Sandia National Laboratories

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

Albuquerque, New Mexico 87185- 0834

date: April 2, 2004
Original August 16, 2002

to: Distribution

from: E. D. Wilkes, MS-0834 (GRAM, Inc.)

subject: Usage of ARPACK eigensolver for linear stability analysis in GOMA (GT-023.1)

keywords: ARPACK, eggroll, eigenvalues, linear stability analysis, Arnoldi methods, Krylov subspace methods

input records: None

Introduction

The Library of Continuation Algorithms (LOCA) is included in the *Goma* source code and supplements the available continuation and linear stability analysis capabilities. This library contains algorithms for simple parameter continuation, arc length continuation, tracking of turning point/fold, pitchfork, and Hopf bifurcations, and an interface to the external ARPACK eigensolver. Details of each of these features are provided in the LOCA 1.0 manual (SAND 2002-0396). This access to ARPACK provides an alternative to the eggroll eigensolver package (included in the source code) for doing linear stability analysis, which offers the following new capabilities:

- Significant improvement in identifying and rejecting non-physical (phantom) eigenvalues.
- Compatibility with all linear solvers supported in *Goma* (except FRONT).
- Access to the two-parameter Cayley transformation algorithm as well as shift-and-invert.
- The ability to do stepwise linear stability analysis during continuation, either after every step or every N steps (without using an external script).
- 3D of 2D (normal mode) stability analysis is supported.
- All of the above can now be done in parallel.

User Interface Description

The LOCA interface to ARPACK uses the existing `Eigensolver Specifications` section of the input deck. Some of the input cards serve the same purpose with either eggroll or ARPACK, but there are also several additional cards which apply only to one or the other. Refer to Chapter 4 of the Advanced Capabilities manual for detailed descriptions of these cards and their eigensolver applicability. For an input file which has already been set up for linear stability analysis with eggroll, only minimal changes would be required to run the problem with ARPACK. The user can specify an ARPACK algorithm with the cards:

```
Linear Stability = {yes | 3D}

...

Eigen Algorithm = {si | cayley}
```

and adding cards as necessary for the chosen algorithm.

Tutorial description

This tutorial will demonstrate the usage of the new combined continuation and linear stability analysis (LSA) features in *Goma* which utilize the ARPACK/PARPACK eigensolver package through an interface provided in LOCA.

The Rayleigh-Benard test problem chosen for this tutorial consists of a rectangular channel of fluid which is three times as wide as it is high. The bottom surface is heated to maintain a desired temperature, while the top surface is held at a lower reference temperature. The side walls are adiabatic. A reference pressure of zero is specified at one point. No-slip and no-penetration BC's are imposed on the top and bottom, but only no-penetration is applied on the side walls. The fluid density is taken to be a linear function of temperature (constant volume expansion), while the other properties are considered constant. The ensuing vertical temperature gradient then gives rise to buoyancy-driven flow, which appears as circular flow in the channel. There are no free surfaces, so mesh equations are not used. The parameter of interest is the temperature difference across the domain. For this formulation, this is equal to the Rayleigh number of the flow because the other quantities evaluate to unity in CGS units. This will be used as the continuation parameter for these examples. This problem was previously studied by Burroughs et. al. (1999), who included analytical eigenvalues for the four leading eigenmodes at two Rayleigh numbers.

The files provided with the tutorial distribution are as follows:

- rb.mat (Material property file for all input files)
- rb.exoII (Starting ExodusII FEM input file)
- rb.b (brkfix file for rb.exoII - needed for parallel runs
- start-ac.in (Input file for initial continuation without LOCA)
- start.in (Input file for initial continuation with LOCA)
- eggroll.in (Input file for LSA at Ra=1500 with eggroll)
- si.in (Input file for LSA at Ra=1500 with ARPACK/shift & invert)
- cayley.in (Input file for LSA at Ra=1500 with ARPACK Cayley transformation)
- cont.in (Input file for continuation with LSA to Ra=2000 with ARPACK)
- cont-3d.in (Input file for 3D of 2D LSA at Ra=1500 with eggroll)
- si-3d.in (Input file for 3D of 2D LSA at Ra=1500 with ARPACK/shift & invert)
- cayley-3d.in (Input file for 3D of 2D LSA at Ra=1500 with ARPACK/Cayley)
- cont-3d.in (input file for continuation to Ra=2000 with 3D of 2D LSA)

Requirements

A recent version of *Goma* is required for this tutorial, along with ARPACK. The continuation examples can also be run in parallel, which will also require brkfix and PARPACK (included with the ARPACK distribution). The linear solvers used are UMFPACK and AZTEC gmres, which must be linked in to *Goma*. *Goma*'s memory requirement is about 85MB for this problem. The input and output solution files can be viewed using BLOT. The commands to be given for each step are given on the indented lines below. Either of the first two examples (start-ac.in or start.in) must be performed first, in order to generate the solution file needed by all of the other examples.

The source code for ARPACK and relevant documentation can be downloaded from:

http://www.caam.rice.edu/software/ARPACK

Procedures

1) Initial continuation without LOCA {Input file: start-ac.in}

This example uses the old zero-order continuation algorithm (from ac_conti.c) to continue in Rayleigh number (Ra) from 500 to 1500 in constant steps of 500. Here, the initial guess is all zeros and the Aztec GMRES solver is used. LSA will not be done in this case.

To run, type:

```
goma -i start-ac.in
```

Newton convergence is attained in four iterations for each step, and the velocity field is still very small.

To run in parallel on N processors, first break the input file as follows:

```
brk -n <N> rb.b rb.exoII
```

Then type:

```
mpirun -np <N> goma -i start-ac.in
```

The result should be almost the same. If you wish to view the solution output Exodus file out.exoII, first run fix to combine the output files:

```
fix -n 2 out
```

NOTE: The ASCII output has been suppressed for these examples. If you want to generate the ASCII solution output, then just provide a file name (e.g. `out.dat`) at the `SOLN file` input card.

2) Initial continuation with LOCA {Input file: start.in}

Here, the same continuation will be done using LOCA's analogous zero-order algorithm. This is invoked simply by changing the `Continuation` card from `zero` to `loca` - this is the only difference between the two input files.

To run, type:

```
goma -i start.in
```

The screen output is formatted differently with and without LOCA, but the solution proceeds identically, as indicated by the agreement in all six displayed norms (and number of Aztec solver iterations) at each Newton iteration.

This run can also be done in parallel (after running brk as above):

```
mpirun -np <N> goma -i start.in
```

NOTE: Before proceeding to the rest of the examples, stop and save the output file that was just generated (which will be used as a restart file):

```
cp out.exoII rb-in.exoII
```

Also, since some of these can be run in parallel, if you plan to do so go ahead and brk this file now (for <N> processors):

```
brk -n <N> rb.b rb-in.exoII
```

3) LSA with eggroll at Ra=1500 {Input file: eggroll.in}

This example will take the converged solution for Ra=1500 and call eggroll to check the eigenvalues. When using eggroll, your `Linear Solver` choice must be `umf` or `umff`, and you cannot use it in parallel. Moreover, eggroll is not supported for continuation of any kind; this has resulted in Perl scripts being created to get eigenvalues after a continuation run has been done. This will no longer be necessary.

To run, type:

```
goma -i eggroll.in
```

Since this example starts from a converged solution, one nearly trivial Newton iteration is taken here. The familiar four-line warning about user-supplied BC's appears, then the stability system is assembled (J and B matrices) and eggroll is called. Here, the UMFPACK working arrays may be re-dimensioned, with several warning lines appearing as IDIM and XDIM are increased. This is a known peculiarity which will be addressed at some point, but Goma still proceeds just fine.

The first five eigenvalues reported are (all real):

```
-1.637450     -3.088563     -4.118804     -7.254228     -8.299721
```

Burroughs et. al. (1999) report the analytical eigenvalues as:

```
-1.637        -3.088        -4.118        -7.254
```

Note that the first four reported eigenvalues match up with the analytical numbers to nearly the given precision. Here, the fifth eigenvalue is included because it is suspect. The reporting of eigenvalues such as these tends to be inconsistent and sensitive to platform type and input parameters -- see the following section for a discussion. One indication of this is the large difference in residual error compared to the first four values. It has been observed that slight differences in the conditions of this run may result in different anomalous reported eigenvalues, possibly within the numerical range of the known leading values. This type of behavior is characteristic of eggroll, and was one of the major motivations for implementing ARPACK in *Goma*.

This run generates a series of ExodusII output files for the eigenvectors with names of the type "LSA_1_of_5_out.exoII". This is a default naming convention for eggroll.

Now, let's see how ARPACK performs in the same situation ...

4) LSA with ARPACK (shift & invert) at Ra=1500 {Input file: si.in}

The only way one can do a true comparison of eggroll and ARPACK on the same problem is for a single step (no continuation) with UMFPACK as the linear solver (which means serial only). When these conditions are met, there is an easy way to switch from eggroll to ARPACK - just add this card to the `Eigensolver Specifications` section:

```
Eigen Algorithm          = si
```

This is the only difference between the files eggroll.in and si.in.

To run the same problem with ARPACK, type:

```
goma -i si.in
```

The format will be different with ARPACK, but a lot of the data displayed is similar. The UMFPACK reallocation warning may appear as noted above; if it does, just disregard it for now.

Now, the first four calculated eigenvalues are:

```
-1.63745      -3.08856      -4.11880      -7.25423
```

These are virtually the same as calculated by eggroll. Also note that no value was found near -8.3.

ARPACK offers an option to specify a custom base name for the eigenvector output files. If one is not specified, then a different default base name is used and the names are of the type "LSA_mode0.exoII". Another difference is that the eigenvalue is recorded as the time stamp value in the files generated with ARPACK (For eggroll, this value is always zero). When a complex eigenvector is recorded with ARPACK, it will be written in two consecutively numbered files. The first will contain the real part of the eigenvector with the real part of the eigenvalue as the time stamp, and the imaginary parts are written to the next file.

5) LSA with ARPACK (Cayley transformation) at Ra=1500 {Input file: cayley.in}

The LOCA interface to ARPACK allows either the shift & invert method (as in eggroll) or a Cayley transformation method to be used to calculate eigenvalues. Without going into detail, the Cayley algorithm uses two shift parameters (sigma and mu) to perform transformations. For now, let's just use default values (set to 100 and 1000).

The Cayley method is invoked by changing the `Eigen Algorithm` card input from `si` to `cayley`.

To run, type:

```
goma -i cayley.in
```

The only glaring difference between the two ARPACK methods for this case is in the way the error residuals are calculated, thus resulting in a shorter list of converged eigenvalues. Note that the first four reported eigenvalues are almost exactly the same as for shift & invert!

6) Continuation with stepwise LSA using ARPACK (Cayley transformation) from Ra=1500 to 2000 {Input file: cont.in}

The LOCA interface to ARPACK now allows eigenvalue calculation at each step of a continuation run. ARPACK also allows solution of the linear eigensystem with any linear solver supported by Goma (except front).

Now, let's change the solver to Aztec and fire up a continuation run.

To run, type:

```
goma -i cont.in
```

For the first step, the results are the same as with cayley.in. There are a number of differences between the input files, though. Obviously, the whole `Continuation Specifications` section has been reinserted. The `Linear Solver` was changed from `umf` to `gmres` (as noted). Also, several new input cards have been added to the `Eigensolver Specifications` section, rather than just falling back on the defaults:

```
Eigen Cayley Sigma                = 10.0
```

[default: 0 for shift & invert, 100 for Cayley]

```
Eigen Cayley Mu                    = 0.0
```

[default: N/A for shift & invert, 1000 for Cayley]

```
Eigen Relative tolerance           = 1.0e-8
```

[default: 1.0e-6]

```
Eigen Linear Solver tolerance      = 1.0e-8
```

[default: 1.0e-6]

```
Eigenvalue output frequency        = 1
```

[default: 1]

```
Eigenvector output frequency       = 1
```

[default: 1]

```
Eigenvector output file            = rbev.exoII
```

[default: LSA.exoII]

Note that even though the ARPACK shift values and tolerances were changed from the defaults (used by cayley.in), the calculated eigenvalues did not change at all! By comparison, eggroll is known to be agonizingly sensitive to these types of variations.

The continuation proceeds to Ra=1750 and 2000 with no problems. At 1750, the calculated leading eigenvalues are:

```
  +0.319870     -1.71595     -1.79698     -6.70707
```

At Ra=2000, these values are:

```
  +2.15629     +0.373087     -0.415512    -5.41704
```

Burroughs et. al. (1999) report the analytical value at Ra=2000 as:

```
  +2.157        +0.3738      -0.4151      -5.415
```

The results reported there also show that this problem is much more sensitive to mesh refinement at Ra=2000 than at Ra=1500, so the larger error (especially for the second

value ~ 0.2%) is to be expected. To illustrate this, the calculated eigenvalues at the same mesh size (31x61 nodes) in that study were:

```
+2.100        +0.1338       -0.4113       -6.046
```

Note that the names of the eigenvector output files are now based on the input name: rbev_mode0.exoII, etc. These files will contain one eigenvector for each continuation step (or each N steps if the input `Eigenvector output frequency` was set to a value N>1); each step is time-stamped with the eigenvalue for that mode and step. They can be viewed (e.g. with BLOT) to examine the leading disturbance modes, which tend to drive the flow in circular patterns (rolls) along the length of the channel.

Now, try running this case in parallel on N processors (if your platform permits):

```
brk -n N rb.b rb-in.exoII   [if not already done]
mpirun -np N goma -i cont.in
```

The results are virtually identical, but total run time is reduced. Although an eigensolve in ARPACK may take more time than one in eggroll for the same conditions, some of the lost ground can be regained by running in parallel (not an option with eggroll). As before, the output file names are "multiplex"ed and include "_1of2", etc. but can be recombined with fix.

7) 3D of 2D LSA with eggroll at Ra=1500 {Input: eggroll-3d.in}

Here, the original implementation of 3D of 2D stability analysis is used to evaluate the effects on 2D base flows of disturbances in the third (transverse) direction by normal mode analysis. This requires several changes in the input file (compare the two files eggroll.in and eggroll-3d.in). These include:

* Set: `Linear Stability = 3D`

* Add a list of `Eigen Wave Numbers`

* Set: `Coordinate System = PROJECTED_CARTESIAN`

* Add a third velocity equation, e.g.:

```
EQ = momentum3  Q2_LSA  U3     Q2_LSA     1.  1.  1.  1.  1.  0.
```

* Add appropriate boundary conditions for third velocity (W).

This has been set up in the input file. The input Exodus file for the 2D problem will suffice.

To run, type:

```
goma -i eggroll-3d.in
```

The input wave numbers are set to 0.0, 0.3, and 1.0 for this example. As a check on the algorithm, the zero wavenumber should give the same results as regular (2D) stability. The first four calculated eigenvalues for the first wavenumber (0.0) are:

```
-1.637450      -3.088563      -4.118804      -7.254228
```

These happen to be identical to those for the corresponding 2D case (eggroll.in), except that the "anomalous" eigenvalue near -8.3 is now absent. Note also that ARPACK has not found an eigenvalue with this value...

At the next wavenumber (0.3), the values are:

```
-1.574391      -2.953918      -4.098376      -7.016650
```

At the last wavenumber (1.0), the values are:

```
-1.193320      -1.963979      -4.071935      -5.200401
```

These results show that these eigenmodes may depend to some extent on the 3D wavenumber, and that a flow which is stable to all 2D disturbances could be unstable to a transverse disturbance - a major motivation for doing this type of stability analysis.

Note that the eigenvector output file names are now appended to include the wavenumber as follows: "LSA_1_of_5_wn=0.3_out-3d.exoII". Thus, a separate output file is created for each requested record mode at each requested wavenumber (a total of 15 files in this case). These eigenvectors now include the transverse (W) velocity disturbances, and as such are larger than those previously generated.

How will ARPACK handle this problem? Let's find out...

8) 3D of 2D LSA with ARPACK (shift & invert) at Ra=1500 {Input: si-3d.in}

Since this case does not involve continuation, it is possible to run the same problem with either eggroll or ARPACK. As before, just set the `Eigen Algorithm` card to `si`. Also, since the previous examples gave an indication of where the leading eigenvalues are, let's move the shift values closer to this range (`Sigma = 10, Mu = 0`).

To run, type:

```
goma -i si-3d.in
```

Here, the first four calculated eigenvalues for wavenumber 0.0 are:

```
-1.63745      -3.08856      -4.11880      -7.25423
```

These values agree with the previous 2d case (si.in) to at least five significant digits.

The values for wavenumber 0.3 are:

```
-1.57439     -2.95392     -4.09838     -7.01665
```

The values for wavenumber 1.0 are:

```
-1.19332     -1.96398     -4.07193     -5.20040
```

While analytical values are not available for the latter two wavenumbers, the results of the two methods agree to at least four significant digits.

The eigenvalue output files generated with ARPACK follow the same naming convention with or without 3D of 2D, and like those with eggroll, they are augmented with the wave number. These names are of the type "LSA_mode0_wn=0.3.exoII", which derived from the default base name "LSA.exoII".

9) 3D of 2D LSA with ARPACK (Cayley transformation) at Ra=1500 {Input: cayley-3d.in}

Here, ARPACK will be called for the same problem, but the `Cayley` transformation algorithm (sigma and mu) will be specified. Also, a custom name will be specified for the eigenvector output files.

To run, type:

```
goma -i cayley-3d.in
```

Here, the first four eigenvalues for wavenumber 0.0 are virtually identical to the previous cases, but there are slight numerical differences between the two ARPACK methods.

The values for wavenumber 0.3 are:

```
-1.57439     -2.95392     -4.09838     -7.01665
```

The values for wavenumber 1.0 are:

```
-1.19332     -1.96398     -4.07193     -5.20040
```

Now, the eigenvector output file names utilize the base name `rbev.exoII` from the input file, and are of the type "rbev_mode0_wn=0.3.exoII" - the extensions for the mode and wavenumber are the same. Note that the mode number is zero-based, whereas with eggroll it is one-based and the number of requested modes is included.

10) Continuation with stepwise 3D of 2D LSA using ARPACK (Cayley transformation) from Ra=1500 to Ra=2000 {Input: cont-3d.in}

Now, continuation will be combined with 3D of 2D stability analysis. This feature is invoked by including both the `Continuation Specifications` and `Eigensolver Specifications` sections of the input file. The one requirement is that LOCA handle the continuation ("Continuation = loca").

To run, type:

```
goma -i cont-3d.in
```

Here is a summary of the first four computed eigenvalues for each of the nine calls to ARPACK:

Ra	N	E0	E1	E2	E3
1500	0	-1.63745	-3.08856	-4.11880	-7.25423
1500	0.3	-1.57439	-2.95392	-4.09838	-7.01665
1500	1	-1.19332	-1.96398	-4.07193	-5.20040
1750	0	+0.31987	-1.71595	-1.79698	-6.70707
1750	0.3	+0.38966	-1.56597	-1.77387	-6.43695
1750	1	+0.81120	-0.46911	-1.73705	-4.38137
2000	0	+2.15629	+0.37308	-4.15514	-5.41704
2000	0.3	+2.23251	+0.39994	-2.52311	-5.41632
2000	1	+2.69285	+0.96031	+0.32545	-3.58971

The values for wavenumber 0 can be compared to the corresponding cases without the 3D of 2D algorithm. The values for Ra=1500 can be compared to the non-continuation examples, and to the analytical values reported by Burroughs et. al. (1999).

Note especially that the leading eigenvalues may increase with the wavenumber (at least initially), as shown for Ra=2000. This could be an important consideration, e.g. for process design.

Also, try this run in parallel:

```
mpirun -np 2 goma -i cont-3d.in
```

The results should be the same.

Notes on reading eigenvalue tables

As pointed out in the previous section, anomalous eigenvalues are sometimes reported as found. This is a characteristic of the Arnoldi algorithm, but occurs much more frequently with eggroll than with ARPACK; this was a major motivation for installing ARPACK. These “anomalous” eigenvalues can often be distinguished from the actual values in that they tend to appear and/or disappear from run to run of the same

problem, and their residuals (far right column) may differ significantly in magnitude. When it is suspected that the reported eigenvalues for a given problem include anomalous values, try repeating the calculation with slightly different values for the shift parameters and/or the number of iterations, or try running on a different platform or machine if available. The real eigenvalues will tend to persist from run to run, while the anomalous values will tend to be more volatile.

Also, it is common for eigenvalues to appear twice in the table. This could be an actual repeated real eigenvalue or a complex conjugate pair. For purely real eigenvalues, the reported complex part will usually be identically zero, but may also be a very small number. True complex pairs will have the actual calculated complex part in the second column (once positive, once negative), and this value should not be sensitive to the variations described above. Further guidance on eigenvalue analysis can be found in the *Goma* Advanced Capabilities manual.

Reference

Burroughs, E. A., Romero, L. A, Salinger, A. G., "Large Scale Eigenvalue Calculations for Computing the Stability of Buoyancy Driven Flows," Journal of Computational Physics, Dec. 2000