# *Degeneracy Hunter*
## A Tool for NLP Diagnostics

**Alexander W. Dowling**

Carnegie Mellon University

June 5th, 2014

# Agenda

- What are degenerate equations?
  - Motivating examples
- *Degeneracy Hunter* Algorithm
  - MILP formulation to find irreducible sets of degenerate equations
- Case Study: ASU Optimization
  - 30% speed-up by removing degenerate equations

# Degenerate Equations

- Definitions:
  - Redundant equations
  - Linearly dependent equations
  - Rank deficient Jacobian for the active set

- Consequences:
  - Singular Jacobian → Newton step fails
  - *Linearly independence constraint qualification* violated → non-unique KKT multipliers

# Separation Example

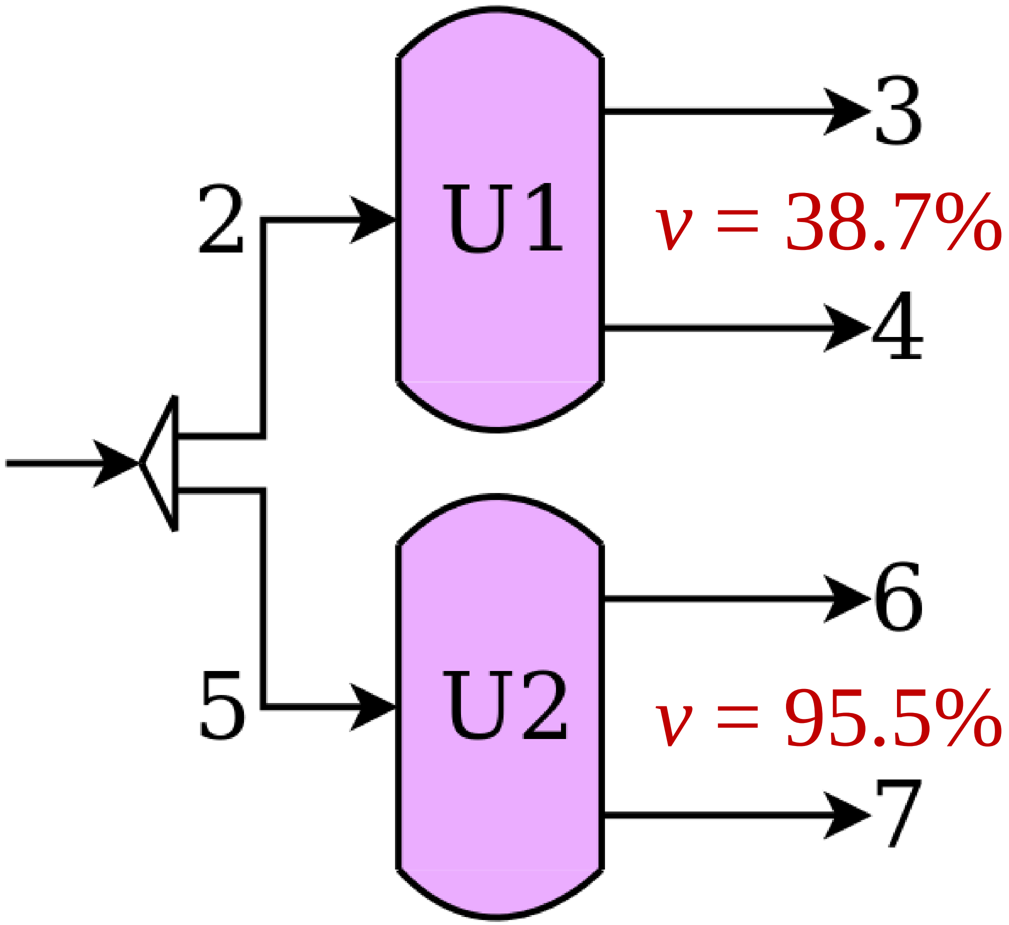
## Specifications

**Feed**: 1 mol/time
55% A, 45% B

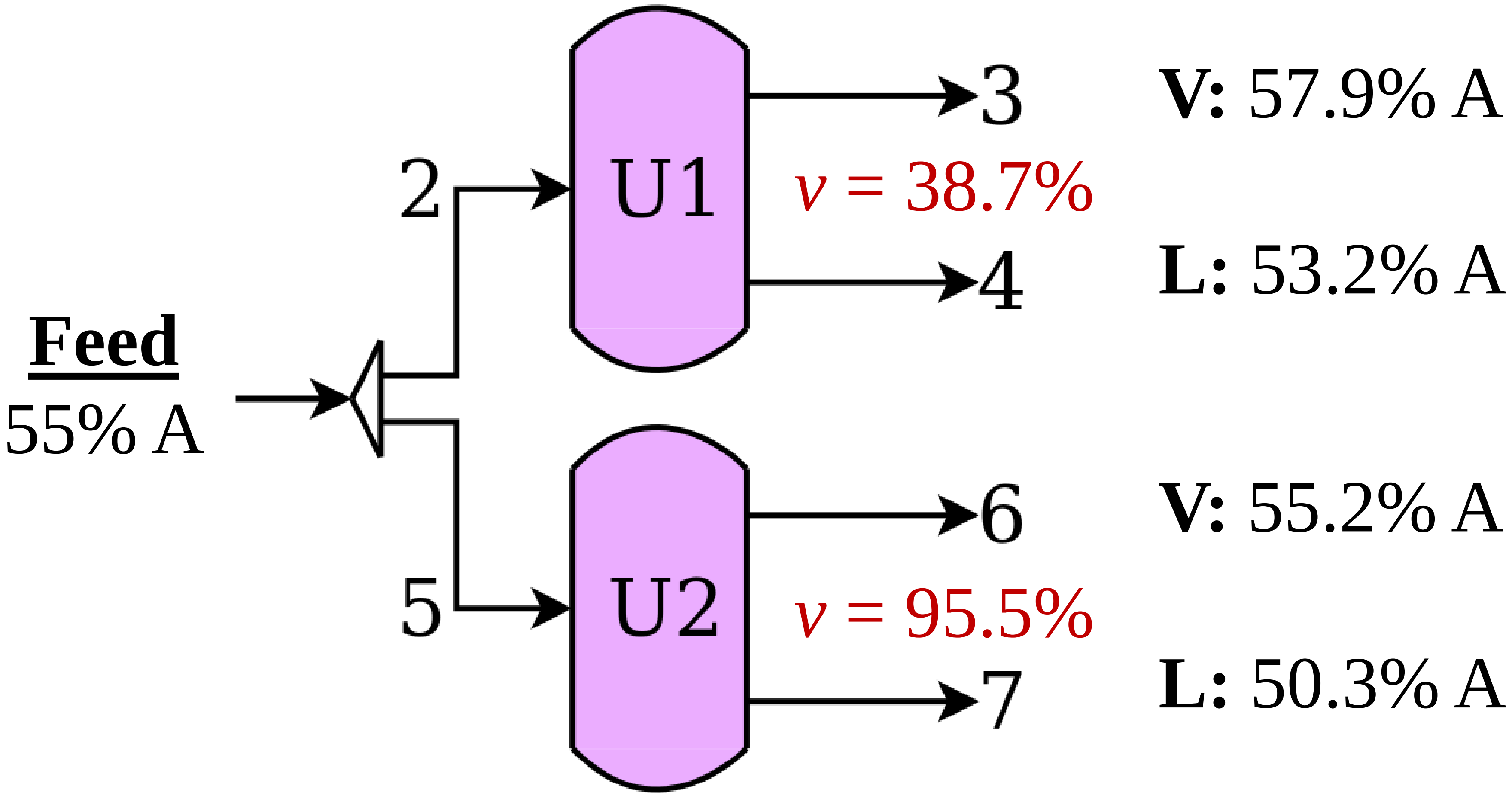
**Equilibrium Data**:

| | $K_A$ | $K_B$ |
|---|---|---|
| U1 | 1.088 | 0.9 |
| U2 | 1.099 | 0.9 |

**Cost**:
U1: $1.50 / mole feed
U2: $1.00 / mole feed



# Separation Example



# Separation Example Model

Set Definitions

$$s \in \{Streams\},\ c \in \{Components\},\ u \in \{Units\}$$
$$s^l, s^v \in \{Outlet\}(u)$$

Component Mole Balance

$$\sum_{s \in \{Inlet\}(u)} f_{s,c} = \sum_{s \in \{Outlet\}(u)} f_{s,c}, \quad \forall u, c$$

Overall Mole Balance

$$\sum_{s \in \{Inlet\}(u)} F_s = \sum_{s \in \{Outlet\}(u)} F_s, \quad \forall u$$

Vapor-liquid Equilib.

$$y_{s^v,c} = K_{u,c}\, x_{s^l,c}, \quad \forall u, c$$

Summation

$$\sum_c (y_{s^v,c} - x_{s^l,c}) = 0, \quad \forall u$$

Component Flowrate Def.

$$F_s x_{s,c} = f_{s,c}, \quad \forall s, c$$

# Separation Example Model

Set Definitions

$$s \in \{Streams\},\ c \in \{Components\},\ u \in \{Units\}$$
$$s^l, s^v \in \{Outlet\}(u)$$

Objective Function

$$Obj = \sum_u Cost_u F_{s^{inlet(u)}}$$

Feed Basis

$$F_2 + F_5 = 1$$

Recovery Definition

$$r_A = \frac{\sum_u f_{s^v,A}}{f_{2,A} + f_{5,A}}$$

Purity Definition

$$p_A \sum_u F_{s^v} = \sum_u f_{s^v,A}$$

# Separation Example Results

minimize Cost

s.t. $r_A \geq 60\ \%$

$\mathbf{p_A \geq 56\ \%}$

$F \geq 0$

U1: 0.505 mol/time

U2: 0.495 mol/time

Obj: 1.253 \$/time

minimize Cost

s.t. $r_A \geq 60\ \%$

$p_A \geq 55\ \%$

$\mathbf{F \geq 0}$

U1: 0.000 mol/time

U2: 1.000 mol/time

Obj: 1.000 \$/time

# Rank Check

Problem 1: Flows into both units

Size of Jacobian: 28

Rank of Jacobian: 28

Problem 2: No flow into Unit 1

Size of Jacobian: **28**

Rank of Jacobian: **27**

Rank Deficiency!

Jacobian contains weakly active inequality constraints and no bounds

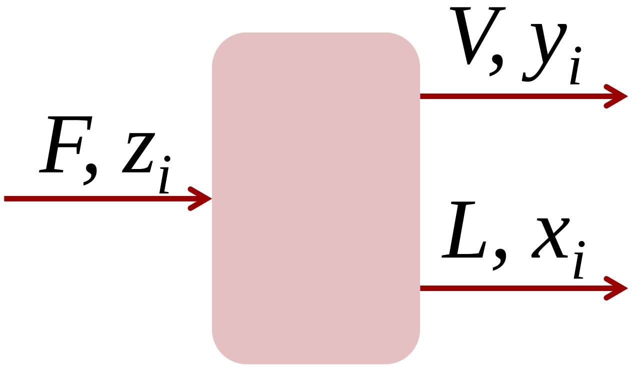
# Proof: Flash Degeneracies

$$F = V + L$$

$$F z_i = V y_i + L x_i, \quad \forall i$$

$$y_i = K_i x_i, \quad \forall i$$

$$\sum_i (y_i - x_i) = 0$$




Jacobian =

| | V | $y_1$ | $y_n$ | L | $x_1$ | $x_n$ | F | $z_1$ | $z_n$ | $K_1$ | $K_n$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Total MB | 1 | | | 1 | | | -1 | | | | |
| MB 1 | $y_1$ | $\cancel{V}$ | | $x_1$ | $\cancel{L}$ | | $z_1$ | $\cancel{-F}$ | | | |
| MB n | $y_n$ | | $\cancel{V}$ | $x_n$ | | $\cancel{L}$ | $z_n$ | | $\cancel{-F}$ | | |
| VLE 1 | | -1 | | | $K_1$ | | | | | $x_1$ | |
| VLE n | | | -1 | | | $K_n$ | | | | | $x_n$ |
| Sum. | | 1 | 1 | | -1 | -1 | | | | | |

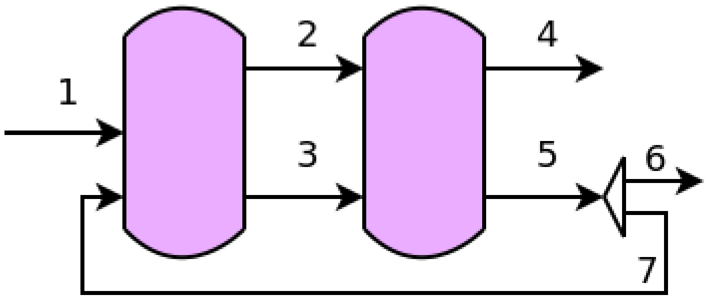
**Mole balances are degenerate when $F = L = V = 0$ and $\sum_i z_i = \sum_i x_i = \sum_i y_i$**

# Types of Degeneracies

- "Global" / "structural"
  - Always degenerate
  - Example: over specified system (next one)

- "Local" / "point-wise"
  - Only occur for specific values of variables
  - Example: Flash calculations + no flow

# Pressure Example




$$P_2 = P_3$$
$$P_4 = P_5$$
$$P_6 = P_7$$

Unit outlets are at same pressure

$$P_5 = P_7$$

No pressure drop in the splitter

$$P_1 \geq P_3$$
$$P_2 \geq P_5$$
$$P_7 \geq P_3$$

Pressure cannot increase in flash vessels

$$P_1 = \bar{P}$$

Specification

Over-specified: 7 variables and 8 equations

# Common Sources of Degenerate Equations in Flowsheet Problems

- Modeler error
  - Including redundant equations
  - Programming mistake (set specification)
- Zero flowrate and equilibrium calculations
- Pressure specifications and recycle loops

# “On the Fly” Fixes

- CONOPT – Active Set Method
  - Detect linearly dependent constraints and remove them from the active set
  - Determining the active set in combinatorial (i.e. difficult) – use heuristics

- IPOPT – Interior Point Method
  - Detect linearly dependent constraints using linear algebra routines
  - Use structural regularization to “knock out” degenerate equations
  - Active IPOPT development by Wei Wan (PhD student)

# Alternative: Reformulation

- **Hypothesis**: “On the Fly” fixes…
  - Add significantly computational costs
  - Don’t always work

- Alternative: Model Reformulation
  - Remove degenerate equations from the model before optimization
  - Difficult for problems with 1000’s of equations

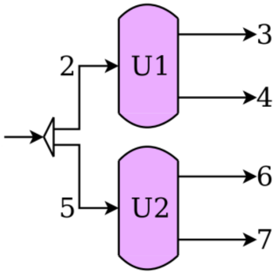
# Naive Approach

Idea: Simply delete degenerate equations detected by factorization (linear algebra routines) before solving optimization problem

1. Load initial point
2. Detect degenerate equations (factorize)
3. Remove degenerate equations
4. Solve NLP

# Naive Approach

Idea: Simply delete degenerate equations detected by factorization (linear algebra routines) before solving optimization problem



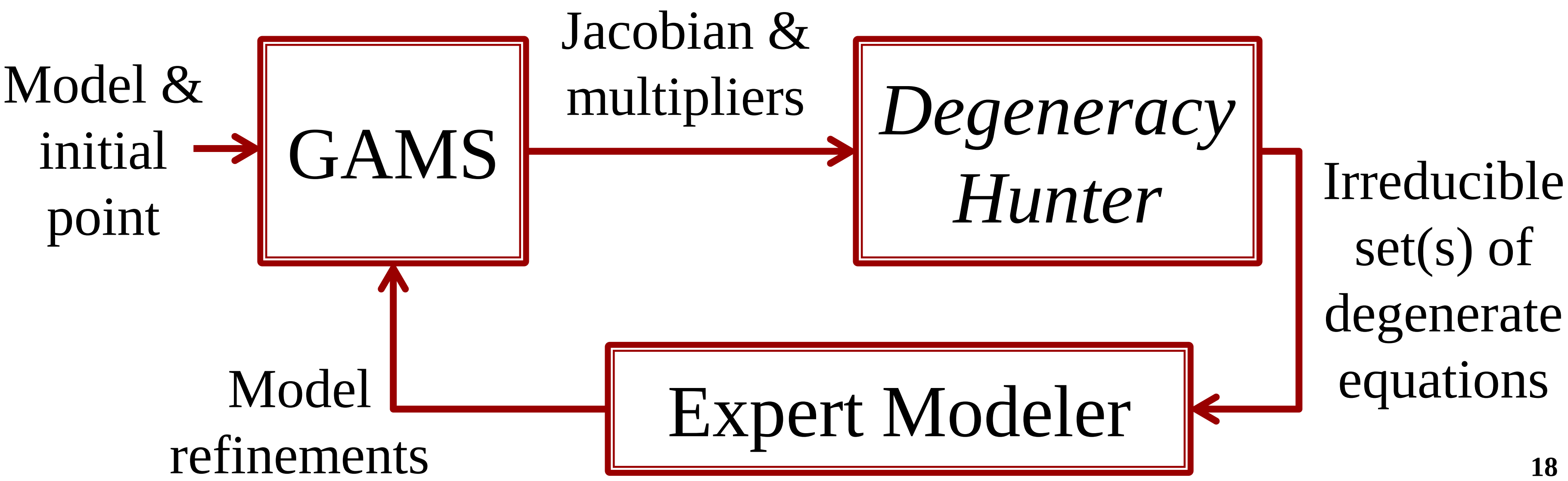

Separation Example

1. Modify model: $Obj = \sum_u Cost_u F_{s^v} - 100 p_A$
2. Consider Problem 2 solution as initial point
3. Delete component mole balance for A in U1
4. Result: creation of A, destruction of B

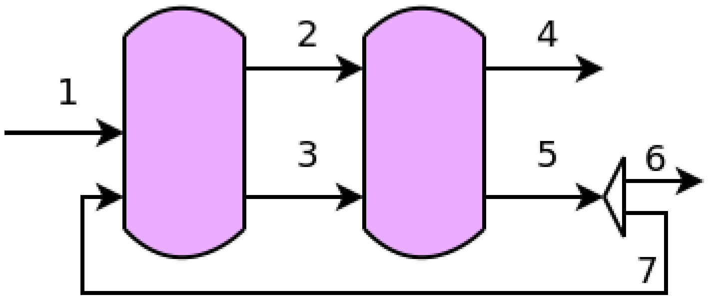
# *Degeneracy Hunter*

- Goal:
  - Provide modeler with irreducible sets of degenerate equations for a generic NLP

- Workflow:



# Pressure Example




$$\boxed{P_2 = P_3}$$
$$P_4 = P_5$$
$$P_6 = P_7$$

Unit outlets are at same pressure

$$\boxed{P_5 = P_7}$$

No pressure drop in the splitter

$$P_1 \geq P_3$$
$$\boxed{\begin{array}{c} P_2 \geq P_5 \\ P_7 \geq P_3 \end{array}}$$

Pressure cannot increase in flash vessels

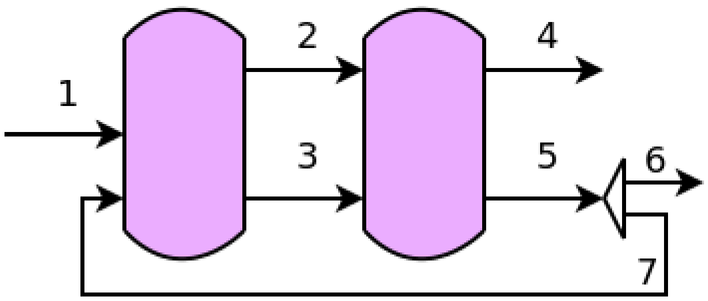
$$P_1 = \bar{P}$$

Specification

Irreducible Set of Degenerate Equations: $\boxed{\phantom{xxxxxxxx}}$

# Pressure Example

## Option 1



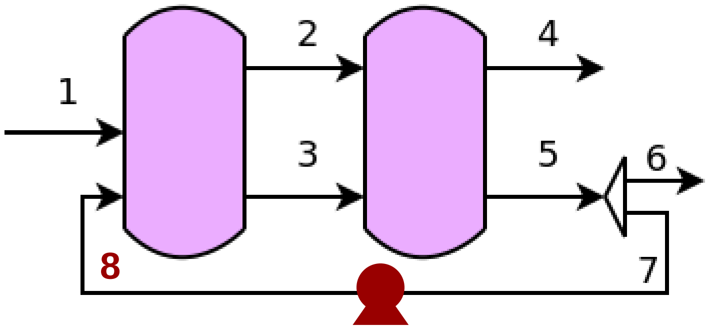

$$P_1 = P_3$$

$$P_3 = P_5$$

$$P_5 = P_7$$

$$\cancel{P_7 \geq P_3}$$

## Option 2




$$P_1 \geq P_3$$

$$P_2 \geq P_5$$

$$P_5 \geq P_7$$

$$P_8 \geq P_3$$

# *Degeneracy Hunter*

- Goal:
  - Provide modeler with irreducible sets of degenerate equations for a generic NLP

- Algorithm:
  1. Import derivative information from GAMS
  2. Determine active set*
  3. Factorize active set Jacobian to find non-pivot columns (equations)
  4. Solve MILP for each non-pivot column to determine irreducible set of degenerate equations

* User specifies if bounds and/or weakly active constraints should be considered

# MILP for Irreducible Set

For each $j$ in {non-pivot equations}, solve:

$$
\begin{aligned}
\min \quad & \sum_{i=1}^{n_{npe}} y_i \\
\text{s.t.} \quad & A_{dh}^T x = 0 \\
& -My_i \leq x_i \leq My_i \ , \quad \forall i = 1, ..., n_{npe} \\
& x_j = 1
\end{aligned}
$$

| | |
|---|---|
| $A_{dh}$ | Jacobian of the active set |
| $i$ | Equations in the active set |
| $y_i$ | Is eqn $i$ in the degenerate set (binary)? |
| $x_i$ | Singular vector component for eqn $i$ |

# *Degeneracy Hunter* Options

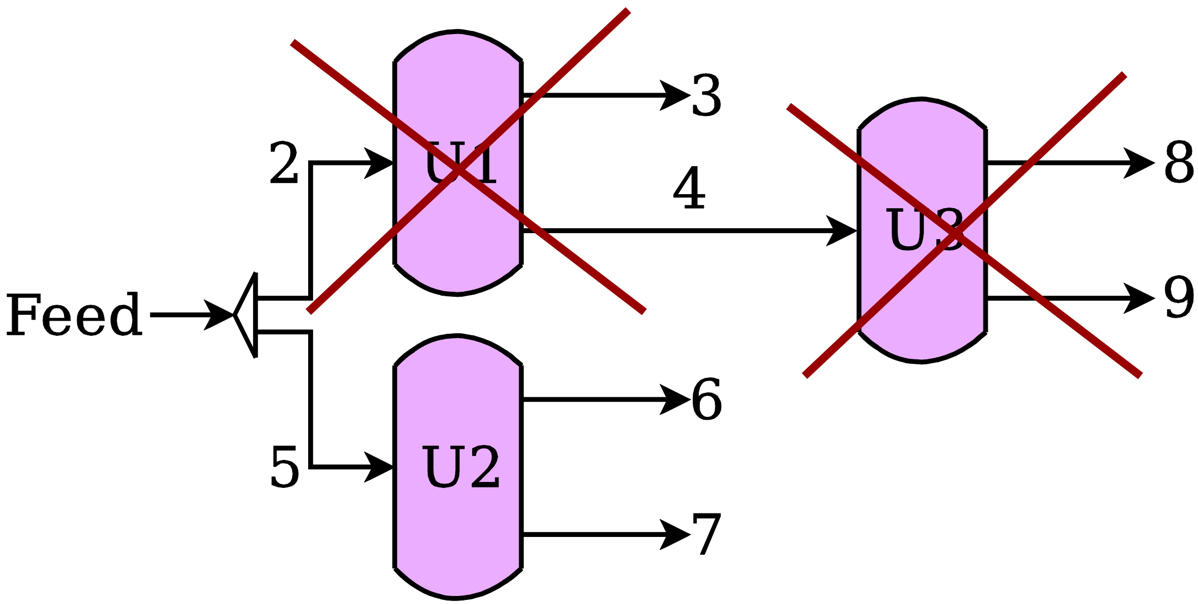
- Use Sparse linear algebra? **Yes**/no
  - `module.sparse`
- Threshold for classifying multipliers as weakly active (default: $\mathbf{10^{-10}}$)
  - `module.multTol`
- Tolerance for rank (dense only, default: $\mathbf{10^{-10}}$)
  - `module.rankTol`
- Verbose output (for debugging)? Yes/**no**
  - `module.verbose`

# *Degeneracy Hunter* Options

- Should weakly active constraints be considered when checking for degeneracies?
  - `module.degenHunt.weakAct`
- Should variable bounds be considered when checking for degeneracies?
  - `module.degenHunt.varBounds`
- Use MILP formulation? **Yes**/no
  - `module.degenHunt.optimal`
- Tolerance for displaying equations in MILP version (default: $\mathbf{10^{-6}}$)
  - `module.degenHunt.tol`

# *Degeneracy Hunter* Demo

Separation example with 3 units




Costs:

| | |
|---|---|
| U1: | 1.50 $ / mol feed |
| U2: | 1.00 $ / mol feed |
| U3: | 0.50 $ / mol feed |

**Min** **Cost**

**s.t.** $r_A \geq 90\%$

$p_A \geq 55.1\%$

Usage:

```
degeneracyHunter3(flagWeakActive,
    flagVarBounds, outputFileName)
```

# Degeneracy Hunter Demo

```
********** Equations **********
Total number: 38

Number of equality constraints: 38
Number of STRONGLY active equality constraints: 10
Number of WEAKLY active equality constraints: 28

Number of inequality constraints: 0
Number of ACTIVE inequality constraints: 0
Number of STRONGLY ACTIVE inequality constraints: 0
Number of WEAKLY ACTIVE inequality constraints: 0
Number of INACTIVE inequality constraints: 0

********** Variables **********
Total number: 43
Number of ACTIVE variable bounds: 19
Number of STRONGLY ACTIVE variable bounds: 5
Number of WEAKLY ACTIVE variable bounds: 18

********** Degrees of Freedom **********
At analyzed point, considering...
   all ACTIVE inequalities and bounds: -14
   only STRONGLY ACTIVE inequalities and bounds: 0
```

# Degeneracy Hunter Demo

```
****************************************
********** Classic Degeneracy Hunter **********
****************************************
Analyzed Jacobian contains all active inequality constraints
  and NO variable bounds

Rank of analyzed Jacobian: 37
Estimated number of dependent equations (using rank): 1
Estimated number of dependent equations (using dense rank calc.): 2

********** Suspect Equations (and Bounds) **********
Index     Active    Type      Name

24        Strong    Eqlty.    EqStrMoleFrac(S4,B)

34        Strong    Eqlty.    EqStrMoleFrac(S9,B)
```

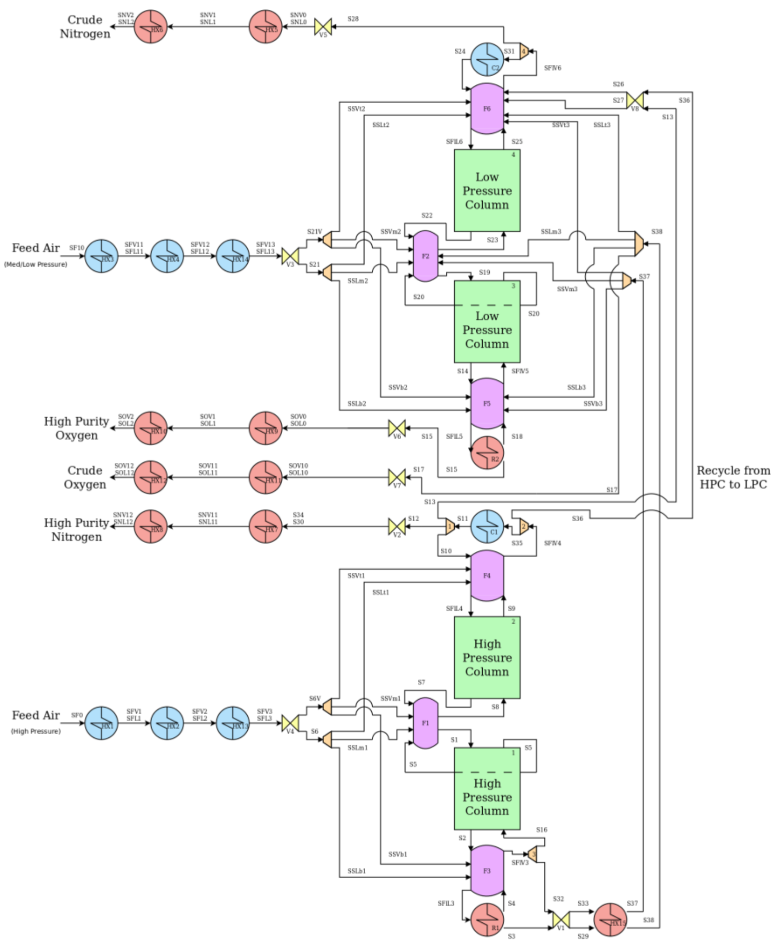
Sparse rank calculation is an approximation

# Degeneracy Hunter Demo

```
Consider degeneracy with EqStrMoleFrac(S4,B)
Optimal: This equation is part of a degenerate set with 9 equations (total)
-1.000000          EqUnitComponentMoleBalance(1,A)
-1.000000          EqUnitComponentMoleBalance(1,B)
1.000000           EqUnitMoleBalance(1)
-1.000000          EqStrMoleFrac(S2,A)
-1.000000          EqStrMoleFrac(S2,B)
1.000000           EqStrMoleFrac(S3,A)
1.000000           EqStrMoleFrac(S3,B)
1.000000           EqStrMoleFrac(S4,A)
1.000000           EqStrMoleFrac(S4,B)

Consider degeneracy with EqStrMoleFrac(S9,B)
Optimal: This equation is part of a degenerate set with 9 equations (total)
-1.000000          EqUnitComponentMoleBalance(3,A)
-1.000000          EqUnitComponentMoleBalance(3,B)
1.000000           EqUnitMoleBalance(3)
-1.000000          EqStrMoleFrac(S4,A)
-1.000000          EqStrMoleFrac(S4,B)
1.000000           EqStrMoleFrac(S8,A)
1.000000           EqStrMoleFrac(S8,B)
1.000000           EqStrMoleFrac(S9,A)
1.000000           EqStrMoleFrac(S9,B)
```

# ASU Case Study




Double-column ASU with multistream heat exchanger

Completely equation based

- Embedded thermo.
- Heat integration
- **Non-convex!**

15,000+ variables and constraints

# Initialization Procedure

**Section 0**

**Section 1** — Ideal Thermo & Shortcut Cascade

↓

**Section 2**
**Section 3** — CEOS Thermo & Shortcut Cascade

↓

**Section 4** — CEOS Thermo & MESH Cascade

↓

**Section 5**
**Section 6** — Decompose Heat Exchange Units & Reoptimize

Repeat with different combinations of initial values and bounds

Sort local solutions by final obj. function value

# ASU Case Study

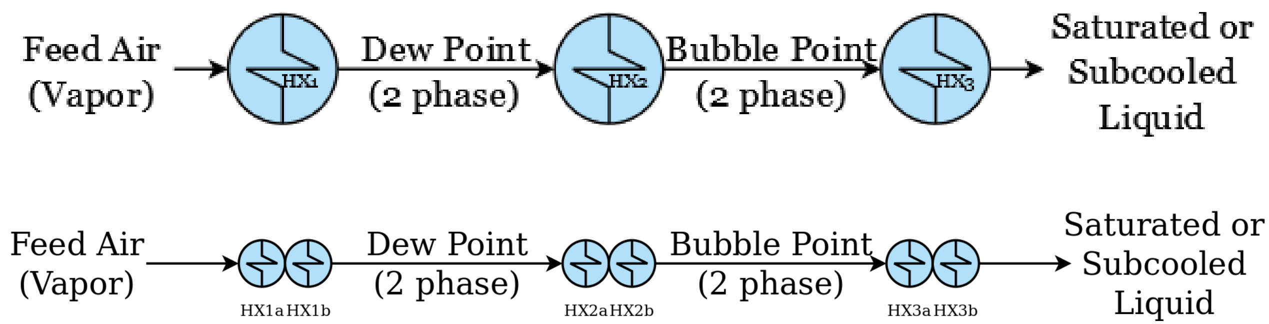
Multistart procedure:

- 288 initialization points considered for each case

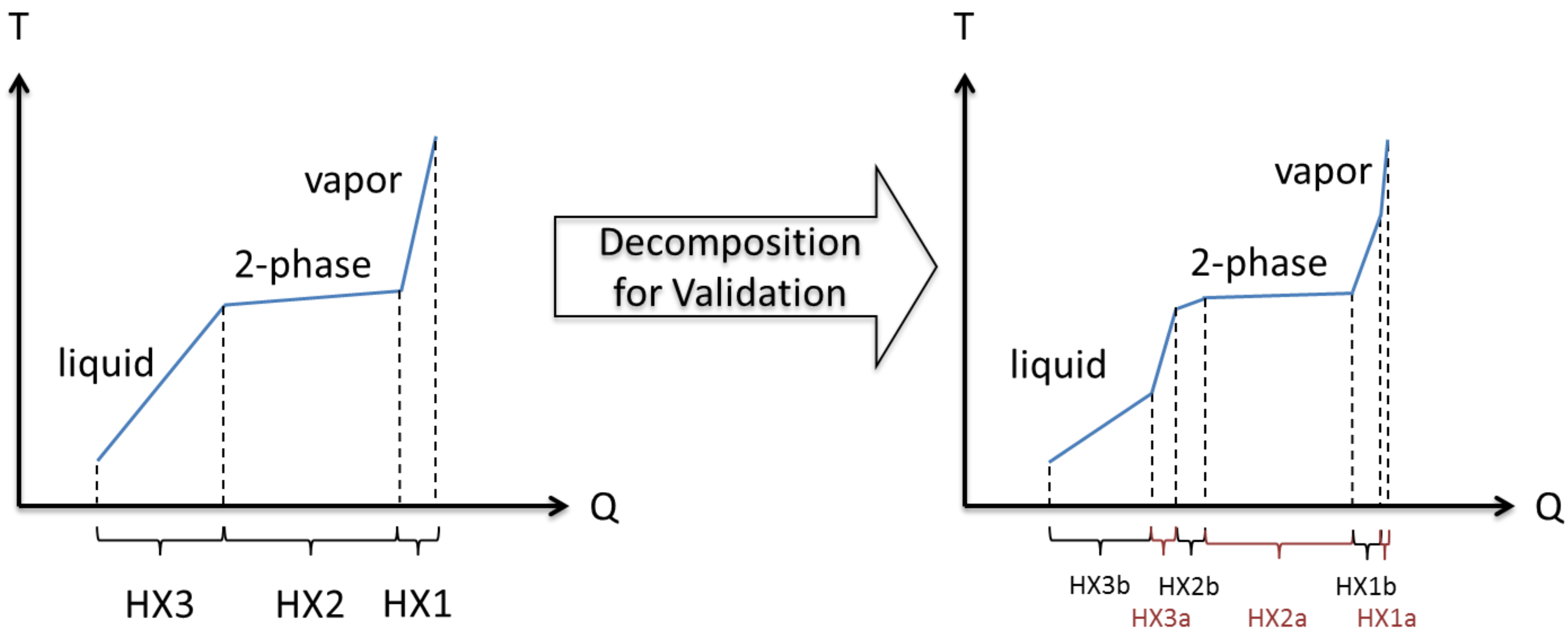
| Description | Number of "High Quality" Optimal Solutions | Best Obj. | Avg Time (s) | Time Red. |
|---|---|---|---|---|
| Based Case | 200 | 0.16821 | 909.6 | -- |

Disclaimer: These results are ***preliminary***.

# Heat Exchanger Decomposition




**Equal ΔT for each subunit of a large heat exchange unit**

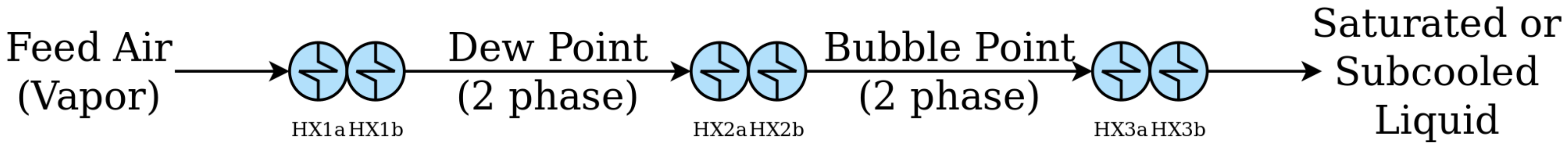


# Heat Exchanger Decomposition

$$u \in \{\text{Heat Exchange Units}\}$$
$$s \in \{\text{Heat Exchange Subunits}\}(u)$$
$$n := \text{number of subunits per large unit}$$

Degenerate $$\boxed{T_u^{in} - T_u^{out} = \Delta T, \quad \forall u}$$ Degenerate

$$T_s^{in} - T_s^{out} = \Delta T/n, \quad \forall s(u), u$$



# ASU Case Study

Multistart methods used to good local solutions

- 288 initialization points considered for each case

| Description | Number of “High Quality” Optimal Solutions | Best Obj. | Avg Time (s) | Time Red. |
|---|---|---|---|---|
| Based Case | 200 | 0.16821 | 909.6 | -- |
| + Removed `EqCalcDeltaT` (modeler error: extra equality constraint) | 199 | 0.16832 | 788.1 | 13.4% |

# Add'n Subunit Equations

$$u \in \{\text{Heat Exchange Units}\}$$
$$s \in \{\text{Heat Exchange Subunits}\}(u)$$
$$n := \text{number of subunits per large unit}$$

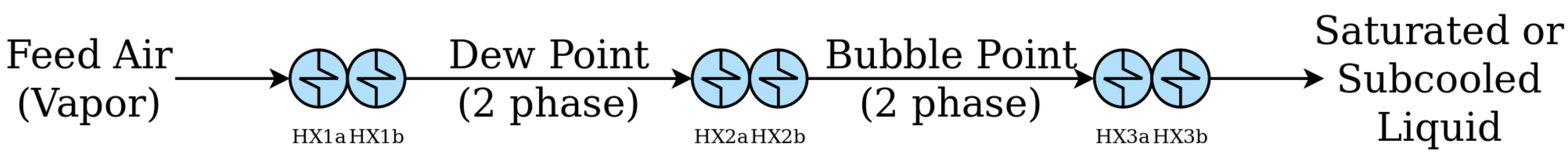
**Cooling Units**

$$Q_s^{in} = 0$$
$$Q_s^{out} > 0$$
$$\boxed{T_s^{in} - T_s^{out} \geq 0}$$

**Occasionally Degenerate**

**Heating Units**

$$Q_s^{in} \geq 0$$
$$Q_s^{out} = 0$$
$$\boxed{T_s^{out} - T_s^{in} \geq 0}$$



# ASU Case Study

Multistart methods used to good local solutions

- 288 initialization points considered for each case

| Description | Number of “High Quality” Optimal Solutions | Best Obj. | Avg Time (s) | Time Red. |
|---|---|---|---|---|
| Based Case | 200 | 0.16821 | 909.6 | -- |
| + Removed `EqCalcDeltaT` (modeler error: extra equality constraint) | 199 | 0.16832 | 788.1 | 13.4% |
| + Removed `EqCoolSHtEx` & `EqHeatSHtEx` (redundant inequality cons.) | 199 | 0.16842 | 699.1 | 23.1% |

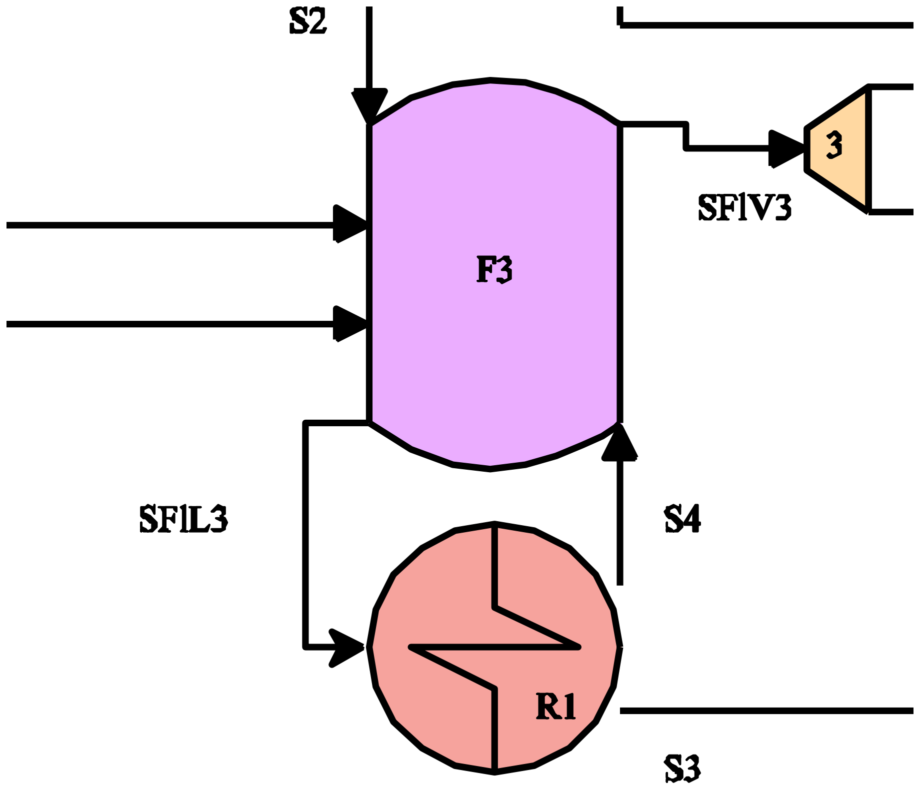
# Degeneracy Hunter Output

```
Consider degeneracy with EqPresSHtEx(SHX354,V353,S3)

Elapsed time is 4.007682 seconds.

Optimal: This equation is part of a degenerate set
with 11 equations (total)
1.000000     EqPRelThrmE(F3,SFlV3,SFlL3)
-1.000000    EqPRelThrmE(SHX351,V351,L351)
-1.000000    EqPRelThrmE(SHX352,V352,L352)
-1.000000    EqPRelThrmE(SHX353,V353,L353)
-1.000000    EqPRelThrmE(SHX354,S4,S3)
1.000000     EqPRel1Flsh(F3)
1.000000     EqPFlashLgc(F3,S4)
1.000000     EqPresSHtEx(SHX351,SFlL3,L351)
1.000000     EqPresSHtEx(SHX352,V351,L352)
1.000000     EqPresSHtEx(SHX353,V352,L353)
1.000000     EqPresSHtEx(SHX354,V353,S3)
```

# Pressure Recycle




$$\cancel{P_{SFlL3} \leq P_{S4}}$$

$$P_{SFlL3} = P_{S3}$$

$$P_{S3} = P_{S4}$$

# ASU Case Study

Multistart methods used to good local solutions

- 288 initialization points considered for each case

| Description | Number of "High Quality" Optimal Solutions | Best Obj. | Avg Time (s) | Time Red. |
|---|---|---|---|---|
| Based Case | 200 | 0.16821 | 909.6 | -- |
| + Removed `EqCalcDeltaT` (modeler error: extra equality constraint) | 199 | 0.16832 | 788.1 | 13.4% |
| + Removed `EqCoolSHtEx` & `EqHeatSHtEx` (redundant inequality cons.) | 199 | 0.16842 | 699.1 | 23.1% |
| + Removed pressure recycles for condensers & reboilers | 208 | 0.16824 | 640.2 | **29.6%** |

# Conclusions

- Degenerate equations are prevalent in process flowsheet models and degrade optimizer performance
- Irreducible sets of degenerate equations help expert modelers with reformulation
- Finding irreducible sets of degenerate equations can be posed as a MILP
- Removing degenerate equations resulting in a **30% reduction** in CPU time in ASU optimization example
  - Disclaimer: *Preliminary* results