# *Jacobian Matrix Analysis Tool for RELAP5-3D*

**International RELAP5 User Meeting**
**Dr. George Mesina**

October 6, 2016

Idaho National Laboratory

# *Outline*

- Purpose: Improve Consistency
- RELAP5 Discrete Field Equations
- Alternate Form for Comparison
- Jacobian Tool Input Fluid States
- Refactoring and Comparisons
- Future Work

# Continued Improvement of RELAP5-3D

- Recently, INL has focused development efforts on improving RELAP5-3D code performance
  - Numerous User Problem corrections
  - Greater code release testing
  - Verification Testing
    - Normal runs, restart, backup, multi-case, multi-deck
  - Restart and plotting upgrades
  - Improved memory management
    - memory leak, pointer nullifying, proper deallocation, initializing
  - Fluid table interpolation and consistency improvements

# *Can Jacobian Matrix Construction be Improved?*

- Jacobian Matrix is the focal point of the solution to the TH governing equations
- How consistent is the numerical approximation?
- If not, what are the inconsistencies?

- If it can be improved, it might produce more exact answers, reduce mass error, lead to fewer backups and time-step cuts

- Idea: Build a consistency checking program, the Jacobian Tool
- Compare RELAP5-3D 5x5 Jacobian matrix to one calculated by the Jacobian Tool
- Make the comparison for many different fluid states

# Mass & Energy Governing Equations

- $\frac{\partial(\alpha_g \rho_g X_n)}{\partial t} = -\frac{1}{A}\frac{\partial}{\partial x}(\alpha_g \rho_g X_n v_g A)$
- $\frac{\partial(\alpha_g \rho_g U_g)}{\partial t} + P\frac{\partial \alpha_g}{\partial t} - Q_{ig} + Q_{gf} - \Gamma_{ig} h_g^{*} = -\frac{1}{A}\frac{\partial \alpha_g \rho_g U_g v_g A}{\partial x} - \frac{P}{A}\frac{\partial(\alpha_g v_g A)}{\partial x} + \Gamma_w h_g^{\prime} + Q_{wg} + DISS_g$
- $\frac{\partial(\alpha_f \rho_f U_f)}{\partial t} + P\frac{\partial \alpha_f}{\partial t} - Q_{if} - Q_{gf} + \Gamma_{ig} h_f^{*} = -\frac{1}{A}\frac{\partial \alpha_f \rho_f U_f v_f A}{\partial x} - \frac{P}{A}\frac{\partial(\alpha_f v_f A)}{\partial x} - \Gamma_w h_f^{\prime} + Q_{wf} + DISS_f$
- $\frac{\partial(\alpha_g \rho_g)}{\partial t} - \Gamma_g = -\frac{1}{A}\frac{\partial}{\partial x}(\alpha_g \rho_g v_g A)$
- $\frac{\partial(\alpha_f \rho_f)}{\partial t} + \Gamma_g = -\frac{1}{A}\frac{\partial}{\partial x}(\alpha_f \rho_f v_f A)$

Conserved Quantities

$$F = \begin{bmatrix} \alpha_g \rho_g X_n \\ \alpha_g \rho_g U_g \\ \alpha_f \rho_f U_f \\ \alpha_g \rho_g \\ \alpha_f \rho_f \end{bmatrix}$$

Primary Variables

$$x = \begin{bmatrix} X_n \\ U_g \\ U_f \\ \alpha_g \\ P \end{bmatrix}, \quad v = \begin{bmatrix} v_g \\ v_f \end{bmatrix}$$

# *Mass & Energy Governing Equations*

- Terms on left side of governing equations at new time level, n+1, in the semi-implicit time advancement.
- **Velocities** and time-level terms n on right
- $\boldsymbol{\partial F / \partial t + W} + \mathbf{\Gamma} + \mathbf{Q} = \mathrm{B} + \mathrm{g}\,(\boldsymbol{v})$
- Goal: convert PDEs to form

  $[\boldsymbol{\partial F / \partial} \mathbf{x} + \boldsymbol{W} + \boldsymbol{\partial}\mathbf{\Gamma} / \boldsymbol{\partial}\mathbf{x} + \boldsymbol{\partial} Q / \boldsymbol{\partial}\mathbf{x}][\partial \mathbf{x}_{\mathbf{i}} / \partial \mathrm{t}]$

  $= \mathrm{B} + \mathrm{g}\,(\boldsymbol{v})$
- Start w/ conserved terms

| Conserved, | PV Work, | Mass Transfer, | Energy Transfer, |
|---|---|---|---|
| | 0 | 0 | 0 |
| | | 0 | |
| | | 0 | |
| | 0 | | 0 |
| | 0 | | 0 |

# RELAP5-3D Solution Strategy

| | Expanded by Chain Rule | Until all terms have a factor of |
|---|---|---|
| | | |
| | | |
| | | |
| | | |
| | | |

# RELAP5-3D Solution Strategy

- All **non-primary** quantities that occur at new time level, **n+1**, in semi - implicit advancement method are converted similarly
  - Use **Taylor polynomial**
  - **Temperatures**, **densities**, and **heat and mass transfer**
- T↓f↑**n+1** ≈T↓f↑**n** +Δt∑i=1↑5▒(∂T↓f /∂**x** ↓**i** )↑n (∂**x** ↓**i** /∂t )↑**n+1** ,T↓g↑**n** **+1** ≈T↓g↑**n** +Δt∑i=1↑5▒(∂T↓g /∂**x** ↓**i** )↑n (∂**x** ↓**i** /∂t )↑**n+1**
- T↓↑**s,n+1** ≈T↓↑s,**n** +Δt∑i=1↑5▒(∂T↑s /∂**x** ↓**i** )↑n (∂**x** ↓**i** /∂t )↑**n+1**
- ρ↓f↑**n+1** ≈ρ↓f↑**n** +Δt∑i=1↑5▒(∂ρ↓f /∂**x** ↓**i** )↑n (∂**x** ↓**i** /∂t )↑**n+1** ,ρ↓g↑**n** **+1** ≈ρ↓g↑**n** +Δt∑i=1↑5▒(∂ρ↓g /∂**x** ↓**i** )↑n (∂**x** ↓**i** /∂t )↑**n+1**
- RELAP5 formulas for heat and mass transfer terms.

| | |
|---|---|
| | |
| | |

# RELAP5-3D Approx. for Heat & Mass Transfer

- In Vol. 1, Sec. 3.1.3, only temperatures are at new time in semi-implicit temporal discretization for $Q_{if}$, $Q_{ig}$, $Q_{gf}$, and $\Gamma_{ig}$.
  - Can substitute temperature Taylor Polynomials.
- Example $Q_{gf} = \left(\frac{P-P_s}{P}\right) H_{gf} (T_g - T_f) \Rightarrow Q_{gf}^{\mathbf{n+1}} \approx \left[\frac{P^n - P_s^n}{P^n}\right] H_{gf}^n \left(T_g^{\mathbf{n+1}} - T_f^{\mathbf{n+1}}\right)$
  - $T_f^{\mathbf{n+1}} \approx T_f^{\mathbf{n}} + \Delta t \sum_{i=1}^{5} \left(\frac{\partial T_f}{\partial \mathbf{x}_{\mathbf{i}}}\right)^n \left(\frac{\partial \mathbf{x}_{\mathbf{i}}}{\partial t}\right)^{\mathbf{n+1}}$, $T_g^{\mathbf{n+1}} \approx T_g^{\mathbf{n}} + \Delta t \sum_{i=1}^{5} \left(\frac{\partial T_g}{\partial \mathbf{x}_{\mathbf{i}}}\right)^n \left(\frac{\partial \mathbf{x}_{\mathbf{i}}}{\partial t}\right)^{\mathbf{n+1}}$
  - $Q_{gf}^{\mathbf{n+1}} \approx \left[\frac{P^n - P_s^n}{P^n}\right] H_{gf}^n \left[T_g^{\mathbf{n}} - T_f^{\mathbf{n}} + \Delta t\left(\sum_{i=1}^{5} \left[\frac{\partial T_g}{\partial \mathbf{x}_{\mathbf{i}}}\right]^n \left[\frac{\partial \mathbf{x}_{\mathbf{i}}}{\partial t}\right]^{\mathbf{n+1}} - \left[\frac{\partial T_f}{\partial \mathbf{x}_{\mathbf{i}}}\right]^n \left[\frac{\partial \mathbf{x}_{\mathbf{i}}}{\partial t}\right]^{\mathbf{n+1}}\right)\right]$
  - $= Q_{gf}^{\mathbf{n}} + \Delta t \sum_{i=1}^{5} Q_{gf,i}^n \left[\frac{\partial \mathbf{x}_{\mathbf{i}}}{\partial t}\right]^{\mathbf{n+1}}$
- $\mathbf{Q_{if}^{n+1}} \approx Q_{if}^n + \Delta t \sum_{i=1}^{5} \left[\frac{\partial Q_{if}}{\partial \mathbf{x}_{\mathbf{i}}}\right]^n \left[\frac{\partial \mathbf{x}_{\mathbf{i}}}{\partial t}\right]^{n+1}$, $\mathbf{Q_{ig}^{n+1}} \approx Q_{ig}^n + \Delta t \sum_{i=1}^{5} \left[\frac{\partial Q_{ig}}{\partial \mathbf{x}_{\mathbf{i}}}\right]^n \left[\frac{\partial \mathbf{x}_{\mathbf{i}}}{\partial t}\right]^{n+1}$
- $\mathbf{Q_{gf}^{n+1}} \approx Q_{gf}^n + \Delta t \sum_{i=1}^{5} \left[\frac{\partial Q_{gf}}{\partial \mathbf{x}_{\mathbf{i}}}\right]^n \left[\frac{\partial \mathbf{x}_{\mathbf{i}}}{\partial t}\right]^{n+1}$, $\mathbf{\Gamma_{ig}^{n+1}} \approx \Gamma_{ig}^n + \Delta t \sum_{i=1}^{5} \left[\frac{\partial \Gamma_{ig}}{\partial \mathbf{x}_{\mathbf{i}}}\right]^n \left[\frac{\partial \mathbf{x}_{\mathbf{i}}}{\partial t}\right]^{n+1}$

# RELAP5 Semi-Implicit Solution Strategy

- **Approximations** to derivatives w.r.t. $\mathbf{x}_i$ are obtained from **equations of state**.
- Time level n terms to right. Collect coefficients, $A_{k,i}^{n}$, of $\partial \mathbf{x}_i / \partial t$ of kth equation.
  - $\sum_{i=1}^{5} A_{k,i}^{n} [\partial \mathbf{x}_i / \partial t]^{\mathbf{n+1}} \approx b_k^{n} + g(\boldsymbol{v}^{\mathbf{n+1}})$
- Numerically approximate governing equations by:
  - $\mathbf{A}^{\mathbf{n}} [d\mathbf{x} / d\mathbf{t}]^{\mathbf{n+1}} \approx \mathbf{b}^{\mathbf{n}} + \mathbf{g}(\boldsymbol{v}^{\mathbf{n+1}})$ (1)
    - $\mathbf{A}^{\mathbf{n}}$ is the 5x5 Jacobian matrix for the PDEs
- Combine (1) w/ RELAP5 **momentum equations** and apply the RELAP5 two-stage semi-implicit solution process to obtain $[d\mathbf{x} / d\mathbf{t}]^{\mathbf{n+1}}$
- Time step: $\mathbf{x}^{\mathbf{n+1}} = \mathbf{x}^{\mathbf{n}} + \Delta \mathbf{t} [d\mathbf{x} / d\mathbf{t}]^{\mathbf{n+1}}$

# Is Discretization Consistent with Orig. PDEs?

- For terms like $\partial F_k/\partial \mathbf{x}_i$ , how close is a finite difference approximation to the RELAP5-3D calculation of the expanded one obtained using the equations of state?
- For 1st conserved quantity: $\frac{\partial F_1}{\partial t} = \frac{\partial F_1}{\partial \mathbf{x}} \frac{\partial \mathbf{x}}{\partial t} = \sum_{i=1}^{5} \frac{\partial F_1}{\partial \mathbf{x}_i} \frac{\partial \mathbf{x}_i}{\partial t}$
- $\frac{\partial F_1}{\partial \mathbf{x}_i} = \left[\begin{matrix} \frac{\partial(\alpha_g \rho_g X_n)}{\partial X_n} & \frac{\partial(\alpha_g \rho_g X_n)}{\partial U_g} & \frac{\partial(\alpha_g \rho_g X_n)}{\partial U_f} & \frac{\partial(\alpha_g \rho_g X_n)}{\partial \alpha_g} & \frac{\partial(\alpha_g \rho_g X_n)}{\partial P} \end{matrix}\right]$
- **RELAP5-3D form with derivatives from EOS,** $\left\{\frac{\partial F_1}{\partial \mathbf{x}_i}\right\}_R = \left[\begin{matrix} \alpha_g \left(\rho_g + X_n \frac{\partial \rho_g}{\partial X_n}\right) & \alpha_g X_n \frac{\partial \rho_g}{\partial U_g} & 0 & \rho_g X_n & \alpha_g X_n \frac{\partial \rho_g}{\partial P} \end{matrix}\right]$
- **First-order finite difference form, where** $y_i = y_0 + \Delta \mathbf{x}_i$, $\left\{\frac{\partial F_1}{\partial \mathbf{x}_i}\right\}_N = \left[\begin{matrix} \frac{F_1(y_1) - F_1(y_0)}{\Delta X_n} & \frac{F_1(y_2) - F_1(y_0)}{\Delta U_g} & \frac{F_1(y_3) - F_1(y_0)}{\Delta U_f} & \frac{F_1(y_4) - F_1(y_0)}{\Delta \alpha_g} & \frac{F_1(y_5) - F_1(y_0)}{\Delta \mathbf{P}} \end{matrix}\right]$

# *Finite Difference Method (FDM)*

- Same kind of comparison for every $\partial \boldsymbol{F}_k / \partial \mathbf{x}_i$ , k = 1, … , 5.
- FDM must select size of perturbation appropriately
  - $\Delta \mathbf{x}_i \approx \mathbf{10}^{-6}\, \mathbf{x}_i$
- FDM must select perturbed point, $y_i = y_0 + \Delta \mathbf{x}_i$ , carefully
  1. 0.0 ≤ $(X_n + \Delta X_n)$ ≤ 1.0
  2. 0.0 ≤ $(\alpha_g + \Delta \alpha_g)$ ≤ 1.0
  3. P$(y_0 + \Delta$P$) < P_{crit}$
  4. $X_n = 0$ at $y_0$ , then $X_n = 0$ at $y_0 + \Delta \mathbf{x}_i$
  5. The state of each fluid phase at $y_i$ (e.g. subcooled, metastable) must remain the same as at $y_0$ .
- If $y_i$ fails any of conditions 1 through 5, $\Delta \mathbf{x}_i \approx -\mathbf{10}^{-6}\, \mathbf{x}_i$ , $y_i = y_0 + \Delta \mathbf{x}_i$ .
- Must also compare RELAP5-3D calculation of energy and mass transfer terms to FDM.

# Mass Transfer Comparison

| Partial | RELAP5-3D | Finite Difference |
|---|---|---|
| | | |
| | | |
| | | |
| | 0 | |
| | | |

# Refactor PRESEQ for Jacobian Comparison

- Three forms of the Jacobian Matrix
  - PDE: $\boldsymbol{\partial F/\partial t} + \boldsymbol{W} + \boldsymbol{\Gamma} + \mathbf{Q} = \mathrm{B} + \mathrm{g}(\boldsymbol{v})$
  - RELAP5-3D: $\mathrm{A}_{\mathrm{A}} = \mathbf{A}_{\mathbf{C,w,A}}^{\mathbf{e}} + \mathbf{W}_{\mathbf{A}} + \mathbf{\Gamma}_{\mathbf{w,A}} + \mathbf{Q}_{\mathbf{A}}$
  - FDM: $\mathrm{A}_{\mathrm{N}} = [\boldsymbol{\partial}\mathbf{F}_{\mathbf{w}}/\boldsymbol{\partial}\mathbf{x}]_{\mathbf{N}} + \mathbf{W}_{\mathbf{N}} + \mathbf{\Gamma}_{\mathbf{w,N}} + \mathbf{Q}_{\mathbf{N}}$
- Jacobian tool must build and compare:
  - $[\mathbf{A}_{\mathbf{C,w,A}}^{\mathbf{e}} + \mathbf{W}_{\mathbf{A}}]$ to $[(\boldsymbol{\partial}\mathbf{F}_{\mathbf{w}}/\boldsymbol{\partial}\mathbf{x})_{\mathbf{N}} + \mathbf{W}_{\mathbf{N}}]$, $\mathbf{\Gamma}_{\mathbf{w,A}}$ to $\mathbf{\Gamma}_{\mathbf{w,N}}$, and $\mathbf{Q}_{\mathbf{A}}$ to $\mathbf{Q}_{\mathbf{N}}$.
- PRESEQ builds the Jacobian Matrix, $\mathrm{A}_{\mathrm{A}}$, in RELAP5-3D
  - Jacobian tool must use the same exact coding as RELAP5-3D subroutine PRESEQ for proper comparison
- PRESEQ coding was refactored to build separate submatrices of $\mathrm{A}_{\mathrm{A}}$.
  - These are added together before Gaussian elimination begins.
- Common coding and supporting memory in a shared module

# Input for Jacobian Tool

- The Jacobian tool does not have RELAP5-3D input processor
  - Needs a fluid state to evaluate fluid properties & FDM
  - Want minimum input to generate required fluid state data
  - Study of Jacobian Matrix led to 10 values:

| | | |
|---|---|---|
| P | vlm()%p | Total Pressure |
| | vlm()%voidg | Void (volume) fraction |
| | vlm()%quala | Noncondensable quality |
| | vlm()%tempg | Gas Temperature, or “delta temperature” of gas |
| | vlm()%tempf | Liquid Temp., or “delta temperature” of liquid |
| | vlm()%hgf | Direct heating heat transfer coefficient per unit volume |
| | vlm()%hif | Liquid interfacial heat transfer coefficient per unit volume |
| | vlm()%hig | Gas interfacial heat transfer coefficient per unit volume |
| F | N/A | Delta flag specifies that temperature deltas from saturation, rather than phasic temps. |

# Generating fluid states from input

- Internally RELAP5-3D uses pressure, P, and specific internal energy, U, rather than pressure and temperatures, T.
- To get specific internal energies from P and T input
  - Calls to tstate then statep generates data for single fluid species
  - If noncondensable present, must separately calculate saturation pressure first
- Input checked for limit violations such as
  - $0.0 \le X_n \le 1.0$, $0.0 \le \alpha_g \le 1.0$
  - Temperatures and pressures within limits of TPF property file
- For finite differences, the same checks are applied to perturbed fluid state points, $y_i = y_0 + \Delta \mathbf{x}_i$ .
  - If $y_i$ violates any of the checks, it is redefined as $y_i = y_0 - \Delta \mathbf{x}_i$ .

# Study Using a Collection of Input Fluid States

- Jacobian Tool input file has header information that includes the title, file names, and number of fluid states
  - Arrays are allocated to this size
- Perl script generated initial study fluid states. Specification:
  - 6 Pressures from 14.7 to 3000 PSI
  - 6 void fractions 0.0, 0.01, 0.1, 0.5, 0.99, 1.0
  - 6 noncondensable qualities, same as void fraction
  - 7 liquid delta-temperatures -100.0, -50.0, -10.0, -1.0, 0.0, 1.0, 10.0
  - 7 gas delta-temperatures 5.0, -1.0, 0.0, 1.0, 100.0, 500.0
  - Fixed values of Hif, Hig, Hgf, and delta-flag = 1.0
- Run with Jacobian Tool

# *Jacobian Tool Description*

- Reads input fluid state
- Builds RELAP5-3D Jacobian submatrices
- Builds Finite Difference Jacobian submatrices
- Compares element by element recording
  - Largest difference above a tolerance
  - Number above the tolerance
  - Number of Input Fluid State with largest difference
- Combines submatrices into Jacobian for PDEs
  - Calculates condition number
- Output differences & condition number for each fluid state input
- Summary of worst conditions

# Results and Conclusion

- 4 Inconsistencies in temporal terms
- 8 Inconsistencies in mass and energy terms
  - Some probably be caused by void ramp inconsistency
- Other issues discovered
  - Flexible Wall issue
  - Jacobian condition number very large
- Future work will include studying these inconsistencies and finding ways to improve them.