which were added to the *Goma* software during the time period 1999-2000. This report describes the same capabilities, each still presented in a separate chapter with one or more example problems to illustrate the proper usage of the added capability. In all cases, the capabilities have been expanded since their original implementation in the code and the presentation has been supplemented as necessary. But in addition, the presentation of the *Goma* input records has been expanded (Schunk, et. al., 2013), and this has been carried over to the input records described in each chapter of this report. This report, as its predecessor, should be used in conjunction with the companion *Goma* Version 6.0 Users Manual (SAND2013-1844) to which references are made.

## 1.1 Additional Sections in *Goma* Input File

Each added capability requires the addition of an optional section to the *Goma* input file to define the parameters of the capability. The conventional order of sections in the input file is:

**File Specifications**
**General Specifications**
**Time Integration Specifications**
**Solver Specifications**
**Boundary Condition Specifications**
**Rotation Specification**
**Problem Description**
**Post Processing Specifications**
**Post Processing Fluxes**

Each advanced capability section may be added between the **Time Integration** and **Boundary Condition Section**; its position may interchange with the **Solver Section**, coming either before or after.

***Time Integration Specifications***
***Solver Specifications (*or place below new sections*)***
    **Augmenting Conditions Specifications**
    **Continuation Specifications**
    **Hunting Specifications**
    **Eigensolver Specifications**
***Solver Specifications (*or place above new sections*)***

***Boundary Condition Specifications***

## 1.2 *Goma* Simulations in this Report

Subdirectories for all simulations presented in Chapters 2, 3 and 4 are provided in the examples subdirectory of the advanced_capabilities directory of the Goma software distribution. Results presented in this report were calculated with the 10/1/2006 version of *Goma* 5.0.

# 2 Augmenting Conditions

The augmenting condition capability in *Goma* enables the addition of supplemental constraints and auxiliary conditions to the problem via standardized and user-defined function(s) (in files *mm_augc_util.c* and *user_ac.c* respectively). These supplemental constraints (equations) are used to define relationships between unknowns, boundary condition data, material properties, and virtually any other extracted quantity from *Goma* post processing routines. Examples include:

**Goma unknowns:**
Specify a relationship between two or more unknowns, e.g. the distance between two nodes remains fixed at a specified value.

**Material properties:**
Solve for a material property value such that another condition is met, e.g., find the liquid viscosity at which the velocity gradient at a wall becomes zero, or to maintain a relationship between the temperature, pressure, and density.

**Boundary condition floats:**
Constrain the problem so that a specified relationship always exists between boundary condition data floats (which may not necessarily be Dirichlet conditions), i.e., the parameters of a boundary condition specification, or between parameters of different boundary conditions.

**Postprocessing constraints:**
Postprocessing of solutions can define auxiliary variables that allow feedback of integrated results into the solution as constraints. Examples include integral constraints on surface forces, heat flux through a wall, volume flux and species flux.

**Other constraints:**
Examples include constraints on volume, mass and component mass.

**Mixed constraints:**
Augmenting conditions can invoke any combination of the above types in a single auxiliary constraint. Thus, a relationship can be specified between any of the *Goma* unknowns, material properties, boundary condition floats, and postprocessed variables.

The *Goma* augmenting condition capability handles all of the above types. Providing the variable of interest is passed to the augmenting condition user-definition routine (described below) via a data structure or argument, the augmenting condition can be applied. The number of augmenting conditions is unlimited but must be set by the user. An important consideration in the addition of augmenting conditions however is increased computational cost (time and storage) as the number

of augmenting conditions increases. There are no limits on the density (number of non-zeroes) of the augmenting condition residual equation or the level of interaction between the unknowns.

Two pieces of information must be specified to add an augmenting condition to a *Goma* analysis: identification of the AC and its parameters (Section 2.3) and the addition of new unknowns to the *Goma* system of equations (Section 2.4). This latter step consists of creating a function which specifies the relationship between the variables of interest. The extra unknowns must be equal in number to the augmenting conditions and be one of four types: boundary condition data floats, material properties, volume constraints or flux constraints. Volume and flux augmenting condition standardized relationships have already been defined in function **std_aug_cond** (file *mm_augc_util.c*); the user must specify the functional relationship for boundary conditions and material properties in the **user_aug_cond_residuals** function (file *user_ac.c).*

## 2.1 Augmented System Solution

Adding extra equations to the original *Goma* system of equations yields an augmented set of equations. In *Goma*, the augmented system of equations is solved by a bordering algorithm (Chan and Resasco, 1986). This method is essentially a block elimination of the augmented equations.

The original *Goma* system has the advantage that it is sparse, usually with small bandwidth. This leads to computational benefits (speed and storage) that can be realized when using direct frontal and iterative solvers. The bordered algorithm is suitable for the augmented system of equations because the decomposition of the original *Goma* set of equations is already required; one additional, but much smaller, decomposition for the augmented equations adds little calculational cost compared to the original system decomposition. If there are N augmenting conditions, the bordered algorithm solves the augmented system of equations with one decomposition and N+1 back-substitutions of the *Goma* original system, plus one NxN system solution. One further advantage of the bordered algorithm is that the augmenting equations can have much wider bandwidth than the original system without substantial growth in the computer time and memory for decomposition.

In the case where the original system is singular but the augmented system is not, the bordered algorithm will fail to solve the system. An instance in which this occurs is the tracking of a turning point in a parameter. Here, the original *Goma* set of equations is singular while the augmented system is not.

## 2.2 Required Specifications in the *Goma* Input File

A new section must be added to the *Goma* input file to identify the new unknowns and other needed data. This section is required only when augmenting conditions are present in the numerical model and has the following form:

```
------------------------------------------------------------
                Augmenting Condition Specifications
------------------------------------------------------------
Number of augmenting conditions  = -1
AC = BC 29 1
AC = MT 4 1300
END OF AC
```

## 2.3 Augmenting Condition Specification

Augmenting conditions (AC) are one of four types: boundary condition (BC), material property (MT), volume constraint (VC) or flux condition (FC). The AC specification identifies which of these four types is active and the location at which it is applied. The parameter(s) of the specification are also required. The form of the BC and MT specifications are similar and are given in Section 2.3.2; the form of the VC and FC specifications are different and require more parameters as described in Sections 2.3.3 and 2.3.4, respectively.

### 2.3.1 Augmenting Conditions Initial Guess

| Augmenting Conditions Initial Guess = {read \| other} |
|---|

#### Description/Usage

This card is used to direct the specification of the initial guess for values of the augmenting condition parameters.

| | |
|---|---|
| **read** | Initial guesses are read from the ASCII file identified on the *GUESS file* card. |
| ***other*** | Any *other* string that **is not *read***, or if this card is missing altogether, the initial guesses for the augmenting condition values will be obtained from the input deck itself or the material file. |

Note that the computed values of the augmenting conditions are automatically written at the end of the *SOLN file*, so it is usually necessary when employing this option to copy the *SOLN file* to the *GUESS file* before restarting.

#### Examples

In the following example, the initial guesses for the augmenting condition values will be obtained from the input deck itself or the material file.

```
Augmenting Conditions Initial Guess = none
```

### Technical Discussion

No discussion.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 2.3.2 Number of augmenting conditions

| Number of augmenting conditions = <integer> |
|---|

### Description/Usage

This card allows the user to specify the number of augmenting condition cards to be read. This card must be present in order to run a problem employing augmenting conditions.

Description of the input parameter is as follows:

| | |
|---|---|
| <integer> | *N*, an integer parameter that is either the number of AC cards to be read or -1. |

*Goma* will try to read *N* cards that start with: *AC* = between this card and a line having only the string *END OF AC*. If *Goma* finds fewer than *N* cards before encountering this string, it will stop with an error. More than *N* cards is fine; the excess are ignored. If *N* is -1, *Goma* will read all the augmenting condition cards between this card and the *END OF AC* card.

### Examples

This example shows a common usage for this card:

```
Number of augmenting conditions = -1
AC = FC  1 0 0 VOLUME_FLUX 4 {-PI}
```

```
AC = FC   1 1 0 FORCE_TANGENT1 3 {3*5.0}
END OF AC
```

In this example, *Goma* will read and execute two different augmenting conditions.

### Technical Discussion

When doing arc length continuation with augmenting conditions, one additional AC record will be automatically created so the number of reported augmenting conditions will be incremented to N+1. In this case, N is still the number of AC cards to be read, which would not include this additional AC (for which there is no input card).

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 2.3.3 AC (*Boundary Condition*)

| AC = BC <bc_id> <integer> <float_list> |
|---|

### Description/Usage

This type of augmenting condition allows the user to connect an additional constraint equation (user-supplied) to a specific boundary condition float parameter which is allowed to vary as an additional degree of freedom during the solution process. The constraint equation is added to the function `user_aug_cond_residuals` in the file *user_ac.c*. A discussion of this function specification and examples is supplied in later sections of this manual.

A description of the card syntax follows:

| | |
|---|---|
| **BC** | A mandatory string indicating that the augmenting condition is attached to a boundary condition parameter. |

| | |
|---|---|
| <bc_id> | An integer parameter identifying the boundary condition index whose parameter is going to be used as the additional unknown. The first index is zero, starting from the first read boundary condition in the input file, and proceeding sequentially upwards. The Technical Discussion describes a method for automatically determining <bc_id>. |
| <integer> | A parameter identifying the index of the float parameter that is to be varied on the boundary condition specification identified by <bc_id>. The leftmost float value is assigned index zero and the index increases sequentially left to right. |
| <float_list> | A list of float parameters that can be used in `user_aug_cond_residuals` to evaluate the augmenting conditions. They are stored in sequence in the array `augc[i].DataFlt`. |

## Examples

For the following set of boundary condition cards:

```
Number of BC = -1
BC = U NS 10 0.0
BC = V NS 10 0.0
BC = W NS 10 0.0 1.0
```

an example augmenting condition card of type BC is:

```
AC = BC 2 0 1.0 0.25
```

The augmenting condition card attaches the (unspecified, in this example) constraint to the first float parameter on the BC = W NS 10 card. Note that there are two float parameters on this card - the first is the specified z-velocity component on node set 10 and the second parameter is required when a Dirichlet condition is attached to an augmenting equation. Ordinarily, Dirichlet conditions are applied by direct substitution, but when the second parameter is present, the condition is included as a residual equation along with all the other residual equations. Naturally, it is this latter form that should be used when an augmenting condition is attached to the boundary condition. The two float parameters supplied on the card can be used in `user_aug_cond_residuals` as the variables `augc[0].DataFlt[0] = 1.0, augc[0].DataFlt[1] = 0.25`.

## Technical Discussion

- The function `user_aug_cond_residuals` is passed an integer index iAC. This is the index of each AC card in order of its appearance in the input deck (with the zero being assigned to the first AC card that appears). The arbitrary float

parameters supplied on the card in <float_list> can be accessed in this function in the array `augc[iAC].DataFlt`. The variable `DataFlt[0]` corresponds to the first float parameter in <float_list> and so on.

- Often the augmenting condition constraint written in function `user_aug_cond_residuals` must make use of the values of nodal unknowns. These can be determined using the function `Index_Solution` as follows:

  The boundary condition index of any boundary condition can be found with the following feature. The NS or SS designator string on the boundary condition in question should be changed (temporarily) to NC or SC, respectively. Then *Goma* should be run with the arguments as follows:

  ```
  goma -i input_file -a -bc_list
  ```

  This invokes a code fork that does not run the problem but analyses the input deck and prints out the index of the boundary conditions flagged by the previous procedure. The values of these indices can be used directly in the augmenting condition cards for <bc_id>

- The augmenting condition constraints are additional residual equations solved simultaneously with the other residual equations associated with the nodal degrees of freedom. The unknown degrees of freedom associated with these equations are the boundary condition float values identified on each augmenting condition card. The new equations also introduce a different structure to the matrix being solved. The formerly sparse, banded Jacobian matrix has now been augmented by rows and columns on its periphery that are potentially populated. This requires a bordered matrix algorithm for the solution of this type of matrix. A consequence of this algorithm is that the residual and update norms of the augmenting conditions are computed separately. The user will see these norms appear as a second row of output at each and every iteration. If these are missing, the augmented system is not being solved.

- Note that the value of the augmenting condition parameter values can be displayed after each iteration by setting *Debug = 1* in the input deck.

## Theory

No Theory.

## FAQs

No FAQs.

### References

No References.

## 2.3.4 AC (*Material property*)

| AC = MT <mat_id> <material_prop_tag> <float_data_list> |
|---|

### Description/Usage

This augmenting condition type attaches a material property (viscosity, thermal conductivity, surface tension, etc.) to an augmenting constraint. The value of the material property value is permitted to vary as any other degree of freedom in order that the user-supplied augmenting condition is satisfied.

Definitions of the input parameters are as follows:

| | |
|---|---|
| MT | A string designator identifying the AC card as a material property type. |
| <mat_id> | An integer parameter identifying the material whose property is being varied. |
| <material_prop_tag> | An integer parameter that associates an integer tag with a type of material property. The material property value designated by this tag will be varied in the solution process. The following tables show the association between material property and integer tag. |
| <float_data_list> | A list of float parameters that can be used in `user_aug_cond_residuals` to evaluate the augmenting conditions. They are stored in sequence in the array augc[i].DataFlt. |

| GENERAL PHYSICAL PROPERTIES | TAG |
|---|---|
| THERMAL_CONDUCTIVITY | 1100 |
| ELECTRICAL_CONDUCTIVITY | 1200 |
| VISCOSITY | 1300 |

| **GENERAL PHYSICAL PROPERTIES** | **TAG** |
|---|---|
| SURFACE_TENSION | 1400 |
| HEAT_CAPACITY | 1500 |
| VOLUME_EXPANSION | 1600 |
| DENSITY | 1700 |
| POROSITY | 1800 |
| PERMEABILITY | 1900 |
| REL_GAS_PERM | 2000 |
| REL_LIQ_PERM | 2100 |
| SATURATION | 2200 |
| MELTING_POINT_LIQUIDUS | 2500 |
| MELTING_POINT_SOLIDUS | 2600 |
| FLOWINGLIQUID_VISCOSITY | 2700 |

| Generalized Newtonian Models | TAG |
|---|---|
| MU0 | 4000 |
| NEXP | 4100 |
| MUINF | 4200 |
| LAM | 4300 |
| AEXP | 4400 |
| ATEXP | 4500 |
| WLFC2 | 4550 |
| TAU_Y | 4600 |
| FEXP | 4610 |
| MAXPACK | 4800 |
| FICKDIFF_X | 4810 |
| FICKDIFF_Y | 4820 |

| Viscoelastic Models | TAG |
|---|---|
| TIME_CONST | 5000[a] |
| WT_FUNC | 5100[a] |
| ALPHA | 5200[a] |
| PTT_XI | 5300[a] |
| PTT_EPS | 5400[a] |

a.This is the TAG_ID for the first viscoelastic mode; subsequent mode TAG_IDs are incremented by 1, e.g., TAGC_TIME_CONST(mode i) = TAGC_TIME_CONST + i

| Solid Elastic Models | TAG |
|---|---|
| LAME_MU | 6000 |

| Solid Elastic Models | TAG |
|---|---|
| LAME_MU_CONTACT_LINE_G0 | 6001 |
| LAME_MU_CONTACT_LINE_G1 | 6002 |
| LAME_MU_CONTACT_LINE_R0 | 6003 |
| LAME_LAMBDA | 6100 |
| CONV_LAG_VELX | 6201 |
| CONV_LAG_VELY | 6202 |
| CONV_LAG_VELZ | 6203 |
| CONV_LAG_ROTRATE | 6221 |
| CONV_LAG_ROT_X0 | 6222 |
| CONV_LAG_ROT_Y0 | 6223 |
| CONV_LAG_ROT_Z0 | 6224 |
| RS_LAME_MU | 6300 |
| RS_LAME_LAMBDA | 6400 |
| POISSON | 6600 |
| STRSS_FR_SOL_VOL_FRAC | 6610 |

This is only a partial list of the available material property tags. To find the latest tag definitions (those active in your version of *Goma*), look in the *Goma* header file *mm_mp_const.h*. The initial value of the extra unknowns is taken from those values given in the input, material definition, and geometry definition (if any) files.

## Examples

The following is an example this AC card:

```
AC = MT 1 1400
```

This card indicates that this augmenting condition constraint is associated with the surface tension value found in the material file designated for material 1.

Note that the value of surface tension originally supplied in the material final will be used as a starting guess. However, at the end of the computation a different value for surface tension will have been determined and save in the memory location for this

material property. If an ASCII output file is requested (SOLN file = ), the value of the surface tension will appear at the end of the file following the output of the nodal unknown vector.

### Technical Discussion

See the technical discussion appearing in the documentation for the AC(*Boundary Condition*) card.

### Theory

No Theory.

### FAQs

No FAQs.

### References

Gates, I.D., D.A. Labreche, and M.M. Hopkins, "Advanced Capabilities in GOMA 3.0 - Augmenting Conditions, Automatic Continuation, and Linear Stability Analysis," SAND2000-2465, Albuquerque, NM, (2001).

## 2.3.5 AC (*Flux Condition*)

| AC = FC <mat_id> <bc_id> <data_float_index> <FC_type> [species_id] <sideset> <flux_value> |
|---|

### Description/Usage

This card attaches a boundary condition float parameter to an integrated flux constraint on a boundary sideset. The flux constraint consists of requiring a specific value for some integrated quantity over the sideset, for example, force, heat flux or fluid flowrate. A number of standard flux constraints have been provided so that a user-defined flux is not necessary. During the solution process the boundary condition parameter is allowed to vary as the additional degree of freedom associated with the flux constraint.

Definitions of the input parameters are as follows:

**FC** Mandatory string indicating that this augmenting condition is of variety *Flux Condition*.

| | |
|---|---|
| <mat_id> | An integer parameter identifying the identification number of the material on which the flux integration will be performed. Material properties needed in evaluating the flux constraint will be obtained from this material and only those elements belonging to this material will contribute to the flux integral |
| <bc_id> | An integer parameter giving the index of the boundary condition card whose parameter is being used as the new degree of freedom. Numbering begins with zero, starting with the first boundary condition in the input file and proceeding sequentially upward with each read boundary condition card. |
| <data_float_index> | An integer parameter that identifies the boundary condition parameter that will be varied. It is an index that starts at zero with the leftmost float value on the <bc_id> boundary condition card and increments upward from right to left. |
| <FC_type> | A string parameter specifying one of the standard flux constraint equations. Choices are (all in capitals):<br>FORCE_X<br>FORCE_Y<br>FORCE_Z<br>FORCE_NORMAL<br>FORCE_TANGENT1<br>FORCE_TANGENT2<br>HEAT_FLUX<br>VOLUME_FLUX<br>SPECIES_FLUX<br>CHARGED_SPECIES_FLUX<br>CURRENT<br>PORE_LIQ_FLUX<br>TORQUE<br>AVERAGE_CONC<br>SURF_DISSIP<br>AREA<br>VOL_REVOLUTION<br>CURRENT_FICKIAN<br>NEG_LS_FLUX<br>POS_LS_FLUX<br>N_DOT_X<br>ELEC_FORCE_NORMAL<br>ELEC_FORCE_TANGENT1<br>ELEC_FORCE_TANGENT2<br>ELEC_FORCE_X |

ELEC_FORCE_Y
ELEC_FORCE_Z
NET_SURF_CHARGE
DELTA
ACOUSTIC_FLUX_NORMAL
ACOUSTIC_FLUX_TANGENT1
ACOUSTIC_FLUX_TANGENT2
ACOUSTIC_FLUX_X
ACOUSTIC_FLUX_Y
ACOUSTIC_FLUX_Z
ACOUSTIC_INTENSITY
LS_DCA

See below for a detailed description of each of these constraints.

[species_id] This is an integer parameter that identifies the species component that is evaluated in the SPECIES_FLUX constraint. When other flux constraints are used this parameter should not appear.

<side set> An integer parameter that identifies the side set over which the flux condition will be integrated. Those elements that belong to <mat_id> and have faces included in <side set> will be evaluated in the integration.

<flux_value> A float parameter that specifies the value that is assigned to the flux integral. At the end of the solution procedure, the flux integral will be equal to this float value.

## Examples

The following are two examples of this AC type:

```
Number of augmenting conditions = -1
AC = FC 1 0 0 VOLUME_FLUX 4 {-PI}
AC = FC 1 1 0 FORCE_TANGENT1 3 {3*5.0}
END OF AC
```

## Technical Discussion

- The following list describes precisely someof the preceding **FC_type** constraint equations. In the following, Φ is the <flux_value> parameter and Γ is the side set specified above.

  For material blocks with ARBITRARY mesh motion:

FORCE_X: $$\Phi = \int_{\Gamma} (n \cdot \Pi - \rho(u - u_m)(n \cdot u)) \cdot e_x d\Gamma$$

FORCE_Y: $$\Phi = \int_{\Gamma} (n \cdot \Pi - \rho(u - u_m)(n \cdot u)) \cdot e_y d\Gamma$$

FORCE_Z: $$\Phi = \int_{\Gamma} (n \cdot \Pi - \rho(u - u_m)(n \cdot u)) \cdot e_z d\Gamma$$

FORCE_NORMAL: $$\Phi = \int_{\Gamma} (n \cdot \Pi - \rho(u - u_m)(n \cdot u)) \cdot n d\Gamma$$

FORCE_TANGENT1: $$\Phi = \int_{\Gamma} (n \cdot \Pi - \rho(u - u_m)(n \cdot u)) \cdot t_1 d\Gamma$$

FORCE_TANGENT2: $$\Phi = \int_{\Gamma} (n \cdot \Pi - \rho(u - u_m)(n \cdot u)) \cdot t_2 d\Gamma$$

HEAT_FLUX: $$\Phi = \int_{\Gamma} (n \cdot q - (\rho C(u - u_m) \cdot n)) d\Gamma$$

VOLUME_FLUX: $$\Phi = \int_{\Gamma} -(u - u_m) \cdot n d\Gamma$$

SPECIES_FLUX: $$\Phi = \int_{\Gamma} (n \cdot j_i - (c_i(u - u_m) \cdot n)) d\Gamma$$

CHARGED_SPECIES_FLUX

CURRENT

PORE_LIQ_FLUX

where

$\Pi$ = fluid stress tensor
$\rho$ = fluid phase density
$u$ = fluid velocity vector
$u_m$ = mesh velocity vector
$e_x$,$e_y$,$e_z$ = cartesian unit bases
$n$ = normal vector to side set
$t_1$ = first tangent vector to side set
$t_2$ = second tangent vector side set (3D only)
$q$ = diffusive heat flux
$C$ = heat capacity per unit mass
$j_i$ = diffusive flux of species $i$
$c_i$ = concentration of species $i$.

In the case of a LAGRANGIAN material, only the FORCE flux constraints are relevant. They have the same form as the preceding except that there are no convective contributions to the flux and Π represents the solid stress tensor.

- For additional technical notes regarding this augmenting condition type see the AC (Boundary Condition) section.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 2.3.6 AC (*Volume Constraint*)

| AC = VC <mat_id> <vc_type> <bc_id> <data_float_index> <species_id> <volume_value> |
|---|

### Description/Usage

This augmenting condition card allows the user to specify a boundary condition parameter as an unknown parameter in order to satisfy an integrated volumetric constraint. Three types of constraint are allowed: mesh volume, mass in mesh, and mass of species in mesh. All three integrated constraints are computed automatically and so no user-defined function is needed. See below for a mathematical description of each type of integrated constraint.

This card attaches a boundary condition parameter to these integrated constraints as an additional unknown degree of freedom. This parameter is specified in exactly the same way as detailed in the AC (*Boundary Condition*) section.

Definitions of the input parameters are as follows:

| | |
|---|---|
| **VC** | Mandatory string identifying the augmenting condition card as being of type *Volume Constraint*. |

<mat_id> An integer parameter giving the material identification number (element block id in the mesh) over which the integrated volume constraint is to be applied. That is to say, if more than one material occupies a problem domain, a volume constraint can only be applied to the space occupied by one of the materials for any given augmenting condition.

<vc_type> An integer parameter that sets the type of volume constraint as follows:

vc_type = 1 (mesh volume):

$$V_T = \int_V dV \tag{2-1}$$

vc_type = 2 (mass in mesh):

$$V_T = \int_V \rho dV \tag{2-2}$$

where ρ is the fluid density.

vc_type = 3 (mass of species in mesh):

$$V_T = \int_V \rho y_i dV \tag{2-3}$$

where $y_i$ is the mass fraction of the *i*th species.

<bc_id> An integer parameter giving the index of the boundary condition card whose parameter is being used as the new degree of freedom. Numbering begins with zero, starting with the first boundary condition in the input file and proceeding sequentially upward with each read boundary condition card.

<data_float_index> An integer parameter that identifies the boundary condition parameter that will be varied. It is an index that starts at zero with leftmost float value on the <bc_id> boundary condition card and increments upward from right to left.

<species_id> An integer parameter that identifies the species number to be used when evaluating a vc_type = 3 constraint equation. For other types of constraints, it is still

necessary for the syntax, but is unused. Good procedure sets it to zero.

<volume_value> A float parameter that fixes the value of $V_T$ used in the expressions above. That is, it fixes the value that the integrated quantity will have at the end of the computation.

### Examples

The following is an AC (*Volume Constrain*t) card with its accompanying boundary condition card.

```
AC = VC 1 1 2 1 0 3.14156
```

which has the following set of boundary conditions

```
Number of BC = -1
BC = U NS 1 0.0
BC = V NS 1 1.0
BC = CAPILLARY SS 10 1.0 10.0 0.0
```

This augmenting condition then applies to material 1 and will fix the mesh volume of this material to 3.14156. To do this it will vary the external pressure applied via the *CAPILLARY* card (the second float parameter). Note that the value given for this pressure in the input deck serves as a starting guess. At the end of the calculation, a new value for pressure will be in the memory location assigned to this float parameter. If an ASCII solution file is requested (via *SOLN file* =), the parameter value will be written following the output of the nodal unknown vector.

### Technical Discussion

- See the technical discussion appearing in the documentation for the AC (*Boundary Condition*) card.

## Theory

No Theory.

## FAQs

No FAQs.

### References

SAND2000-2465: Gates, I.D., Labreche, D. A., Hopkins, M. M. and Wilkes, E. D., 2001. “Advanced Capabilities in GOMA 3.0 - Augmenting Conditions, Automatic Continuation, and Linear Stability Analysis,” Sandia Technical Report.

## 2.3.7 AC (*Overlapping Grid Boundary Conditions*)

| AC = OV <SSID> <integer1> <integer2><integer3> |
|---|

### Description/Usage

This type of augmenting condition is used to invoke the overlapping grid algorithm for fluid-structure interaction problems, which applies a kinematic constraint along either the moving part of the solid boundary or the zero contour of a phase function field within the fluid which follows this boundary, and enables sensitivities from both phases to be included. When this card is provided in conjunction with an appropriate set of interfacial boundary conditions (see below), this AC is replaced with *m* new augmenting constraints -- two per element side along the side set specified by <SSID>.

This algorithm works with either of the following two sets of interfacial boundary conditions:

LS_NO_SLIP, SOLID_FLUID_CONTACT, FLUID_SOLID_CONTACT

or

LAGRANGE_NO_SLIP, OVERSET(BAAIJENS)_SOLID_FLUID, OVERSET(BAAIJENS)_FLUID_SOLID

The primary difference between these two sets is that the former imposes the kinematic condition from the fluid side and the latter from the solid side. The same physical equations are applied in either case.

A description of the card syntax follows:

| | |
|---|---|
| **OV** | A mandatory string indicating that the augmenting condition is being used to invoke the overlapping grid algorithm. |
| <SSID> | The SS index of the moving solid boundary, taken directly from the ExodusII input file. |

| | |
|---|---|
| <integer1> | The element block index for the solid phase, taken directly from the ExodusII input file. |
| <integer2> | The element block index for the fluid phase, taken directly from the ExodusII input file. |
| <integer3> | Location of the Lagrange multiplier unknowns. For now, this entry must be zero, which indicates that the unknowns are stored in the augmenting conditions. |

## Examples

For a case where a fluid is in element block 1 and the solid is in element block 2 with sideset 10 defined along the moving boundary, the appropriate augmenting condition card of type OV is:

```
AC = OV 10 2 1 0
```

Note that the first three integers come directly from the input mesh file and thus do not depend on which of the above boundary condition sets is chosen. It is only necessary that those BC's which are applied from the solid side have the same side set ID specified.

## Technical Discussion

- If it is necessary to specify any other type of augmenting condition in the same problem, this card must be the last one in the list.
- When this card is present, a check will be done for the presence of any of the above interfacial boundary conditions. An error will occur if none are specified.
- There is a routine which counts the number of element sides in side set <SSID> and allocate the necessary number of augmenting constraints to be used in the program (This changes nAC). *Goma* must be compiled with a sufficient value for the constant MAX_NGV, which must be at least nAC+5. The default value of this constant is set to 10 in the file rf_io_const.h; this value can either be changed there or overridden by specifying "-DMAX_NGV=<#>" in the DEFINES section of the makefile.
- The augmenting condition constraints are additional residual equations solved simultaneously with the other residual equations associated with the nodal degrees of freedom. The unknown degrees of freedom associated with these equations are the boundary condition float values identified on each augmenting condition card. The new equations also introduce a different structure to the matrix being solved. The formerly sparse, banded Jacobian matrix has now been augmented by rows and columns on its periphery that are potentially populated. This requires a

bordering algorithm for the solution of this type of matrix. A consequence of this algorithm is that the residual and update norms of the augmenting conditions are computed separately. The user will see these norms appear as a second row of output at each and every iteration. If these are missing, the augmented system is not being solved.

- Note that the value of the augmenting condition parameter values, which in this case are the Lagrange multipliers representing interfacial forces, can be displayed after each iteration by setting *Debug = 1* in the input deck. However, since these values are stored as global-type unknowns, the values cannot be readily referenced to their locations along the solid boundary. This information is recorded in a separate file “overlap_ac_id.dat”.

### Theory

No Theory.

### FAQs

No FAQs.

### References

Baaijens, F. P. T. “A fictitious domain/mortar element method for fluid-structure interaction,” *Int. J. Numer. Meth. Fluids*. **35**, 2001, 743-761.

## 2.3.8 AC (Phase Velocity of Level Set)

| AC = LSV <integer_list> <VX\|VY\|VZ> <float1> |
|---|

### Description/Usage

This type of augmenting condition is used to connect a boundary condition to an augmenting constraint on the average velocity of one of the phases of the level set field. This is useful for performing a simulation in the reference frame of a moving material when the velocity is not known *a priori*, such as a bubble rising through a quiescent fluid.

The <integer_list> has four parameters; definitions of the input parameters are as follows:

**LSV | LS_VELOCITY**

A mandatory string indicating the name of the augmenting condition card.

<integer1> An integer parameter giving the material identification number over which the phase volume constraint is applied. That is to say, if more than one material is present, the phase velocity constraint can only be applied to the space occupied by the level set phase in one material.

<integer2> An integer parameter giving the index of the boundary condition card whose parameter is being used as the new degree of freedom. Numbering begins with zero, starting with the first boundary condition in the input file. For this augmenting condition, this is generally a velocity boundary condition.

<integer3> An integer parameter that identifies the floating point parameter of the boundary condition that will be varied. It is an index that starts at zero with the left-most float value on the <int2> boundary condition cast and increments upward from left to right.

<integer4> An integer parameter which specifies the level set phase whose velocity will be constrained by this augmenting condition.

- > 0 The constraint will be placed on the phase where the level set function has positive values.
- < 0 The constraint will be placed on the phase where the level set function has negative phase.

<VX|VY|VZ> This string parameter specifies which component of the velocity will be constrained by the augmenting condition.

- VX The x component of the phase velocity will be constrained.
- VY The y component of the phase velocity will be constrained.
- VZ The z component of the phase velocity will be constrained.

<float1> A float parameter that fixes the volume averaged velocity of the level set phase specified by <int4>.

## Examples

The following is an example of using this augmenting condition to perform a calculation in the moving reference frame of the negative level set phase where the reference frame velocity is not known *a priori*:

```
AC = LSV 1 0 0 -1 VX 0.0
```

Applied to boundary condition:

```
BC = U NS 4 0.0 1.0
```

The x component of the velocity of the negative level set phase of material #1 is set to zero and the first floating point parameter of the first velocity boundary condition is used as the new degree of freedom. Note that the Dirichlet boundary condition has the additional float parameter 1.0 so that the boundary condition equation is retained as a residual equation. The initial guess for the boundary velocity in this case is 0.

## Technical Discussion

The velocity of the specified phase of the level set field is calculated using a volume average over the domain where the level set fill function has positive or negative values as specified. The boundary between the two phases is defined as the zero contour line of that fill function. The ‘thickness’ of the region around the embedded interface where material properties transition between theproperties of the two pure phases is specified by the “Level Set Length Scale” card in the input deck. Within the diffuse interface region, the contribution to the averaged velocity integral is weighted by a smoothed Dirac delta function. Additional information about the embedded interface and the form of the delta function are available in the technical discussion section of the “Level Set Length Scale” card.

See the technical discussion section of the AC (Boundary Condition) card for additional information regarding augmenting conditions.

## Theory

No Theory.

## FAQs

No FAQs.

## References

No References.

## 2.3.9 AC (*Periodic Boundary Condition*)

AC = PBC <var_name> <sideset_id1> <sideset_id2>

### Description/Usage

This augmenting condition card allows the user to specify that a field variable has the same value at both ends of a given domain. The ends are specified as side sets, which must have the same number of nodes with the same spacing between them.

Definitions of the input parameters are as follows:

| | |
|---|---|
| **PBC** | Mandatory string identifying the augmenting condition card as being of type *Periodic Boundary Condition*. |
| <var_name> | A string indicating the variable whose value will be matched at the specified periodic boundaries of the problem domain. This variable must be active throughout the region bordered by these two boundaries. |
| <sideset_id1> | An integer parameter that identifies the sideset at one of the two periodic boundaries for the specified variable. This sideset must be coextensive with this domain boundary. |
| <sideset_id2> | An integer parameter that identifies the sideset at the other periodic boundary for the specified variable. This sideset must be coextensive with this domain boundary. |

### Examples

The following is an AC (*Periodic Boundary Condition*) card which specifies periodic boundaries for voltage at sidesets 20 and 30:

```
AC = PBC VOLTAGE 20 30
```

### Technical Discussion

When this augmenting condition is invoked, the degrees of freedom (DOF's) of the specified variable at corresponding points on the two periodic boundaries are matched. During equation assembly, a Lagrange multiplier constraint is applied to each pair of DOF's to enforce the constraint, which in residual form is:

$$\lambda(x_2 - x_1) = 0 \qquad \textbf{(2-4)}$$

where $x_1$ and $x_2$ are the unknown values at corresponding points along the two specified periodic boundaries, and $\lambda$ is the added unknown associated with this constraint. This residual and all relevant sensitivities to problem unknowns are assembled into the appropriate augmenting condition arrays and solved along with any other active augmenting conditions via the internal bordering algorithm within Goma's nonlinear solver.

### Theory

No Theory.

### FAQs

No FAQs.

### References

SAND2000-2465: Gates, I.D., Labreche, D. A., Hopkins, M. M. and Wilkes, E. D., 2001. "Advanced Capabilities in GOMA 3.0 - Augmenting Conditions, Automatic Continuation, and Linear Stability Analysis," Sandia Technical Report.

## 2.3.10 END OF AC

| END OF AC |
|---|

### Description/Usage

This card is the companion card to the *Number of augmenting conditions* card. The *END OF AC* card signals the end of the *Augmenting Conditions Specifications* section of the *Goma* input. When the *Number of augmenting conditions* (= -1) is set to negative one, *Goma* will read all the augmenting condition cards until this card is encountered in the input file. This card may omitted if the integer *N* on the *Number of augmenting conditions* card is not -1.

### Examples

Typical usage of this card is illustrated below:

```
Number of augmenting conditions = -1
.
.
.
END OF AC
```

### Technical Discussion

See companion card *Number of augmenting conditions*.

## Theory

No Theory.

## FAQs

No FAQs.

## References

No References.

## 2.4 Functional Definition of Unknowns in the *Goma user_ac.c* File

The functional form of the augmented constraint equations is defined in the *Goma* user file *user_ac.c*. The augmenting conditions are expressed in residual form in the user function **user_aug_cond_residuals**. By residual form, it is meant that the conditions are assembled/written such that the function equals zero:

$$\Re_{AC}(x,MP,BC_{DF},\ldots) = 0 \qquad \textbf{(2-5)}$$

The number of these extra unknowns must match the number of augmenting conditions specified in the `Augmenting Condition Specifications` card. A listing of the *C*-code for *user_ac.c* shows that it is essentially an empty shell ready for user expansion:

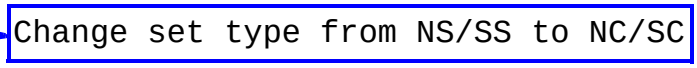
```
void
user_aug_cond_residuals (int iAC,              (index to augmenting conditions)
                         double *x,            (vector of Goma unknowns)
                         double *xdot,
                         double delta_t,
                         double time_value,
                         double **x_sens_p,
                         double *AC,           (vector of augmenting condition, residuals)
                          int *have_bAC, int *have_cAC, int *have_dAC,
                         double **bAC, double **cAC, double **dAC,
                          Exo_DB *exo,
                         Dpi *dpi,
                         Comm_Ex *cx)
{
 dbl inventory, target;

 /**  loop over all AC conditions **/

 if (augc[iAC].Type == AC_USERBC )          /* BC augmenting condition */
   {
    AC[iAC] = BC_Types[4].BC_Data_Float[0]-2.0;  /* Goma Test Suite */
   }
 else if (augc[iAC].Type == AC_USERMAT )      /* MT augmenting condition */
   {
   }

} /* END of routine user_aug_cond_residuals                              */
/*****************************************************************************/
```

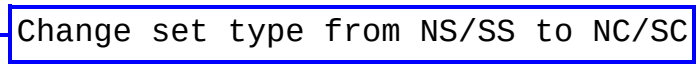
In the *user_ac.c* routine, the (one or more) augmenting conditions are entered as a vector -- `AC[0]`, `AC[1]`, and so on, with a starting index of 0 (i.e., the *C*-convention for indexing). In the listing above, the first augmenting condition residual (AC[0]) specifies that the first float (BC_Data_Float[0]) belonging to the fifth boundary condition (BC_Types[4]) equals 2. This is an example of a BC type augmenting condition (though admittedly trivial since it could have been accomplished without using an augmenting condition).

The boundary condition index is the internal index that *Goma* assigns to a boundary condition definition in the `Boundary Condition Specification` section of the input file. Usually, this value is the number of the boundary condition counting from the first condition down in the input file (starting from number 0). However, an easy means for finding the ***bc_id_index*** is to use the –`bc_list` command line argument (described below) for *Goma*.

To obtain the ***bc_id_index***, the user marks all boundary conditions whose indices are to be identifed by changing the BC set type from `NS` to `NC` or `SS` to `SC`. For example, in the Boundary Condition section of the *Goma* input file:

```
Number of BC = -1
BC = PLANE SS 1 {a1} {b1} {c1} {d1}
BC = PLANE SS 2 {a2} {b2} {c2} {d2}
BC = PLANE SS 3 {a3} {b3} {c3} {d3}
BC = PLANE SS 4 {a4} {b4} {c4} {d4}
$
BC = U NC 3 {speed}
BC = V NS 3 0.0
BC = U NC 1 0.0
BC = V NS 1 0.0
BC = U NS 2 0.0
BC = V NS 2 0.0
BC = V NS 4 0.0
BC = U NC 4 0.0
END OF BC
$
```



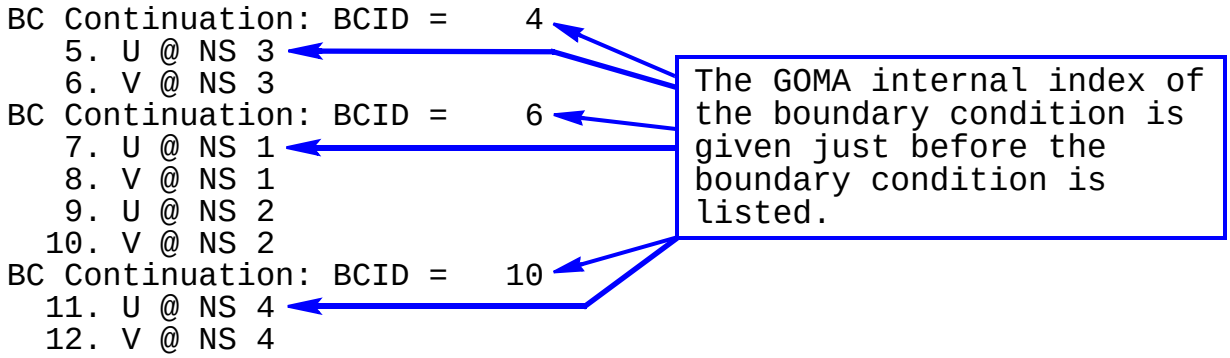




Next, GOMA is run with the –`bc_list` argument, that is:

```
goma –a –i input_file –bc_list
```

The output is:

```
Number of boundary conditions = 12
   1. PLANE @ SS 1
   2. PLANE @ SS 2
   3. PLANE @ SS 3
   4. PLANE @ SS 4
```




```
Number of rotation conditions = 0

Number of materials           = 1

Number of equations           = 3 (coating_liq)
-bc_list request.

         goma done.
```

After the user finds the boundary condition indices needed, the set types should be changed back to their original BC type values (`NS/SS`).

To illustrate this indexing, consider the following examples. A BC augmenting condition that specifies that the first data float belonging to the boundary condition with ***bc_id_index*** 6 and the third data float belonging to the boundary condition with ***bc_id_index*** 11 equal each other is:

AC[0] = BC_Types[6].BC_Data_Float[0] – BC_Types[11].BC_Data_Float[2];

A mixed constraint augmenting condition might specify that the second data float of the boundary condition with ***bc_id_index*** 2 is a function, `f()`, of the density of material 1. This is given by:

AC[1] = BC_Types[2].BC_Data_Float[1] – f(mp_glob[0]->density);

It should be clear from these examples that defining augmenting conditions requires some knowledge of the data structures in *Goma*. For boundary conditions, all data floats are contained in the `BC_Types[]` data structure, while material information is contained in the `mp_glob[]`, `gn_glob[], ve_glob[]`, and `elc_glob[]` data structures for material, generalized newtonian, viscoelastic and elastic properties, respectively. Postprocessed variables, such as flux quantities, are held in the `Post_Processing` data structure. These and other data are made available to users in the *user_aug_cond_residuals* function through the header (C include) files;

all external variables can be readily accessed. The variables/structures used as arguments in the function call should normally not need to be changed by users to acquire access to data in functional relationships among augmenting conditions. If the functionality of *user_aug_cond_residuals* were to be changed, argument changes may be need but this should not be undertaken without consulting the *Goma* development group.

For BC and MT type augmenting conditions, the user function is entered in the appropriate section of the if-block (between curly braces). For the VC and FC type of augmenting conditions, **no** changes need be made to the *user_aug_cond_residuals* function as these are standardized functions that have been entered into algorithms and placed in the **std_augc_cond** function in *mm_augc_util.c*.

When *Goma* is run with augmenting conditions, after the augmented system converges, the extra unknowns are written to the standard output. At present, the values of the extra unknowns are not written to the Exodus II output file. To initialize the extra unknowns for a new calculation, the values of the boundary condition data floats and material properties from a past converged solution can be inserted directly into the *Goma* input and material definition files.

## 2.5 Continuation with Augmenting Conditions

*Goma* has the capability to continue in one or more parameters (discussed in Chapter 3) together with augmenting conditions. Currently, the supported continuation types include `zero`, `first`, `hzero`, `hfirst`, and `loca` (LOCA methods `zero`, `first`, and `alc`), and multiple continuation conditions can be specified with CC or HC cards, or with user-defined continuation functions as explained in Chapter 3. The only requirements are that the

`Augmenting Condition Specifications` section is in the input file and augmenting conditions of types BC or MT are defined in *user_ac.c* as described above. At each step of the solution path, the values of the extra unknowns are updated and displayed to the standard output. A special method has been provided to handle arc length continuation (LOCA method = `alc`) when augmenting conditions are used by treating the arc length equation as an additional AC; this method is invoked internally when the input file specifies both `alc` continuation and a list of AC's and requires no intervention by the user (e.g. no new AC card).

Both first-order continuation (by any available method) and arc length continuation involve sensitivity resolves to calculate $d\mathbf{x}/d\lambda$ after each step, which are used to generate an initial guess for the solution at the following step. These resolves are done without including any augmenting condition effects and therefore the sensitivities are not rigorously correct, but in most cases the resulting initial solution guess will still be sufficiently accurate to yield acceptable convergence on the next continuation step.

## 2.6 Examples

### 2.6.1 Augmenting condition on lid speed in the lid driven cavity problem

This example demonstrates a simple use of augmenting conditions by constraining the velocity on the boundary of lid driven cavity. (Under normal circumstances, the user would change boundary velocity by changing the boundary condition.) The input file for augmenting the steady state problem, `ldc.input-augc`, has the AC Specification section shown below; the BC's are shown for reference as a boundary condition is changed.

```
------------------------------------------------------------
       Augmenting Condition Specifications
------------------------------------------------------------
Number of augmenting conditions  = -1
AC = BC 4 0
END OF AC

------------------------------------------------------------
         Boundary Condition Specifications
------------------------------------------------------------
Number of BC                        = -1
BC = PLANE SS 1 {a1} {b1} {c1} {d1}
BC = PLANE SS 2 {a2} {b2} {c2} {d2}
BC = PLANE SS 3 {a3} {b3} {c3} {d3}
```

```
BC = PLANE SS 4 {a4} {b4} {c4} {d4}
$
BC = U NS 3 {speed}
BC = V NS 3 0.0
BC = U NS 1 0.0
BC = V NS 1 0.0
BC = U NS 2 0.0
BC = V NS 2 0.0
BC = U NS 4 0.0
BC = V NS 4 0.0
$
END OF BC
```

In this example the augmenting condition defined in *user_ac.c* replaces the *speed* variable on NS 3 (at the top of the cavity or lid), changing the x-velocity from 1.0 (in *geometry.in*) to 2.0 (in the augmenting condition). The function entered in *user_aug_cond_residuals* is identical to the augmenting condition shown in the code "snippet" at the beginning of Section 2.4.

The *Goma* screen output for this problem is:

```
                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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
10:05:30 [0] 2.6e-12 2.3e-11 5.4e-12 1.2e+02 1.2e+04 1.2e+03  1  4.0e-02/7.0e-02
         AC 1.0e-00 1.0e-00 1.0e-00 1.0e-00 1.0e-00 1.0e-00     3.0e-02/0.0e+00
10:05:30 [1] 1.8e-15 1.1e-13 6.6e-15 1.6e+01 1.6e+03 1.6e+02  1  4.0e-02/6.0e-02
         AC 6.6e-12 6.6e-12 6.6e-12 6.6e-12 6.6e-12 6.6e-12     3.0e-02/1.0e-02


scaled solution norms   1.666382e+02  1.148502e+01  3.876488e+01


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

        BC[   4] DF[   0]= 2.000000e+00

-done


Proc 0 runtime:       0.01 Minutes.
```

With the addition of an augmenting condition, the Newton iteration history lists the residuals and corrections for the initial system of equations as well as for the augmented system. The output indicates the balanced number of ACs and unknowns and shows the data float (DF) value for BC_ID= 4 being set to 2.0. As such, this augmenting condition resets the lid speed, enforcing the data float on boundary condition 4 to be equal to 2.

### 2.6.2 Pressure difference augmenting condition in slot coating model

This example demonstrates the use of an augmenting condition to compute the vacuum pressure required on the upstream meniscus of a slot coating die to hold the meniscus fixed at its initial location.

The AC Section and the relevant portion of the BC Section of the *Goma* input file `scpd.input` is:

```
-------------------------------------------------------------
        Augmenting Conditions Specifications
-------------------------------------------------------------
Number of augmenting conditions  = 1
AC = BC 24 1
END OF AC

-------------------------------------------------------------
          Boundary Condition Specifications
-------------------------------------------------------------
Number of BC =  -1

BC = KINEMATIC  SS  23  0.0
BC = CAPILLARY  SS  23  {surface_tension}  {vacuum}  0.0
BC = CA  NS  700  0.90757  0.0  -1.0  0.0
BC = CA  NS  100  { 3.14159 - 2.05949 }  0.0  0.0  0.0
BC = KINEMATIC  SS  6  0.0
BC = CAPILLARY  SS  6  {surface_tension}  0.0  0.0
BC = CAP_ENDFORCE  NS  1400  1.0  0.0  0.0  {surface_tension}

END OF BC
```

The augmenting condition input identifies the capillary pressure on the upstream meniscus (***bc_id_index*** = 1 for sideset 23, i.e., the second BC data parameter - the applied pressure on that boundary, currently set to -11444.0 in the included file `slot.geom`), as the unknown in this problem. The BC augmenting condition is replacing the capillary stress condition on sideset 2. This problem is run using the -a command line argument, which invokes *Aprepro* prior to launching *Goma*. All expressions in curly braces are then evaluated through standard arithmetic or definitions in the included file `slot.geom`, e.g. {`surface_tension`} is replaced with its assigned numerical value of 61.0.

The constraint equation associated with this augmenting condition has been extracted from its definition in the *user_ac.c* file and is shown below:

```
i = index_solution(2, MESH_DISPLACEMENT1, 0, 0, -1);
AC[0] = x[i];
```

*Goma*'s `index_solution` function is designed to extract an index which points to the value of a particular variable in the solution vector, *x*. In this case, *i* is the storage location for mesh displacement at node 2, the dynamic contact line. The second line of the functional definition forces the value of mesh displacement at node 2 to be zero, i.e., no movement of the contact line, and is expressed in residual form.

The Newton iteration history output by *Goma* is:

```
      R e s i d u a lC 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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
10:36:16 [0] 3.7e-04 2.3e-02 1.3e-03 2.1e+03 2.8e+04 3.6e+03  1  2.8e+00/4.8e+00
        AC 0.0e+00 0.0e+00 0.0e+00 7.9e+00 7.9e+00 7.9e+00     2.0e+00/1.6e-01
10:36:26 [1] 3.2e-04 5.7e-03 9.4e-04 4.0e+02 1.1e+04 6.6e+02  1  2.8e+00/4.7e+00
        AC 0.0e+00 0.0e+00 0.0e+00 4.6e+00 4.6e+00 4.6e+00     2.0e+00/1.5e-01
10:36:36 [2] 1.0e-03 1.2e-03 1.0e-03 2.7e+01 2.1e+03 7.5e+01  1  2.8e+00/4.7e+00
        AC 0.0e+00 0.0e+00 0.0e+00 1.5e+00 1.5e+00 1.5e+00     2.0e+00/1.6e-01
10:36:46 [3] 7.3e-08 1.7e-06 2.1e-07 2.8e-01 8.8e+00 6.9e-01  1  2.8e+00/4.6e+00
        AC 0.0e+00 0.0e+00 0.0e+00 5.0e-04 5.0e-04 5.0e-04     2.0e+00/1.5e-01
10:36:55 [4] 8.6e-12 1.4e-10 1.9e-11 2.7e-05 9.4e-04 6.5e-05  1  2.8e+00/4.6e+00
        AC 0.0e+00 0.0e+00 0.0e+00 3.5e-08 3.5e-08 3.5e-08     2.0e+00/1.5e-01
scaled solution norms   1.647387e+04  3.019041e+02  1.399954e+03

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

        BC[  24] DF[   1]=-1.144884e+04

-done


Proc 0 runtime:     0.83 Minutes.
```

This indicates that the capillary pressure (vacuum) required to keep the static contact line fixed is -11448.84.

# 3 Automatic Continuation

Automatic continuation refers to the family of algorithms that allow tracking steady-state solution paths as a set of one or more parameters are varied. *Goma* is currently capable of zero order, first order, and arc length continuation in a single parameter set, and bifurcation tracking (turning point, pitchfork, or Hopf) in two parameter sets. Continuation can be carried out in four types of parameters:

**Material Properties**

**Material Property Tags** are obtained from Goma source file mm_mp_const.h. A partial list is given in the previous chapter.(

**Boundary Condition Floats**

Continuation can be carried out in any boundary condition data float, i.e., the boundary condition parameters, making continuation in any boundary data and geometry simple. The definition of a boundary condition has the form:

**BC = BC_NAME BC_TYPE BC_ID DATAFLOAT1 DATAFLOAT2 . . .**

An example of continuation in a boundary condition: the effect of flow rate on a flow field can be tracked by continuing in the data float that represents velocity in a Dirichlet boundary condition.

**Material Property User Model Floats**

This option is used when the user supplies a model for a material property in the file “user_mp.c” which includes a list of one or more input data floats, and one of these floats is the desired continuation parameter.

**Augmenting Conditions**

There are two types of augmenting condition values which can be used as continuation parameters: the constant value (e.g. flux, volume) which is to be imposed, and one of a list of optional floats which may be specified with a user-defined augmenting condition (file “user_ac.c”).

Following is a summary of solution prediction algorithms used in *Goma*’s continuation schemes.

| Continuation Method | Characteristic Algorithm |
|---|---|
| zero | Single-parameter zeroth order continuation<br>$x^{PREDICTED}(\lambda^{NEW}) = x^{OLD}(\lambda^{OLD})$ |

| | |
|---|---|
| first | Single-parameter first order continuation<br>$x^{PREDICTED}(\lambda^{NEW}) = x^{OLD}(\lambda^{OLD}) + \Delta\lambda \frac{\partial x}{\partial \lambda}$ |
| hzero | Multi-parameter zeroth order continuation<br>$x^{PREDICTED}(\lambda^{NEW}) = x^{OLD}(\lambda^{OLD})$ |
| hfirst | Multi-parameter first order continuation<br>$x^{PREDICTED}(\lambda^{NEW}) = x^{OLD}(\lambda^{OLD}) + \sum \Delta\lambda_j \frac{\partial x}{\partial \lambda_j}$ |

These algorithms are available with or without the Library of Continuation Algorithms (LOCA), which also offers algorithms for arc length continuation and bifurcation tracking, as described above. Details of these algorithms can be found in the LOCA 1.0 manual (SAND 2002-0396).

## 3.1 Required Specifications in the *Goma* Input File

A new section must be added to the *Goma* input file to identify the continuation method, continuation type, the continuation parameter, and other needed data. It is required only when automatic continuation is used in the numerical model and has the following form:

```
-------------------------------------------------------------
                Continuation Specifications
-------------------------------------------------------------
Continuation                          = zero
Continuation Type                     = MT
Boundary condition ID                 = 4
Boundary condition data float tag     = 0
Material id                           = 1
Material property tag                 = 1700
Material property tag subindex        = 0
Initial parameter value               = 0.0
Final parameter value                 = 800.0
delta_s                               = 20.0
Maximum number of path steps          = 20
Minimum path step                     = 1.0e-05
```

```
Maximum path step                    = 100.0
Continuation Printing Frequency      = 1
```

The input records above control the continuation process. Specifically, this example indicates that zeroth-order continuation in material property density (tag 1700) will be simulated with *Goma*. (Boundary condition records must be present even though their input is ignored, and vice versa for material property records when the Continuation Type is BC.) The material subindex is not currently used. Density will be varied between 0 and 800 in material 1 using an initial path length/ step size of 20.0 and minimum and maximum values of 1.0e-5 and 100.0, respectively. A maximum of 20 steps will be taken and every solution step will be written to the database. Each entry of the Continuation Specifications is discussed further in Section 3.2.

There are additional input cards which are either optional or required only in certain cases; these will also be described in Section 1.2.

## 3.2 Continuation Specifications

This section of input records is used to direct all automatic continuation procedures. The entire section is completely optional. Basically, automatic continuation can be accomplished in steady state simulations (see *Time Integration* card) through any one or combination of parameters. These parameters can be any one or combination of the input floats required on the boundary condition cards (see Section 4.10) or material property cards (see Chapter 5). The cards in this section are used to specify the parameters that will be marched automatically, the method of marching (e.g. zero-order, first-order, multiparameter first-order, etc.), the limits of parameter values, and other sundry options. Much of this capability can now be managed from the LOCA library package (Library of Continuation Algorithms - Salinger, et al. 2002).

### 3.2.1 Continuation

| Continuation = {zero \| first \| hzero \| hfirst \| loca} |
|---|

#### Description/Usage

This card is required only for continuation problems. It is used to specify the continuation method to be used. Valid options are:

**zero** Zero-order continuation.

**first** First-order continuation.

**hzero** Zero-order hunting.

| | |
|---|---|
| **hfirst** | First-order hunting. |
| **loca** | Use the Library of Continuation Algorithms (LOCA). |

If this card is not present, then none of the other *Continuation Specifications* cards in this section or *Hunting Specifications* cards in the Hunting section are needed.

## Examples

Sample card for zero-order single-parameter continuation:

```
Continuation = zero
```

Sample card for first-order hunting (multi-parameter continuation):

```
Continuation = hfirst
```

Sample card for any LOCA continuation routine:

```
Continuation = loca
```

## Technical Discussion

Continuation refers to solving a given problem at a series of steps in which one of more numerical values (other than time) appearing in the governing equations and/or boundary conditions is varied between specified limits at specified step increments. If any LOCA algorithm is to be used, refer to the next paragraph; otherwise, set as follows: When there is only one such value, or continuation parameter, option **zero** or **first** is chosen and the information required to identify the parameter and its values is provided in the following cards in this section. When there are two or more parameters which are to be simultaneously varied using hunting continuation, option **hzero** or **hfirst** is chosen and the relevant information for each parameter is provided in the *Hunting Specifications* section.

To use any of the continuation routines in LOCA, option **loca** is chosen and the inputs required in this section will depend on the chosen routine. LOCA offers a number of continuation methods which can also be used with multiple parameters independently of the hunting algorithms.

The zero-order algorithms use the converged solution from each step as the initial guess for the next step, while the first-order algorithms perform a resolve to obtain a parameter sensitivity vector and use this vector to estimate the solution at the next step.

## Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.2 Continuation Type

| Continuation Type = {BC \| MT \| AC \| UM \| UF \| AN} |
|---|

### Description/Usage

This card is required for all continuation methods (including LOCA methods) to identify the type of continuation parameter. Valid options are:

| | |
|---|---|
| **BC** | Boundary condition input float. |
| **MT** | Pre-defined constant material property. |
| **AC** | Augmenting condition (constant value or input float). |
| **UM** | User-defined material property model input float. |
| **UF** | User-defined continuation function list. |
| **AN** | Angular continuation parameter |

The option selected determines which of the subsequent cards in this section are necessary to uniquely identify the continuation parameter.

### Examples

For a parameter which appears in a boundary condition card, use:

```
Continuation Type = BC
```

For a constant material property, use:

```
Continuation Type = MT
```

For a parameter which appears in an augmenting condition card, use:

```
Continuation Type = AC
```

For a parameter used in a user-defined material property model, use:

```
Continuation Type = UM
```

For a user-supplied continuation function set, use:

```
Continuation Type = UF
```

If the continuation parameter will be an angle and trigonometric functions of it will be needed, use:

```
Continuation Type = AN
```

### Technical Discussion

To use the **BC** option, a BC card must be provided for the relevant boundary conditions on the *Boundary Condition Specification* section.

To use the **MT** option, a tag for the relevant property must be defined in the file `mm_mp_const.h`.

To use the **AC** option, an AC card must be provided for the relevant augmenting condition in the *Augmenting Conditions Specifications* section.

To use the **UM** option, a property tag must be defined (as for **MT**), a user model and parameter list must be specified for the relevant property in file `user_mp.c` and this *USER* model must be specified with the correct number of parameter values in the relevant material (*.mat) file.

To use the **UF** option, a list of user-defined continuation conditions (functions) must be provided in the function *update_user_parameter*, which is in the file `user_continuation.c`.

The **AN** (angular parameter) option works differently in that the quantities to be updated use trigonometric functions (e.g. sin, cos) of this angle, rather than the angle itself. These functions are specified in the CC or TC cards which follow - these cards have completely different interpretations than with any of the other *Continuation Type* options (see the entry for the CC card). At least one such card must be provided.

## Theory

No Theory.

## FAQs

No FAQs.

### References

No References.

## 3.2.3 Number of user continuation functions

| Number of user continuation functions = <integer> |
|---|

### Description/Usage

This card is required for all continuation problems when the selected *Continuation Type* is "UF".

| | |
|---|---|
| <integer> | the number of functions provided by the user in the function "update_user_parameter" (in file user_continuation.c). These functions identify quantities (e.g. BC floats, material properties) which must be updated on each continuation step and then calculates new values for them on each parameter update. |

### Examples

If there are three quantities to be updated at each continuation parameter step, use:

```
Number of user continuation functions = 3
```

### Technical Discussion

In order to use the "UF" continuation type, a list of functions which calculate each update value as a function of the continuation parameter λ must be provided and compiled in file "user_continuation.c". This approach differs from using multiple continuation conditions (specified by CC cards) in that the continuation parameter specified in the input deck is not used directly, but instead through these functions, parameter updates are performed. Thus, the BC/MT ID cards are not actually used (as this information is provided along with the functions), but the range and step size input cards are used. λ can then be used as a progress parameter (i.e. range from 0 to 1) as long as the user-provided functions are based on the λ range specified in these cards.

The number of user continuation functions must be provided in advance so that the correct number of copies of the *User_Continuation_Info* structure (cpuc) can be allocated.

The file user_continuation.c provides templates for two different user continuation function sets: update_user_parameter (for the first parameter) and

update_user_TP_parameter (for the second parameter). The latter is used only with LOCA bifurcation tracking algorithms; a different card is used to enumerate these functions.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.4 Boundary condition ID

| Boundary condition ID = <integer> |
|---|

### Description/Usage

This card is required for continuation problems when the specified *Continuation Type* is **BC** or **AC**.

| | |
|---|---|
| <integer> | the zero-based numerical position of the BC or AC card (within the list in the *Boundary* or *Augmenting Condition Specifications* section) which contains the continuation parameter. If there are *N* BC or AC cards, the top card would be number 0 and the bottom card would be number N-1. |

### Examples

If the continuation parameter is included in the third BC card from the top of the list, this card should be:

```
Boundary condition ID = 2
```

If the continuation parameter is included in the first (or only) AC card, use:

```
Boundary condition ID = 0
```

### Technical Discussion

Note that the BC list written to the screen when *Goma* is launched with the -bc_list command line option (which prints the active boundary conditions with their assigned indices) is one-based, such that the above BC would be numbered 3. If this list is used to determine the ID number, be careful to subtract one from the number on the screen.

Note that the BC ID cards for continuation are being “borrowed” for similar use by augmenting conditions, in lieu of creating a second set of cards.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References

## 3.2.5 Boundary condition data float tag

| Boundary condition data float tag = <integer> |
|---|

### Description/Usage

This card is required for continuation problems when the specified *Continuation Type* is **BC** or **AC**. It is used to identify which of the floating-point inputs on the targeted BC (or user-supplied AC) card is to be used as the continuation parameter.

| | |
|---|---|
| <integer> | *n*, the zero-based numerical position (from left to right) of this value among the float inputs defined for this BC or user-supplied AC. |
| | -1, the constant value (e.g. volume, flux) is specified |

### Examples

A typical BC card may have a defined input format as follows:

```
BC = {BC_name} <integer1> <integer2> <float1> <float2> <float3>
```

If the desired continuation parameter is <float3> above, then use:

```
Boundary condition data float tag = 2
```

If you are using an augmenting condition and wish to continue in its constant value, use:

```
Boundary condition data float tag = -1
```

If you have specified a user-defined AC in file `user_ac.c` which has a list of two floats and wish to continue in the first of these floats, use:

```
Boundary condition data float tag = 0
```

### Technical Discussion

There are many different input formats defined for BC cards. If in doubt, look up the correct format for the specific BC of interest.

Note that the BC ID cards for continuation are being "borrowed" for similar use by augmenting conditions, in lieu of creating a second set of cards.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.6 Material id

| Material id = <integer> |
|---|

### Description/Usage

This card is required for continuation problems when the specified *Continuation Type* is either "MT" or "UM". It identifies which of the materials specified for the problem has the property which will be used as the continuation parameter, and corresponds to the block number assigned to this material in the input Exodus file. If there is only one

material, this number will always be 1. If there are two or more materials, they would be numbered starting with 1.

### Examples

For a problem with only one material, use:

```
Material id = 1
```

If there are several materials and the continuation parameter is a property of aluminum (with properties given in file “aluminum.mat”) which occupies block 4 of the input Exodus file, there will be a subsection for aluminum in the Problem Description section which will start with the card:

```
MAT = aluminum 4
```

In this case, use:

```
Material id = 4
```

### Technical Discussion

This index number differs from most others in that it is one-based (to be consistent with the input file) even though the Goma internal index is zero-based. Accordingly, there is never a case where material 0 would be specified.

## Theory

No Theory.

## FAQs

No FAQs.

## References

No References.

## 3.2.7 Material property tag

| Material property tag = <integer> |
|---|

### Description/Usage

This card is required for continuation problems when the specified *Continuation Type* is “MT” or “UM”. It identifies the property of the relevant material to be used as the continuation parameter. <integer> is the 4-digit property tag number assigned to that property in the file “mm_mp_const.h” as follows:

```
#define TAGC_<PROPERTY_NAME> xxxx
```

### Examples

If the continuation parameter is heat capacity (TAGC_HEAT_CAPACITY = 1600), use:

```
Material property tag = 1600
```

### Technical Discussion

Although a card for the chosen material property is still required in the *..mat file for the relevant material, the property value specified there is overwritten by the continuation parameter.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.8 Material property tag subindex

| Material property tag subindex = <integer> |
|---|

### Description/Usage

This card is required for continuation problems when the specified *Continuation Type* is “UM”. It identifies the continuation parameter by its zero-based position (left to right) within a list of float parameters specified for a user-defined property of a material.

### Examples

Consider a case where the desired continuation parameter is the temperature dependence of surface tension. Currently, doing this in Goma requires a surface tension model to be supplied in “user_mp.c” such as:

```
sigma = param[0] - param[1] * T;
dsigma_dT = -param[1];
```

Here, the continuation parameter would be param[1]. To employ this model, the *..mat file would include the card:

```
Surface Tension = USER <float1> <float2>
```

To specify param[1] as the continuation parameter, use:

```
Material property tag subindex = 1
```

### Technical Discussion

The number of floats assigned for a given user property model is determined by counting the number supplied with the property model card in the *.mat file. If N floats are given there, then they are assigned indices from 0 to N-1 in the param[] array. Accordingly, the subindex number must not exceed N-1, or an error will occur.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.9 Initial parameter value

| Initial parameter value = <float> |
|---|

### Description/Usage

This card is required for all continuation problems. <float> is the value assigned to the chosen continuation parameter on the first step

### Examples

To continue in a chosen parameter from 1 to 5 in increments of 0.5, use:

```
Initial parameter value = 1.0
```

### Technical Discussion

No technical discussion.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.10 Final parameter value

| Final parameter value = <float> |
|---|

### Description/Usage

This card is required for all continuation problems. <float> is the parameter value at which continuation is to end.

### Examples

To continue in a chosen parameter from 1 to 5 in increments of 0.5, use:

```
Final parameter value = 5.0
```

### Technical Discussion

The final parameter value may be higher or lower than the initial parameter value, allowing for continuation in either direction. The size of the final step will be adjusted if necessary to reach this exact value. When doing LOCA arc length continuation, however, this can only approximately be done.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.11 delta_s

| delta_s = <float> |
|---|

### Description/Usage

This card is required for all continuation problems. It specifies the size of the first parameter step to be taken; viz. $\lambda_1 = \lambda_0 +$ delta_s.

<float> the size of the first parameter step.

### Examples

To continue in a chosen parameter from 1 to 5 at increments of 0.5, use:

```
delta_s = 0.5
```

### Technical Discussion

If a continuation step fails when using LOCA with a constant specified step, the step size is initially halved and then allowed to increase back up to *delta_s* over a number of steps.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.12 Maximum number of path steps

| Maximum number of path steps = <integer> |
|---|

### Description/Usage

This card is required for all continuation problems. <integer> is the maximum number of continuation steps which may be taken. Continuation will stop after this number of steps even if the final parameter value is not reached.

### Examples

To continue in a chosen continuation parameter from 1 to 5 at increments of 0.5 for a problem which has difficulty converging, and allow a maximum of 25 steps, use:

```
Maximum number of path steps = 25
```

### Technical Discussion

The initial continuation step, and every subsequent step attempt (successful or not), count toward this maximum.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.13 Maximum path value

| Maximum path value = N/A |
|---|

### Description/Usage

This card is no longer used. The “Final parameter value” card now indicates the end of the continuation parameter range.

### Examples

None.

### Technical Discussion

None.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.14 Minimum path step

| Minimum path step = <float> |
|---|

### Description/Usage

This card is required for all continuation problems. <float> is the absolute value of the smallest allowable increment in the continuation parameter between steps.

### Examples

To set a minimum parameter step size of 0.1, use:

```
Minimum path step = 0.01
```

This can also be entered in scientific notation as follows:

```
Minimum path step = 1.0e-2
```

### Technical Discussion

When convergence failure occurs on a continuation step, it is re-attempted at half of the previous step size until convergence is achieved. The value set for this cards places a limit on how small a step can be taken, and aborts continuation if the step size falls below this value. This can be used to avoid a large number of unnecessary step attempts, such as when the parameter reaches a problem stability limit.

When using arc length continuation, an approximate limit is placed on the arc length step size.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.15 Maximum path step

| Maximum path step = <float> |
|---|

### Description/Usage

This card is required for all continuation problems. <float> is the absolute value of the maximum allowable increment in the continuation parameter between steps.

### Examples

To allow the continuation parameter step size to increase but not allow a step size larger than 2, use:

```
Maximum path step = 2.0
```

### Technical Discussion

For LOCA continuation problems, the function simple_step_control determines a step size for each step after the first. For continuation without LOCA, the function path_step_control performs this task. In either case, if a step size is computed which exceeds this limit, it is truncated to this value. For LOCA arc length continuation, an approximate limit is placed on the arc length step size.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.16 Path step parameter

| Path step parameter = N/A |
|---|

### Description/Usage

This card is no longer applicable.

### Examples

None.

### Technical Discussion

None.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.17 Path step error

| Path step error = N/A |
|---|

### Description/Usage

This card is no longer applicable.

### Examples

None.

### Technical Discussion

None.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.18 Continuation Printing Frequency

| Continuation Printing Frequency = <integer> |
|---|

### Description/Usage

This card is required for all continuation problems. It is used to specify that the solution is to be written to the EXODUS (*.exoII) and ASCII (*.dat) output files after a specified number of steps.

<integer> *N*, frequency of writing a continuation step

### Examples

To write the solution after every step, use:

```
Continuation Printing Frequency = 1
```

To write the solution after every third step, use

```
Continuation Printing Frequency = 3
```

To output the solution only after the last step, use:

```
Continuation Printing Frequency = N+1
```

where *N* is the maximum number of path steps (the last step is always written).

### Technical Discussion

This card can be used to avoid generating very large output files, or when only the final step is of interest.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.19 Second frequency

| Second frequency = N/A |
|---|

### Description/Usage

This card is no longer applicable. Printing frequency is now determined solely by the Continuation Printing Frequency card.

### Examples

None.

### Technical Discussion

None.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.20 LOCA method

| LOCA method = {ss \| zero \| first \| alc \| tp \| pf\| hp} |
|---|

### Description/Usage

This optional card is used to specify one of the LOCA continuation algorithms. Valid options are:

| | |
|---|---|
| **ss** | Steady state continuation, order to be indicated by the *Continuation order* card. |
| **zero** | Zero-order continuation. |
| **first** | First-order continuation. |
| **alc** | Arc length continuation. |
| **tp** | Turning point (fold) bifurcation tracking. |
| **pf** | Pitchfork (symmetry breaking) bifurcation tracking. |
| **hp** | Hopf bifurcation tracking. |

This card is pertinent only if "loca" is indicated in the *Continuation* card. If not present, the default is zero.

### Examples

To do first-order continuation using the LOCA algorithm, use:

```
LOCA method = first
```

To do turning point tracking in LOCA, use:

```
LOCA method = tp
```

### Technical Discussion

Zero and first order continuation methods in LOCA are similar to those in *ac_conti.c*.

Arc length continuation is a bordered method to map solution in parameter-solution space; this method has additional optional input cards.

Turning point tracking is a bordered method to map a stability boundary in a two-parameter space; this method has additional required input cards.

Pitchfork tracking also has required cards (as for turning point tracking) to identify the second parameter, and requires an asymmetry eigenvector to be input via a separate EXODUS II file.

Hopf bifurcation tracking also requires cards to identify the second parameter, an initial guess of the bifurcation frequency (Ω), and the names of two EXODUS II input files for the real and imaginary parts of the initial complex null vector such that the corresponding eigenvalue is Ωi with zero real part. These eigenvectors would be generated by an eigensolve at a known Hopf point. To use this algorithm, the Aztec "komplex" library (currently residing in Trilinos) must be compiled in with *Goma* and the KOMPLEX flag included in the DEFINES list.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.21 Continuation order

| Continuation order = {0 \| 1 \| 2} |
|---|

### Description/Usage

This card is optional and is used to indicate one of the LOCA continuation methods. Valid options are:

| | |
|---|---|
| 0 | Zero-order continuation. |
| 1 | First-order continuation. |

| | |
|---|---|
| 2 | Arc length continuation. |

This card is pertinent only when the chosen *LOCA method* is **ss**. If pertinent and not present, the default is zero.

### Examples

For LOCA first-order continuation, use:

```
Continuation order = 1
```

For LOCA arc length continuation, use:

```
Continuation order = 2
```

### Technical Discussion

This card merely provides an alternate way of choosing the continuation order.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.22 LOCA print level

| |
|---|
| LOCA print level = {0 \| 2 \| 5 \| 8} |

### Description/Usage

This optional card is used to control the amount of screen output generated by the LOCA continuation algorithms. Valid options are:

| | |
|---|---|
| 0 | No screen output at all, not even error messages. |
| 2 | Minimal screen output, including error messages. |
| 5 | Most print statements executed. |

| | |
|---|---|
| 8 | All print statements executed. |

This card is pertinent only if "loca" is indicated in the *Continuation* card. If not present, the default is 5.

### Examples

To suppress all screen output generated by LOCA, use:

```
LOCA print level = 0
```

To include all screen output generated by LOCA, use:

```
LOCA print level = 8
```

### Technical Discussion

All print statements in LOCA are carried out conditionally on ProcID=0 and on the print level in effect (set by this card) being greater than or equal to a value assigned to that statement. Print statements in the interface to LOCA (file ac_loca.c) are not affected by this level at this time.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.23 Perturbation magnitude

| |
|---|
| Perturbation magnitude = <float> |

### Description/Usage

This optional card is used to control the perturbation parameter used by LOCA bordering algorithms for sensitivity calculations. If the card is not present, this parameter defaults to a value of 1.0e-9

### Examples

To set the perturbation parameter to 1.0e-4, use:

```
Perturbation magnitude = 1.0e-4
```

### Technical Discussion

A typical sensitivity resolve in LOCA is of the form J du/dp = - dR/dp, where u is the unknown vector and d/dp is a derivative with respect to some continuation parameter. Here, dR/dp would be found as [ R(p+ε) - R(ε)] / ε, where ε is a function of the perturbation magnitude. Theory suggests that a good typical value may be about 1.0e-6. However, practice has shown that some free surface problems may require lower values, and some other algorithms (such as pitchfork tracking) may require higher values. The is one input that can be adjusted for any particular problem.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.24 LOCA print level

| LOCA print level = {0 \| 2 \| 5 \| 8} |
|---|

### Description/Usage

This optional card is used to control the amount of screen output generated by the LOCA continuation algorithms. Valid options are:

| | |
|---|---|
| 0 | No screen output at all, not even error messages. |
| 2 | Minimal screen output, including error messages. |
| 5 | Most print statements executed. |
| 8 | All print statements executed. |

This card is pertinent only if “loca” is indicated in the *Continuation* card. If not present, the default is 5.

### Examples

To suppress all screen output generated by LOCA, use:

```
LOCA print level = 0
```

To include all screen output generated by LOCA, use:

```
LOCA print level = 8
```

### Technical Discussion

All print statements in LOCA are carried out conditionally on ProcID=0 and on the print level in effect (set by this card) being greater than or equal to a value assigned to that statement. Print statements in the interface to LOCA (file ac_loca.c) are not affected by this level at this time.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.25 ALC Desired solution fraction

| ALC Desired solution fraction = <float> |
|---|

### Description/Usage

This card is optional. It is used only in the LOCA arc length continuation algorithm to establish scaling factors for the solution vector in the arc length equation.

| | |
|---|---|
| <float> | the fraction of arc length due to the parameter. If not present, the default value is 0.5 (equal contributions from the parameter and the solution). |

### Examples

To specify a target solution contribution to the arc length equation of 30%, use:

```
ALC Desired solution fraction = 0.3
```

### Technical Discussion

This algorithm borders the problem with a linearized arc length equation:

$$\Delta\lambda(d\lambda/ds)_0 + \Delta\mathbf{x}(d\mathbf{x}/ds)_0 = \Delta s \quad \textbf{(3-1)}$$

where $\lambda$ is the continuation parameter, $\mathbf{x}$ is the solution vector, and S is the arc length. When the parameter contribution becomes too large, such as in the vicinity of a turning point, the effectiveness of the bordering algorithm is diminished and convergence becomes more difficult. This is taken care of by scaling and periodically rescaling the solution contribution to achieve a desired ratio.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.26 ALC Max. parameter sensitivity

| ALC Max. parameter sensitivity = <float> |
|---|

### Description/Usage

This card is optional. It is used only in the LOCA arc length continuation algorithm to establish scaling factors for the solution vector in the arc length equation.

| | |
|---|---|
| <float> | the fraction of arc length due to the parameter that will not be exceeded; when this fraction is found to exceed <float>, a new scaling factor is calculated to reset the fraction to the value designated on the *ALC Desired solution fraction* card. |

If not present, the default value is 1.0 (which disables rescaling).

### Examples

To specify a maximum parameter contribution to the arc length equation of 80%, use:

```
ALC Max. parameter sensitivity = 0.8
```

### Technical Discussion

This algorithm borders the problem with a linearized arc length equation:

$$\Delta\lambda(d\lambda/ds)_0 + \Delta\mathbf{x}(d\mathbf{x}/ds)_0 = \Delta s \quad \textbf{(3-2)}$$

where $\lambda$ is the continuation parameter, $\mathbf{x}$ is the solution vector, and S is the arc length. When the parameter contribution becomes too large, such as in the vicinity of a turning point, the effectiveness of the bordering algorithm is diminished and convergence becomes more difficult. This is taken care of by scaling and periodically rescaling the solution contribution to achieve a desired ratio, provided a value less than one is specified on this card.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.27 ALC Tangent factor exponent

| ALC Tangent factor exponent = <float > |
|---|

### Description/Usage

This card is optional and used only in the LOCA arc length continuation algorithm. It is used to provide more control of step size when doing continuation near a turning point.

| | |
|---|---|
| <float> | a non-negative exponent used to specify the desired degree of control. If not present, this value defaults to zero and additional control is suppressed. |

### Examples

To specify a tangent factor exponent of one, use:

```
ALC Tangent factor exponent = 1.0
```

### Technical Discussion

Following each arc length step taken (after the first step), a tangent factor is calculated as the direction cosine between the current and previous solution vectors. This provides an indication of how the solution path is changing with the continuation parameter: if this value is nearly one, the solution is changing nearly linearly with the parameter; smaller values indicate more nonlinear solution sensitivity. The purpose of the tangent factor is to detect when the solution path is approaching a turning point bifurcation, so that smaller steps can be taken. When a positive value is specified, the direction cosine is raised to this power and the result is used as a multiplicative factor to reduce the size of the next step which would otherwise be attempted. Thus, higher exponent values will cause a larger number of steps to be taken as the solution continues around turning points.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.28 ALC Tangent factor step limit

| |
|---|
| ALC Tangent factor step limit = <float> |

### Description/Usage

This card is optional and is used only in the LOCA arc length continuation algorithm.

| | |
|---|---|
| <float> | a lower bound for the tangent factor. If the step just completed produced a tangent factor larger than this limit, it will be repeated at a smaller step size. If not present, this value defaults to zero and this step size limit is not imposed. |

This option is needed only for problems which involve very large changes in solution behavior with the continuation parameter which would preclude continuation past a turning point bifurcation in the event of a single step being large enough that convergence could not be attained beyond that point. Therefore, this option should rarely be needed.

### Examples

If step sizes are to be kept small enough that the tangent factor does not fall below 0.5, use:

```
ALC Tangent factor step limit = 0.5
```

### Technical Discussion

The tangent factor is the direction cosine between the solution vectors at the start and end of each continuation step (after the first). If an exponent is specified on the *ALC Tangent factor exponent* card, the factor is raised to that power. If the tangent factor thus obtained falls below the value (<float>) specified on this card, the step is treated as a failed step and is repeated at half of the initial size, just as if it hadn't converged.

## Theory

No Theory.

## FAQs

No FAQs.

## References

No References.

## 3.2.29 Number of continuation conditions

| Number of continuation conditions = <integer> |
|---|

### Description/Usage

This card is required only when using a LOCA continuation algorithm for a problem in which two or more quantities must be updated on each continuation parameter step. This enables multiple updates by a method similar to hunting. If a positive <integer> value is specified, say N, then there are N such quantities (including the continuation parameter) to be updated, and N-1 continuation condition (CC=...) cards must follow this card. Valid values of <integer> are:

| | |
|---|---|
| N>0 | Read N-1 CC cards. |
| 0 | No additional continuation conditions. |
| -1 | Count CC cards (end list with "END OF CC" card). |
| -2 | Read hunting condition (HC=...) cards instead of CC cards. |

The "-1" option works the same way as for similar cards (e.g. BC, HC). If not present, this number defaults to zero (specified parameter is the only continuation condition).

### Examples

If a problem contains four boundary conditions which include the continuation parameter as an argument, use:

```
Number of continuation conditions = 4
```

This would be followed by a list of three CC cards.

Alternatively, use the count option as follows:

```
Number of continuation conditions = -1
```

This would be followed by the same three CC cards, then an *END OF CC* card.

If HC cards for these BC's already exist and you want to read them in lieu of CC cards for continuation in LOCA, use:

```
Number of continuation conditions = -2
```

Note that this would require four HC cards for the same problem.

### Technical Discussion

When multiple continuation conditions are to be used in LOCA, the correct number of copies of the cpcc structure must be allocated -- this card indicates how many such copies are needed and sets the global variable nCC. When -1 is specified, the CC cards are first counted and nCC is set to one more than the number of cards present because the entries for the first copy (cpcc[0]) are taken from the previous continuation parameter cards (unlike in hunting, the entries on these cards are not overwritten).

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.30 CC

| CC = {char_string} <integer_list> <float_list> |
|---|

### Description/Usage

This card is required for each additional quantity (material property parameter or boundary condition card argument) which must be updated each time the continuation parameter is changed in any LOCA continuation algorithm. It is used to provide all of the information required to uniquely identify the quantity to be updated and the relationship which must be maintained between its value and the continuation parameter as the latter changes.

The general syntax for this card is shown above; the specific number of inputs required varies according to the option entered for the parameter selected for continuation. This parameter is {char_string}, indicating the type of quantity to be updated. The valid options are:

| | |
|---|---|
| **BC** | Boundary condition float argument. |
| **MT** | Constant material property. |
| **AC** | Augmenting condition (constant value or float argument). |

**UM** User-defined material property model float argument.

**AN** Angular continuation parameter.

The required number and meaning of the <integer> entries depends on {char_string}. NB, the range flag will always be the last integer entry on the card and is used to indicate the meaning of <float1> and <float2> (if required) in establishing the relationship between the update value $v_i$ and the continuation parameter $\lambda$. Note that the first two integers identify the quantity to be updated, and the third describes the relationship between this quantity and the continuation parameter. The following definitions apply to all *Continuation Type* options except AN:

**BC**

integer1 BCID - Zero-based position of relevant BC card.

integer2 DFID - Zero-based float argument number on this BC card.

integer3 Range flag, as follows:

- 0 $v_i = \lambda$, floats not needed.
- 1 range of $v_i$ specified
  - <float1> - $v_i$ minimum
  - <float2> - $v_i$ maximum
- 2 linear variation
  - <float1> - initial $v_i$ value
  - <float2> - $dv_i/d\lambda$
- 3 polynomial variation ($v_i = C_1 + C_2 * \lambda^{C_3}$)
  - <float1> - $C_1$
  - <float2> - $C_2$
  - <float3> $C_3$

**MT**

integer1 MTID - One-based material number index.

integer2 MPID - Property tag number (assigned in *mm_mp_const.h*).

integer3 Range flag, as follows:

- 0 $v_i = \lambda$, floats not needed.
- 1 range of $v_i$ specified
  - <float1> - $v_i$ minimum
  - <float2> - $v_i$ maximum
- 2 linear variation
  - <float1> - initial $v_i$ value
  - <float2> - $dv_i/d\lambda$
  - <float2> - $dv_i/d\lambda$
- 3 polynomial variation ($v_i = C_1 + C_2 * \lambda^{C_3}$)
  - <float1> - $C_1$
  - <float2> - $C_2$

<float3> $C_3$

**AC**

integer1 BCID - Zero-based position of relevant AC card.

integer2 DFID - Either zero-based data float (for user-supplied AC's) or -1 to indicate the AC constant value (e.g. volume, flux).

integer3 Range flag, as follows:

0 $v_i = \lambda$, floats not needed.

1 range of $v_i$ specified
<float1> - $v_i$ minimum
<float2> - $v_i$ maximum

2 linear variation
<float1> - initial $v_i$ value
<float2> - $dv_i/d\lambda$
<float2> - $dv_i/d\lambda$

3 polynomial variation ($v_i = C_1 + C_2 * \lambda \hat{} C_3$)
<float1> - $C_1$
<float2> - $C_2$
<float3> $C_3$

**UM**

integer1 MTID - One-based material number index.

integer2 MPID - Property tag number (assigned in *mm_mp_const.h*).

integer3 MDID - Zero-based user model float argument number.

integer4 Range flag, as follows:

0 $v_i = \lambda$, floats not needed.

1 range of $v_i$ specified
<float1> - $v_i$ minimum
<float2> - $v_i$ maximum

2 linear variation
<float1> - initial $v_i$ value
<float2> - $dv_i/d\lambda$
<float2> - $dv_i/d\lambda$

3 polynomial variation ($v_i = C_1 + C_2 * \lambda \hat{} C_3$)
<float1> - $C_1$
<float2> - $C_2$
<float3> $C_3$

When the continuation parameter is an angle and the *Continuation Type* option selected is AN, the definition of the "Range flag" integer (the last one in each of the above lists) is as follows:

0 $v_i = C_1 + C_2 \sin \lambda$.
<float1> - C1
<float2> - C2

1 $v_i = C_1 + C_2 \cos \lambda$
<float1> - $C_1$
<float2> - $C_2$

2 $v_i = C_1 + C_2 \tan \lambda$
<float1> - $C_1$
<float2> - $C_2$

3 $v_i = C_1 + C_2 \sin \lambda + C_3 \cos \lambda$
<float1> - $C_1$
<float2> - $C_2$
<float3> $C_3$

It is not currently possible to combine angular functions of the continuation parameter with linear/polynomial ones. Note that continuation parameter angles are specified in degrees; *Goma* converts them internally to radians.

There are no defaults for any entries on this card.

## Examples

When one condition is a BC float (fifth BC in list, second of three floats) with the same value as the continuation BC parameter, use:

```
CC = BC 4 1 0
```

If this BC float value is the negative of the continuation BC parameter, which goes from 5 to 10, use:

```
CC = BC 4 1 1 -5.0 -10.0
```

or:

```
CC = BC 4 1 2 -5.0 -1.0
```

where the continuation parameter is the one specified in the preceding cards.

If the constant value of the first AC in the input list is half of the continuation BC parameter, which goes from 5 to 10, use:

```
CC = AC 0 -1 2 2.5 0.5
```

### Technical Discussion

*CC* cards represent one of the options offered in the LOCA interface which perform the function of hunting. When all quantities to be updated at each step are linear functions of the continuation parameter (i.e. $v_i = m\lambda + b$ ), these cards can provide the necessary information with less user input. If any of these values are nonlinear functions other than the two-term polynomial and trigonometric functions specified above (i.e. $v_i = \log\lambda$ ), it will be necessary to create a user-defined continuation condition list and compile it in the function *update_user_parameter* (file `user_continuation.c`). This can be done in lieu of providing CC cards for any LOCA continuation or bifurcation tracking problem.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.31 END OF CC

| END OF CC |
|---|

### Description/Usage

This card is the companion card to the *Number of continuation conditions* card. The *END OF CC* card signals the end of a list of CC cards in the *Continuation Specifications* section of the *Goma* input. When the *Number of continuation conditions* (= -1) is set to negative one, *Goma* will read all the continuation condition cards until this card is encountered in the input file. This card may omitted if the integer *N* on the *Number of continuation conditions* card is not -1.

### Examples

Typical usage of this card is illustrated below:

```
Number of augmenting conditions = -1
.
.
```

```
.
END OF AC
```

### Technical Discussion

See companion card *Number of continuation conditions*.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.32 TP Continuation Type

| TP Continuation Type = {BC \| MT \| AC \| UM \| UF \| AN} |
|---|

### Description/Usage

This card is required only for the LOCA bifurcation tracking algorithms to identify the type of the second (turning point) continuation parameter. Valid options are:

| | |
|---|---|
| BC | Boundary condition input float. |
| MT | Pre-defined constant material property. |
| AC | Augmenting condition (constant value or input float). |
| UM | User-defined material property model input float. |
| UF | User-defined continuation function list. |
| AN | Angular continuation parameter. |

The option selected determines which of the subsequent cards in this section are necessary to uniquely identify the second continuation parameter.

NOTE: The “second continuation parameter” is different from an additional condition on the (first) continuation parameter; in fact, it may have its own additional (and separate) list of continuation conditions, which would be specified with TC cards.

## Examples

If the second parameter is from a boundary condition, use:

```
TP Continuation Type = BC
```

If it is a constant material property, use:

```
TP Continuation Type = MT
```

If it is an augmenting condition constant value or input float, use:

```
TP Continuation Type = AC
```

If it is an input float to a user-defined material property model, use:

```
TP Continuation Type = UM
```

If it is a user-defined continuation function list, use:

```
TP Continuation Type = UF
```

For an angular parameter with trigonometric functions specified in TC cards, use:

```
TP Continuation Type = AN
```

## Technical Discussion

To use the “BC” option, a BC card must be provided for the relevant boundary conditions on the Boundary Condition Specification section.

To use the “MT” option, a tag for the relevant property must be defined in the file “mm_mp_const.h”.

To use the “AC” option, an AC card must be provided for the relevant augmenting conditions on the Augmenting Condition Specification section.

To use the “UM” option, a property tag must be defined (as for “MT”), a user model and parameter list must be specified for the relevant property in file “user_mp.c” and this “USER” model must be specified with the correct number of parameter values in the relevant material (*..mat) file.

To use the “UF” option, a list of user-defined continuation conditions (functions) must be provided in the function “update_user_TP_parameter”, which is in the file user_continuation.c.

To use the “AN” option, the angle is specified in degrees (which are internally converted to radians). One or more TC cards which specify the update quantities as trigonometric functions of this angle must be provided.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.33 Number of user TP continuation functions

| Number of user TP continuation functions = <integer> |
|---|

### Description/Usage

This card is required for all LOCA bifurcation tracking problems when the selected *TP Continuation Type* is “UF”.

| | |
|---|---|
| <integer> | the number of functions provided by the user in the function “update_user_TP_parameter” (in file user_continuation.c). These functions identify quantities (e.g. BC floats, material properties) which must be updated on each continuation step and then calculates new values for them on each update of the second (TP) parameter. |

### Examples

If there are three quantities to be updated for the second (TP) parameter at each continuation step, use:

```
Number of user TP continuation functions = 3
```

### Technical Discussion

In order to use the “UF” continuation type, a list of functions which calculate each update value as a function of the continuation parameter $\lambda$ must be provided and

compiled in file “user_continuation.c”. This approach differs from using multiple TP continuation conditions (specified by TC cards) in that the continuation parameter specified in the input deck is not used directly, but instead through these functions, to perform parameter updates. Thus, the TP versions of the BC/MT ID cards are not actually used (as this information is provided along with the functions), but the range and step size input cards are used. λ can then be used as a progress parameter (i.e. range from 0 to 1) as long as the user-provided functions are based on the λ range specified in these cards.

The number of user TP continuation functions must be provided in advance so that the correct number of copies of the *User_Continuation_Info* structure (tpuc) can be allocated.

The file user_continuation.c provides templates for two different user continuation function sets: update_user_parameter (for the first parameter) and update_user_TP_parameter (for the second parameter). This card pertains only to the first one. The other one is used only with LOCA bifurcation tracking algorithms.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.34 TP Boundary condition ID

| TP Boundary condition ID = <integer> |
|---|

### Description/Usage

This card is required for all LOCA bifurcation tracking algorithms when the selected *TP Continuation Type* is BC or AC. <integer> is the zero-based numerical position of the BC or AC card (within the list in the Boundary or Augmenting Condition Specifications section) which contains the second (TP) continuation parameter. If there are N BC or AC cards, the top card would be number 0 and the bottom card would be number N-1.

### Examples

If the second (TP) parameter is the third float specified on the sixth BC card from the top of the list, use:

```
TP Boundary condition ID = 5
```

If it is in the second AC card fron the top, use:

```
TP Boundary condition ID = 1
```

### Technical Discussion

Note that the BC list written to the screen when Goma is launched with the command line option `-bc_list` is one-based, such that the above BC would be numbered 6. If this list is used to determine the ID number, be careful to subtract one from the number on the screen.

Note that the BC ID cards for continuation are being "borrowed" for similar use by augmenting conditions, in lieu of creating a second set of cards.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.35 TP BC data float tag

| TP BC data float tag = <integer> |
|---|

### Description/Usage

This card is required for all LOCA bifurcation tracking algorithms when the specified *TP Continuation Type* is BC or AC. It is used to identify which of the floating-point inputs in the specified BC or user-supplied AC card is to be used as the second (TP) continuation parameter. <integer> is the zero-based numerical position of this value (left to right) among the float inputs defined for this BC or user-supplied AC. If the

continuation parameter is the constant value (e.g. volume, flux) for an AC, then <integer> should be -1.

### Examples

A typical BC card may have a defined input format as follows:

```
BC = {BCname} <int1> <int2> <float1> <float2> <float3> ...
```

If the desired second (TP) continuation parameter is <float3> above, use:

```
TP BC data float tag = 2
```

If if is the first data float in a list specified for a user-supplied AC, use:

```
TP BC data float tag = 0
```

If it is the constant value for an AC, use

```
TP BC data float tag = -1
```

### Technical Discussion

There are many different input formats defined for BC cards. If in doubt, look up the correct format for the specific BC of interest.

Note that the BC ID cards for continuation are being “borrowed” for similar use by augmenting conditions, in lieu of creating a second set of cards.

## Theory

No Theory.

## FAQs

No FAQs.

## References

No References.

## 3.2.36 TP parameter material id

| TP parameter material id = <integer> |
|---|

### Description/Usage

This card is required for LOCA bifurcation tracking algorithms when the specified *TP Continuation Type* is either MT or UM. It identifies which of the materials specified for the problem has the property which will be used as the second (TP) continuation parameter, and corresponds to the block number assigned to this material in the input Exodus file. If there is only one material, this number will always be 1. If there are two or more materials, they would be numbered starting with 1.

### Examples

For a bifurcation tracking problem with only one material, use:

```
TP parameter material id = 1
```

If there are several materials and the second (TP) continuation parameter is a property of aluminum (with properties given in file “aluminum.mat”) which occupies block 4 of the input Exodus file, there will be a subsection for aluminum in the Problem Description section which will start with the card:

```
MAT = aluminum 4
```

In this case, use:

```
TP parameter material id = 4
```

### Technical Discussion

This index number differs from most others in that it is one-based (to be consistent with the input file) even though the Goma internal index is zero-based. Accordingly, there is never a case where material 0 would be specified.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.37 TP parameter material property tag

| TP parameter material property tag = <integer> |
|---|

### Description/Usage

This card is required for all LOCA bifurcation tracking algorithms when the specified *TP Continuation Type* is MT or UM. It identifies the property of the relevant material to be used as the second (TP) continuation parameter. <integer> is the 4-digit property tag number assigned to that property in the file "mm_mp_const.h" as follows:

```
#define TAGC_<PROPERTY_NAME> xxxx
```

### Examples

If the second (TP) continuation parameter is heat capacity (TAGC_HEAT_CAPACITY = 1600), use:

```
TP parameter material property tag = 1600
```

### Technical Discussion

Although a card for the chosen material property is still required in the *..mat file for the relevant material, the property value specified there is overwritten by the TP continuation parameter.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.38 TP_Material_property_tag_subindex

| TP Material property tag subindex = <integer> |
|---|

### Description/Usage

This card is required for all LOCA bifurcation tracking algorithms when the selected *TP Continuation Type* is UM. <integer> is the zero-based index of the floating-point input for a user-defined material property model which is to be used as the second (TP) continuation parameter.

### Examples

For a simple linear surface tension model $\sigma = a - bT$, the model and its temperature sensitivity would be provided in function "usr_surface_tension (in user_mp.c). To invoke this model, this line would be placed in the *.mat file:

```
Surface Tension = USER <a> <b>
```

To specify the temperature coefficient <b> as the second (TP) continuation parameter, use:

```
TP Material property tag subindex = 1
```

### Technical Discussion

The UM continuation type requires the user to define a material property model and at least one float argument. This input enables access to any argument in the list for selection as the second (TP) continuation parameter.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.39 Initial guess of TP parameter

| Initial guess of TP parameter = <float> |
|---|

### Description/Usage

This card is required for all LOCA bifurcation tracking algorithms. <float> is an initial estimate of the second (TP) parameter value at which a solution bifurcation (e.g. turning point) occurs when the primary parameter is at its specified initial value.

### Examples

To specify an initial guess of 50 for the TP parameter at the specified initial value of the primary parameter, use:

```
Initial guess of TP parameter = 50.0
```

### Technical Discussion

This input provides a starting point for the nonlinear iterations at the first continuation step which lead to a solution of a bordered (augmented) equation system on a bifurcation path (which yields the actual TP parameter value). On subsequent steps, the previous TP parameter value is used as the initial guess. The second (TP) parameter and any continuation conditions specified for it are updated separately from the primary parameter.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.40 TP parameter final value

| TP parameter final value = <float> |
|---|

### Description/Usage

This card is required for all LOCA bifurcation tracking algorithms. <float> is used for two things: 1) the tracking run is considered completed if the TP parameter reaches this value, even if the primary parameter has not yet reached its specified final value, and 2) It is used to compute either the ratio or final value (when the other one is given) for each specified TP continuation condition.

### Examples

If the TP parameter value cannot exceed 100, use:

```
TP parameter final value = 100.0
```

### Technical Discussion

When continuation conditions are assigned to the second (TP) parameter, a range for its value is needed. This range is just the final value from this card minus the initial guess value. When the flag input on a TC card is 1, the range specified for that condition is divided by the TP parameter range to find the update ratio $dv_i / d\lambda_{TP}$, which is used to increment each condition value $v_i$ as new values of the TP continuation parameter $\lambda_{TP}$ are found. Therefore, if the initial value of $\lambda_{TP}$ is not accurately known, it would be safer to change the flag on the TC card to 2 so that the update ratio can be precisely specified.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.41 Null vector restart file

| Null vector restart file = <string> |
|---|

### Description/Usage

This card is used to specify the name of an EXODUS II file containing an initial null vector to be used by the LOCA bifurcation tracking algorithms as follows:

For turning point tracking, this input is optional; if omitted, the algorithm will automatically generate an initial guess for this vector.

For pitckfork tracking, this input is required, and the input vector is also used as an asymmetry vector.

For Hopf tracking, this input is required, and the input vector is the real part of a complex null vector. The corresponding imaginary part is required to be input in a second EXODUS II file - the *Null vector imaginary restart file* card is also required.

### Examples

During a previous run, a turning point was located and confirmed by an eigensolve. The eigenvector was written to "eig_mode1.exoII". To start the turning point tracking algorithm using this null vector, use:

```
Null vector restart file = eig_mode1.exoII
```

During a previous run, a Hopf point was located and confirmed by an eigensolve, which reported the leading eigenvalue as 0 +/- 0.5i. The complex leading eigenvecor was saved in files named "eig_mode1.exoII" and "eig_mode2.exoII". This card should read:

```
Null vector restart file = eig_mode2.exoII
```

### Technical Discussion

If the input EXODUS II file contains more than one time or parameter step, only the last one in the file will be read.

For pitchfork tracking, the problem should be solved on a symmetric FEM mesh (centered at zero).

For Hopf tracking, the Aztec "komplex" library (currently residing in Trilinos) must be compiled in with *Goma* and the KOMPLEX flag included in the DEFINES list. Also, the imaginary eigenvalue associated with this eigenvector must be given in the *Initial Hopf frequency* card.

If running in parallel, this file must be broken up with the brk utility prior to running *Goma*.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.42 Null vector imaginary restart file

| Null vector imaginary restart file = <string> |
|---|

### Description/Usage

This card is required when using the LOCA Hopf bifurcation tracking algorithm. It is used to specify the name of an EXODUS II file containing the imaginary part of an initial null vector to be used by this algorithm, where the real part of this vector is contained in the EXODUS II file named in the *Null vector restart file* card and the corresponding imaginary eigenvalue is provided in the *Initial Hopf frequency* card.

### Examples

During a previous run, a Hopf point was located and confirmed by an eigensolve, which reported the leading eigenvalue as 0 +/- 0.5i. The complex leading eigenvecor was saved in files named “eig_mode1.exoII” and “eig_mode2.exoII”. This card should read:

```
Null vector imaginary restart file = eig_mode2.exoII
```

### Technical Discussion

If the input EXODUS II file contains more than one time or parameter step, only the last one in the file will be read.

For Hopf tracking, the Aztec “komplex” library (currently residing in Trilinos) must be compiled in with *Goma* and the KOMPLEX flag included in the DEFINES list. Also,

the imaginary eigenvalue associated with this eigenvector must be given in the *Initial Hopf frequency* card.

If running in parallel, this file must be broken up with the brk utility prior to running *Goma*.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.43 Initial Hopf frequency

| Initial Hopf frequency = <float> |
|---|

### Description/Usage

This card is required when using the LOCA Hopf bifurcation tracking algorithm. It is used to specify the pure imaginary eigenvalue which corresponds to the input initial null vector for this algorithm, which is read in from the files specified in the *Null vector restart file* and *Null vector imaginary restart file* cards.

### Examples

During a previous run, a Hopf point was located and confirmed by an eigensolve, which reported the leading eigenvalue as 0 +/- 0.5i. To start or restart the Hopf algorithm from that point, this card should read:

```
Initial Hopf frequency = -0.5
```

### Technical Discussion

Experience appears to indicate that the Hopf algorithm tends to seek the negative value of omega, hence faster convergence of the first step may be achieved by specifying the negative value in this card (although either positive or negative will work).

If the input ExodusII file contains more than one time or parameter step, only the last one in the file will be read, so the value specified in this card should be the one for that step.

For Hopf tracking, the Aztec "komplex" library (currently residing in Trilinos) must be compiled in with Goma and the KOMPLEX flag included in the DEFINES list. Also, the imaginary eigenvalue associated with this eigenvector must be given in the "Initial Hopf frequency" card.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.44 Null vector save file

| Null vector save file = <string> |
|---|

### Description/Usage

This optional card is used to specify the name of an EXODUS II file to which the final null vector is to be written when using the LOCA bifurcation tracking algorithms. This vector could then be used to restart the tracking problem from the previous solution. If this card is not present, the default is that the null vector is not saved.

### Examples

To save the null vector from a turning point tracking run to "nv1.exoII", use:

```
Null vector save file = nv1.exoII
```

To save the null vector from a Hopf tracking run to "nv1.exoII" (real) and "nv2.exoII" (imaginary), this card should read:

```
Null vector save file = nv1.exoII
```

### Technical Discussion

The file named in this card is written to only after LOCA completes its run. Intermediate null vectors are not saved unless stability analysis is performed along with continuation. When the Hopf tracking algorithm is used, only the real part of the null vector is written to this file.

If running in parallel, one file is created per processor; these files can be rejoined with the fix utility.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.45 Null vector imaginary save file

| Null vector imaginary save file = <string> |
|---|

### Description/Usage

This optional card is used to specify the name of an EXODUS II file to which the imaginary part of the final null vector is to be written when using the LOCA Hopf bifurcation tracking algorithm. This vector could then be used (along with the separately-saved real part) to restart the Hopf tracking problem from the previous solution. If this card is not present, the default is that the null vector is not saved.

### Examples

To save the null vector from a Hopf tracking run to “nv1.exoII” (real) and “nv2.exoII” (imaginary), this card should read:

```
Null vector save file = nv2.exoII
```

### Technical Discussion

The file named in this card is written to only after LOCA completes its run. Intermediate null vectors are not saved unless stability analysis is performed along with continuation.

If running in parallel, one file is created per processor; these can be rejoined with the fix utility.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 3.2.46 Number of TP continuation conditions

| Number of TP continuation conditions = <integer> |
|---|

### Description/Usage

This card is required only when using a LOCA bifurcation tracking algorithm for a problem in which two or more quantities associated with the second (TP) parameter must be updated on each continuation parameter step. This enables multiple updates by a method similar to hunting. If a positive <integer> value is specified, say N, then there are N such quantities (including the second (TP) continuation parameter) to be updated, and N-1 TP continuation condition (TC=...) cards must follow this card. Valid values of <integer> are:

| | |
|---|---|
| N>0 | Read N-1 TC cards. |
| 0 | No additional TP continuation conditions. |
| -1 | Count TC cards (end list with “END OF TC” card). |

The “-1” option works the same way as for similar cards (e.g. BC, HC). If not present, this number defaults to zero (specified TP parameter is the only TP continuation condition).

## Examples

If a problem contains four boundary conditions which include the second (TP) continuation parameter as an argument, use:

```
Number of TP continuation conditions = 4
```

This would be followed by a list of three TC cards.

Alternatively, use the count option as follows:

```
Number of TP continuation conditions = -1
```

This would be followed by the same three TC cards, then an *END OF TC* card.

## Technical Discussion

When multiple TP continuation conditions are to be used in LOCA, the correct number of copies of the tpcc structure must be allocated -- this card indicates how many such copies are needed and sets the global variable nTC. When -1 is specified, the TC cards are first counted and nTC is set to one more than the number of cards present because the entries for the first copy (tpcc[0]) are taken from the previous TP continuation parameter cards (unlike in hunting, the entries on these cards are not overwritten). While continuation conditions for the primary parameter can be specified using hunting (HC) cards, this cannot be done for the second (TP) parameter. Therefore, -2 is not a valid option for this card -- TP cards must be provided if applicable.

## Theory

No Theory.

## FAQs

No FAQs.

## References

No References.

## 3.2.47 TC

| TC = {string} <int1> <int2> <int3> <int4> <float1> <float2> |
|---|

### Description/Usage

This card is required for each additional quantity (material property parameter or boundary condition card argument) which must be updated each time the second (TP) continuation parameter changes in any LOCA continuation algorithm. It is used to provide all of the information required to uniquely identify the quantity to be updated and the relationship which must be maintained between its value and the TP continuation parameter as the latter changes. The syntax shown above indicates the maximum number of inputs required on this card; as discussed below, some of these entries are required only in certain cases.

{string}indicates the type of quantity to be updated. The valid options are:

| | |
|---|---|
| BC | Boundary condition float argument. |
| MT | Constant material property. |
| AC | Augmenting condition (constant value or float argument). |
| UM | User-defined material property float argument. |

The required number and meaning of the <int> entries depend on {string} as follows:

{string} = BC: Three integers

| | |
|---|---|
| int1 | BCID - Zero-based position of relevant BC card. |
| int2 | DFID - Zero-based float argument number on this BC card. |
| int3 | Range flag (see below). |

{string} = MT: Three integers

| | |
|---|---|
| int1 | MTID - One-based material number index. |
| int2 | MPID - Property tag number (assigned in "mm_mp_const.h"). |
| int3 | Range flag (see below). |

{string} = AC: Three integers

| | |
|---|---|
| int1 | BCID - Zero-based position of relevant AC card. |

| | |
|---|---|
| int2 | DFID - Either zero-based data float (for user-supplied AC's) or -1 to indicate the AC constant value (e.g. volume, flux). |
| int3 | Range flag (see below) |

{string} = UM: Four integers

| | |
|---|---|
| int1 | MTID - One-based material number index. |
| int2 | MPID - Property tag number (assigned in "mm_mp_const.h"). |
| int3 | MDID - Zero-based user model float argument. |
| int4 | Range flag (see below). |

The range flag is always the last integer entry on the card and is used to indicate the meaning of <float1>, <float2>, and <float3> (if required) in establishing the relationship between the update value $v_i$ and the TP continuation parameter $\lambda_{tp}$. For all *TP Continuation Type* options other than AN, the range flag options are as follows:

| | |
|---|---|
| flag = 0 | $v_i = \lambda_{tp}$ (floats not needed). |
| flag = 1 | Range of $v_i$: from <float1> to <float2>. |
| flag = 2 | <float1> is initial $v_i$ value; <float2> is $dv_i/d\lambda_{tp}$. |
| flag = 3 | $v_i$ = <float1> + <float2> * $\lambda_{tp}$^<float3> |

When the second continuation parameter is an angle and the *TP Continuation Type* selected is AN, then the flag definitions are as follows (also see the CC card entry):

| | |
|---|---|
| flag = 0 | $v_i$ = <float1> + <float2> sin $\lambda_{tp}$ |
| flag = 1 | $v_i$ = <float1> + <float2> cos $\lambda_{tp}$ |
| flag = 2 | $v_i$ = <float1> + <float2> tan $\lambda_{tp}$ |
| flag = 3 | $v_i$ = <float1> + <float2> sin $\lambda_{tp}$+ <float3> cos $\lambda_{tp}$ |

There are no defaults for any entries on this card.

## Examples

When one condition is a BC float (third BC in list, second of two floats) with the same value as the TP continuation BC parameter, use:

```
TC = BC 2 1 0
```

If this BC float value is the negative of the TP continuation parameter, which ranges from 1 to 2, use:

```
TC = BC 2 1 1 -1.0 -2.0
```

or:

```
TC = BC 2 1 2 -1.0 -1.0
```

where the TP continuation parameter is the one specified in the preceding cards.

If the constant value of the first AC in the input list is half of the TP continuation BC parameter, which goes from 5 to 10, use:

```
TC = AC 0 -1 2 2.5 0.5
```

### Technical Discussion

TC cards represent one of the options offered in the LOCA interface which perform the function of hunting. When all quantities to be updated at each step are linear functions of the second (TP) continuation parameter (i.e. $v_i = m\lambda_{TP} + b$), these cards are used to provide the necessary information. If these quantities are nonlinear functions of the second parameter other than those provided for above, it will be necessary to specify them in function “update_user_TP_parameter” (file user_continuation.c). This can be done in lieu of providing TC cards for any LOCA bifurcation tracking problem.

## Theory

No Theory.

## FAQs

No FAQs.

## References

No References.

### 3.2.48 END OF TC

| END OF TC |
|---|

#### Description/Usage

This card is the companion card to the *Number of TP continuation conditions* card. The *END OF TC* card signals the end of a list of TC cards in the *Continuation Specifications* section of the *Goma* input. When the *Number of TP continuation conditions* (= -1) is set to negative one, *Goma* will read all the second parameter (TP) continuation condition cards until this card is encountered in the input file. This card may omitted if the integer *N* on the *Number of TP continuation conditions* card is not equal to -1, or if there is no TC card list or no second continuation parameter.

#### Examples

Typical usage of this card is illustrated below:

```
Number of TP continuation conditions = -1
.
.
.
END OF TC
```

#### Technical Discussion

This card is applicable only with the LOCA bifurcation tracking algorithms (turning point, pitchfork, Hopf). See companion card *Number of continuation conditions*.

#### Theory

No Theory.

#### FAQs

No FAQs.

#### References

No References.

## 3.3 Single Parameter Continuation via the Command Line

After a continuation run, *Goma* writes a text file called *goma-cl.txt* that lists the command line

arguments for continuing directly from the command line. This makes it easy for writing scripts that can do multiple continuation sequences without having to provide multiple input files. There may be some benefit for use with other *Goma* wrap-arounds such as Dakota.

Continuation command line arguments take precedence over the values provided in the input file. Continuation is enabled even if **`Continuation = none`** is in the input file. For the `Continuation Specifications` section above (Section ?), *goma-cl.txt* contains the following after running the continuation sequence once:

```
goma -a -i input -cb 0.000000e+00 -ce 8.000000e+02 \
                 -cd 2.000000e+01 -cn 20 \
                 -cm 0 -ct 2 \
                 -c_mn 1 -c_mp 1700
```

These are the command line arguments required to replicate the main continuation commands specified in the input file. The backslashes indicate that all terms should appear on one line. All the continuation command line arguments (list by typing `goma –help`) are as follows:

```
-cb FLT          Continuation: Start value
-ce FLT          Continuation: Final value
-cd FLT          Continuation: Path step, ds
-cn INT          Continuation: Max number of path steps
-cm INT          Continuation: Method
-ct INT          Continuation: Type
-c_bc INT        Continuation: Boundary condition ID
-c_df INT        Continuation: BC Data Float ID
-c_mn INT        Continuation: Material ID
-c_mp INT        Continuation: Material property ID
```

The method and type flags are as follows: 0 for the method (`cm`) indicates zeroth order continuation and 1 first order continuation, while for the type, 1 indicates a boundary condition and 2 a material property. If the continuation parameter is a boundary condition data float, the *Goma* boundary condition and data float indices must be provided on the command line or in the input file. Similarly, if the continuation parameter is a material property, the material number and property tag must be provided on the command line or in the input file.

## 3.4 Multi-parameter Continuation with Hunting

If the continuation method card (Section 3.2.1) is set to `hzero` or `hfirst`, that is

Continuation = *hzero/hfirst*

the hunting continuation capability is used. This allows multi-parameter continuation in either

boundary condition data floats or material properties or both. `hzero` and `hfirst` refer to the zeroth order and the first order methods, respectively. Both methods are enabled with a ramping feature so that all parameters can be ramped from a start to an end value in the specified number of steps. Hunting continuation can be used to continue in a single parameter instead of the standard `zero` and `first` methods. The maximum number of steps and printing frequency are specified in the `Continuation Specifications` section (?? and ??) described above.

When the hunting capability is being used, *Goma* ignores the `Continuation Type`, `Boundary condition ID, Boundary condition data float tag, Material id, Material property tag, Material property tag subindex, Initial parameter value, Final parameter value,` and `delta_s` cards in the `Continuation Specifications` section. (Those input records not ignored are the `Continuation, Maximum number of path steps, Minimum` and `Maximum path steps,` and `Continuation Printing Frequency`.)

A new section must be added to the *Goma* input file to identify the hunting continuation parameters, their start and end value, and step size information. It has the following form:

```
-------------------------------------------------------------
                   Hunting Specifications
-------------------------------------------------------------
Number of hunting conditions  = -1
HC = BC 4    0 1 0.0   1.0  1.0 0.0001 1000.0
HC = MT 1 1700 1 0.0 100.0 10.0 0.0001 1000.0
END OF HC
```

In the above example, two hunting conditions are specified. One indicates continuation in a boundary condition data float and another in the density. Other entries are described below.

## 3.5 Hunting Specifications

### 3.5.1 Number of hunting conditions

| Number of hunting conditions = <int> |
|---|

#### Description/Usage

This card is required when multiple parameter continuation is to be performed in either of these two cases: (1) The Goma hunting routine is invoked by setting the *Continuation* card to hzero or hfirst, or (2) Continuation is to be performed in LOCA

(*Continuation* = loca) and the *Number of continuation conditions* card is set to -2, indicating that the continuation conditions are to be taken from HC cards instead of CC cards. In either case, if <int> is positive then <int> cards will be read below this card, and if <int> is -1 then the HC cards between this card and the *END OF HC* card will be counted and read.

### Examples

Consider a continuation problem in which the continuation parameter is used by three different boundary conditions, hence three continuation or hunting conditions are required. For either of the two cases described above, use:

```
Number of hunting conditions = 3
```

to be followed by three HC cards (one for each of these BC's).

Alternatively, count the cards by using:

```
Number of hunting conditions = -1
```

to be followed by the same three HC cards, then an "END OF HC" card.

### Technical Discussion

A backward compatibility routine has been provided such that LOCA can use either HC cards or CC cards. Accordingly, when HC cards are used (even with LOCA), the cards which identify and set values and step sizes for the continuation parameter in the *Continuation Specifications* section, while still required to be present, are overwritten with the values provided in the first HC card.

## Theory

No Theory.

## FAQs

No FAQs.

## References

No References.

## 3.5.2 HC

| HC = {string} <int1> <int2> <int3> <float1> <float2> <float3> <float4> <float5> |
|---|

### Description/Usage

This card is required for each of two or more quantities which must be updated at each continuation step when the Goma hunting routine is invoked (*Continuation* = hzero or hfirst) or when LOCA is invoked (*Continuation* = loca) and *Number of continuation conditions* is set to -2 to indicate that HC cards (rather than CC cards) are to be read. The arguments contain the information required to uniquely identify the quantity to be updated and set its value range, initial step size, and step size limits. These arguments are as follows:

{string} indicates the type of quantity to be updated. The valid options are:

| | |
|---|---|
| BC | Boundary condition float argument. |
| MT | Constant material property. |
| AC | Augmenting condition (constant value or flaot argument). |
| UM | User-defined material property model float argument. |

NOTE: When "UM" is chosen, a fourth <int> argument will be needed!

The required number and meaning of the <int> entries depend on {string} as follows:

{string} = BC: Three integers

| | |
|---|---|
| int1 | BCID - Zero-based position of relevant BC card. |
| int2 | DFID - Zero-based float argument number on this BC card. |
| int3 | Step control flag (see below). |

{string} = MT: Three integers

| | |
|---|---|
| int1 | MTID - One-based material index number. |
| int2 | MPID - Property tag number (assigned in "mm_mp_const.h"). |
| int3 | Step control flag (see below). |

{string} = AC: Three integers:

| | |
|---|---|
| int1 | BCID - Zero-based position of relevant AC card. |
| int2 | DFID - Either zero-based data float (for user-supplied AC's) or -1 to indicate the AC constant value (e.g. volume, flux). |
| int3 | Step control flag (see below). |

{string} = UM: Four integers

| | |
|---|---|
| int1 | MTID - One-based material index number |
| int2 | MPID - Property tag number (assigned in "mm_mp_const.h"). |
| int3 | MDID - Zero-based user model float argument number. |
| int4 | Step control flag (see below). |

The step control flag is always the last integer entry on the card. Valid options are:

| | |
|---|---|
| 0 | No step control |
| 1 | Use step control |

When step control is used, the step size is set to a constant value equal to the range (end_value - start_value) divided by one less than the maximum number of steps. Otherwise, the step size is recalculated at each step based on the Newton convergence rate.

The float entries are as follows:

| | |
|---|---|
| float1 | start_value: Value on first continuation step. |
| float2 | end_value: Value at the end of the continuation run. |
| float3 | start_step: Step size taken after first continuation step. |
| float4 | min_step_value: Minimum allowable step size. |
| float5 | max_step_value: Maximum allowable step size. |

If the step size calculated for the next step exceeds max_step_value, it is reset to this value. If it falls below min_step_value, continuation is aborted.

There are no defaults for any entries on this card.

### Examples

Consider a continuation problem in which three sides of a box are to be held at a constant temperature T, which is to be increased from 50 to 100 in constant steps of 10. This is specified with Dirichlet BC cards at the top of the BC list.

The corresponding HC cards would then be used:

```
HC = BC 0 0 1 50.0 100.0 10.0 10.0 10.0
HC = BC 1 0 1 50.0 100.0 10.0 10.0 10.0
HC = BC 2 0 1 50.0 100.0 10.0 10.0 10.0
```

To allow the step size to start at 10 and range from 5 to 20, use for the first card:

```
HC = BC 0 0 0 50.0 100.0 10.0 5.0 20.0
```

and make the corresponding changes to the other two cards.

### Technical Discussion

HC cards differ from CC cards in that step control is done independently for each card (thus care must be taken to ensure that the updated quantities change in consistent linear proportions to each other) and in that they can be used with or without LOCA.

An alternative to HC cards is to use user-defined continuation functions, which allow nonlinear functions to be specified (see file “user_continuation.c”).

## Theory

No Theory.

## FAQs

No FAQs.

## References

No References.

### 3.5.3 END OF HC

| END OF HC |
|---|

#### Description/Usage

This card is the companion card to the *Number of hunting conditions* card. The *END OF HC* card signals the end of the *Hunting Conditions Specifications* section of the *Goma* input. When the *Number of hunting conditions* (= -1) is set to negative one, *Goma* will read all the hunting condition cards until this card is encountered in the input file. This card may omitted if the integer *N* on the *Number of hunting conditions* card is not -1.

#### Examples

Typical usage of this card is illustrated below:

```
Number of hunting conditions = -1
.
.
.
END OF HC
```

#### Technical Discussion

See companion card *Number of hunting conditions*.

#### Theory

No Theory.

#### FAQs

No FAQs.

#### References

No References.

## 3.6 Continuation condition (CC) cards

### 3.6.1 Purpose

Some continuation problems involve the requirement to update a number of different conditions of the problem each time a continuation step is taken. For example, if there are three boundary condition cards which have a common float input, then each must be updated at every step. It may also be necessary to match some quantity on both sides of a material interface. Or, it may be of interest to examine the response of a system to two variables changing simultaneously, even if the physics of the problem does not demand it.

Hunting conditions, as discussed in Section 1.5, provide a mechanism for fulfilling this need by providing ID and step information for each quantity to be updated (using HC cards). However, this requires the use of one of two designated *Goma* hunting algorithms (hzero and hfirst), which are handled separately from all other continuation algorithms. In order to have the ability to handle multiple updates per step with the more advanced algorithms in LOCA, it is necessary to integrate this feature with each of the algorithms. For many such problems, the quantities to be updated maintain a linear relationship to each other (i.e. y=mx+b) as they change, either by design or by necessity. In fact, this linear relationship is required for the hunting algorithms to work.

### 3.6.2 Usage

The approach taken for LOCA is to designate one of the update quantities as the “master” and place the ID and step information for it in the cards pertaining to a single continuation parameter (which the hunting algorithms require but do not use). Any other update quantities, or “continuation conditions”, are considered “slaves” and identified by inserting a CC card, which need only include the necessary ID information and a means to specify the linear relationship between changes in it and changes in the “master”, as described in Section 1.2. Thus, there is one fewer CC card required. A global variable indicates how many continuation conditions are in effect, so that when an an update call comes from LOCA, the correct number of parameter update calls are made (even if only one).

Here is how the previous example continuation section could be restated using CC cards:

```
-----------------------------------------------------------

 Continuation Specifications

-----------------------------------------------------------

Boundary condition ID = 4

Boundary condition data float tag= 0

...
```

```
Initial parameter value= 0.0

Final parameter value= 10.0

delta_s= 1.0

Minimum path step= 0.0001

Maximum path step= 1000.0

LOCA method= zero

Number of continuation conditions= 2 (or -1)

CC = MT 1 1700 2 0.0 10.0

END OF CC
```

The CC card here is set up such that MT 1 ID 1700 starts at 0.0 and changes by 10x whenever BC 4 float 0 changes by x. Note also that the number of continuation conditions is 2, but only one CC card is required because the information for the first (the “master”) is taken from the preceding cards.

To maintain backward compatibility, LOCA can use an existing set of hunting (HC) cards in lieu of converting them to CC cards, and will extract the information it needs properly. To do so, it is only necessary to include the card:

```
Number of continuation conditions= -2
```

which will tell the input parser to read hunting cards even if a hunting algorithm is not specified by the Continuation card.

## 3.7 User-defined continuation conditions

### 3.7.1 Purpose

When each quantity which must be updated during continuation bears a linear (y=b+mx) relationship to the designated continuation parameter, then these relationships can be adequately specified with either HC or CC cards. The CC and TC cards can also be used to specify a two-term polynomial relationship of the type y=b+mx^n, or relationships involving the trigonometric functions (e.g. sin, cos) of an angular continuation parameter. However, some problems may require more complicated relationships between these values, such as multiple-term polynomials, exponentials, etc. In other cases, the previous solution may be required to obtain new values. Such

requirements would then require a mechanism for the user to construct custom continuation conditions for the problem at hand. This mechanism is provided in the file user_continuation.c.

## 3.7.2 Usage

The file user_continuation.c contains two template functions for the user to supply custom continuation conditions: update_user_parameter (for the primary continuation parameter) and update_user_TP_parameter (for the second parameter used in LOCA bifurcation tracking algorithms). Detailed instructions and example entries are provided in the comments of this file.

To use this feature, first identify all quantities $q_i$ which are to be updated at each continuation step. Then designate a single scalar continuation parameter λ such that each update quantity can be readily expressed as a function of λ (and possibly the solution), viz.

$$q_i = f(\lambda, x, \dot{x}, x_{AC}) \tag{3-3}$$

where x is the current solution vector, xdot is its time derivative, and x_AC is the vector of extra unknowns when augmenting conditions are used. Note that l may or may not be one of the update quantities $q_i$. It may be used, for example, as a "progress parameter", as theta is for hunting, or it can be a dimensionless group (such as Reynolds number) which is based on the input quantities but may not be equal to any of them. When user-defined continuation is specified for a parameter, the parameter ID card (e.g. BCID, MTID) card inputs are not used (but must still be present). However, the step inputs (Initial and final values, starting, minimum, and maximum step size) are used for λ. All other necessary information will be placed in the relevant user continuation function and compiled into the code; no HC/CC/TC cards are needed.

The procedure for supplying the continuation conditions consists of three steps for each quantity: 1) providing the necessary type and ID information, 2) defining the new value in terms of λ and the solution, and 3) pasting in a standard update call. This may be done for either or both parameters for LOCA bifurcation tracking algorithms. Once these functions are entered, it is then necessary to recompile Goma to include them. User-defined continuation is then invoked by specifying continuation type "UF" (user functions) in the input file and providing the number of continuation functions (conditions) entered for the relevant parameter. For example, to continue in web speed for a slot coater, the relevant update quantities are the two Dirichlet BC's for the web and the fluid in contact with it, and since it is necessary to vary the inflow proportionally to maintain constant downstream film thickness, each of the three floats of a GD_PARAB BC also. Thus, there are five quantities. Here is how the functions would be entered when l is the web speed:

```
{
  static int first_cp = 1;
  int first_tp = -1;
  int n=0;
  int Type, BCID, DFID, MTID, MPID, MDID;
  double value;
```

```
/* Declare any additional variables here */
  double f2, f3, f4;


/* If using this function, comment out this line. */
/*EH(-1, "No user continuation conditions entered!");*/

/* Evaluate any intermediate quantities and/or assign constants here */
  f2 = -9.684754406;
  f3 =  318.0179944;
  f4 = -2486.458128;

/* Enter each continuation condition in sequence in this space */

/* Condition 0:  Slip velocity */

/* Condition 0:  Slip velocity */

/* ID's */
  Type = BC;
  BCID = 13;
  DFID = 1;

/* Value */
  value = lambda;

/* Update call - copy from example */
  n = do_user_update(n, first_cp, first_tp, Type, BCID, DFID,
                     MTID, MPID, MDID, value, cx, exo, dpi);

/* Condition 1:  Fluid velocity along web */

/* ID's */
  Type = BC;
  BCID = 14;
  DFID = 0;

/* Value */
  value = lambda;

/* Update call */
  n = do_user_update(n, first_cp, first_tp, Type, BCID, DFID,
                     MTID, MPID, MDID, value, cx, exo, dpi);

/* Condition 2:  Inlet velocity GD_PARAB first float */

/* ID's */
  Type = BC;
  BCID = 11;
  DFID = 0;

/* Value */
  value = f2 * lambda;

/* Update call */
  n = do_user_update(n, first_cp, first_tp, Type, BCID, DFID,
                     MTID, MPID, MDID, value, cx, exo, dpi);

/* Condition 3:  Inlet velocity GD_PARAB second float */
```

```
/* ID's */
  Type = BC;
  BCID = 11;
  DFID = 1;

/* Value */
  value = f3 * lambda;

/* Update call */
  n = do_user_update(n, first_cp, first_tp, Type, BCID, DFID,
                     MTID, MPID, MDID, value, cx, exo, dpi);

/* Condition 4:  Inlet velocity GD_PARAB third float */

/* ID's */
  Type = BC;
  BCID = 11;
  DFID = 2;

/* Value */
  value = f4 * lambda;


/* Update call */
  n = do_user_update(n, first_cp, first_tp, Type, BCID, DFID,
                     MTID, MPID, MDID, value, cx, exo, dpi);

/* Done */
  first_cp = 0;
  return;

}
```

Then, to invoke this function, include the following Continuation input cards:

```
Continuation Type                          = UF

Number of user continuation functions      = 5
```

# 3.8 Examples

## 3.8.1 Zeroth order continuation in lid speed in lid driven cavity problem

In this example, the continuation parameter is the lid speed. The relevant sections of the input file for continuation, `ldc.input-c0ls`, are as follows:

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

```
Continuation                     = zero
Continuation Type                = BC
Boundary condition ID            = 4
Boundary condition data float tag= 0
Material id                      = 1
Material property tag            = 1700
Material property tag subindex   = 0
Initial parameter value          = {speed}
Final parameter value            = {10.0*speed}
delta_s                          = {1.0*speed}
Maximum number of path steps     = 10
Minimum path step                = 1.0e-05
Maximum path step                = 2.0
Continuation Printing Frequency  = 1

-------------------------------------------------------------
        Boundary Condition Specifications
-------------------------------------------------------------
.
BC = U NS 3 {speed}
```

The Aprepro variable {speed} is set equal to 1.0 in the include file geometry.in.

The *Goma* output for this zeroth order continuation is:

```
    ---------------------------------
        Zero Order Continuation:
        Step number:    1 of   10 (max)
        Attempting solution at:
        BCID=  4 DFID=    0 Parameter=  1.000000e+00 delta_s= 1.000000e+00

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
11:33:27 [0] 2.6e-12 2.2e-11 5.3e-12 1.9e-10 1.3e-08 1.2e-09  1  4.0e-02/6.0e-02
scaled solution norms   3.386794e+01  4.267119e-01  2.028370e+00

        Step accepted, parameter = 1.000000e+00

        ---------------------------------
        Zero Order Continuation:
        Step number:    2 of   10 (max)
        Attempting solution at:
        BCID=  4 DFID=    0 Parameter=  2.000000e+00 delta_s= 1.000000e+00

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
11:33:27 [0] 9.4e-02 1.9e+00 3.5e-01 3.4e+01 4.9e+02 7.0e+01  1  4.0e-02/6.0e-02
11:33:27 [1] 3.1e-04 1.4e-02 1.2e-03 1.3e-01 4.0e+00 4.2e-01  1  4.0e-02/6.0e-02
```

```
11:33:27 [2] 6.5e-09 3.6e-07 2.6e-08 2.5e-06 8.3e-05 8.9e-06  1  4.0e-02/6.0e-02
11:33:27 [3] 1.1e-16 7.2e-15 3.8e-16 1.3e-14 7.3e-13 5.8e-14  1  4.0e-02/5.0e-02
scaled solution norms   6.808972e+01  8.532607e-01  4.056800e+00

        Step accepted, parameter = 2.000000e+00

        ----------------------------------
        Zero Order Continuation:
        Step number:    3 of   10 (max)
        Attempting solution at:
        BCID=  4 DFID=    0 Parameter=  4.000000e+00 delta_s= 2.000000e+00

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
11:33:27 [0] 1.9e-01 3.8e+00 6.9e-01 6.9e+01 9.7e+02 1.4e+02  1  4.0e-02/6.0e-02
11:33:28 [1] 1.2e-03 5.6e-02 4.7e-03 5.2e-01 1.6e+01 1.7e+00  1  4.0e-02/6.0e-02
11:33:28 [2] 1.0e-07 5.8e-06 4.2e-07 4.0e-05 1.3e-03 1.4e-04  1  4.0e-02/5.0e-02
11:33:28 [3] 1.7e-16 1.9e-14 1.0e-15 9.1e-14 6.2e-12 5.3e-13  1  5.0e-02/5.0e-02
scaled solution norms   1.376161e+02  1.705662e+00  8.115703e+00

        Step accepted, parameter = 4.000000e+00

        ----------------------------------
        Zero Order Continuation:
        Step number:    4 of   10 (max)
        Attempting solution at:
        BCID=  4 DFID=    0 Parameter=  6.000000e+00 delta_s= 2.000000e+00

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
11:33:28 [0] 1.9e-01 3.8e+00 6.9e-01 7.0e+01 9.7e+02 1.4e+02  1  4.0e-02/6.0e-02
11:33:28 [1] 1.3e-03 5.6e-02 4.7e-03 5.4e-01 1.6e+01 1.7e+00  1  4.0e-02/6.0e-02
11:33:28 [2] 1.0e-07 5.8e-06 4.2e-07 4.0e-05 1.3e-03 1.4e-04  1  4.0e-02/6.0e-02
11:33:28 [3] 5.3e-16 2.6e-14 1.4e-15 1.1e-13 7.8e-12 6.8e-13  1  4.0e-02/6.0e-02
scaled solution norms   2.086216e+02  2.557158e+00  1.218044e+01

        Step accepted, parameter = 6.000000e+00

        ----------------------------------
        Zero Order Continuation:
        Step number:    5 of   10 (max)
        Attempting solution at:
        BCID=  4 DFID=    0 Parameter=  8.000000e+00 delta_s= 2.000000e+00

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
11:33:28 [0] 1.9e-01 3.8e+00 6.8e-01 7.2e+01 9.7e+02 1.4e+02  1  4.0e-02/6.0e-02
11:33:29 [1] 1.3e-03 5.6e-02 4.7e-03 5.6e-01 1.6e+01 1.7e+00  1  4.0e-02/6.0e-02
11:33:29 [2] 1.0e-07 5.8e-06 4.2e-07 4.0e-05 1.3e-03 1.4e-04  1  4.0e-02/5.0e-02
```

```
11:33:29 [3] 4.5e-16 3.3e-14 1.7e-15 1.4e-13 4.8e-12 4.1e-13  1  5.0e-02/5.0e-02
scaled solution norms   2.811481e+02  3.408727e+00  1.625473e+01

        Step accepted, parameter = 8.000000e+00

         ******** LAST PATH STEP!

        ----------------------------------
        Zero Order Continuation:
        Step number:    6 of   10 (max)
        Attempting solution at:
        BCID=  4 DFID=    0 Parameter=  1.000000e+01 delta_s= 2.000000e+00

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
11:33:29 [0] 1.9e-01 3.8e+00 6.8e-01 7.4e+01 9.8e+02 1.4e+02  1  3.0e-02/4.0e-02
11:33:29 [1] 1.3e-03 5.6e-02 4.7e-03 5.8e-01 1.6e+01 1.7e+00  1  3.0e-02/4.0e-02
11:33:29 [2] 1.0e-07 5.7e-06 4.2e-07 4.0e-05 1.3e-03 1.4e-04  1  3.0e-02/4.0e-02
11:33:29 [3] 4.6e-16 3.9e-14 2.0e-15 2.1e-13 1.8e-11 1.6e-12  1  3.0e-02/3.0e-02
scaled solution norms   3.552368e+02  4.259312e+00  2.034230e+01


Step accepted, parameter = 1.000000e+01.


         I will continue no more!
         No more continuation for you!

-done

Proc 0 runtime:       0.04 Minutes.
```

Shown below are plots of the STREAM function at four "`TIMES`", where the time value actually corresponds to values of the continuation parameter. From left to right and top to bottom, STREAM function plots for a lid `speed` of 1, 2, 4 and 10 are shown.

### 3.8.2 First order continuation in density in the lid driven cavity problem

In this example, the continuation command line feature is used. The input file is exactly the same as the the one in the previous example (Section ??). The *Goma* command line which will be used is:

```
goma -a -i input -cb 0.000000e+00 -ce 8.000000e+02 \

     -cd 5.000000e+01 -cn 10 \

     -cm 1 -ct 2 \

     -c_mn 1 -c_mp 1700
```

The input file for Problem 3.8.1 used a BC continuation type. Note that this problem will use a

MT continuation type since the command line takes overrides or takes precedence over the input file. The continuation parameter is the density (property tag 1700) in material number 1. The initial step size is 50.0 and maximum steps is 10. Also note that the continuation method (`-cm`) will be first order rather than zero order as specified in the input file.

The *Goma* Newton iteration history is:

```
         ----------------------------------
          First Order Continuation:
          Step number:    1 of   10 (max)
          Attempting solution at:
          MTID=  0 MPID= 1700 Parameter=  0.000000e+00 delta_s= 5.000000e+01

                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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
12:28:58 [0] 7.4e-04 1.6e-02 1.7e-03 1.8e-01 4.2e+00 4.8e-01  1  4.0e-02/5.0e-02
12:28:58 [1] 3.9e-17 3.8e-15 2.0e-16 1.2e-14 6.2e-13 5.2e-14  1  4.0e-02/6.0e-02
scaled solution norms   3.369280e+01  4.267761e-01  2.028496e+00


First Order Continuation resolve time:  3.0e-02
         Step accepted, parameter = 0.000000e+00

         ----------------------------------
         First Order Continuation:
         Step number:    2 of   10 (max)
         Attempting solution at:
         MTID=  0 MPID= 1700 Parameter=  5.000000e+01 delta_s= 5.000000e+01

                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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
12:28:58 [0] 6.4e-03 2.8e-01 2.3e-02 2.2e+00 6.0e+01 6.5e+00  1  5.0e-02/6.0e-02
12:28:58 [1] 4.4e-05 1.8e-03 1.5e-04 1.5e-02 4.4e-01 5.2e-02  1  4.0e-02/6.0e-02
12:28:58 [2] 1.3e-09 8.1e-08 5.4e-09 5.1e-07 1.7e-05 1.6e-06  1  4.0e-02/6.0e-02
12:28:59 [3] 5.9e-17 3.9e-15 2.1e-16 1.6e-14 8.5e-13 7.4e-14  1  4.0e-02/6.0e-02
scaled solution norms   4.439571e+01  4.605871e-01  2.209286e+00


First Order Continuation resolve time:  3.0e-02
         Step accepted, parameter = 5.000000e+01

         ----------------------------------
         First Order Continuation:
         Step number:    3 of   10 (max)
         Attempting solution at:
         MTID=  0 MPID= 1700 Parameter=  1.500000e+02 delta_s= 1.000000e+02

                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)
```

```
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
12:28:59 [0] 2.4e-02 9.6e-01 7.8e-02 5.3e+00 1.9e+02 1.9e+01  1  4.0e-02/5.0e-02
12:28:59 [1] 1.8e-03 1.2e-01 8.4e-03 1.3e+00 3.0e+01 3.4e+00  1  5.0e-02/5.0e-02
12:28:59 [2] 3.0e-05 1.7e-03 1.2e-04 7.7e-03 3.9e-01 3.4e-02  1  4.0e-02/6.0e-02
12:28:59 [3] 1.3e-08 4.9e-07 3.7e-08 2.8e-06 1.2e-04 1.1e-05  1  4.0e-02/6.0e-02
12:28:59 [4] 9.9e-16 4.5e-14 3.8e-15 5.5e-13 1.2e-11 1.4e-12  1  4.0e-02/6.0e-02
scaled solution norms   7.097491e+01  8.127580e-01  3.517973e+00


First Order Continuation resolve time:  3.0e-02

Step accepted, parameter = 1.500000e+02.


... Steps 4 through 8 skipped ....


       ---------------------------------
        First Order Continuation:
        Step number:    9 of   10 (max)
        Attempting solution at:
        MTID=  0 MPID= 1700 Parameter=  7.500000e+02 delta_s= 1.000000e+02

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
12:29:02 [0] 1.4e-03 1.4e-01 7.4e-03 7.9e-01 5.5e+01 4.8e+00  1  3.0e-02/4.0e-02
12:29:02 [1] 6.8e-06 5.1e-04 3.4e-05 3.3e-02 1.0e+00 1.1e-01  1  3.0e-02/4.0e-02
12:29:02 [2] 4.4e-09 2.9e-07 2.0e-08 3.7e-06 1.4e-04 1.3e-05  1  3.0e-02/4.0e-02
12:29:02 [3] 2.1e-16 1.6e-14 8.8e-16 3.7e-13 1.0e-11 1.1e-12  1  3.0e-02/4.0e-02
scaled solution norms   2.150763e+02  6.096488e+00  2.094008e+01


First Order Continuation resolve time:  2.0e-02
        Step accepted, parameter = 7.500000e+02

         ******** LAST PATH STEP!

        ---------------------------------
        First Order Continuation:
        Step number:   10 of   10 (max)
        Attempting solution at:
        MTID=  0 MPID= 1700 Parameter=  8.000000e+02 delta_s= 5.000000e+01

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
12:29:02 [0] 2.9e-04 2.9e-02 1.5e-03 1.8e-01 1.3e+01 1.2e+00  1  3.0e-02/4.0e-02
12:29:02 [1] 2.4e-07 2.0e-05 1.3e-06 1.2e-03 4.1e-02 4.1e-03  1  3.0e-02/4.0e-02
12:29:02 [2] 5.7e-12 3.7e-10 2.3e-11 5.2e-09 2.1e-07 1.9e-08  1  3.0e-02/4.0e-02
scaled solution norms   2.272351e+02  6.691865e+00  2.287778e+01
```

```
First Order Continuation resolve time:  2.0e-02

Step accepted, parameter = 8.000000e+02.


         I will continue no more!
         No more continuation for you!

-done


Proc 0 runtime:       0.08 Minutes.
```

The iteration history for steps 4 through 8, corresponding to path values of 250., 350., 450., 550. and 650. respectively, have been omitted. Convergence of the Newton iteration was similar to those shown above. A plot of the STREAM function for densities 0., 150., 450. and 800. is shown in Figure 3.2 below.

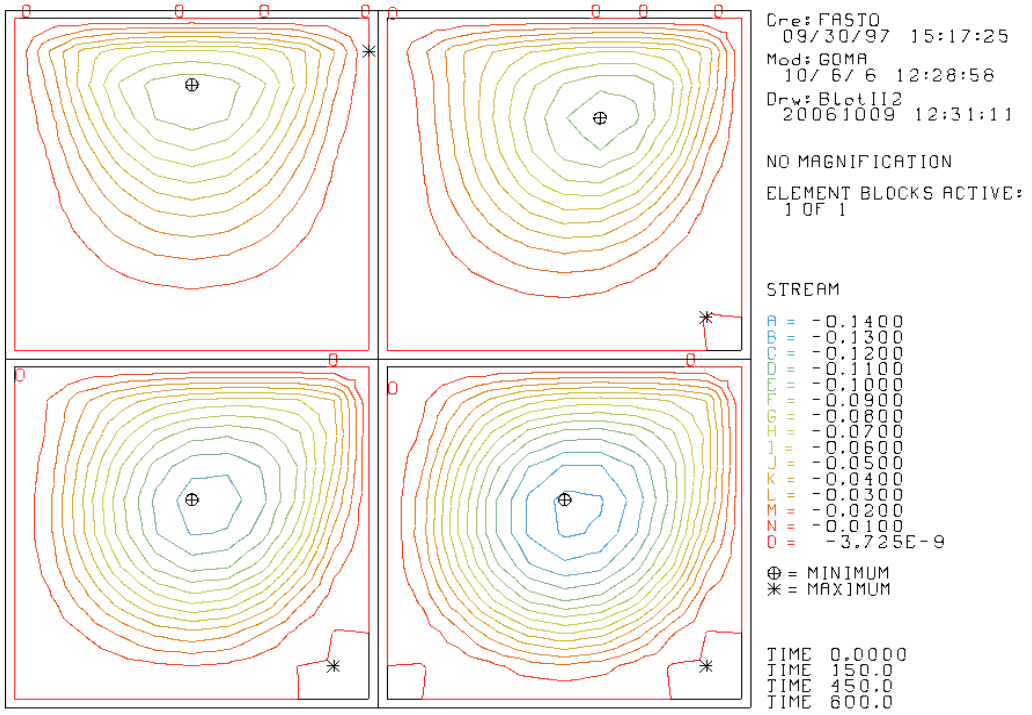



Figure 3.2 STREAM Function for Densities of 0., 150., 450., and 800.

### 3.8.3 First order continuation in vacuum pressure in slot coating model

For a basic description of this problem, a single layer slot coater, refer to *Goma* tutorial GT-002. In this problem, side set 23 is the location of the upstream meniscus, the boundary on which the vacuum pressure is applied. To continue in the vacuum pressure, the `Continuation Specifications` section is added to the input file `sc1.input`. The key cards to note are:

Continuation Type = BC

$$Boundary condition ID= 4

Boundary condition data float tag= 1

Note that the Boundary condition ID card is commented out. Instead, the identity of the capillary boundary condition is indicated by changing:

BC = CAPILLARY **SS** 23 {surface_tension} {vacuum} 0.0

to

BC = CAPILLARY SC 23 {surface_tension} {vacuum} 0.0

in the Boundary Condition Specifications section. *Goma* automatically detects the index of this boundary condition and tags it for continuation. The first two floats, {surface_tension} and {vacuum}, are set in included file slot.geom and inserted by *Aprepro*. The vacuum pressure is specified by the second data float, thus the proper data float tag is 1.

*Goma* is run as usual, that is:

```
goma -a -i sc1.input
```

The *Goma* iteration history is excerpted below:

```
---------------------------------
        First Order Continuation:
        Step number:    1 of   50 (max)
        Attempting solution at:
        BCID= 26 DFID=    1 Parameter= -3.675000e+03 delta_s= 1.000000e+02

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:06:09 [0] 3.7e-04 2.3e-02 1.3e-03 2.1e+03 2.5e+04 3.6e+03  1  2.4e+00/4.8e+00
13:06:17 [1] 3.2e-04 5.7e-03 9.4e-04 4.0e+02 9.9e+03 6.6e+02  1  1.6e+00/4.6e+00
13:06:23 [2] 1.0e-03 1.2e-03 1.0e-03 3.0e+01 1.6e+03 7.2e+01  1  1.6e+00/4.6e+00
13:06:30 [3] 7.5e-08 1.7e-06 2.1e-07 2.8e-01 9.3e+00 7.0e-01  1  1.6e+00/4.6e+00
```

```
13:06:36 [4] 8.8e-12 1.4e-10 1.9e-11 2.4e-05 9.8e-04 6.6e-05  1  1.6e+00/4.6e+00
scaled solution norms   1.646916e+04  3.018376e+02  1.399544e+03


First Order Continuation resolve time:  1.3e+00
        Step accepted, parameter = -1.144400e+04

        ----------------------------------
        First Order Continuation:
        Step number:    2 of   16 (max)
        Attempting solution at:
        BCID= 24 DFID=    1 Parameter= -1.140000e+04 delta_s= 4.400000e+01

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:06:44 [0] 3.4e-06 3.4e-05 4.2e-06 4.4e+00 3.6e+01 5.5e+00  1  1.6e+00/4.6e+00
13:06:50 [1] 9.9e-10 1.3e-08 2.2e-09 3.7e-03 1.5e-01 7.8e-03  1  1.6e+00/4.6e+00
13:06:57 [2] 4.9e-14 2.6e-13 5.0e-14 1.0e-08 1.2e-07 1.2e-08  1  1.6e+00/4.6e+00
scaled solution norms   1.642634e+04  3.012333e+02  1.395819e+03


First Order Continuation resolve time:  1.3e+00
        Step accepted, parameter = -1.140000e+04

        ----------------------------------
        First Order Continuation:
        Step number:    3 of   16 (max)
        Attempting solution at:
        BCID= 24 DFID=    1 Parameter= -1.130000e+04 delta_s= 1.000000e+02

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:07:05 [0] 1.4e-05 1.7e-04 1.8e-05 2.3e+01 2.0e+02 2.9e+01  1  1.6e+00/4.6e+00
13:07:11 [1] 1.0e-07 3.8e-07 1.1e-07 5.9e-02 1.9e+00 1.3e-01  1  1.6e+00/4.6e+00
13:07:18 [2] 2.0e-12 1.9e-11 3.7e-12 8.0e-06 2.7e-04 1.5e-05  1  1.6e+00/4.6e+00
scaled solution norms   1.632909e+04  2.998605e+02  1.387368e+03


First Order Continuation resolve time:  1.3e+00
        Step accepted, parameter = -1.130000e+04

        ----------------------------------
        First Order Continuation:
        Step number:    4 of   16 (max)
        Attempting solution at:
        BCID= 24 DFID=    1 Parameter= -1.120000e+04 delta_s= 1.000000e+02

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
```

```
13:07:26 [0] 1.1e-05 1.7e-04 1.7e-05 2.6e+01 1.9e+02 3.3e+01  1  1.6e+00/4.6e+00
13:07:32 [1] 1.7e-08 2.6e-07 4.0e-08 5.7e-02 2.2e+00 1.3e-01  1  1.6e+00/4.6e+00
13:07:39 [2] 1.5e-13 3.9e-12 4.3e-13 1.6e-06 3.7e-05 2.5e-06  1  1.6e+00/4.6e+00
scaled solution norms   1.623191e+04  2.984892e+02  1.378943e+03


First Order Continuation resolve time:  1.3e+00
        Step accepted, parameter = -1.120000e+04

        ----------------------------------
        First Order Continuation:
        Step number:    5 of   16 (max)
        Attempting solution at:
        BCID= 24 DFID=    1 Parameter= -1.110000e+04 delta_s= 1.000000e+02

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:07:47 [0] 1.0e-05 1.7e-04 1.6e-05 3.1e+01 2.1e+02 3.9e+01  1  1.6e+00/4.6e+00
13:07:53 [1] 1.3e-07 5.1e-07 1.9e-07 2.1e-01 3.1e+00 2.7e-01  1  1.6e+00/4.6e+00
13:07:59 [2] 2.7e-12 2.7e-11 5.3e-12 8.5e-06 5.2e-04 2.5e-05  1  1.6e+00/4.6e+00
scaled solution norms   1.613483e+04  2.971207e+02  1.370550e+03


First Order Continuation resolve time:  1.3e+00
        Step accepted, parameter = -1.110000e+04

        ----------------------------------
        First Order Continuation:
        Step number:    6 of   16 (max)
        Attempting solution at:
        BCID= 24 DFID=    1 Parameter= -1.100000e+04 delta_s= 1.000000e+02

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:08:08 [0] 1.6e-05 1.9e-04 2.2e-05 2.5e+01 1.9e+02 3.1e+01  1  1.6e+00/4.6e+00
13:08:14 [1] 3.4e-07 9.6e-07 3.8e-07 6.8e-01 6.8e+00 9.8e-01  1  1.6e+00/4.6e+00
13:08:20 [2] 7.5e-12 7.8e-11 1.5e-11 8.2e-05 1.9e-03 1.3e-04  1  1.6e+00/4.6e+00
scaled solution norms   1.603786e+04  2.957550e+02  1.362190e+03


First Order Continuation resolve time:  1.3e+00
        Step accepted, parameter = -1.100000e+04

        ----------------------------------
        First Order Continuation:
        Step number:    7 of   16 (max)
        Attempting solution at:
        BCID= 24 DFID=    1 Parameter= -1.090000e+04 delta_s= 1.000000e+02

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:08:28 [0] 1.3e-05 1.8e-04 1.9e-05 3.0e+01 2.0e+02 3.7e+01  1  1.6e+00/4.6e+00
13:08:35 [1] 2.2e-07 4.7e-07 2.2e-07 5.8e-01 2.9e+00 7.3e-01  1  1.6e+00/4.6e+00
13:08:41 [2] 3.6e-07 3.6e-07 3.6e-07 8.0e-01 5.7e+00 1.1e+00  1  1.6e+00/4.6e+00
13:08:47 [3] 3.0e-08 3.0e-08 3.0e-08 6.7e-02 4.8e-01 9.4e-02  1  1.6e+00/4.6e+00
13:08:54 [4] 7.5e-14 7.7e-13 1.4e-13 9.1e-07 1.8e-05 1.4e-06  1  1.6e+00/4.6e+00
scaled solution norms   1.594099e+04  2.943911e+02  1.353857e+03


First Order Continuation resolve time:  1.3e+00
        Step accepted, parameter = -1.090000e+04

        ----------------------------------
        First Order Continuation:
        Step number:    8 of   16 (max)
        Attempting solution at:
        BCID= 24 DFID=    1 Parameter= -1.080000e+04 delta_s= 1.000000e+02

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:09:02 [0] 9.1e-06 1.8e-04 1.6e-05 3.4e+01 2.2e+02 4.0e+01  1  1.6e+00/4.6e+00
13:09:08 [1] 4.9e-08 2.5e-07 5.6e-08 9.2e-02 6.9e-01 1.2e-01  1  1.6e+00/4.6e+00
13:09:15 [2] 2.8e-13 4.0e-12 6.1e-13 3.3e-06 8.3e-05 5.6e-06  1  1.6e+00/4.6e+00
scaled solution norms   1.584426e+04  2.930274e+02  1.345549e+03


First Order Continuation resolve time:  1.3e+00
        Step accepted, parameter = -1.080000e+04

        ----------------------------------
        First Order Continuation:
        Step number:    9 of   16 (max)
        Attempting solution at:
        BCID= 24 DFID=    1 Parameter= -1.070000e+04 delta_s= 1.000000e+02

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:09:23 [0] 1.7e-05 2.0e-04 2.3e-05 2.3e+01 2.1e+02 2.9e+01  1  1.6e+00/4.6e+00
13:09:29 [1] 1.8e-07 5.8e-07 2.0e-07 2.7e-01 2.5e+00 3.9e-01  1  1.6e+00/4.6e+00
13:09:36 [2] 3.4e-12 4.5e-11 7.3e-12 2.0e-05 9.7e-04 5.3e-05  1  1.6e+00/5.5e+00
scaled solution norms   1.574764e+04  2.916613e+02  1.337256e+03


First Order Continuation resolve time:  1.6e+00
        Step accepted, parameter = -1.070000e+04

        ----------------------------------
        First Order Continuation:
        Step number:   10 of   16 (max)
        Attempting solution at:
```

```
        BCID= 24 DFID=    1 Parameter= -1.060000e+04 delta_s= 1.000000e+02

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:09:45 [0] 1.3e-05 1.9e-04 2.0e-05 3.0e+01 2.2e+02 3.5e+01  1  2.1e+00/4.6e+00
13:09:52 [1] 9.0e-08 3.7e-07 9.7e-08 3.3e-01 1.5e+00 4.4e-01  1  1.6e+00/4.6e+00
13:09:58 [2] 1.3e-12 1.4e-11 2.8e-12 2.7e-05 3.8e-04 3.8e-05  1  1.6e+00/4.6e+00
scaled solution norms   1.565115e+04  2.902953e+02  1.328985e+03


First Order Continuation resolve time:  1.3e+00
Step accepted, parameter = -1.060000e+04

        ----------------------------------
        First Order Continuation:
        Step number:   11 of   16 (max)
        Attempting solution at:
        BCID= 24 DFID=    1 Parameter= -1.050000e+04 delta_s= 1.000000e+02

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:10:06 [0] 1.0e-05 1.9e-04 1.8e-05 3.3e+01 2.3e+02 4.0e+01  1  1.6e+00/4.6e+00
13:10:13 [1] 3.4e-07 7.4e-07 3.6e-07 3.3e-01 3.4e+00 4.5e-01  1  1.6e+00/4.6e+00
13:10:19 [2] 8.9e-12 8.9e-11 1.9e-11 6.0e-05 3.1e-03 1.7e-04  1  1.6e+00/4.6e+00
scaled solution norms   1.555480e+04  2.889310e+02  1.320744e+03


First Order Continuation resolve time:  1.3e+00
        Step accepted, parameter = -1.050000e+04

        ----------------------------------
        First Order Continuation:
        Step number:   12 of   16 (max)
        Attempting solution at:
        BCID= 24 DFID=    1 Parameter= -1.040000e+04 delta_s= 1.000000e+02

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:10:27 [0] 1.8e-05 2.1e-04 2.5e-05 3.4e+01 2.3e+02 3.9e+01  1  1.6e+00/4.6e+00
13:10:34 [1] 1.2e-06 1.7e-06 1.2e-06 1.1e+00 1.1e+01 1.6e+00  1  1.6e+00/4.6e+00
13:10:40 [2] 9.1e-11 1.0e-09 1.9e-10 7.1e-04 3.1e-02 1.9e-03  1  1.6e+00/4.6e+00
13:10:46 [3] 9.6e-15 2.1e-13 1.1e-14 2.4e-10 2.5e-08 8.9e-10  1  1.6e+00/4.6e+00
scaled solution norms   1.545860e+04  2.875696e+02  1.312536e+03


First Order Continuation resolve time:  1.3e+00
        Step accepted, parameter = -1.040000e+04

        ----------------------------------
```

```
        First Order Continuation:
        Step number:   13 of   16 (max)
        Attempting solution at:
        BCID= 24 DFID=    1 Parameter= -1.030000e+04 delta_s= 1.000000e+02

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:10:54 [0] 1.2e-05 2.0e-04 2.0e-05 1.8e+02 3.0e+03 2.2e+02  1  1.6e+00/4.6e+00
13:11:01 [1] 7.8e-05 3.2e-04 9.4e-05 2.2e+02 5.9e+03 3.0e+02  1  1.6e+00/4.6e+00
13:11:07 [2] 3.6e-04 1.4e-03 4.7e-04 6.4e+01 3.5e+03 1.5e+02  1  1.6e+00/4.6e+00
13:11:14 [3] 3.7e-05 3.0e-04 6.1e-05 8.9e+01 1.0e+03 1.1e+02  1  1.6e+00/4.6e+00
13:11:20 [4] 3.0e-05 1.8e-04 4.5e-05 2.5e+01 3.3e+02 3.1e+01  1  1.6e+00/4.6e+00
13:11:26 [5] 4.2e-06 1.7e-05 5.1e-06 3.6e+01 5.9e+02 4.3e+01  1  1.6e+00/4.6e+00
13:11:33 [6] 3.8e-06 1.5e-05 5.1e-06 1.6e+02 2.7e+03 2.0e+02  1  1.6e+00/4.6e+00
13:11:39 [7] 8.6e-05 2.9e-04 9.9e-05 3.0e+02 4.2e+03 3.5e+02  1  1.6e+00/4.6e+00
13:11:45 [8] 6.1e-05 4.6e-04 9.8e-05 1.4e+02 3.8e+03 2.0e+02  1  1.6e+00/4.7e+00
13:11:52 [9] 2.2e-04 5.8e-04 2.3e-04 3.3e+02 4.0e+03 4.0e+02  1  1.6e+00/4.6e+00
scaled solution norms   1.536244e+04  2.861387e+02  1.304128e+03


First Order Continuation resolve time:  1.3e+00

        Failed to converge:

        Decreasing step-length to 5.000000e+01.


              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:11:59 [0] 2.9e-06 5.0e-05 5.0e-06 9.5e+00 1.3e+02 1.2e+01  1  1.6e+00/4.6e+00
13:12:06 [1] 1.2e-07 5.4e-07 1.8e-07 2.7e-01 5.2e+00 3.7e-01  1  1.6e+00/4.6e+00
13:12:12 [2] 6.0e-11 6.5e-10 1.3e-10 1.1e-03 2.2e-02 1.4e-03  1  1.6e+00/4.6e+00
13:12:18 [3] 3.4e-15 2.1e-13 6.9e-15 2.1e-08 3.9e-07 2.6e-08  1  1.6e+00/4.6e+00
scaled solution norms   1.541057e+04  2.868915e+02  1.308449e+03


First Order Continuation resolve time:  1.3e+00
        Step accepted, parameter = -1.035000e+04

        ----------------------------------
        First Order Continuation:
        Step number:   14 of   16 (max)
        Attempting solution at:
        BCID= 24 DFID=    1 Parameter= -1.030000e+04 delta_s= 5.000000e+01

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:12:26 [0] 2.4e-06 5.4e-05 5.3e-06 6.1e+01 1.1e+03 7.5e+01  1  1.6e+00/4.6e+00
13:12:33 [1] 1.4e-05 4.6e-05 1.6e-05 7.4e+01 1.2e+03 8.9e+01  1  1.6e+00/4.6e+00
```

```
13:12:39 [2] 1.5e-05 5.7e-05 1.8e-05 3.8e+01 5.9e+02 4.5e+01  1  1.6e+00/4.6e+00
13:12:46 [3] 3.6e-06 1.5e-05 4.7e-06 1.3e+03 2.2e+04 1.6e+03  1  1.6e+00/4.6e+00
13:12:52 [4]
P_0: Volume change  -0.398988
P_0: Deformation Gradient -4.469441 -3.247984 in elem [218]
P_0: Deformation Gradient -3.382808 -1.897544 in elem [218]

        Failed to converge:

        Decreasing step-length to 2.500000e+01.


              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:12:52 [0] 5.8e-07 1.3e-05 1.3e-06 2.4e+01 4.1e+02 2.9e+01  1  1.6e+00/4.6e+00
13:12:58 [1] 1.7e-06 6.5e-06 2.0e-06 2.2e+01 3.8e+02 2.7e+01  1  1.6e+00/4.6e+00
13:13:05 [2] 1.8e-06 6.5e-06 2.1e-06 3.6e+01 6.3e+02 4.4e+01  1  1.6e+00/4.6e+00
13:13:11 [3] 4.4e-06 1.5e-05 5.1e-06 1.9e+01 3.3e+02 2.4e+01  1  1.6e+00/4.6e+00
13:13:18 [4] 1.5e-06 4.6e-06 1.6e-06 2.3e+01 3.9e+02 2.7e+01  1  1.6e+00/4.6e+00
13:13:24 [5] 1.4e-06 5.5e-06 1.6e-06 3.3e+01 5.7e+02 4.0e+01  1  1.6e+00/4.6e+00
13:13:30 [6] 4.0e-06 1.3e-05 4.7e-06 2.0e+01 3.6e+02 2.5e+01  1  1.6e+00/4.6e+00
13:13:37 [7] 1.8e-06 5.0e-06 2.0e-06 2.2e+02 3.7e+03 2.6e+02  1  1.6e+00/4.6e+00
13:13:43 [8] 1.3e-04 5.1e-04 1.5e-04 9.4e+01 3.2e+03 1.6e+02  1  1.7e+00/4.6e+00
13:13:49 [9] 8.0e-05 4.1e-04 9.9e-05 5.8e+02 1.7e+04 9.8e+02  1  1.7e+00/4.8e+00
scaled solution norms   1.538602e+04  2.861380e+02  1.305031e+03

P_0: Volume change  -1.154287
P_0: Deformation Gradient -2.414944 0.223015 in elem [218]
P_0: Deformation Gradient -1.183234 0.468009 in elem [218]

        Failed to converge:

        Decreasing step-length to 1.250000e+01.

 X: C step-length reduced below minimum.

    Program terminated.
```

*Goma* converges on a solution for the first 12 steps of constant size 100.0, then has a deformation gradient failure on step 13 while trying to reach a vacuum of -10300. The step size is then halved and Goma converges at -10350, then another attempt at reaching -10300 is made. *Goma* fails to converge because the mesh contains numerous badly shaped elements in the upstream meniscus region; *Goma* stops with a Deformation Gradient error.

The solutions are stored to the output exodusII data base file. Shown below are plots of the stream function for vacuum pressures of -11400, -11100, -10700 and -10350 in order from left to right and top to bottom. As the upstream meniscus draws in closer to the feedslot, the mesh is compressed and distorts badly. The reduction in path length is directly related to the mesh quality; cessation of the analysis occurs when the mesh can no longer satisfy conditions in the domain.

This problem could have been remeshed when the vacuum pressure was within the -10600 to -10400 range and thus been carried further.

### 3.8.4 Zero order multiple parameter continuation in slot coating model

For a basic description of this problem, a single layer slot coater, refer to *Goma* tutorial GT-002. In this problem, side set 2 is the location of the upstream meniscus, the boundary on which the vacuum pressure is applied. The substrate (web) is denoted by node set 4. The liquid viscosity has material property tag 1300. To continue in the vacuum, web speed, and viscosity, both the `Continuation` and `Hunting Specifications` sections must be contained in the *Goma* input file `sc2.input`. The Continuation Specifications section is needed to define path parameters and print frequency; most importantly, the continuation method must be set to one of the hunting options:

```
------------------------------------------------------------
                  Continuation Specifications
------------------------------------------------------------
  Continuation                    = hzero
```

The `Hunting Specifications` section must be added to the input file to define the parameters of the continuation:

```
------------------------------------------------------------
                  Hunting Specifications
------------------------------------------------------------
  Number of hunting conditions  = -1
  HC = BC 24 1 1 -11444.0  -11700.0    16.0  3.2       32.0
  HC = BC 13 1 1     50.0       54.0     0.25 0.1        1.0
  HC = BC 14 0 1     50.0       54.0     0.25 0.1        1.0
  HC = MT  1 1300 1 0.25         0.41    0.16 0.01     100.0
  END OF HC
```

The first hunting condition specifies that the first parameter to be continued in belongs to the boundary condition with *Goma* index 24 and is the data float with index 1 (the second data float). The third integer specifies that *Goma* ramps from the start value, given as –11444.0, to the final value, given as -11700.0, with equal step sizes. The step size, given as 16.0, is ignored because STEP_CONTROL is on. The minimum and maximum step sizes are specified as 3.2 and 32.0 respectively. The second hunting condition indicates that the second continuation parameter belongs to the boundary condition with *Goma* index 13 and is the data float with index 1 (the second data float), which in this case is the web speed. This parameter is also used in the following boundary condition (index 14, float #0), so the third HC card is identical to the second save for these two entries. In both of these cases, *Goma* will ramp from 50.0 to 54.0 with equal step sizes. Note that if the Newton iterations fail to converge, the step size of the parameters will be halved despite STEP_CONTROL being on. The last hunting condition indicates that the fourth continuation parameter is a material property with assigned index 1300, that property being the viscosity. *Goma* will ramp this property from 0.25to 0.41.

*Goma* is run as usual, that is:

```
goma –a –i sc2.input
```

The *Goma* output is:

```
--------------------------------
        Zero Order Hunting:
        Step number:    1 of  100 (max)
        Attempting solution at: theta = 0
BCID= 24 DFID=    1 Parameter= -1.144400e+04 delta_s= 1.600000e+01
        BCID= 13 DFID=    1 Parameter=  5.000000e+01 delta_s= 2.500000e-01
        BCID= 14 DFID=    0 Parameter=  5.000000e+01 delta_s= 2.500000e-01
```

```
        MTID=  0 MPID= 1300 Parameter=  2.500000e-01 delta_s= 1.000000e-02

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:25:51 [0] 3.7e-04 2.3e-02 1.3e-03 2.1e+03 2.5e+04 3.6e+03  1  2.3e+00/4.7e+00
13:25:59 [1] 3.2e-04 5.7e-03 9.4e-04 4.0e+02 9.9e+03 6.6e+02  1  1.6e+00/4.6e+00
13:26:05 [2] 1.0e-03 1.2e-03 1.0e-03 3.0e+01 1.6e+03 7.2e+01  1  1.6e+00/4.6e+00
13:26:11 [3] 7.5e-08 1.7e-06 2.1e-07 2.8e-01 9.3e+00 7.0e-01  1  1.6e+00/4.6e+00
13:26:18 [4] 8.8e-12 1.4e-10 1.9e-11 2.4e-05 9.8e-04 6.6e-05  1  1.7e+00/4.6e+00
scaled solution norms   1.646916e+04  3.018376e+02  1.399544e+03

        Step accepted, theta (proportion complete) = -0.000000e+00
        Step accepted, BCID= 24 DFID=    1 Parameter= -1.144400e+04
        Step accepted, BCID= 13 DFID=    1 Parameter=  5.000000e+01
        Step accepted, BCID= 14 DFID=    0 Parameter=  5.000000e+01
        Step accepted, MTID=  0 MPID= 1300 Parameter=  2.500000e-01

        ----------------------------------
        Zero Order Hunting:
        Step number:    2 of   17 (max)
        Attempting solution at: theta = 0.0625
        BCID= 24 DFID=    1 Parameter= -1.146000e+04 delta_s= 1.600000e+01
        BCID= 13 DFID=    1 Parameter=  5.025000e+01 delta_s= 2.500000e-01
        BCID= 14 DFID=    0 Parameter=  5.025000e+01 delta_s= 2.500000e-01
        MTID=  0 MPID= 1300 Parameter=  2.600000e-01 delta_s= 1.000000e-02

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:26:25 [0] 3.1e-04 1.1e-02 1.0e-03 1.3e+03 2.0e+05 6.0e+03  1  1.6e+00/4.6e+00
13:26:31 [1] 2.8e-04 4.6e-03 4.4e-04 5.3e+02 5.6e+03 7.0e+02  1  1.6e+00/4.6e+00
13:26:37 [2] 9.0e-05 2.9e-04 1.1e-04 1.5e+02 1.0e+03 1.9e+02  1  1.6e+00/4.6e+00
13:26:44 [3] 1.0e-05 1.9e-05 1.0e-05 5.9e+00 4.5e+01 6.9e+00  1  1.6e+00/4.6e+00
13:26:50 [4] 4.4e-07 5.5e-07 4.4e-07 9.5e-01 5.5e+00 1.3e+00  1  1.6e+00/4.6e+00
13:26:56 [5] 1.6e-11 1.3e-10 3.1e-11 2.4e-04 4.0e-03 3.6e-04  1  1.6e+00/4.6e+00
scaled solution norms   1.674300e+04  3.095957e+02  1.425321e+03

        Step accepted, theta (proportion complete) = 6.250000e-02
        Step accepted, BCID= 24 DFID=    1 Parameter= -1.146000e+04
        Step accepted, BCID= 13 DFID=    1 Parameter=  5.025000e+01
        Step accepted, BCID= 14 DFID=    0 Parameter=  5.025000e+01
        Step accepted, MTID=  0 MPID= 1300 Parameter=  2.600000e-01

        ----------------------------------
        Zero Order Hunting:
        Step number:    3 of   17 (max)
        Attempting solution at: theta = 0.125
        BCID= 24 DFID=    1 Parameter= -1.147600e+04 delta_s= 1.600000e+01
        BCID= 13 DFID=    1 Parameter=  5.050000e+01 delta_s= 2.500000e-01
        BCID= 14 DFID=    0 Parameter=  5.050000e+01 delta_s= 2.500000e-01
        MTID=  0 MPID= 1300 Parameter=  2.700000e-01 delta_s= 1.000000e-02
```

```
                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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:27:03 [0] 3.1e-04 1.1e-02 1.0e-03 8.4e+02 2.0e+05 6.1e+03  1  1.7e+00/4.6e+00
13:27:10 [1] 1.9e-04 4.3e-03 3.8e-04 2.9e+02 5.3e+03 5.5e+02  1  1.6e+00/4.6e+00
13:27:16 [2] 3.7e-05 1.6e-04 4.3e-05 1.2e+02 7.5e+02 1.7e+02  1  1.6e+00/4.6e+00
13:27:22 [3] 5.4e-06 8.8e-06 5.5e-06 4.9e+00 5.5e+01 6.4e+00  1  1.6e+00/4.6e+00
13:27:29 [4] 5.6e-06 5.7e-06 5.6e-06 5.8e+00 6.4e+01 8.2e+00  1  1.6e+00/4.6e+00
13:27:35 [5] 1.4e-06 1.4e-06 1.4e-06 1.4e+00 1.5e+01 2.0e+00  1  1.6e+00/4.6e+00
13:27:41 [6] 8.6e-09 9.9e-09 8.7e-09 8.1e-03 5.6e-02 1.1e-02  1  1.7e+00/4.6e+00
13:27:48 [7] 1.6e-14 2.7e-13 2.0e-14 4.8e-08 1.8e-06 1.1e-07  1  1.6e+00/4.6e+00
scaled solution norms   1.702251e+04  3.174808e+02  1.452151e+03

        Step accepted, theta (proportion complete) = 1.250000e-01
        Step accepted, BCID= 24 DFID=    1 Parameter= -1.147600e+04
        Step accepted, BCID= 13 DFID=    1 Parameter=  5.050000e+01
        Step accepted, BCID= 14 DFID=    0 Parameter=  5.050000e+01
        Step accepted, MTID=  0 MPID= 1300 Parameter=  2.700000e-01

        ----------------------------------
        Zero Order Hunting:
        Step number:    4 of   17 (max)
        Attempting solution at: theta = 0.1875
        BCID= 24 DFID=    1 Parameter= -1.149200e+04 delta_s= 1.600000e+01
        BCID= 13 DFID=    1 Parameter=  5.075000e+01 delta_s= 2.500000e-01
        BCID= 14 DFID=    0 Parameter=  5.075000e+01 delta_s= 2.500000e-01
        MTID=  0 MPID= 1300 Parameter=  2.800000e-01 delta_s= 1.000000e-02

                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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:27:55 [0] 3.1e-04 1.1e-02 1.0e-03 8.8e+02 2.0e+05 6.1e+03  1  1.6e+00/4.6e+00
13:28:01 [1] 2.1e-04 3.9e-03 3.7e-04 4.5e+02 6.7e+03 6.3e+02  1  1.6e+00/4.6e+00
13:28:07 [2] 3.1e-05 2.5e-04 4.5e-05 3.5e+01 7.5e+02 6.2e+01  1  1.6e+00/4.6e+00
13:28:14 [3] 4.3e-05 5.8e-05 4.4e-05 6.6e+01 9.5e+02 9.5e+01  1  1.6e+00/4.6e+00
13:28:20 [4] 7.2e-05 7.9e-05 7.3e-05 9.9e+01 1.4e+03 1.5e+02  1  1.7e+00/4.6e+00
13:28:26 [5] 2.4e-05 3.0e-05 2.4e-05 2.3e+01 1.9e+02 3.7e+01  1  1.6e+00/4.6e+00
13:28:33 [6] 9.8e-07 1.6e-06 9.9e-07 9.7e-01 1.4e+01 1.2e+00  1  1.6e+00/4.6e+00
13:28:39 [7] 1.2e-10 1.3e-09 2.7e-10 1.7e-03 4.0e-02 2.8e-03  1  1.6e+00/4.6e+00
13:28:45 [8] 3.2e-15 2.1e-13 6.4e-15 5.3e-10 3.3e-08 1.2e-09  1  1.6e+00/4.6e+00
scaled solution norms   1.730803e+04  3.254630e+02  1.479894e+03

        Step accepted, theta (proportion complete) = 1.875000e-01
        Step accepted, BCID= 24 DFID=    1 Parameter= -1.149200e+04
        Step accepted, BCID= 13 DFID=    1 Parameter=  5.075000e+01
        Step accepted, BCID= 14 DFID=    0 Parameter=  5.075000e+01
        Step accepted, MTID=  0 MPID= 1300 Parameter=  2.800000e-01

        ----------------------------------
        Zero Order Hunting:
        Step number:    5 of   17 (max)
        Attempting solution at: theta = 0.25
        BCID= 24 DFID=    1 Parameter= -1.150800e+04 delta_s= 1.600000e+01
```

```
        BCID= 13 DFID=    1 Parameter=  5.100000e+01 delta_s= 2.500000e-01
        BCID= 14 DFID=    0 Parameter=  5.100000e+01 delta_s= 2.500000e-01
        MTID=  0 MPID= 1300 Parameter=  2.900000e-01 delta_s= 1.000000e-02

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:28:52 [0] 3.1e-04 1.1e-02 1.1e-03 8.7e+02 2.1e+05 6.3e+03  1  1.6e+00/4.6e+00
13:28:59 [1] 2.3e-04 4.0e-03 4.0e-04 7.1e+02 6.8e+03 9.4e+02  1  1.7e+00/4.7e+00
13:29:05 [2] 1.9e-05 2.2e-04 3.4e-05 5.2e+01 6.0e+02 7.2e+01  1  1.7e+00/4.6e+00
13:29:11 [3] 2.2e-05 4.0e-05 2.5e-05 9.9e+01 7.9e+02 1.4e+02  1  1.6e+00/4.6e+00
13:29:18 [4] 3.7e-05 5.1e-05 3.9e-05 1.7e+02 1.2e+03 2.3e+02  1  1.6e+00/4.6e+00
13:29:24 [5] 1.3e-05 1.7e-05 1.3e-05 5.7e+01 3.2e+02 7.8e+01  1  1.6e+00/5.2e+00
13:29:31 [6] 8.4e-07 1.1e-06 8.5e-07 3.6e+00 1.6e+01 4.9e+00  1  1.7e+00/4.6e+00
13:29:37 [7] 7.4e-11 7.6e-10 1.7e-10 2.1e-03 4.1e-02 3.4e-03  1  1.6e+00/4.6e+00
13:29:44 [8] 3.6e-15 2.2e-13 7.4e-15 2.7e-10 3.5e-08 1.2e-09  1  1.6e+00/4.6e+00
scaled solution norms   1.759941e+04  3.335869e+02  1.508707e+03

        Step accepted, theta (proportion complete) = 2.500000e-01
        Step accepted, BCID= 24 DFID=    1 Parameter= -1.150800e+04
        Step accepted, BCID= 13 DFID=    1 Parameter=  5.100000e+01
        Step accepted, BCID= 14 DFID=    0 Parameter=  5.100000e+01
        Step accepted, MTID=  0 MPID= 1300 Parameter=  2.900000e-01

        ----------------------------------
        Zero Order Hunting:
        Step number:    6 of   17 (max)
        Attempting solution at: theta = 0.3125
        BCID= 24 DFID=    1 Parameter= -1.152400e+04 delta_s= 1.600000e+01
        BCID= 13 DFID=    1 Parameter=  5.125000e+01 delta_s= 2.500000e-01
        BCID= 14 DFID=    0 Parameter=  5.125000e+01 delta_s= 2.500000e-01
        MTID=  0 MPID= 1300 Parameter=  3.000000e-01 delta_s= 1.000000e-02

              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:29:51 [0] 3.1e-04 1.0e-02 1.1e-03 8.8e+02 2.1e+05 6.4e+03  1  1.9e+00/4.6e+00
13:29:57 [1] 2.1e-04 4.4e-03 4.3e-04 9.2e+02 1.0e+04 1.6e+03  1  1.6e+00/5.2e+00
13:30:04 [2] 7.1e-05 3.9e-04 8.1e-05 5.1e+02 1.3e+04 8.1e+02  1  1.6e+00/4.8e+00
13:30:11 [3]
P_0: Volume change  -1.263624
P_0: Deformation Gradient -0.816547 -1.326580 in elem [218]
P_0: Deformation Gradient -1.208836 -0.994729 in elem [218]

        Failed to converge:
Decreasing step-length to 8.000000e+00.
Decreasing step-length to 1.250000e-01.
Decreasing step-length to 1.250000e-01.
Decreasing step-length to 5.000000e-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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:30:11 [0] 2.7e-04 6.3e-03 8.0e-04 4.3e+02 1.1e+05 3.2e+03  1  1.6e+00/4.6e+00
13:30:18 [1] 6.5e-05 1.2e-03 1.2e-04 2.6e+02 3.5e+03 3.7e+02  1  1.6e+00/4.6e+00
13:30:24 [2] 1.3e-04 4.1e-04 1.4e-04 6.0e+01 9.3e+02 7.5e+01  1  1.6e+00/4.6e+00
13:30:30 [3] 2.5e-05 9.4e-05 2.9e-05 2.8e+01 3.8e+02 4.4e+01  1  1.6e+00/4.6e+00
13:30:37 [4] 1.9e-05 2.9e-05 1.9e-05 7.0e+00 2.9e+02 1.7e+01  1  1.6e+00/4.6e+00
13:30:43 [5] 2.7e-05 2.8e-05 2.7e-05 7.2e+00 3.3e+02 1.9e+01  1  1.6e+00/4.6e+00
13:30:49 [6] 1.3e-05 1.4e-05 1.3e-05 3.9e+00 1.7e+02 1.0e+01  1  1.6e+00/4.6e+00
13:30:56 [7] 1.7e-06 2.1e-06 1.7e-06 5.5e-01 2.3e+01 1.4e+00  1  1.6e+00/4.6e+00
13:31:02 [8] 6.5e-10 8.4e-09 1.6e-09 2.0e-03 6.4e-02 3.3e-03  1  1.6e+00/4.6e+00
13:31:08 [9] 1.0e-14 3.2e-13 2.2e-14 6.6e-08 8.7e-07 7.8e-08  1  1.7e+00/4.6e+00
scaled solution norms   1.774661e+04  3.376645e+02  1.523374e+03

        Step accepted, theta (proportion complete) = 2.812500e-01
        Step accepted, BCID= 24 DFID=    1 Parameter= -1.151600e+04
        Step accepted, BCID= 13 DFID=    1 Parameter=  5.112500e+01
        Step accepted, BCID= 14 DFID=    0 Parameter=  5.112500e+01
        Step accepted, MTID=  0 MPID= 1300 Parameter=  2.950000e-01

        ----------------------------------
        Zero Order Hunting:
        Step number:    7 of   17 (max)
        Attempting solution at: theta = 0.3125
        BCID= 24 DFID=    1 Parameter= -1.152400e+04 delta_s= 8.000000e+00
        BCID= 13 DFID=    1 Parameter=  5.125000e+01 delta_s= 1.250000e-01
        BCID= 14 DFID=    0 Parameter=  5.125000e+01 delta_s= 1.250000e-01
        MTID=  0 MPID= 1300 Parameter=  3.000000e-01 delta_s= 5.000000e-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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:31:16 [0] 2.7e-04 6.2e-03 8.1e-04 5.3e+02 1.1e+05 3.2e+03  1  2.4e+00/4.6e+00
13:31:23 [1] 6.4e-05 1.1e-03 1.2e-04 2.4e+02 1.7e+03 2.9e+02  1  1.6e+00/4.6e+00
13:31:29 [2] 2.5e-06 2.6e-05 4.2e-06 1.3e+01 1.1e+02 1.9e+01  1  1.6e+00/4.6e+00
13:31:35 [3] 5.4e-06 5.6e-06 5.4e-06 6.0e+00 5.1e+01 8.6e+00  1  1.6e+00/4.6e+00
13:31:42 [4] 5.8e-06 5.8e-06 5.8e-06 6.5e+00 5.3e+01 9.3e+00  1  1.7e+00/4.6e+00
13:31:48 [5] 1.5e-06 1.5e-06 1.5e-06 1.6e+00 1.2e+01 2.4e+00  1  1.6e+00/4.6e+00
13:31:54 [6] 6.1e-08 6.2e-08 6.1e-08 6.7e-02 4.8e-01 9.7e-02  1  1.6e+00/4.6e+00
13:32:01 [7] 2.5e-13 2.7e-12 5.5e-13 5.7e-06 1.4e-04 9.5e-06  1  1.6e+00/4.6e+00
scaled solution norms   1.789429e+04  3.417586e+02  1.538235e+03

        Step accepted, theta (proportion complete) = 3.125000e-01
        Step accepted, BCID= 24 DFID=    1 Parameter= -1.152400e+04
        Step accepted, BCID= 13 DFID=    1 Parameter=  5.125000e+01
        Step accepted, BCID= 14 DFID=    0 Parameter=  5.125000e+01
        Step accepted, MTID=  0 MPID= 1300 Parameter=  3.000000e-01

        ----------------------------------
        Zero Order Hunting:
        Step number:    8 of   17 (max)
        Attempting solution at: theta = 0.34375
        BCID= 24 DFID=    1 Parameter= -1.153200e+04 delta_s= 8.000000e+00
        BCID= 13 DFID=    1 Parameter=  5.137500e+01 delta_s= 1.250000e-01
```

```
        BCID= 14 DFID=    0 Parameter=  5.137500e+01 delta_s= 1.250000e-01
        MTID=  0 MPID= 1300 Parameter=  3.050000e-01 delta_s= 5.000000e-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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:32:08 [0] 2.7e-04 6.2e-03 8.2e-04 4.4e+02 1.1e+05 3.2e+03  1  1.6e+00/4.6e+00
13:32:14 [1] 5.7e-05 1.0e-03 1.1e-04 2.2e+02 1.6e+03 2.6e+02  1  1.6e+00/4.6e+00
13:32:20 [2] 1.6e-06 1.4e-05 2.7e-06 8.2e+00 4.3e+01 1.1e+01  1  1.6e+00/4.6e+00
13:32:27 [3] 1.2e-05 1.2e-05 1.2e-05 5.5e+01 3.2e+02 7.4e+01  1  1.6e+00/4.6e+00
13:32:33 [4] 1.0e-05 1.0e-05 1.0e-05 4.9e+01 2.7e+02 6.6e+01  1  1.6e+00/4.6e+00
13:32:39 [5] 2.8e-06 2.9e-06 2.8e-06 1.3e+01 6.9e+01 1.8e+01  1  1.6e+00/4.6e+00
13:32:46 [6] 5.3e-08 6.0e-08 5.3e-08 2.4e-01 1.0e+00 3.2e-01  1  1.6e+00/4.6e+00
13:32:52 [7] 2.6e-13 2.7e-12 5.6e-13 8.0e-06 1.7e-04 1.4e-05  1  1.6e+00/4.6e+00
scaled solution norms   1.804189e+04  3.458843e+02  1.553344e+03

        Step accepted, theta (proportion complete) = 3.437500e-01
        Step accepted, BCID= 24 DFID=    1 Parameter= -1.153200e+04
        Step accepted, BCID= 13 DFID=    1 Parameter=  5.137500e+01
        Step accepted, BCID= 14 DFID=    0 Parameter=  5.137500e+01
        Step accepted, MTID=  0 MPID= 1300 Parameter=  3.050000e-01

        ----------------------------------
        Zero Order Hunting:
        Step number:    9 of   17 (max)
        Attempting solution at: theta = 0.375
        BCID= 24 DFID=    1 Parameter= -1.154000e+04 delta_s= 8.000000e+00
        BCID= 13 DFID=    1 Parameter=  5.150000e+01 delta_s= 1.250000e-01
        BCID= 14 DFID=    0 Parameter=  5.150000e+01 delta_s= 1.250000e-01
        MTID=  0 MPID= 1300 Parameter=  3.100000e-01 delta_s= 5.000000e-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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:32:59 [0] 2.7e-04 6.2e-03 8.2e-04 4.7e+02 1.1e+05 3.3e+03  1  1.6e+00/4.6e+00
13:33:05 [1] 5.8e-05 9.8e-04 1.1e-04 2.6e+02 1.8e+03 3.2e+02  1  1.6e+00/4.6e+00
13:33:12 [2] 1.8e-06 1.2e-05 2.5e-06 7.3e+00 9.0e+01 1.0e+01  1  1.6e+00/4.6e+00
13:33:18 [3] 1.4e-05 1.4e-05 1.4e-05 1.8e+01 1.7e+02 2.5e+01  1  1.6e+00/4.6e+00
13:33:24 [4] 1.5e-05 1.5e-05 1.5e-05 2.0e+01 1.8e+02 2.8e+01  1  1.6e+00/4.6e+00
13:33:31 [5] 3.9e-06 4.1e-06 3.9e-06 4.8e+00 3.4e+01 6.8e+00  1  1.6e+00/4.6e+00
13:33:37 [6] 1.1e-09 1.1e-08 2.4e-09 2.7e-02 7.1e-01 4.8e-02  1  1.6e+00/4.6e+00
13:33:44 [7] 4.3e-15 3.6e-13 1.4e-14 6.9e-08 1.4e-06 1.2e-07  1  1.6e+00/4.6e+00
scaled solution norms   1.818862e+04  3.500440e+02  1.568713e+03

        Step accepted, theta (proportion complete) = 3.750000e-01
        Step accepted, BCID= 24 DFID=    1 Parameter= -1.154000e+04
        Step accepted, BCID= 13 DFID=    1 Parameter=  5.150000e+01
        Step accepted, BCID= 14 DFID=    0 Parameter=  5.150000e+01
        Step accepted, MTID=  0 MPID= 1300 Parameter=  3.100000e-01

        ----------------------------------
        Zero Order Hunting:
        Step number:   10 of   17 (max)
```

```
        Attempting solution at: theta = 0.40625
        BCID= 24 DFID=    1 Parameter= -1.154800e+04 delta_s= 8.000000e+00
        BCID= 13 DFID=    1 Parameter=  5.162500e+01 delta_s= 1.250000e-01
        BCID= 14 DFID=    0 Parameter=  5.162500e+01 delta_s= 1.250000e-01
        MTID=  0 MPID= 1300 Parameter=  3.150000e-01 delta_s= 5.000000e-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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:33:50 [0] 2.7e-04 6.2e-03 8.3e-04 4.6e+02 1.1e+05 3.3e+03  1  1.6e+00/4.6e+00
13:33:57 [1] 5.3e-05 9.7e-04 1.0e-04 2.0e+02 1.8e+03 2.7e+02  1  1.6e+00/4.6e+00
13:34:03 [2] 8.1e-06 1.6e-05 8.2e-06 4.0e+01 2.3e+02 5.8e+01  1  1.6e+00/4.6e+00
13:34:09 [3] 3.2e-07 3.8e-07 3.2e-07 1.6e+00 5.9e+00 2.1e+00  1  1.6e+00/4.6e+00
13:34:16 [4] 9.6e-12 9.8e-11 2.2e-11 3.5e-04 7.4e-03 5.9e-04  1  1.6e+00/4.6e+00
scaled solution norms   1.833358e+04  3.542310e+02  1.584320e+03

        Step accepted, theta (proportion complete) = 4.062500e-01
        Step accepted, BCID= 24 DFID=    1 Parameter= -1.154800e+04
        Step accepted, BCID= 13 DFID=    1 Parameter=  5.162500e+01
        Step accepted, BCID= 14 DFID=    0 Parameter=  5.162500e+01
        Step accepted, MTID=  0 MPID= 1300 Parameter=  3.150000e-01

        ----------------------------------
        Zero Order Hunting:
        Step number:   11 of   17 (max)
        Attempting solution at: theta = 0.4375
        BCID= 24 DFID=    1 Parameter= -1.155600e+04 delta_s= 8.000000e+00
        BCID= 13 DFID=    1 Parameter=  5.175000e+01 delta_s= 1.250000e-01
        BCID= 14 DFID=    0 Parameter=  5.175000e+01 delta_s= 1.250000e-01
        MTID=  0 MPID= 1300 Parameter=  3.200000e-01 delta_s= 5.000000e-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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:34:23 [0] 2.8e-04 6.2e-03 8.4e-04 4.8e+02 1.1e+05 3.3e+03  1  1.6e+00/4.6e+00
13:34:29 [1] 4.9e-05 9.4e-04 1.0e-04 2.6e+02 2.0e+03 3.3e+02  1  1.6e+00/4.6e+00
13:34:35 [2] 7.5e-06 1.6e-05 7.6e-06 1.1e+01 8.1e+01 1.5e+01  1  1.6e+00/4.6e+00
13:34:42 [3] 2.3e-07 5.0e-07 3.2e-07 1.2e+00 4.7e+00 1.7e+00  1  1.6e+00/4.6e+00
13:34:48 [4] 4.7e-12 7.4e-11 1.3e-11 1.7e-04 3.5e-03 2.6e-04  1  1.6e+00/4.6e+00
scaled solution norms   1.847567e+04  3.584484e+02  1.600171e+03

        Step accepted, theta (proportion complete) = 4.375000e-01
        Step accepted, BCID= 24 DFID=    1 Parameter= -1.155600e+04
        Step accepted, BCID= 13 DFID=    1 Parameter=  5.175000e+01
        Step accepted, BCID= 14 DFID=    0 Parameter=  5.175000e+01
        Step accepted, MTID=  0 MPID= 1300 Parameter=  3.200000e-01

        ----------------------------------
        Zero Order Hunting:
        Step number:   12 of   17 (max)
        Attempting solution at: theta = 0.46875
        BCID= 24 DFID=    1 Parameter= -1.156400e+04 delta_s= 8.000000e+00
        BCID= 13 DFID=    1 Parameter=  5.187500e+01 delta_s= 1.250000e-01
```

```
        BCID= 14 DFID=    0 Parameter=  5.187500e+01 delta_s= 1.250000e-01
        MTID=  0 MPID= 1300 Parameter=  3.250000e-01 delta_s= 5.000000e-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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:34:55 [0] 2.8e-04 6.2e-03 8.5e-04 5.1e+02 1.1e+05 3.3e+03  1  1.6e+00/4.8e+00
13:35:01 [1] 4.4e-05 9.3e-04 9.6e-05 1.3e+02 1.8e+03 2.2e+02  1  2.3e+00/4.6e+00
13:35:08 [2] 3.9e-06 1.1e-05 4.2e-06 2.2e+01 1.3e+02 3.3e+01  1  1.7e+00/4.6e+00
13:35:15 [3] 6.9e-07 7.1e-07 6.9e-07 7.9e-01 4.7e+00 1.1e+00  1  1.6e+00/4.6e+00
13:35:21 [4] 4.9e-07 4.9e-07 4.9e-07 5.7e-01 3.9e+00 8.2e-01  1  1.6e+00/4.6e+00
13:35:28 [5] 1.2e-08 1.2e-08 1.2e-08 1.3e-02 8.3e-02 1.9e-02  1  1.6e+00/4.6e+00
13:35:34 [6] 7.7e-15 3.0e-13 1.8e-14 2.3e-07 5.8e-06 3.9e-07  1  1.6e+00/4.6e+00
scaled solution norms   1.861387e+04  3.626915e+02  1.616249e+03

        Step accepted, theta (proportion complete) = 4.687500e-01
        Step accepted, BCID= 24 DFID=    1 Parameter= -1.156400e+04
        Step accepted, BCID= 13 DFID=    1 Parameter=  5.187500e+01
        Step accepted, BCID= 14 DFID=    0 Parameter=  5.187500e+01
        Step accepted, MTID=  0 MPID= 1300 Parameter=  3.250000e-01

        ---------------------------------
        Zero Order Hunting:
        Step number:   13 of   17 (max)
        Attempting solution at: theta = 0.5
        BCID= 24 DFID=    1 Parameter= -1.157200e+04 delta_s= 8.000000e+00
        BCID= 13 DFID=    1 Parameter=  5.200000e+01 delta_s= 1.250000e-01
        BCID= 14 DFID=    0 Parameter=  5.200000e+01 delta_s= 1.250000e-01
        MTID=  0 MPID= 1300 Parameter=  3.300000e-01 delta_s= 5.000000e-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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:35:41 [0] 2.8e-04 6.2e-03 8.6e-04 4.4e+02 1.1e+05 3.3e+03  1  2.0e+00/4.6e+00
13:35:48 [1] 4.2e-05 9.2e-04 9.6e-05 2.4e+02 1.9e+03 3.0e+02  1  1.6e+00/4.6e+00
13:35:54 [2] 4.3e-06 1.3e-05 4.5e-06 9.1e+00 7.0e+01 1.1e+01  1  1.6e+00/4.6e+00
13:36:00 [3] 2.3e-06 2.4e-06 2.3e-06 1.2e+01 5.8e+01 1.6e+01  1  1.6e+00/4.6e+00
13:36:07 [4] 6.7e-07 6.8e-07 6.7e-07 3.4e+00 1.6e+01 4.6e+00  1  1.6e+00/4.6e+00
13:36:13 [5] 1.9e-07 1.9e-07 1.9e-07 9.9e-01 4.8e+00 1.3e+00  1  1.6e+00/4.6e+00
13:36:19 [6] 2.8e-12 2.7e-11 6.0e-12 9.9e-05 2.4e-03 1.8e-04  1  1.7e+00/4.6e+00
scaled solution norms   1.874697e+04  3.669595e+02  1.632547e+03

        Step accepted, theta (proportion complete) = 5.000000e-01
        Step accepted, BCID= 24 DFID=    1 Parameter= -1.157200e+04
        Step accepted, BCID= 13 DFID=    1 Parameter=  5.200000e+01
        Step accepted, BCID= 14 DFID=    0 Parameter=  5.200000e+01
        Step accepted, MTID=  0 MPID= 1300 Parameter=  3.300000e-01

        ---------------------------------
        Zero Order Hunting:
        Step number:   14 of   17 (max)
        Attempting solution at: theta = 0.53125
        BCID= 24 DFID=    1 Parameter= -1.158000e+04 delta_s= 8.000000e+00
```

```
        BCID= 13 DFID=    1 Parameter=  5.212500e+01 delta_s= 1.250000e-01
        BCID= 14 DFID=    0 Parameter=  5.212500e+01 delta_s= 1.250000e-01
        MTID=  0 MPID= 1300 Parameter=  3.350000e-01 delta_s= 5.000000e-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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:36:26 [0] 2.8e-04 6.2e-03 8.6e-04 4.8e+02 1.1e+05 3.3e+03  1  1.6e+00/4.6e+00
13:36:33 [1] 4.3e-05 9.2e-04 9.6e-05 1.9e+02 1.9e+03 2.4e+02  1  1.6e+00/4.6e+00
13:36:39 [2] 2.4e-06 1.2e-05 2.8e-06 1.4e+01 8.5e+01 1.9e+01  1  1.6e+00/4.6e+00
13:36:45 [3] 2.9e-06 2.9e-06 2.9e-06 3.7e+00 2.6e+01 5.2e+00  1  1.6e+00/4.6e+00
13:36:52 [4] 1.5e-06 1.5e-06 1.5e-06 1.9e+00 1.3e+01 2.7e+00  1  1.6e+00/4.6e+00
13:36:58 [5] 3.4e-08 3.5e-08 3.4e-08 4.0e-02 2.1e-01 5.9e-02  1  1.7e+00/4.6e+00
13:37:05 [6] 7.2e-14 9.8e-13 1.6e-13 2.4e-06 5.9e-05 4.0e-06  1  1.6e+00/4.6e+00
scaled solution norms   1.887396e+04  3.712456e+02  1.649036e+03

        Step accepted, theta (proportion complete) = 5.312500e-01
        Step accepted, BCID= 24 DFID=    1 Parameter= -1.158000e+04
        Step accepted, BCID= 13 DFID=    1 Parameter=  5.212500e+01
        Step accepted, BCID= 14 DFID=    0 Parameter=  5.212500e+01
        Step accepted, MTID=  0 MPID= 1300 Parameter=  3.350000e-01

        ----------------------------------
        Zero Order Hunting:
        Step number:   15 of   17 (max)
        Attempting solution at: theta = 0.5625
        BCID= 24 DFID=    1 Parameter= -1.158800e+04 delta_s= 8.000000e+00
        BCID= 13 DFID=    1 Parameter=  5.225000e+01 delta_s= 1.250000e-01
        BCID= 14 DFID=    0 Parameter=  5.225000e+01 delta_s= 1.250000e-01
        MTID=  0 MPID= 1300 Parameter=  3.400000e-01 delta_s= 5.000000e-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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:37:11 [0] 2.8e-04 6.2e-03 8.7e-04 4.3e+02 1.1e+05 3.3e+03  1  1.6e+00/4.6e+00
13:37:18 [1] 4.9e-05 9.3e-04 9.8e-05 2.1e+02 2.3e+03 2.8e+02  1  1.6e+00/4.6e+00
13:37:24 [2] 3.5e-06 5.0e-05 7.5e-06 7.9e+00 1.1e+02 1.2e+01  1  1.6e+00/4.6e+00
13:37:31 [3] 3.6e-06 4.7e-06 3.6e-06 1.9e+01 1.3e+02 2.6e+01  1  1.6e+00/4.6e+00
13:37:37 [4] 2.3e-06 2.9e-06 2.3e-06 1.2e+01 6.4e+01 1.7e+01  1  1.6e+00/4.6e+00
13:37:43 [5] 4.3e-07 4.4e-07 4.3e-07 2.4e+00 1.1e+01 3.2e+00  1  1.6e+00/4.6e+00
13:37:50 [6] 1.4e-11 1.4e-10 3.1e-11 5.3e-04 1.4e-02 9.9e-04  1  1.6e+00/4.6e+00
scaled solution norms   1.899387e+04  3.755401e+02  1.665677e+03

        Step accepted, theta (proportion complete) = 5.625000e-01
        Step accepted, BCID= 24 DFID=    1 Parameter= -1.158800e+04
        Step accepted, BCID= 13 DFID=    1 Parameter=  5.225000e+01
        Step accepted, BCID= 14 DFID=    0 Parameter=  5.225000e+01
        Step accepted, MTID=  0 MPID= 1300 Parameter=  3.400000e-01

        ----------------------------------
        Zero Order Hunting:
        Step number:   16 of   17 (max)
        Attempting solution at: theta = 0.59375
```

```
        BCID= 24 DFID=    1 Parameter= -1.159600e+04 delta_s= 8.000000e+00
        BCID= 13 DFID=    1 Parameter=  5.237500e+01 delta_s= 1.250000e-01
        BCID= 14 DFID=    0 Parameter=  5.237500e+01 delta_s= 1.250000e-01
        MTID=  0 MPID= 1300 Parameter=  3.450000e-01 delta_s= 5.000000e-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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:37:57 [0] 2.8e-04 6.2e-03 8.8e-04 4.9e+02 1.1e+05 3.3e+03  1  1.6e+00/4.6e+00
13:38:03 [1] 4.4e-05 1.1e-03 9.8e-05 2.3e+02 2.0e+03 2.9e+02  1  1.6e+00/4.6e+00
13:38:09 [2] 2.5e-06 2.2e-05 3.5e-06 1.5e+01 1.2e+02 2.1e+01  1  1.6e+00/4.6e+00
13:38:16 [3] 3.6e-06 3.8e-06 3.6e-06 4.9e+00 3.5e+01 7.0e+00  1  1.6e+00/4.6e+00
13:38:22 [4] 2.3e-06 2.3e-06 2.3e-06 3.2e+00 2.2e+01 4.5e+00  1  1.6e+00/4.6e+00
13:38:28 [5] 4.3e-07 4.4e-07 4.3e-07 6.0e-01 4.0e+00 8.5e-01  1  1.6e+00/4.6e+00
13:38:35 [6] 1.2e-11 1.3e-10 2.6e-11 4.3e-04 1.1e-02 7.3e-04  1  1.6e+00/4.6e+00
scaled solution norms   1.910589e+04  3.798342e+02  1.682424e+03

        Step accepted, theta (proportion complete) = 5.937500e-01
        Step accepted, BCID= 24 DFID=    1 Parameter= -1.159600e+04
        Step accepted, BCID= 13 DFID=    1 Parameter=  5.237500e+01
        Step accepted, BCID= 14 DFID=    0 Parameter=  5.237500e+01
        Step accepted, MTID=  0 MPID= 1300 Parameter=  3.450000e-01

        ----------------------------------
        Zero Order Hunting:
        Step number:   17 of   17 (max)
        Attempting solution at: theta = 0.625
        BCID= 24 DFID=    1 Parameter= -1.160400e+04 delta_s= 8.000000e+00
        BCID= 13 DFID=    1 Parameter=  5.250000e+01 delta_s= 1.250000e-01
        BCID= 14 DFID=    0 Parameter=  5.250000e+01 delta_s= 1.250000e-01
        MTID=  0 MPID= 1300 Parameter=  3.500000e-01 delta_s= 5.000000e-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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
13:38:42 [0] 2.8e-04 6.3e-03 8.9e-04 4.0e+02 1.1e+05 3.3e+03  1  1.6e+00/4.6e+00
13:38:48 [1] 4.1e-05 9.9e-04 9.5e-05 1.5e+02 2.0e+03 2.4e+02  1  1.6e+00/4.6e+00
13:38:54 [2] 2.9e-06 1.5e-05 3.3e-06 9.4e+00 1.1e+02 1.2e+01  1  1.6e+00/4.6e+00
13:39:01 [3] 3.8e-06 4.0e-06 3.8e-06 2.1e+01 1.1e+02 2.9e+01  1  1.6e+00/4.6e+00
13:39:07 [4] 2.5e-06 2.5e-06 2.5e-06 1.4e+01 7.1e+01 2.0e+01  1  1.6e+00/4.6e+00
13:39:13 [5] 5.4e-07 5.5e-07 5.4e-07 3.1e+00 1.5e+01 4.2e+00  1  1.6e+00/4.6e+00
13:39:20 [6] 8.0e-08 8.1e-08 8.0e-08 4.6e-01 2.3e+00 6.3e-01  1  1.6e+00/4.6e+00
13:39:26 [7] 4.9e-13 5.2e-12 1.1e-12 1.9e-05 5.4e-04 3.7e-05  1  1.6e+00/4.6e+00
scaled solution norms   1.920959e+04  3.841233e+02  1.699247e+03

        Step accepted, theta (proportion complete) = 6.250000e-01
        Step accepted, BCID= 24 DFID=    1 Parameter= -1.160400e+04
        Step accepted, BCID= 13 DFID=    1 Parameter=  5.250000e+01
        Step accepted, BCID= 14 DFID=    0 Parameter=  5.250000e+01
        Step accepted, MTID=  0 MPID= 1300 Parameter=  3.500000e-01

        Failed to reach end of hunt in maximum number of successful steps (17).
        Sorry.
```

The Hunting Continuation proceeds as designed, with one deformation gradient failure at Step 6, until the specified maximum number of path steps (17) is reached. Attempting to proceed beyond this point would result in repeated deformation gradient failures, indicating the need for regular remeshing and remapping. The path (continuation) variables at this point are: vacuum pressure of -11604.0, web speed of -52.5 and viscosity of 0.35.

The solutions are stored to the output exodusII data base file `sc2-out.exoII`.

Shown below are plots of the stream function for sets of operating conditions for the slot coater; the order of these plots are from left to right and top to bottom.

vacuum pressure (shown in Fig 3.4): -11460, -11524, -11564, -11596

web speed: 50.25, 51.25, 51.875, 52.375

viscosity: 0.26, 0.30, 0.325, 0.345

As the upstream meniscus draws in closer to the feedslot, the mesh is compressed and distorts badly. The reduction in path length is directly related to the mesh quality; cessation of the analysis occurs when the mesh can no longer satisfy conditions in the domain. If this problem was remeshed when the conditions were in the neighborhood of web speed .16, vacuum pressure -3750 and viscosity 2.5, the continuation could be restarted and carried further to the target goals on these three parameters.

It is also possible to run this problem with LOCA using continuation conditions (CC cards) rather than with the hunting routine. An input file `sc2l.input` is provided for this purpose; it is left to the reader to run this file and observe the equivalent solution.

# 4 Linear Stability Analysis

Stability analysis addresses the following question: After disturbing a given steady base state with random disturbances, do the disturbances grow or decay in time? If the disturbances vanish and the base state persists, then the system is stable. Otherwise the disturbances grow and the state changes to a new state that can be physically undesirable.

Stability analysis tests the stability of a steady-state flow solution by adding disturbances to it. By invoking the method of small disturbances and representing a generic disturbance as a superposition of normal modes, the stability problem takes the form of an asymmetric and singular generalized eigenvalue problem. This eigenvalue problem characterizes the stability of the base state subjected to the imposed disturbance. The disturbance modes are the eigenvectors and the growth or decay rates are the eigenvalues. The sign of the real parts of the eigenvalues (growth rate if positive; decay rate if negative) tell whether the flow is either stable or not. If one or more of the real parts of the eigenvalues are positive, the base flow is unstable; otherwise the base flow is stable.

The stability problem arising from incompressible fluid mechanics contains a few troublesome features that must be taken into account. The first concerns the properties of the two, usually large, matrices involved in linear stability analysis, The Jacobian matrix (sensitivities of the residuals to the unknowns), is in general asymmetric, as are shifted combinations with the mass matrix (sensitivities of the residuals to time-dependent unknowns). This means that fast symmetric eigenvalue extraction methods are not suitable. The second is that the mass matrix is singular. This is because both Dirichlet boundary conditions and the incompressibility constraint contain no time dependence. This gives rise to ‘infinite’ growth (or decay) rates, since the speed of sound is infinite in incompressible media (Christodoulou, 1988). The ‘infinite magnitude’ eigenvalues must be filtered out so that the eigenvalue extractor finds the physically relevant, most dangerous eigenvalues. A simple manipulation that maps ‘infinite’ eigenvalues to zero is the shift-and-invert transformation. The ‘shift’ is usually set near to the anticipated values of the leading eigenvalues. Most often, the shifts are set to the values of the leading eigenvalues available at a ‘close’ parameter value. More details about the nature of the matrix stability problem are provided in Coyle (1984), Christodoulou (1988), and Gates (1998).

The method used in *Goma* to extract eigenvalues is Arnoldi's method (Saad, 1992). It makes use of projections of the shifted matrix onto a subspace spanned by iterates of the power method. The method belongs to the class of Krylov subspace methods. Saad (1992) provides a very thorough discourse on these methods. Arnoldi (1951) presented the procedure as a means for reducing dense matrices to Hessenberg form. The interested user can consult Saad (1992), Christodoulou (1988), and Gates (1998) for more details on the algorithm implemented in *Goma*.

There are now two eigensolvers available for *Goma*. Eggroll, which has been available for some time, is included in the source code, and ARPACK (Lehoucq and Sorensen, 1996, Lehoucq and Scott, 1997) can be linked in as an external library. Both eigensolvers use Arnoldi-based Krylov

subspace methods to extract the first few (n) eigenvalues of interest; however, the LOCA driver for ARPACK uses either a one-parameter shift-and-invert algorithm or a two-parameter Cayley transformation algorithm, whereas eggroll only uses shift-and-invert. Moreover, eggroll cannot be accessed during continuation, and requires UMFPACK to perform the necessary solves of the shifted linear eigensystem (which precludes running in parallel). ARPACK can use any linear solver supported by *Goma* (except FRONT), and with the provided sub-package PARPACK can be used for parallel eigensolves. The interface to ARPACK is contained in LOCA, which enables continuation to be combined with stability analysis at specified step intervals

The sections of this chapter present an explanation of the input specifications (4.1 and 2.2), an example to illustrate the use of LSA and to highlight the output and it's meaning (4.4), and a presentation of several considerations relevant to the use of LSA (4.5). Where necessary, notations will be made in the remainder of this chapter about the applicability of inputs, features, etc. to either one or both eigensolvers.

## 4.1 Required Specifications in the *Goma* Input File

This section describes how to carry out linear stability analysis with *Goma*. The input information that *Goma* requires is supplied in the usual *Goma* input file under the Eigensolver Specifications section. To activate the stability software, the Linear Stability keyword in the Solver Specifications section must be set equal to yes, that is:

```
------------------------------------------------------------
Solver Specifications
------------------------------------------------------------
Linear Stability                = yes
```

A sample of the information/parameters that must be supplied in the Eigensolver Specifications section is listed below. The eigensolver applicability of each input is indicated here by <A> for ARPACK only, <E> for eggroll only, or <AE> for both.

```
------------------------------------------------------------
Eigensolver Specifications
------------------------------------------------------------
Eigen Algorithm = cayley <A>
Eigen Number of modes           = 10 <AE>
Eigen Record modes              = 5 <AE>
Eigen Size of Krylov subspace   = 30 <AE>
Eigen Maximum Iterations        = 2 <E>
Eigen Number of Filter Steps    = 1 <E>
Eigen Recycle                   = no <E>
Eigen Tolerance                 = 1.0e-06 <E>
Eigen Matrix Output= no <AE>
Eigen Initial Vector Weight     = 0.1 <E>
Eigen Initial Shifts            = -50.0 -51.0 -52.0 -54.0 <E>
Eigen Cayley Sigma= 100.0 <A>
```

```
Eigen Cayley Mu= 1000.0 <A>
Eigen Relative tolerance= 1.0e-6 <A>
Eigen Linear Solver tolerance= 1.0e-6 <A>
Eigenvalue output frequency= 1 <A>
Eigenvector output frequency= 1 <A>
Eigenvector output file= eigen.exoII <A>
Eigen Wave Numbers= 0.0 0.1 0.2 1.0 <AE>
```

## 4.2 Eigensolver Specifications

The ability to solve for the stability of a base flow is a very powerful tool. Often, the important characteristics of a flow can be summarized in the answer to the question “is the flow stable?”. Although the following cards are in active use at the time of this writing, sweeping changes are coming to the eigensolver sections of *Goma*. In particular, the old code (called “eggroll”) is being replaced with newer methods (in the ARPACK library), as well as being coupled to the continuation and tracking algorithms (in the LOCA library).

Input specifications for this section of input records is discussed in a separate, comprehensive manual (Gates, et. al., 2000); an update to this manual will be completed during the fall of 2002 (Labreche, et. al., 2002). Either of these manuals contains a thorough discussion of how to successfully compute the stability and interesting modes of an underlying base flow.

### 4.2.1 Eigen Algorithm

| Eigen Algorithm = {cayley \| si} |
|---|

#### Description/Usage

This optional card is used by the ARPACK eigensolver to select between the Cayley transformation algorithm and the shift and invert algorithm. When cayley is selected, *Eigen Cayley Sigma* card and *Eigen Cayley Mu* are the two shift parameters. When the shift and invert (default) algorithm is used, *Eigen Cayley Sigma* is the single shift parameter. The valid options are:

| | |
|---|---|
| cayley | Cayley transformation algorithm. |
| si | Shift and invert algorithm. |

The default value is si.

#### Examples

Here is a sample card:

```
Eigen Cayley Mu = 10.0
```

### Technical Discussion

When the Cayley algorithm is selected, the relative values of *Eigen Cayley Sigma* and *Eigen Cayley Mu* also determine which of the two Cayley transformation methods (A or B) is used by ARPACK.

Another use of this card is to determine whether eggroll or ARPACK will be used for non-continuation problems: If this card is present with a valid option, ARPACK will be used; otherwise eggroll will be used. Continuation runs with LOCA will always use ARPACK, and non-LOCA continuation runs cannot be combined with stability analysis.

This card is not applicable to the eggroll eigensolver, which uses only the shift and invert algorithm.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 4.2.2 Eigen Number of modes

| Eigen Number of modes = <int> |
|---|

### Description/Usage

This optional card specifies how many modes (eigenvalue/eigenvector pairs) are to be computed during linear stability analysis. The input values are:

| | |
|---|---|
| **N** | Number of eigenvalue/eigenvector pairs (modes) to compute. |

The default value for **N** is 10.

### Examples

Here is a sample card:

```
Eigen Number of modes = 10
```

### Technical Discussion

When the *Linear Stability* card indicates stability analysis should be performed, this card is used when computing the spectrum. Generally, the user is indicating that they would like to compute the leading **N** modes. Judicious selection of other pameters is required for this to happen. See the *Eigen Initial Shifts* card, especially.

This card is applicable to both eggroll and ARPACK eigensolvers.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 4.2.3 Eigen Record modes

| Eigen Record modes = <int> |
|---|

### Description/Usage

This optional card determines how many of the converged modes should have their eigenvectors written to exodus II files. The valid input is:

| | |
|---|---|
| **0<=n<=N** | This requests that **n** exodus II outputs be written. **n** can be no larger than **N**, the value supplied in the *Eigen Number of modes* card. |

The default value of **n** is 0.

### Examples

Here is a sample card:

```
Eigen Record modes = 5
```

### Technical Discussion

This card is especially important when the 3D stability of a 2D flow is being computed. Each of the requested normal mode wave numbers receives **n** outputs, so the number of files written can become quite large if the user isn't careful.

This card is applicable to both eggroll and ARPACK eigensolvers. When eggroll is used, the name of the output file is LSA_<i>_of_**n**_<out_name>, where <i> is the ith mode (of **n**), and <out_name> is the name of the regular Exodus II output file (from the *Output Exodus II file* card), including any extension if there was one. In the case of 3D stability of a 2D flow, the output file is LSA_<i>_of_**n**_wn=<f>_<out_name>, where <f> corresponds to the value of the normal mode (see the *Eigen Wave Numbers* card). When ARPACK is used, these files will have the base name specified by the *Eigenvector Output File* card, similarly augmented with mode and wave numbers.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 4.2.4 Eigen Size of Krylov subspace

| Eigen Size of Krylov subspace = <int> |
|---|

### 4.2.4Description/Usage

This optional card specifies the dimension of the Krylov subspace used in the eigensolver. Valid inputs are:

| | |
|---|---|
| **m>0** | A positive integer specifying the maximum dimension of the Krylov subspace. |

The default value of **m** is 30.

### Examples

Here is a sample card:

```
Eigen Size of Krylov subspace = 120
```

### Technical Discussion

A restarted Lanczos-type algorithm is used internally to solve the generalized eigenvalue problem. Specifying a very large **m** can lead to wasted effort, while an **m** that is too small can lead to nonconvergence, or possibly very slow convergence. The optimal value of **m** is completely problem dependent.

This card is applicable to both eggroll and ARPACK eigensolvers. When ARPACK is used, however, this value is also used as the maximum number of inner eigensolver iterations (see the *Eigen Maximum Iterations* card).

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 4.2.5 Eigen Maximum Iterations

| Eigen Maximum Iterations = <int> |
|---|

### Description/Usage

This optional card specifies the maximum number of eggroll eigensolver iterations. Its valid argument is given by:

| | |
|---|---|
| **n>0** | Maximum number of eigensolver iterations. |

The default value of **n** is 100.

### Examples

Here is a sample card:

```
Eigen Maximum Iterations = 32
```

### Technical Discussion

Within the eigensolver is a restarted Lanczos-type method. The maximum number of restarts is **n**. Also, see the *Eigen Size of Krylov subspace* card.

This card is not applicable to the ARPACK eigensolver, for which the maximum number of iterations is set to the Krylov subspace size.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 4.2.6 Eigen Number of Filter Steps

| Eigen Number of Filter Steps = <int> |
|---|

### Description/Usage

This optional card specifies the number of internal filter steps used in the eggroll eigensolver. Its valid syntax is given by:

**n>=0** Number of filter steps to perform.

The default value of **n** is 2.

### Examples

Here is a sample card:

```
Eigen Number of Filter Steps = 4
```

### Technical Discussion

In practice this card seems to have little effect. See the Advanced Capabilities document.

This card is not applicable to the ARPACK eigensolver.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 4.2.7 Eigen Recycle

| Eigen Recycle = { yes \| no } |
|---|

### Description/Usage

This optional card determines whether or not the eigenpairs will be recycled within the eggroll eigensolver algorithm. Valid input is defined by:

| | |
|---|---|
| **yes** | Do recycle eigenpairs. |
| **no** | Do not recycle eigenpairs. |

The default value is **no**.

### Examples

Here is a sample card:

```
Eigen Recycle = yes
```

### Technical Discussion

In practice, this card seems to have little effect.

This card is not applicable to the ARPACK eigensolver.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 4.2.8 Eigen Tolerance

| Eigen Tolerance = <float> |
|---|

### Description/Usage

This optional card specifies the absolute residual tolerance used in the eggroll eigensolver. Valid input are defined as:

**tol>=0.0** The termination criteria used within the eigensolver.

The default value of **tol** is 1.0e-6.

### Examples

Here is a sample card:

```
Eigen Tolerance = 1.0e-16
```

### Technical Discussion

The value of **tol** specifies a termination condition used within the eigensolver. At the time of this writing it was difficult to guarantee that a small **tol** would lead to very accurate eigenpairs with the eggroll solver. This is one of the primary reasons for switching to the ARPACK eigensolver. The correspondence of “internal residual” to “error in eigenpair” is even more difficult with the generalized eigenvalue problem than “residual” to “error in solution vector” with regular linear solvers.

This card is not applicable to the ARPACK eigensolver, which uses a relative tolerance (see the *Eigen Relative tolerance* card).

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 4.2.9 Eigen Matrix Output

| Eigen Matrix Output = { yes \| no } |
|---|

### Description/Usage

This optional card is used to indicate that the Jacobian and mass matrices which are calculated for linear stability analysis should be written to output files.

### Examples

To write the Jacobian and mass matrices when doing LSA, use:

```
Eigen Matrix Output = yes
```

### Technical Discussion

These files can be quite large, so it is advisable to output them only if you need them. The intent of matrices availble is so that the experienced human eigensolver can import the matrices into other tools (i.e., Matlab) to perform their own investigations. Note that when the *Linear Stability* choice is “file” or “3Dfile”, this card must be set to yes in order to create and write the desired stability output files.

This card is applicable to both eggroll and ARPACK. The same matrix output function is called in either case.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 4.2.10 Eigen Initial Vector Weight

| Eigen Initial Vector Weight = <float> |
|---|

### Description/Usage

This optional card allows the user to modify the initial vector used within the eggroll eigensolver to contruct a Krylov subspace. Legal inputs are:

| | |
|---|---|
| **w** | A scale factor for the amount of uniform randomness to apply to each component of the initial vector. |

The default value of **w** is 0.5.

### Examples

Here is a sample card:

```
Eigen Initial Vector Weight = 0.1
```

### Technical Discussion

Because of the way a Krylov space is generated, it is very important that the initial vector contain some component in the direction of the eigenvector. By adding a random vector to the initial vector this can be guaranteed. A random vector r is constructed to have components from U[0,1] (uniform distribution). It is then normalized to have length 1. The initial vector is then modifed as x = **w** * r + (1-**w**) * x. The effect this value has should be minimal for **w** not equal to 0 or 1.

This card is not applicable to the ARPACK eigensolver.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 4.2.11 Eigen Initial Shifts

| Eigen Initial Shifts = <float1> <float2> <float3> <float4> |
|---|

### Description/Usage

This optional card allows the user to supply a set of initial shifts for the shift-and-invert algorithm used in the eggroll eigensolver. Valid syntax is:

| | |
|---|---|
| **f1**, **f2**, **f3**, **f4** | All real values that should be close to the eigenvalue the user is trying to compute. |

The default value(s) are all -1.0.

### Examples

Here is a sample card:

```
Eigen Initial Shifts = -0.001 -0.001 -0.001 -0.001
```

### Technical Discussion

The eigensolver algorithm beings by trying to find eigenvalues near **f1**. If it fails to do so, it will try for eigenvalues near **f2**, and so on. The only time when shifts other than **f1** are used is when **f1** is a truly bad approximation to an eigenvalue. This happens rarely (the locations of the eigenvalues need to be clustered in a particularly bad way).

This card is not applicable to the ARPACK eigensolver.

### Theory

No Theory.

### FAQs

No FAQs.

## References

No References.

## 4.2.12 Eigen Wave Numbers

| Eigen Wave Numbers = <float1> ... <floatN> |
|---|

### Description/Usage

If the type of stability requested in the *Linear Stability* card indicates that 3D stability of a 2D base flow is to be performed, then this card is required. Otherwise it is optional (and ignored). This card is used to specify the wave numbers in the normal (z) direction for 3D stability of a 2D flow. The valid inputs are:

| | |
|---|---|
| $\mathbf{w_1}$, $\mathbf{w_2}$, ..., $\mathbf{w_N}$ | Any number (subject to input line length restrictions) of real numbers that are used as normal wave numbers. |

There is no default value.

### Examples

Here is a sample card:

```
Eigen Wave Numbers = 0.0 0.01 0.1 1.0 2.0 3.0
```

### Technical Discussion

Each $\mathbf{w_i}$ is used in turn as the wave number in the 3rd direction (the normal direction) for which linear stability analysis will be performed. They are equivalent to a wavelength of $2\pi/\mathbf{w_i}$. A wavenumber of $\mathbf{w_i}$=0 corresponds to the regular 2D stability of a 2D base flow.

This card is applicable to both eggroll and ARPACK eigensolvers. However, 3D of 2D stability can only be done in parallel when ARPACK is used.

## Theory

No Theory.

## FAQs

No FAQs.

### References

No References.

## 4.2.13 Eigen Cayley Sigma

| Eigen Cayley Sigma = <float> |
|---|

### Description/Usage

This optional card is used by the ARPACK eigensolver as a shift parameter. When the Cayley transformation is selected (see the *Eigen Algorithm* card), this value and *Eigen Cayley Mu* are the two shift parameters. When the shift and invert (default) algorithm is used, this is the single shift parameter.

The default value is 100.

### Examples

Here is a sample card:

```
Eigen Cayley Sigma = 10.0
```

### Technical Discussion

When the Cayley algorithm is selected, this value and *Eigen Cayley Mu* also determine which of the two Cayley transformation methods (A or B) is used by ARPACK. For either of these methods, sigma must be to the right of (greater than) the real parts of all eigenvalues -- any eigenvalues greater than sigma will not be found and will generate an error message. If this happens, just increase sigma and try again.

This card is not applicable to the eggroll eigensolver, which takes shift values from the *Eigen Initial Shifts* card.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 4.2.14 Eigen Cayley Mu

| Eigen Cayley Mu = <float> |
|---|

### Description/Usage

This optional card is used by the ARPACK eigensolver as the second Cayley shift parameter. When the Cayley transformation is selected (see the *Eigen Algorithm* card), this value and *Eigen Cayley Sigma* are the two shift parameters. When the shift and invert (default) algorithm is used, this card is not applicable.

The default value is 1000.

### Examples

Here is a sample card:

```
Eigen Cayley Mu = 10.0
```

### Technical Discussion

When the Cayley algorithm is selected, this value and *Eigen Cayley Sigma* also determine which of the two Cayley transformation methods (A or B) is used by ARPACK.

This card is not applicable to the eggroll eigensolver, which takes shift values from the *Eigen Initial Shifts* card.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 4.2.15 Eigen Relative tolerance

| Eigen Relative tolerance = <float> |
|---|

### Description/Usage

This optional card specifies the relative residual tolerance used in the ARPACK eigensolver. Valid input are defined as:

| | |
|---|---|
| **tol>=0.0** | The relative termination criterion used within the eigensolver. |

The default value of **tol** is 1.0e-6.

### Examples

Here is a sample card:

```
Eigen Relative tolerance = 1.0e-10
```

### Technical Discussion

The value of **tol** specifies a termination condition used within the eigensolver which is applied to the transformed eigensystem. The correspondence of "internal residual" to "error in eigenpair" is even more difficult with the generalized eigenvalue problem than "residual" to "error in solution vector" with regular linear solvers.

This card is not applicable to the eggroll eigensolver, whose tolerance is specified with the *Eigen Tolerance* card.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 4.2.16 Eigen Linear Solver tolerance

| Eigen Linear Solver tolerance = <float> |
|---|

### Description/Usage

This optional card is used to specify the convergence tolerance used by Aztec linear solvers when solving the shifted linear system used by the ARPACK eigensolver, and is applicable only when an Aztec iterative solver is specified for the steady state problem. The absolute tolerance passed from ARPACK to Aztec is the product of this value and the norm of the mass matrix, which is recalculated at each iteration. Valid input are defined as:

| | |
|---|---|
| **LStol>=0.0** | The base linear solver convergence criterion used within the eigensolver. |

The default value of **LStol** is the value specified for the linear solver (see the *Residual Ratio Tolerance* card) or its default of 1.0e-6.

### Examples

Here is a sample card:

```
Eigen Linear Solver tolerance = 1.0e-10
```

### Technical Discussion

This card value differs from the one in the Solver Specifications section because it has a mass matrix norm factor. It can also be used to set a smaller tolerance for eigensolves than for steady state solves.

This card is not applicable to the eggroll eigensolver, which does not use Aztec.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 4.2.17 Eigenvalue output frequency

| Eigenvalue output frequency = <int> |
|---|

### Description/Usage

This optional card is used by LOCA to determine how often to call the ARPACK eigensolver during a continuation problem. When this value is selected to **n**, ARPACK will be called on the first continuation step and every $\mathbf{n}^{th}$ step thereafter. n can also be set to -1, in which case ARPACK is called only after the last continuation step.

The default value is -1, in which case ARPACK is called on the last continuation step.

### Examples

Here is a sample card:

```
Eigenvalue output frequency = 2
```

### Technical Discussion

This card is useful for continuation problems when it is not desired to calculate eigenvalues at every continuation step, enabling time to be saved.

This card is not applicable to the eggroll eigensolver, which cannot be used during continuation.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 4.2.18 Eigenvector output frequency

| Eigenvector output frequency = <integer> |
|---|

### Description/Usage

This optional card is used by LOCA to determine how often to write out a set of eigenvectors from the ARPACK eigensolver to EXODUS II files during a continuation problem. When this value is selected to **n**, ARPACK eigenvectors will be written on the first continuation step and every $\mathbf{n}^{th}$ step thereafter. **n** can also be set to -1, in which case eigenvectors will be output only on the last continuation step.

The default value is -1, in which case eigenvectors are written on every continuation step.

### Examples

Here is a sample card:

```
Eigenvector output frequency = 2
```

### Technical Discussion

This card is useful for continuation problems when is is not desired to save eigenvectors at every continuation step, thus enabling savings of time and disk space.

This value is independent of the *Eigenvalue output frequency*, so if a frequency **m** is specified for eigenvalues and **n** is not a multiple of **m**, then eigenvectors will be written on step numbers which are common multiples of **m** and **n**.

When requesting linear stability analysis during a LOCA continuation run, the eigenvectors from all continuation steps corresponding to a given mode number (or mode number/wave number combination for 3D of 2D) will be written sequentially to the same EXODUS II file. The time stamp for each step will be the eigenvalue, rather than the continuation parameter value. Complex eigenvectors are written as two steps in the same file: first the real part (time-stamped with the real part of the eigenvalue), then the imaginary part (time-stamped with the imaginary part of the eigenvalue).

This card is not applicable to the eggroll eigensolver, which cannot be used during continuation.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 4.2.19 Eigenvector output file

| Eigenvector output file = <string> |
|---|

### Description/Usage

This optional card is used to select a custom base name for the EXODUS II files which will be created for writing eigenvectors from the ARPACK eigensolver. The name should be entered as follows:

*basename*.exoII  Filenames will be *basename*_mode*i*.exoII

If this card is absent, the default name is LSA.exoII.

### Examples

Here is a sample card:

```
Eigenvector output file = base-EV.exoII
```

### Technical Discussion

The actual filenames are formed from the base filename as follows:

For mode i (zero-based) when not using 3D of 2D LSA:  *basename*_mode*i*.exoII.

When using 3D of 2D LSA with wave number N:  *basename*_mode*i*_*wn*=*N*.exoII.

When requesting linear stability analysis during a LOCA continuation run, the eigenvectors from all continuation steps corresponding to a given mode number (or mode number/wave number combination for 3D of 2D) will be written sequentially to the same EXODUS II file.  The time stamp for each step will be the eigenvalue, rather than the continuation parameter value.  Complex eigenvectors are written as two steps in the same file: first the real part (time-stamped with the real part of the eigenvalue), then the imaginary part (time-stamped with the imaginary part of the eigenvalue).

The inputs *Eigenvalue output frequency* and *Eigenvector output frequency* are independent, so if a frequency **m** is specified for eigenvalues and **n** for eigenvectors, and **n** is not a multiple of **m**, then eigenvectors will be written on step numbers which are common multiples of **m** and **n**.

This card is not applicable to the eggroll eigensolver, which has a different default file naming convention with no options.

### Theory

No Theory.

### FAQs

No FAQs.

### References

No References.

## 4.3 3D of 2D Stability Analysis

### 4.3.1 Theory

For some 2D flow problems, it may be possible that the steady state flow is stable to all disturbances in either flow direction but unstable to disturbances in the third (transverse) direction. This situation could be analyzed by a full 3D simulation with stability analysis, but this would have considerably larger computational expense. An alternate method is to use normal mode analysis (e.g. Coyle 1984, Christodoulou 1988, Gates 1998) to model the transverse disturbances. This approach considers such disturbances as the sum of a series of Fourier normal modes of different wavelengths, or different wavenumbers - in this case, the actual transverse stability is that of the “most dangerous” normal mode -- i.e. the one which yields the largest leading eigenvalue (real part if complex). These authors have developed algorithms to include Fourier factors in the base 2D equations for residual and Jacobian assembly which account for the effects of these normal modes, such that the 2D finite element mesh can still be used and only a third velocity component needs to be added to the eigensystem.

This is the general approach taken in *Goma* for 3D of 2D stability analysis, which makes this method accessible to a wide range of physical problems. Basically, when 3D of 2D stability is requested for a given list of wavenumbers, the steady state problem is first solved as usual, then the augmented eigensystem is assembled using a two-pass approach for both the Jacobian and the mass matrix. All of the necessary factors are included with the basis and weight functions, to

minimize intrusion on the assembly equations.

### 4.3.2 Usage

The 3D of 2D stability algorithm is accessed by the user similarly to basic stability analysis, and requires only a few changes to an input file which is already set up for LSA (using either eggroll or ARPACK). One requirement of this method is that both momentum equations of the 2D problem must be included and must use Q2 interpolation for basis and weight functions; a third momentum equation will then be added as shown in Example 1.4.2.

For a file which is set up for LSA, the following changes must be made:

- Add the “momentum3” equation, using “Q2_LSA” interpolation type.
- Update (add 1 to) number of equations.
- Add BC’s for third velocity (W)
- Change Coordinate System to “PROJECTED_CARTESIAN”
- Change Linear Stability choice to “3D”
- Add the Eigen Wave Numbers card with a list of (1 or more) wavenumbers.

When used with either eigensolver, a list of converged eigenvalues will be generated on screen for each requested wave number, but print formats will differ. If the Eigen Record Modes card is set to n (1 or more), then the eigenvectors for the first n modes will be written to separate ExodusII files. In both cases, the file names are appended with mode and wave numbers, but the base names will differ. Also, eigenvector output files generated with ARPACK will have zero-based mode numbers and the eigenvalue itself is used as the time stamp. For continuation runs with ARPACK, each of these files will contain eigenvectors for the same mode/wavenumber combination at each continuation step. If the run was done in parallel, it will be necessary to run `fix` for any file you wish to view.

## 4.4 Examples

### 4.4.1 Stability of the lid driven cavity problem at Re = 1

The relevant sections of the input file to carry out linear stability analysis are as follows:

```
------------------------------------------------------------
Solver Specifications
------------------------------------------------------------
.
.
Linear Stability                = yes

------------------------------------------------------------
```

```
                    Eigensolver Specifications
-------------------------------------------------------------
Eigen Number of modes            = 10
Eigen Record modes               = 5
Eigen Size of Krylov subspace    = 30
Eigen Maximum Iterations         = 2
Eigen Number of Filter Steps     = 1
Eigen Recycle                    = no
Eigen Tolerance                  = 1.0e-06
Eigen Initial Vector Weight      = 0.1
Eigen Initial Shifts             = -50.0 -51.0 -52.0 -54.0
```

To run the stability software, the `Linear Stability` keyword in the `Solver Specifications` section was set to yes. Given that this problem is small (dimension is 1182), the dimension of the Krylov subspace was taken to be 30. The number of modes wanted was chosen to be 10 with the five most dangerous modes being written to file. The shifts were chosen to be close to the leading eigenvalue found below. This was found by trial and error (refer to discussion in Section 4.5). The initial vector weight was taken to be 0.1. In this problem, changing the initial vector weight did not help speed up the eigenvalue extraction.

*Goma* converges to a solution for the cavity problem; the linear stability analysis is performed on the converged solution. The *Goma* output for both analysis steps is:

```
                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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
11:56:10 [0] 2.6e-12 2.2e-11 5.3e-12 1.9e-10 1.3e-08 1.2e-09  1  5.0e-02/6.0e-02
scaled solution norms   3.386794e+01  4.267119e-01  2.028370e+00

WARNING: Linear Stability Analysis is not (and cannot be) compatible
  with all user-supplied boundary conditions (i.e., BC cards with
  the term USER in them).  If you are using some, and you want LSA,
  then you must modify your boundary conditions accordingly.

Assembling J...
Assembling B...
 Initializing variables and allocating space...
 Arnoldi Eigenvalue Extractor

 Allocated work vectors
 Allocated work martices
 Pass  1  New Shift = -5.000000e+01 +0.000000e+00 i
 Pass  2
 De-allocated work storage
 ARN-IT done.

-------------------------------------------------------------------------------
 Eigensolver required 85 iterations.
 Found 8 converged eigenvalues.
 Leading Eigenvalue  = -5.235398e+01-0.000000e+00 i RES =  4.163666e-47
```

```
   Real           Imag           RES
-5.235398e+01 -0.000000e+00 i  4.163666e-47
-9.219072e+01 +3.614838e-01 i  7.087330e-29
-9.219072e+01 -3.614838e-01 i  7.087330e-29
-1.283342e+02 -0.000000e+00 i  9.032547e-20
-1.544776e+02 -0.000000e+00 i  5.007987e-16
-1.675191e+02 -0.000000e+00 i  1.794834e-13
-1.899878e+02 -8.560254e-01 i  2.109350e-10
-1.899878e+02 +8.560254e-01 i  2.109350e-10
push_mode                          =            5
Writing modes to file ...
                Mode    0 ... recorded.
                Mode    1 ... recorded.
                Mode    2 ... recorded.
                Mode    3 ... recorded.
                Mode    4 ... recorded.
Deallocating memory ... done.

-done


Proc 0 runtime:       0.02 Minutes.
```

The *Goma* steady state solution for this problem was written to the output exodusII file named `out.exoII`. The names of the files containing the five eigenvectors requested for writing (by the `Eigen Record modes` input) are:

LSA_1_of_5_out.exoII

LSA_2_of_5_out.exoII

LSA_3_of_5_out.exoII

LSA_4_of_5_out.exoII

LSA_5_of_5_out.exoII

Each file is an exodusII file that can be read and displayed by BLOT. These eigenvectors correspond to the first five eigenvalues. Note that this means that `LSA_2_of_5_out.exoII` corresponds to **`-9.219072e+01 +3.614838e-01 i`** whereas `LSA_3_of_5_out.exoII` corresponds to **`-9.219072e+01 -3.614838e-01 i`** in this example. A velocity vector plot of the leading mode is shown in Figure 4.1.

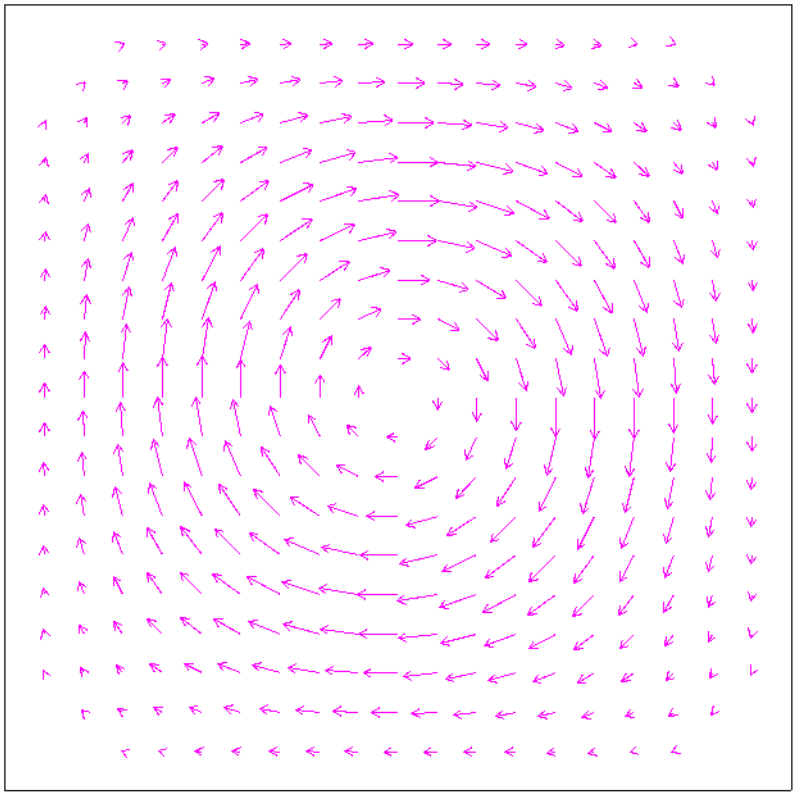


Figure 4.1 Leading eigenvector of lid driven cavity at Re=1.

### 4.4.2 3D of 2D stability of the lid driven cavity problem at Re=1

To access the 3D of 2D stability algorithm for this problem, the previous input file is modified as indicated in Section 2.3.2. The affected cards are:

```
Linear Stability= 3D
Eigen Initial Shifts= -40.0 -51.0 -52.0 -54.0
Eigen Wave Numbers= 0.0 1.0 2.0
BC= W NS 3 0.0
BC= W NS 1 0.0
```

```
BC= W NS 2 0.0
BC= W NS 4 0.0
Coordinate System= PROJECTED_CARTESIAN
Number of EQ = 4
EQ= momentum3  Q2_LSA U3 Q2_LSA    0.  1.  1.  1.  0.  0.
```

Note that the new momentum3 EQ card must be inserted just below the existing momentum2 card -- the three cards must remain together. Also, the first Eigen Initial Shift was changed to accommodate the higher wavenumbers.

The output from *Goma* is:

```
             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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
12:29:02 [0] 9.4e-02 1.9e+00 3.5e-01 3.4e+01 4.9e+02 7.0e+01  1  1.1e-01/1.0e-01
12:29:02 [1] 3.0e-04 1.4e-02 1.2e-03 1.2e-01 4.0e+00 4.2e-01  1  1.0e-01/8.0e-02
12:29:02 [2] 6.5e-09 3.6e-07 2.6e-08 2.5e-06 8.3e-05 8.8e-06  1  1.1e-01/8.0e-02
12:29:03 [3] 4.1e-17 3.5e-15 1.8e-16 8.9e-15 7.6e-13 6.5e-14  1  1.0e-01/9.0e-02
scaled solution norms   3.386794e+01  3.107662e-01  1.730999e+00

WARNING: Linear Stability Analysis is not (and cannot be) compatible
  with all user-supplied boundary conditions (i.e., BC cards with
  the term USER in them).  If you are using some, and you want LSA,
  then you must modify your boundary conditions accordingly.

Solving for wave number 1 out of 3.  WAVE NUMBER = 0
Assembling J (pass 1)...
Assembling J (pass 2)...
Assembling B (pass 1)...
Assembling B (pass 2)...
Entering eggrollinit...
Entering eggrollwrap...
 Initializing variables and allocating space...
 Arnoldi Eigenvalue Extractor

 Allocated work vectors
 Allocated work martices
 Pass  1  New Shift = -4.000000e+01 +0.000000e+00 i
 Pass  2
 De-allocated work storage
 ARN-IT done.

--------------------------------------------------------------------------------
NORMAL MODE WAVE NUMBER = 0
 Eigensolver required 85 iterations.
 Found 8 converged eigenvalues.
 Leading Eigenvalue  = -5.235398e+01-0.000000e+00 i RES =  8.518212e-42
    Real           Imag           RES
 -5.235398e+01 -0.000000e+00 i  8.518212e-42
 -9.219072e+01 +3.614838e-01 i  5.467111e-26
 -9.219072e+01 -3.614838e-01 i  5.467111e-26
```

```
 -1.283342e+02 -0.000000e+00 i  1.335964e-18
 -1.544776e+02 -0.000000e+00 i  1.872965e-14
 -1.675191e+02 -0.000000e+00 i  1.874266e-11
 -1.899878e+02 -8.560254e-01 i  1.267486e-09
 -1.899878e+02 +8.560254e-01 i  1.267486e-09
 push_mode                          =            5
 Writing modes to file ...
                 Mode    0 ... recorded.
                 Mode    1 ... recorded.
                 Mode    2 ... recorded.
                 Mode    3 ... recorded.
                 Mode    4 ... recorded.
Deallocating memory ... done.
Solving for wave number 2 out of 3.  WAVE NUMBER = 1
Assembling J (pass 1)...
Assembling J (pass 2)...
Assembling B (pass 1)...
Assembling B (pass 2)...
Entering eggrollinit...
Entering eggrollwrap...
 Initializing variables and allocating space...
Arnoldi Eigenvalue Extractor

 Allocated work vectors
 Allocated work martices
 Pass  1  New Shift = -4.000000e+01 +0.000000e+00 i
 Pass  2
 De-allocated work storage
 ARN-IT done.

--------------------------------------------------------------------------------
NORMAL MODE WAVE NUMBER = 1
 Eigensolver required 91 iterations.
 Found 10 converged eigenvalues.
 Leading Eigenvalue  = -4.836323e+01+3.013838e-01 i RES =  1.531033e-40
    Real           Imag           RES
 -4.836323e+01 +3.013838e-01 i  1.531033e-40
 -4.836323e+01 -3.013838e-01 i  1.531033e-40
 -4.840189e+01 -0.000000e+00 i  3.570638e-07
 -5.334569e+01 -0.000000e+00 i  2.696544e-33
 -7.887323e+01 -0.000000e+00 i  5.620381e-19
 -7.952717e+01 -0.000000e+00 i  1.478867e-19
 -9.344237e+01 +2.740097e-01 i  3.290522e-13
 -9.344237e+01 -2.740097e-01 i  3.290522e-13
 -9.875473e+01 -0.000000e+00 i  4.547433e-11
 -1.503152e+02 -0.000000e+00 i  3.600397e-11
 push_mode                          =            5
 Writing modes to file ...
                 Mode    0 ... recorded.
                 Mode    1 ... recorded.
                 Mode    2 ... recorded.
                 Mode    3 ... recorded.
                 Mode    4 ... recorded.
Deallocating memory ... done.
Solving for wave number 3 out of 3.  WAVE NUMBER = 2
```

```
Assembling J (pass 1)...
Assembling J (pass 2)...
Assembling B (pass 1)...
Assembling B (pass 2)...
Entering eggrollinit...
Entering eggrollwrap...
 Initializing variables and allocating space...
 Arnoldi Eigenvalue Extractor

 Allocated work vectors
 Allocated work martices
 Pass  1  New Shift = -4.000000e+01 +0.000000e+00 i
 De-allocated work storage
 ARN-IT done.

--------------------------------------------------------------------------------
NORMAL MODE WAVE NUMBER = 2
 Eigensolver required 57 iterations.
 Found 10 converged eigenvalues.
 Leading Eigenvalue  = -4.683098e+01-1.065660e-01 i RES =  2.650689e-41
    Real           Imag           RES
 -4.683098e+01 -1.065660e-01 i  2.650689e-41
 -4.683098e+01 +1.065660e-01 i  2.650689e-41
 -5.631855e+01 -0.000000e+00 i  2.181672e-31
 -7.914762e+01 -0.000000e+00 i  6.955747e-19
 -8.472175e+01 -0.000000e+00 i  9.876808e-18
 -9.740884e+01 +3.948238e-01 i  2.448277e-11
 -9.740884e+01 -3.948238e-01 i  2.448277e-11
 -9.903985e+01 -0.000000e+00 i  3.766791e-10
 -1.286752e+02 +6.173513e-01 i  8.271854e-07
 -1.286752e+02 -6.173513e-01 i  8.271854e-07
 push_mode                          =           5
 Writing modes to file ...
                 Mode    0 ... recorded.
                 Mode    1 ... recorded.
                 Mode    2 ... recorded.
                 Mode    3 ... recorded.
                 Mode    4 ... recorded.
Deallocating memory ... done.

 -done


 Proc 0 runtime:       0.07 Minutes.
```

Here, the leading eigenvalue (real part) is seen to increase with wavenumber over this range, indicationg that some transverse perturbations are less stable than those in the other two directions (although not unstable in this case). This run generated fifteen eigenvector output files (one for each requested mode/wavenumber combination) with names appended accordingly. These names are of the type “LSA_5_of_5_wn=2_out.exoII“; those generated with ARPACK contain this information in a slightly different format. One check on this algorithm is that when the wavenumber is zero (or infinite wavelength), the standard 2D stability results should be recovered

-- comparison of the first set of eigenvalues with those found in Example 2.4.1 reveals that this is the case.

### 4.4.3 Stability of the lid driven cavity with continuation in Re

To perform continuation with stepwise stability analysis, it is necessary to switch from eggroll to ARPACK; this is achieved when LOCA continuation is specified. The input file is modified by removing the Hunting Specifications section (including its header) and adding or changing the following cards:

```
(in Continuation Specifications:)
Continuation= loca
Continuation Type= BC
Boundary condition ID= 4
Boundary condition data float tag= 0
Material id= 1
Material property tag= 1700
Material property tag subindex= 0
Initial parameter value= 1.0
Final parameter value= 5.0
delta_s= 2.0
Maximum number of path steps= 3
Minimum path step= 10.0
Maximum path step= 50.0
Continuation Printing Frequency= 1
LOCA method = zero
(in Eigensolver Specifications:)
Eigen Algorithm= si
Eigen Cayley Sigma= -50.0
```

These cards set up a continuation run in the lid speed (here, taken to be Re) from 1 to 5 in constant steps of 2 by changing the float on BC #4 (fifth from the top) with a call to ARPACK (shift-and-invert method with shift of -50) after each step is completed. The Goma output for this run is as follows:

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Start of Step:     1   Continuation Param =         1 from         1
        Continuation method = Zero-order Continuation
        delta_c_p        = 0.00000e+00
        delta_c_p_old    = 0.00000e+00

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

        ---------------------------------
        Zero order continuation:
        Step number:    1 of    3 (max)
        Attempting solution at:
        BCID=  0 DFID=    0 Parameter= 1.000000e+00 delta_s= 0.000000e+00
```

```
              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)
-------- --- ------- ------- ------- ------- ------- ------- --- ---------------
12:54:31 [0] 2.6e-12 2.2e-11 5.3e-12 1.9e-10 1.3e-08 1.2e-09  1  5.0e-02/6.0e-02scaled
solution norms   3.386794e+01  4.267119e-01  2.028370e+00

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Continuation Step Number     1 was a success: Param =          1
        Order   = 0
        delta_c_p = 0

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


        Starting Eigenvalue Calculation (2 matrix fills and an ARPACK call):
Assembling LSA Jacobian ...
Assembling LSA Mass matrix ...
        SHIFT-n-INVERT: sigma,ncv = -50 30

            Eigensolver Initial Guess Generation
            Requiring 2 extra orders resid reduction: 2.03356e-11

        Before poleze: sigma and mu = -50  0
        After  poleze: sigma and mu = -50  0

        8 converged of 12 candidate eigenvalues found
Printing real eigenvector with value  -52.354
Printing eigenvector pair for complex eigenvalues:  -92.1907 +- 0.361441 i
Printing real eigenvector with value  -128.334
Printing real eigenvector with value  -154.478

 Ritz values (Real,Imag) and direct residuals
 ----------------------------------------
              Col   1       Col   2       Col   3
  Row   1:   -5.23540E+01   0.00000E+00   7.81597E-14
  Row   2:   -9.21907E+01  -3.61441E-01   4.78150E-12
  Row   3:   -9.21907E+01   3.61441E-01   4.78150E-12
  Row   4:   -1.28334E+02   0.00000E+00   9.66338E-13
  Row   5:   -1.54478E+02   0.00000E+00   2.95017E-11
  Row   6:   -1.67519E+02   0.00000E+00   3.55271E-12
  Row   7:   -1.89988E+02  -8.55814E-01   5.56433E-09
  Row   8:   -1.89988E+02   8.55814E-01   5.56433E-09


Maximum number of iterations reached.

Eigenvalue Calculation Summary

The number of Ritz values requested is 10
The number of Arnoldi vectors generated (NCV) is 30
What portion of the spectrum: LR
The number of converged Ritz values is 8
Number of Implicit Arnoldi update iterations taken is 1
```

```
The number of OP*x is 30
The convergence criterion is 1.000000e-06

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Calculating initial guess for next continuation step

   Doing Zeroth-order continuation --
   previous solution used as initial guess

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Start of Step:     2   Continuation Param =         3 from         1
        Continuation method = Zero-order Continuation
        delta_c_p        = 2.00000e+00
        delta_c_p_old    = 0.00000e+00

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

        ----------------------------------
        Zero order continuation:
        Step number:    2 of    3 (max)
        Attempting solution at:
        BCID=  0 DFID=    0 Parameter= 3.000000e+00 delta_s= 2.000000e+00

               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)
-------- ---  -------  -------  -------  -------  -------  -------  ---  ---------------
12:54:32 [0] 4.4e-17 3.7e-15 2.0e-16 1.1e-14 4.7e-13 3.9e-14  1  5.0e-02/5.0e-02scaled
solution norms   3.386794e+01  4.267119e-01  2.028370e+00

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Continuation Step Number     2 was a success: Param =          3
        Order   = 0
        delta_c_p = 2

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


        Starting Eigenvalue Calculation (2 matrix fills and an ARPACK call):
Assembling LSA Jacobian ...
Assembling LSA Mass matrix ...
        SHIFT-n-INVERT: sigma,ncv = -50 30

            Eigensolver Initial Guess Generation
            Requiring 2 extra orders resid reduction: 2.02664e-11

        Before poleze: sigma and mu = -50  0
        After  poleze: sigma and mu = -50  0

        8 converged of 12 candidate eigenvalues found
Printing real eigenvector with value  -52.354
Printing eigenvector pair for complex eigenvalues:  -92.1907 +- 0.361441 i
Printing real eigenvector with value  -128.334
```

```
Printing real eigenvector with value  -154.478

 Ritz values (Real,Imag) and direct residuals
 ----------------------------------------
              Col   1       Col   2       Col   3
  Row   1:   -5.23540E+01   0.00000E+00   1.49214E-13
  Row   2:   -9.21907E+01  -3.61441E-01   1.81333E-12
  Row   3:   -9.21907E+01   3.61441E-01   1.81333E-12
  Row   4:   -1.28334E+02   0.00000E+00   1.25056E-12
  Row   5:   -1.54478E+02   0.00000E+00   5.99698E-12
  Row   6:   -1.67519E+02   0.00000E+00   5.42855E-12
  Row   7:   -1.89988E+02  -8.55814E-01   6.03077E-08
  Row   8:   -1.89988E+02   8.55814E-01   6.03077E-08


Maximum number of iterations reached.

Eigenvalue Calculation Summary

The number of Ritz values requested is 10
The number of Arnoldi vectors generated (NCV) is 30
What portion of the spectrum: LR
The number of converged Ritz values is 8
Number of Implicit Arnoldi update iterations taken is 1
The number of OP*x is 30
The convergence criterion is 1.000000e-06

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Calculating initial guess for next continuation step

   Doing Zeroth-order continuation --
   previous solution used as initial guess

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

Start of Step:     3   Continuation Param =         5 from         3
        Continuation method = Zero-order Continuation
        delta_c_p        = 2.00000e+00
        delta_c_p_old    = 2.00000e+00

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

        ---------------------------------
        Zero order continuation:
        Step number:    3 of    3 (max)
        Attempting solution at:
        BCID=  0 DFID=    0 Parameter= 5.000000e+00 delta_s= 2.000000e+00

              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)
-------- ---  -------  -------  -------  -------  -------  -------  ---  ---------------
12:54:32 [0] 3.9e-17 3.0e-15 1.6e-16 9.4e-15 3.8e-13 3.1e-14  1  5.0e-02/5.0e-02scaled
solution norms   3.386794e+01  4.267119e-01  2.028370e+00
```

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Continuation Step Number     3 was a success: Param =          5
        Order   = 0
        delta_c_p = 2

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~


        Starting Eigenvalue Calculation (2 matrix fills and an ARPACK call):
Assembling LSA Jacobian ...
Assembling LSA Mass matrix ...
        SHIFT-n-INVERT: sigma,ncv = -50 30

            Eigensolver Initial Guess Generation
            Requiring 2 extra orders resid reduction: 1.87703e-11

        Before poleze: sigma and mu = -50  0
        After  poleze: sigma and mu = -50  0

        8 converged of 12 candidate eigenvalues found
Printing real eigenvector with value  -52.354
Printing eigenvector pair for complex eigenvalues:  -92.1907 +- 0.361441 i
Printing real eigenvector with value  -128.334
Printing real eigenvector with value  -154.478

 Ritz values (Real,Imag) and direct residuals
 --------------------------------------------
              Col   1       Col   2       Col   3
  Row   1:   -5.23540E+01   0.00000E+00   9.94760E-14
  Row   2:   -9.21907E+01  -3.61441E-01   5.87378E-12
  Row   3:   -9.21907E+01   3.61441E-01   5.87378E-12
  Row   4:   -1.28334E+02   0.00000E+00   8.52651E-14
  Row   5:   -1.54478E+02   0.00000E+00   1.96110E-12
  Row   6:   -1.67519E+02   0.00000E+00   5.94014E-12
  Row   7:   -1.89988E+02  -8.55814E-01   7.39488E-08
  Row   8:   -1.89988E+02   8.55814E-01   7.39488E-08


 Maximum number of iterations reached.

 Eigenvalue Calculation Summary

 The number of Ritz values requested is 10
 The number of Arnoldi vectors generated (NCV) is 30
 What portion of the spectrum: LR
 The number of converged Ritz values is 8
 Number of Implicit Arnoldi update iterations taken is 1
 The number of OP*x is 30
 The convergence criterion is 1.000000e-06
 Simulation completed continuation in 3 steps
 Final Parameter Value: 5.000000e+00
```

```
          I will continue no more!
          No more continuation for you!

CONTINUATION ROUTINE HAS FINISHED:
         Ending Parameter value     = 5
         Number of steps            = 3
         Number of Matrix fills     = 3
         Number of Residual fills   = 6


-done


Proc 0 runtime:       0.03 Minutes.
```

While ARPACK supports an option to specify a base name for eigenvector output files, one was not selected here and the default "LSA.exoII" was applied. Five of these files were created with names of the type "LSA_mode0.exoII" Note that these modes are zero-based whereas they are one-based with eggroll. Each of these files contain three "time steps" which correspond to the continuation steps taken, and the time stamp for each step is the corresponding eigenvalue. Note also that the computed eigenvalues for the first step agree with those found in Example 4.4.1 (with eggroll).

## 4.5 User Guidance

Linear stability analysis of the solution of a previously unstudied process can be quite difficult without some fortuitous choices in the LSA setup. Experience will erase the uncertainty but the user requires a jump off point. To this end, the following guidance is provided; included are some *do's* and *don'ts* in problem setup.

### 4.5.1 User Boundary Conditions

In general, a LSA should not be performed if user-specified boundary conditions have been used in the *Goma* analysis, since boundary conditions are the main source of difficulties in doing an LSA. To solve for the requested eigenvalues, a matrix is constructed from the time derivatives of the solution variables. The boundary conditions from the associated transient problem are considered, and only those boundary conditions that contain time derivatives are inserted into this matrix. This can lead to unexpected complications: one would not normally consider the time-dependent boundary conditions when solving a steady-state problem. User-supplied boundary conditions should not be used, unless the user is aware of, and can implement, the appropriate time derivatives in the user function necessary to construct the correct matrix.

### 4.5.2 Leading Eigenvalue

The LSA reports a *Leading Eigenvalue* and a list of values up to the **Number of Modes** (Section 2.2.1) selected in the analysis. The *Leading Eigenvalue* is simply the eigenvalue with the largest real part from the list. If the shift has been chosen well (i.e., near the actual leading eigenvalue), then this will be the eigenvalue with the largest real part. However, if the shift was chosen poorly, or a phantom eigenvalue is present (the eigensolver may report an eigenvalue converged when it is not an actual eigenvalue), then the leading Eigenvalue may be misleading. In this case, repeated runs should be performed and the list consulted (ignoring the Leading Eigenvalue reported). This is true to a greater extent with eggroll than ARPACK, as the LOCA routines that check for convergence are more effective in discarding spurious modes, thereby detecting fewer phantom eigenvalues for similar conditions. Thus, the choices of ARPACK shft parameters appears to have a smaller effect on the reliability of the results. But beware that ARPACK may generate an error if it detects an eigenvalue to the right of (larger than) the primary shift parameter sigma; when this happens, just increase sigma and try again. Another way to decrease the effect of spurious modes (for either method) is to increase the Krylov subspace size.

### 4.5.3 Selecting Initial Shifts (eggroll)

The eggroll eigensolver is very sensitive to the user-supplied shifts (Section 2.2.9). The guidance is to pick a set of initial shifts which has at least one shift to the right (in the positive real direction) of the leading eigenvalue. In the example in Section 4.4, the initial shifts selected were -50, -51, -52, -54 while the leading eigenvalue was determined to be -52.35. With time this portion of the LSA module will become more robust, but for the present a trial and error approach is necessary for new models/processes. Following is some assistance to carry out this trial and error procedure.

The first shifts can be selected to span several orders of magnitude, or successive analyses can be used to span several orders of magnitude, e.g., -1 to -1000. (Recall, we are looking for stable solutions, so the eigenvalues will be negative.) By expanding this range, reducing the range, or refining within a selected range, the user should look for consistency in the leading eigenvalue, i.e., the leading eigenvalue should emerge from all the stable modes as the largest real value. Then a final set of shifts can be chosen which is in keeping with the *"... at least one shift to the right of the leading eigenvalue"* approach.

Alternatively, the user could apply "analytic intuition." The eigenvalues being sought imply a time constant, $\tau_i = \frac{1}{\lambda_i}$, for the problem of interest, which is simply

$$\lambda_i = \frac{1}{eigenvalue(i)} \tag{4-1}$$

The decay (or growth in unstable solutions) is given by $e^{\lambda_i t}$. If a user has some insight into the "time constant" for the process being modeled, i.e., seconds or milliseconds, for example, an

estimate of the leading eigenvalue can be made to guide the trial and error search referred to in the above paragraph.

### 4.5.4 Selecting Initial Shifts (ARPACK)

The two Cayley shift parameters, $\sigma$ and $\mu$, are specified using the `Eigen Cayley Sigma` and `Eigen Cayley Mu` input cards, respectively. With the LOCA drivers for ARPACK there are 3 transformations implemented. The Shift-and-Invert behaves the same as the one in eggroll and one can follow the same strategy in choosing the shift paramter (in fact it was added to LOCA expressly for eggroll users), and in this case $\mu$ does not apply. The two Cayley transformation options available through LOCA for driving ARPACK (A and B as discussed below) have different strengths depending on the type of instability that is expected. For reliability in determining the stability of a solution, the LOCA developers recommend Cayley version B. Method A is invoked by choosing $\sigma < \mu$, and Method B is invoked by choosing $\mu < \sigma$.

The Cayley version A has $\lambda < \sigma < \mu$, where $\lambda$ is the expected leading eigenvalue, and is good for finding real eigenvalues or those with relatively small imaginary parts. Since eigenvalues are dimensional quantities of units inverse time, it is not possible to pick good default values for the transformation parameters. The paper by Lehoucq and Salinger (2001) presents some guidance on choosing the Cayley parameters and discusses the trade-offs. If one wants to converge the first n eigenvalues and expects them to have real parts in the range $-10 < \lambda < 0$ and imgainary parts less then 10i, a reasonable choice would be $\sigma$=5 and $\mu$ = 25 n (Remember that $\sigma$ must be to the right of all eigenvalues. Choosing $\mu$ to be 5 times further away from the eigenvalues than $\sigma$ works well. Typical choices for the Kylov subspace size would be 20-50.

The Cayley version B has $\lambda < \sigma$ and usually has $\mu = -\sigma$. This transformation is very robust for finding Hopf instabilities where there is a relatively large imaginary component to the eigenvalue. There are some details of this transformation in the Sandia report by Burroughs et al.(2001). If one expects an instability with pure imaginary $\lambda = \omega i$ (i.e. with period $2i / \omega$) the a good choice would be $\sigma = \omega$ and $\mu = -\sigma$. This transformation corresponds directly to trapezoid rule time integration, with $\sigma = 2/\Delta t$, so one can use intuition on choosing time steps to choose the Cayley parameters. Typical values woiuld be $\sigma$=100, $\mu$=-100. With this transformation, we often need large Krylov subspaces than the other transformations, often 50-200; yet when using an iterative solver this expense is in part mitigated by the fact that the matrices are diagonallyt dominant and solve very quickly. Also, this method is by far the most robust in locating all eigenvalues with positive real part, irrespective of the imaginary part.

# References

[1] Arnoldi, W. E. 1951 The principle of minimized iteration in the solution of the matrix eigenvalue problem. *Quart. Appl. Math.* **9**: 17

[2] Chan, T. F. and Resasco, D. C. 1986 Generalized deflated block – elimination. *SIAM J. Numer. Anal.* **23**: 913.

[3] Christodoulou, K. N. 1988 *Computational Physics of Slide Coating.* Ph.D. Thesis, University of Minnesota. Published by University Microfilms International, Ann Arbor, MI.

[4] Coyle, D. J. 1984 *The Fluid Mechanics of Roll Coating: Steady Flows, Stability, and Rheology.* Ph.D. Thesis, University of Minnesota. Published by University Microfilms International, Ann Arbor, MI.

[5] Gates, I. D. 1999 *Slot Coating Flows: Feasibility, Quality.* Ph.D. Thesis, University of Minnesota. Soon to be available from Published by University Microfilms International, Ann Arbor, MI.

[6] Lehoucq, R. B. and Sorensen, D. C. 1996 Deflation techniques for an implicitly restarted Arnoldi iteration. *SIAM J. Matrix Anal. Appl.* **17**: 789

[7] Lehoucq, R. B. and Scott, J. A. Implicitly restarted Arnoldi methods and eigenvalues of the discretized Navier-Stokes equations. Technical Report SAND97-2712J, Sandia National Laboratories, Albuquerque, New Mexico, 1997.

[8] Saad, Y. 1992 *Numerical Methods for Large Eigenvalue Problems.* Manchester, U.K.: Manchester University Press.

[9] Schunk, P. R., Sackinger, P. A., Rao, R. R., Chen, K. S., Cairncross, R. A., Baer, T. A. and Labreche, D. A. GOMA 6.0 - A Full-Newton Finite Element Program for Free and Moving Boundary Problems with Coupled Fluid/Solid Momentum, Energy, Mass and Chemical Species Transport: User's Guide, SAND2013-1844, Sandia National Laboratories, Albuquerque, NM, 1998.

# Index


Sandia National Laboratories