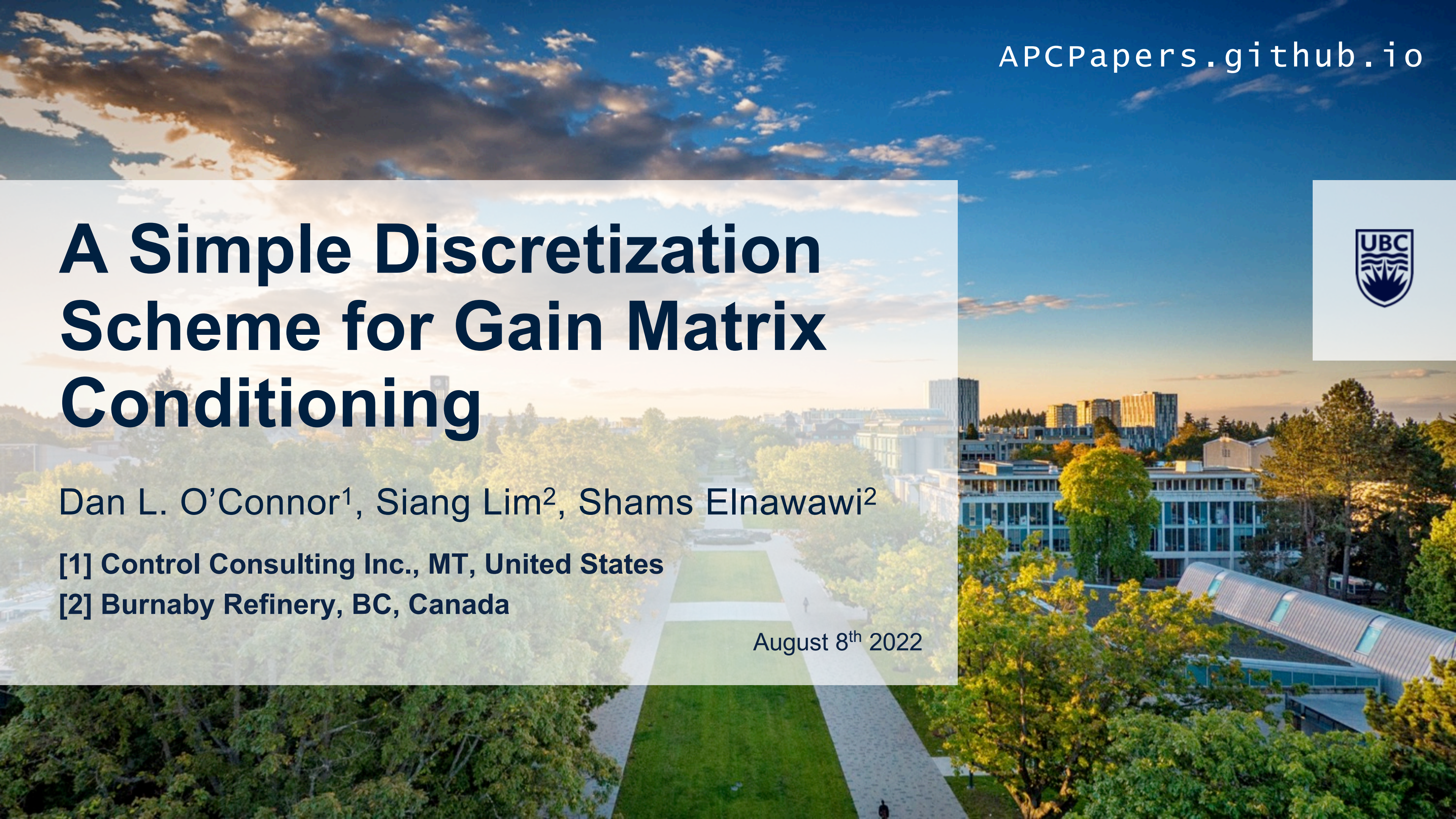
APCPapers.github.io



A Simple Discretization Scheme for Gain Matrix Conditioning

Dan L. O'Connor[1], Siang Lim[2], Shams Elnawawi[2]

[1] Control Consulting Inc., MT, United States

[2] Burnaby Refinery, BC, Canada

August 8th 2022

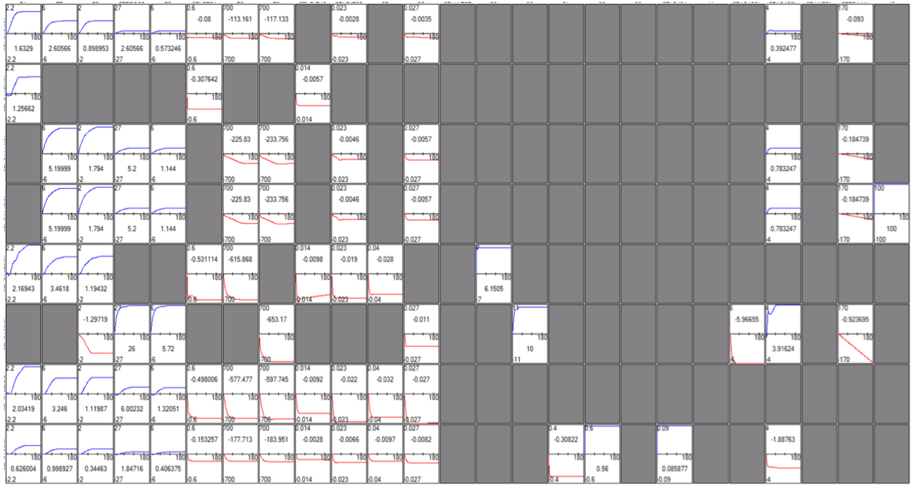
Why we are here today

Motivation:

Classical matrix conditioning techniques are very time-consuming for large gain matrices in practice due to the need for iterative adjustments.

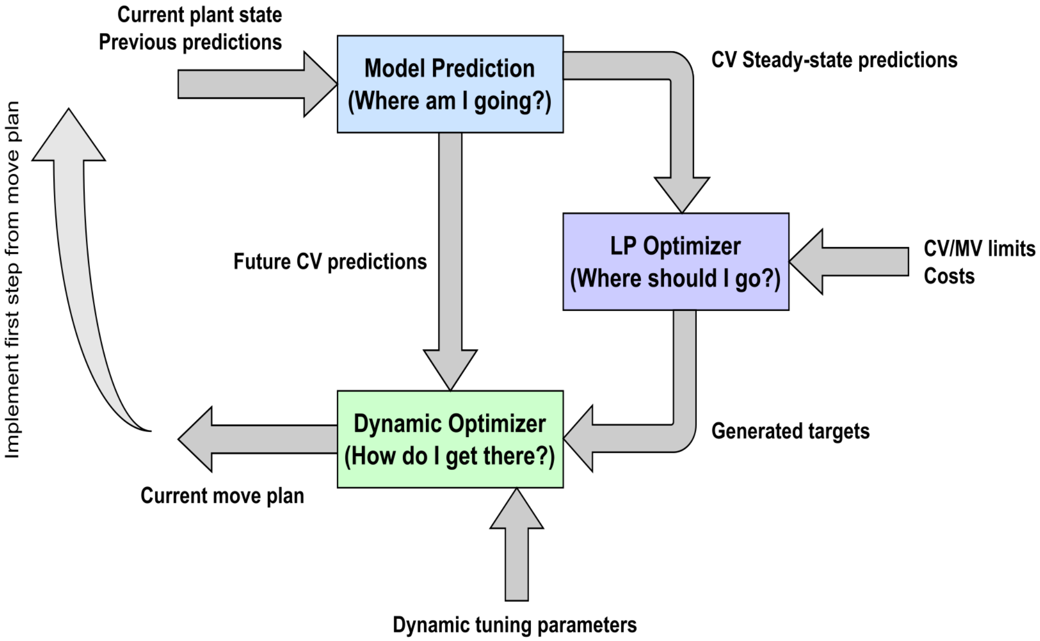
Our Solution:

A novel matrix conditioning technique that avoids iterations by discretizing the gain matrix.



Example of a moderately-sized gain matrix from the Burnaby Refinery.

Industrial Model Predictive Control



- Industrial MPCs have 2 optimizers:
  - Steady-state optimizer for economics
  - Dynamic optimizer for move planning
- Our focus is on the steady-state optimizer (Linear Program, or LP)
- LP uses steady-state model data from system identification

Ranade, S. M., & Torres, E. (2009). From dynamic mysterious control to dynamic manageable control. *Hydrocarbon Processing*, 88(3), 77-81.

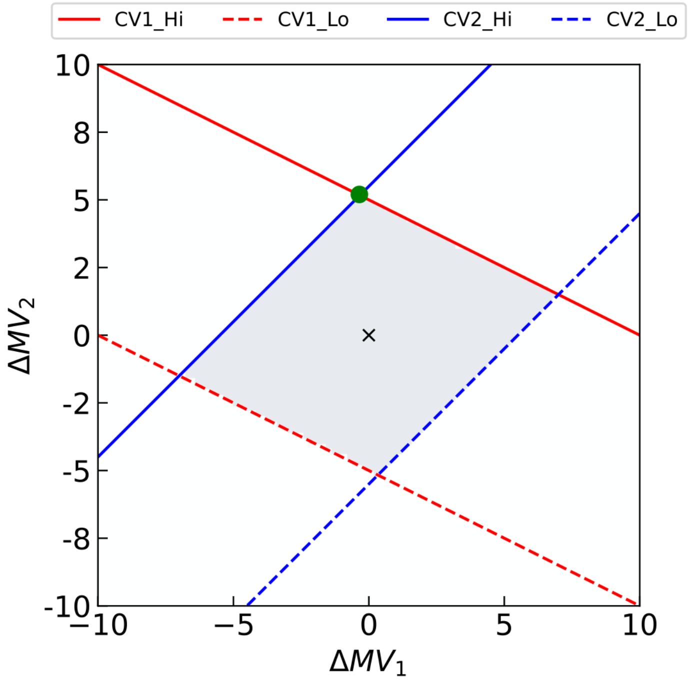
Classifying 2x2 gain interactions

$$G = \begin{array}{cc} \begin{array}{cc} \text{MV1} & \text{MV2} \end{array} & \\ \begin{bmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{bmatrix} & \begin{array}{c} \text{CV1} \\ \text{CV2} \end{array} \end{array}$$

$$\text{RGA} = \frac{1}{1-\left(\frac{G_{21} * G_{12}}{G_{11} * G_{22}}\right)}$$

- The smallest possible gain interaction is 2×2
- Combinations of gain values will give different degrees of variable interaction
- Interaction in 2×2 submatrices can be quantified using RGA or SVD
- Problematic interactions arise when matrices are ill-conditioned, such that $\frac{G_{11}}{G_{21}} \approx \frac{G_{12}}{G_{22}}$

Visualizing the LP solution

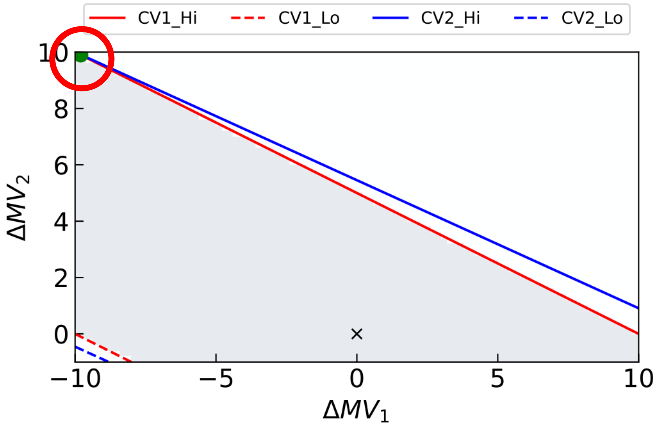


LP feasible region (grey) showing all possible solutions, with current solution in green.

- The LP calculates steady-state targets by solving a cost minimization problem subject to MV and CV limits
- We can visualize a 2x2 LP solution by plotting CV limits as functions of MV moves
- The feasible region shows the optimal solution at an intersection of constraints

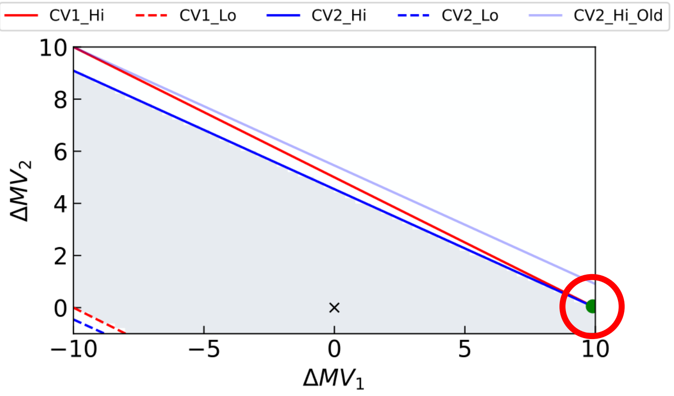
LP solution for change – ill-conditioned

ΔMV_1 = -10.0
ΔMV_2 = +10.0



CV_2 high limit dropped from 1.2 to 1.0

ΔMV_1 = +10.0
ΔMV_2 = +0.05

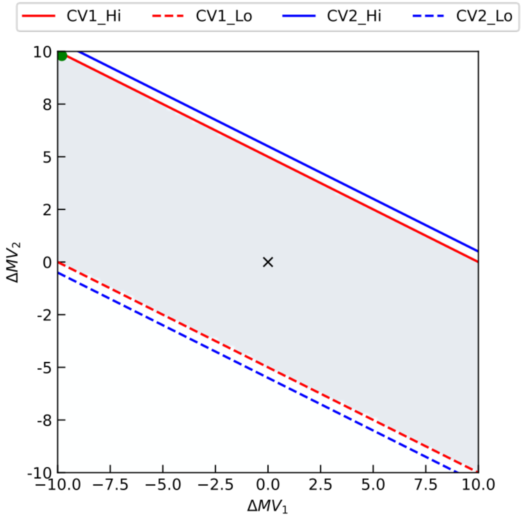


Small perturbations to an ill-conditioned model can result in large solution changes and general LP instability.

How do we repair ill-conditioned submatrices?

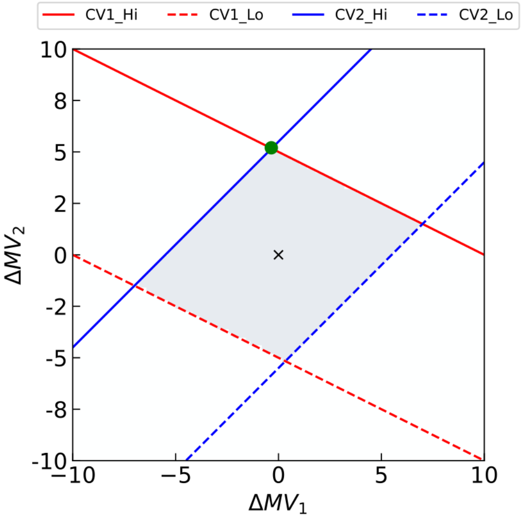
Option 1: force collinearity

- Reduces degrees of freedom
- Only one CV constraint can be satisfied



Option 2: adjust gains to reduce RGA

- Creates a well-conditioned submatrix
- Both CVs can be adequately controlled



Traditional conditioning approach - example

$$G = \begin{array}{c} \\ \\ \\ \\ \end{array}\begin{array}{cccc} \text{MV1} & \text{MV2} & \text{MV3} & \\ \left[\begin{array}{c} 1 \\ 3 \\ 0 \end{array}\right. & \begin{array}{c} 2 \\ 6.1 \\ 7 \end{array} & \left.\begin{array}{c} 0 \\ -1 \\ -1 \end{array}\right] & \begin{array}{c} \text{CV1} \\ \text{CV2} \\ \text{CV3} \end{array} \end{array}$$

Traditional conditioning approach - example

$$G = \begin{array}{cc} \begin{array}{ccc} \text{MV1} & \text{MV2} & \text{MV3} \end{array} & \\ \begin{bmatrix} 1 & 2 & 0 \\ 3 & 6.1 & -1 \\ 0 & 7 & -1 \end{bmatrix} & \begin{array}{c} \text{CV1} \\ \text{CV2} \\ \text{CV3} \end{array} \end{array}$$

Traditional conditioning approach - example

$$G = \begin{array}{cccc} \text{MV1} & \text{MV2} & \text{MV3} & \\ \left[\begin{matrix} 1 \\ 3 \\ 0 \end{matrix}\right. & \begin{matrix} 2 \\ 6.1 \\ 7 \end{matrix} & \left.\begin{matrix} 0 \\ -1 \\ -1 \end{matrix}\right] & \begin{matrix} \text{CV1} \\ \text{CV2} \\ \text{CV3} \end{matrix} \end{array}$$

$$\begin{array}{ccc} \text{MV1} & \text{MV2} & \\ \left[\begin{matrix} 1 \\ 3 \end{matrix}\right. & \left.\begin{matrix} 2 \\ 6.1 \end{matrix}\right] & \begin{matrix} \text{CV1} \\ \text{CV2} \end{matrix} \end{array}$$

RGA = 61

Traditional conditioning approach - example

$$G = \begin{array}{cccc} \text{MV1} & \text{MV2} & \text{MV3} & \\ \left[\begin{matrix} 1 \\ 3 \\ 0 \end{matrix}\right. & \begin{matrix} 2 \\ 6.1 \\ 7 \end{matrix} & \left.\begin{matrix} 0 \\ -1 \\ -1 \end{matrix}\right] & \begin{matrix} \text{CV1} \\ \text{CV2} \\ \text{CV3} \end{matrix} \end{array}$$

$$\begin{array}{ccc} \text{MV1} & \text{MV2} & \\ \left[\begin{matrix} 1 \\ 3 \end{matrix}\right. & \left.\begin{matrix} 2 \\ 6.1 \end{matrix}\right] & \begin{matrix} \text{CV1} \\ \text{CV2} \end{matrix} \end{array}$$

RGA = 61

$$\begin{array}{ccc} \text{MV2} & \text{MV3} & \\ \left[\begin{matrix} 6.1 \\ 7 \end{matrix}\right. & \left.\begin{matrix} -1 \\ -1 \end{matrix}\right] & \begin{matrix} \text{CV2} \\ \text{CV3} \end{matrix} \end{array}$$

RGA = 6.78

Traditional conditioning approach - example

$$G = \begin{array}{cc} \begin{matrix}\text{MV1} & \text{MV2} & \text{MV3}\end{matrix} & \\ \begin{bmatrix}1 & 2 & 0\\ 3 & 6.1 & -1\\ 0 & 7 & -1\end{bmatrix} & \begin{matrix}\text{CV1}\\ \text{CV2}\\ \text{CV3}\end{matrix}\end{array}$$

$$\begin{array}{cc} \begin{matrix}\text{MV1} & \text{MV2}\end{matrix} & \\ \begin{bmatrix}1 & 2\\ 3 & 6.1\end{bmatrix} & \begin{matrix}\text{CV1}\\ \text{CV2}\end{matrix}\end{array}$$

RGA = 61

Adjust to 6.6,
reduce RGA

Traditional conditioning approach - example

$$G = \begin{array}{cccc} \text{MV1} & \text{MV2} & \text{MV3} & \\ \left[\begin{matrix} 1 \\ 3 \\ 0 \end{matrix}\right. & \begin{matrix} 2 \\ 6.6 \\ 7 \end{matrix} & \left.\begin{matrix} 0 \\ -1 \\ -1 \end{matrix}\right] & \begin{matrix} \text{CV1} \\ \text{CV2} \\ \text{CV3} \end{matrix} \end{array}$$

$$\begin{array}{ccc} \text{MV1} & \text{MV2} & \\ \left[\begin{matrix} 1 \\ 3 \end{matrix}\right. & \left.\begin{matrix} 2 \\ 6.6 \end{matrix}\right] & \begin{matrix} \text{CV1} \\ \text{CV2} \end{matrix} \end{array}$$

RGA = 11

Traditional conditioning approach - example

$$G = \begin{array}{cccc} \text{MV1} & \text{MV2} & \text{MV3} & \\ \left[\begin{matrix} 1 \\ 3 \\ 0 \end{matrix}\right. & \begin{matrix} 2 \\ 6.6 \\ 7 \end{matrix} & \left.\begin{matrix} 0 \\ -1 \\ -1 \end{matrix}\right] & \begin{matrix} \text{CV1} \\ \text{CV2} \\ \text{CV3} \end{matrix} \end{array}$$

$$\begin{array}{ccc} \text{MV1} & \text{MV2} & \\ \left[\begin{matrix} 1 \\ 3 \end{matrix}\right. & \left.\begin{matrix} 2 \\ 6.6 \end{matrix}\right] & \begin{matrix} \text{CV1} \\ \text{CV2} \end{matrix} \end{array}$$

RGA = 11

$$\begin{array}{ccc} \text{MV2} & \text{MV3} & \\ \left[\begin{matrix} 6.6 \\ 7 \end{matrix}\right. & \left.\begin{matrix} -1 \\ -1 \end{matrix}\right] & \begin{matrix} \text{CV2} \\ \text{CV3} \end{matrix} \end{array}$$

RGA = 16.5

Fixing that submatrix breaks this one. Previously 6.78

Traditional conditioning approach - example

$$G = \begin{array}{cccc} \text{MV1} & \text{MV2} & \text{MV3} & \\ \left[\begin{matrix} 1 & 2 & 0 \\ 3 & 6.6 & -1 \\ 0 & 7 & -1 \end{matrix}\right] & & & \begin{matrix} \text{CV1} \\ \text{CV2} \\ \text{CV3} \end{matrix} \end{array}$$

$$\begin{array}{cc} \text{MV1} \quad \text{MV2} & \\ \begin{bmatrix} 1 & 2 \\ 3 & 6.6 \end{bmatrix} & \begin{matrix} \text{CV1} \\ \text{CV2} \end{matrix} \end{array}$$

RGA = 11

$$\begin{array}{cc} \text{MV2} \quad \text{MV3} & \\ \begin{bmatrix} 6.6 & -1 \\ 7 & -1 \end{bmatrix} & \begin{matrix} \text{CV2} \\ \text{CV3} \end{matrix} \end{array}$$

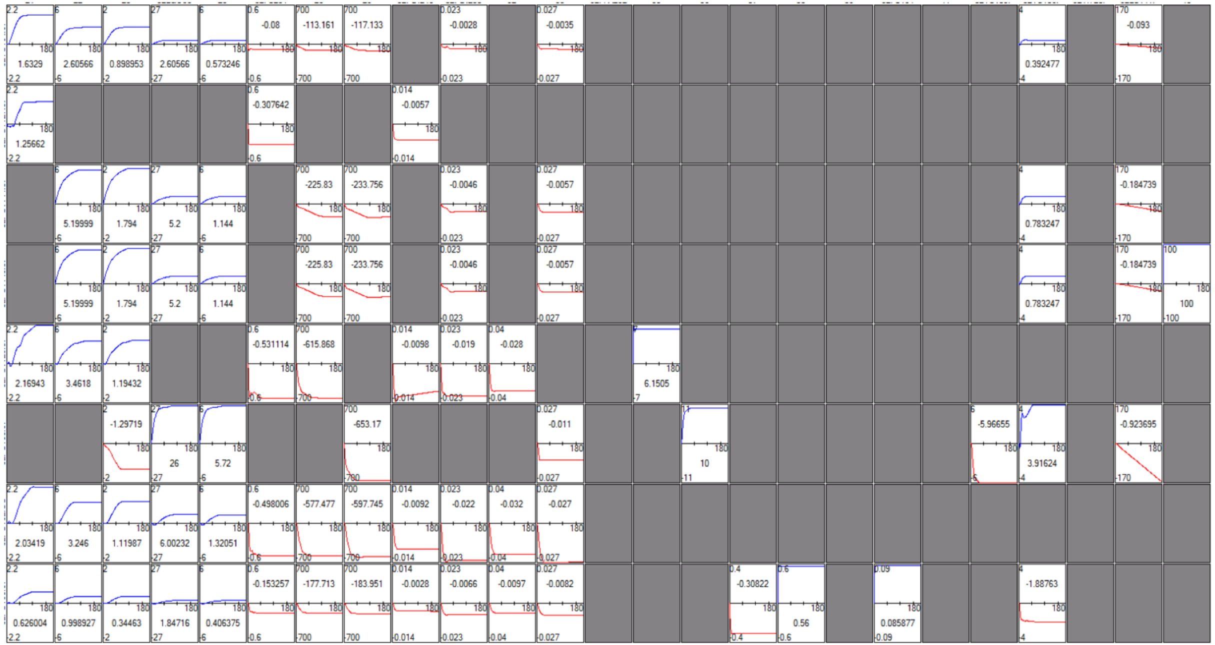
RGA = 16.5

Fixing that submatrix breaks this one. Previously 6.78

Problem: Fixing one submatrix breaks another.

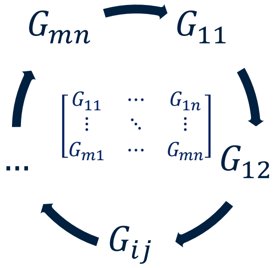
Traditional Solution: Keep iterating through all submatrices in a trial-and-error manner until all submatrices are repaired.

Why is the traditional conditioning approach difficult?



Moderately-sized gain matrix from the Burnaby Refinery. We would need to iteratively check a very large number of 2×2 submatrices.

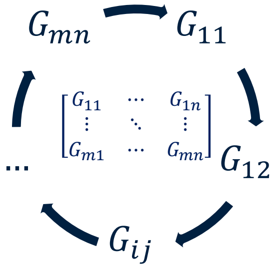
Key Idea: Binning the gains



Motivation:

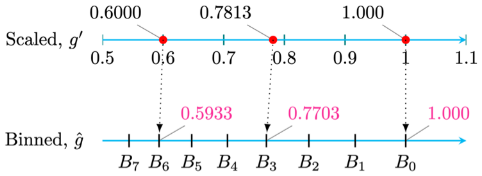
Making gain adjustments to iteratively ‘repair’ each submatrix is very time-consuming and tedious for large models.

Key Idea: Binning the gains



Motivation:

Making gain adjustments to iteratively ‘repair’ each submatrix is very time-consuming and tedious for large models.



Our Solution:

We present a novel binning technique for gain matrix conditioning that is achievable in just a single-pass without iterations.

Gain binning procedure

- **Step 1:** ‘Normalize’ gain matrix to [-1,1]

Gain binning procedure

- **Step 1:** ‘Normalize’ gain matrix to [-1,1]
- **Step 2:** Define an $\mathrm{RGA}_{\mathrm{max}}$ threshold; generate grid of bin values

Gain binning procedure

- **Step 1:** ‘Normalize’ gain matrix to [-1,1]
- **Step 2:** Define an $\mathrm{RGA}_{\max}$ threshold; generate grid of bin values
- **Step 3:** Adjust each ‘normalized’ gain to the nearest bin

Gain binning procedure

- **Step 1:** ‘Normalize’ gain matrix to [-1,1]
- **Step 2:** Define an $\mathrm{RGA}_{\mathrm{max}}$ threshold; generate grid of bin values
- **Step 3:** Adjust each ‘normalized’ gain to the nearest bin
- **Step 4:** That’s it.

  Binning is done in one-pass. No iterations.

Generate grid of binned gains in [0,1]

How?

Use binning equation, from rearranging the RGA definition *(detailed derivation in paper)*

$$B_i = (1 - \mathrm{RGA}_{\max}^{-1})^i \cdot B_0$$

Generate grid of binned gains in [0,1]

How?

Use binning equation, from rearranging the RGA definition *(detailed derivation in paper)*

$$B_i = (1 - \mathrm{RGA}_{\max}^{-1})^i \cdot B_0$$

Example:

$\mathrm{RGA}_{\max} = 12$ is a reasonable choice in practice. $B_0 = 1$ *(since scaled gains are in [-1,1])*

$$B_1 = (1 - 12^{-1})^1 = 0.9167$$

Generate grid of binned gains in [0,1]

How?

Use binning equation, from rearranging the RGA definition *(detailed derivation in paper)*

$$B_i = (1 - \mathrm{RGA}_{\max}^{-1})^i \cdot B_0$$

Example:

$\mathrm{RGA}_{\max} = 12$ is a reasonable choice in practice. $B_0 = 1$ *(since scaled gains are in [-1,1])*

$$B_1 = (1 - 12^{-1})^1 = 0.9167$$

$$B_2 = (1 - 12^{-1})^2 = 0.8403$$

Generate grid of binned gains in [0,1]

How?

Use binning equation, from rearranging the RGA definition *(detailed derivation in paper)*

$$B_i = (1 - \mathrm{RGA}_{\max}^{-1})^i \cdot B_0$$

Example:

$\mathrm{RGA}_{\max} = 12$ is a reasonable choice in practice. $B_0 = 1$ *(since scaled gains are in [-1,1])*

$$B_1 = (1 - 12^{-1})^1 = 0.9167$$

$$B_2 = (1 - 12^{-1})^2 = 0.8403$$

$$B_i = \ \ldots$$

Generate grid of binned gains in [0,1]

How?

Use binning equation, from rearranging the RGA definition *(detailed derivation in paper)*

$$B_i = (1 - \mathrm{RGA}_{\max}^{-1})^i \cdot B_0$$

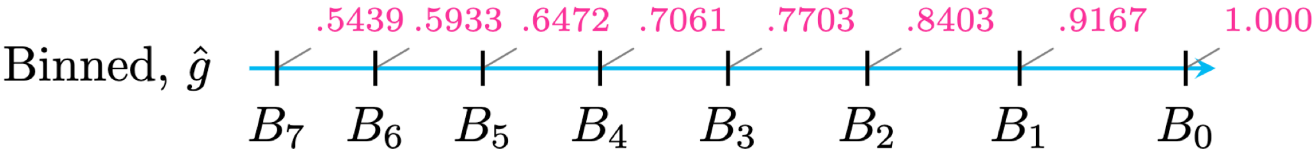
Example:

$\mathrm{RGA}_{\max} = 12$ is a reasonable choice in practice. $B_0 = 1$ *(since scaled gains are in [-1,1])*

$$B_1 = (1 - 12^{-1})^1 = 0.9167$$

$$B_2 = (1 - 12^{-1})^2 = 0.8403$$

$$B_i = \ \dots$$



Adjust each scaled gain to the nearest bin

Adjust absolute value of scaled gain to closest bin value while preserving signs.

Adjust each scaled gain to the nearest bin

Adjust absolute value of scaled gain to closest bin value while preserving signs.

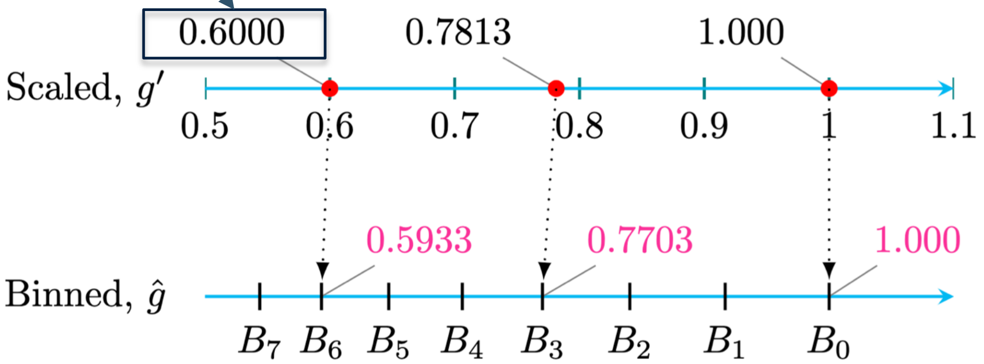
For example…

$$G' = \begin{bmatrix} -1.000 & 0.6000 \\ 1.000 & -0.7813 \end{bmatrix}$$

Adjust each scaled gain to the nearest bin

Adjust absolute value of scaled gain to closest bin value while preserving signs.

$$G' = \begin{bmatrix} -1.000 & \boxed{0.6000} \\ 1.000 & -0.7813 \end{bmatrix}$$



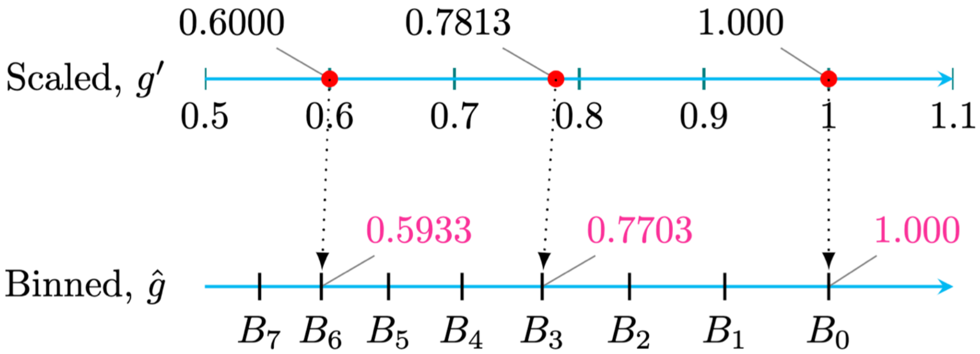
Adjust each scaled gain to the nearest bin

Adjust absolute value of scaled gain to closest bin value while preserving signs.

$$G' = \begin{bmatrix} -1.000 & \boxed{0.6000} \\ 1.000 & -0.7813 \end{bmatrix}$$

$$\Big\downarrow \ \textit{Binning}$$

$$\hat{G} = \begin{bmatrix} -1.000 & \boxed{0.5933} \\ 1.000 & -0.7703 \end{bmatrix}$$



The binned matrix contains discrete values generated in the binning grid, rather than continuous values on the number line.

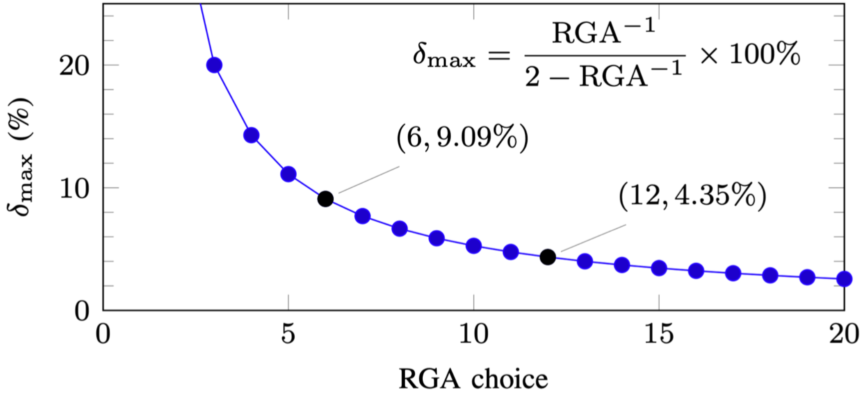
Binning guarantees 2 desirable properties

Property 1: The maximum possible gain adjustment is bounded by $\mathrm{RGA}_{\max}$

Binning guarantees 2 desirable properties

Property 1: The maximum possible gain adjustment is bounded by $\mathrm{RGA}_{\max}$

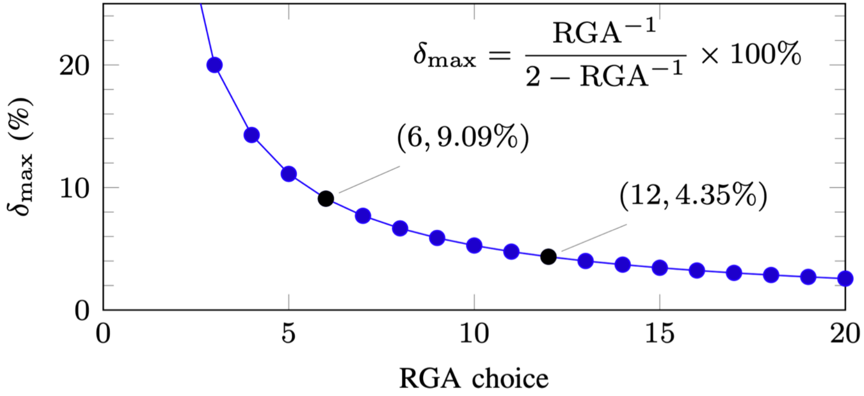
- For $\mathrm{RGA}_{\max} = 12$, the maximum change is only 4.35%.
- The lower the RGA requirements, the higher the max possible change.



Binning guarantees 2 desirable properties

Property 1: The maximum possible gain adjustment is bounded by $\mathrm{RGA}_{\mathrm{max}}$

- For $\mathrm{RGA}_{\mathrm{max}} = 12$, the maximum change is only 4.35%.
- The lower the RGA requirements, the higher the max possible change.



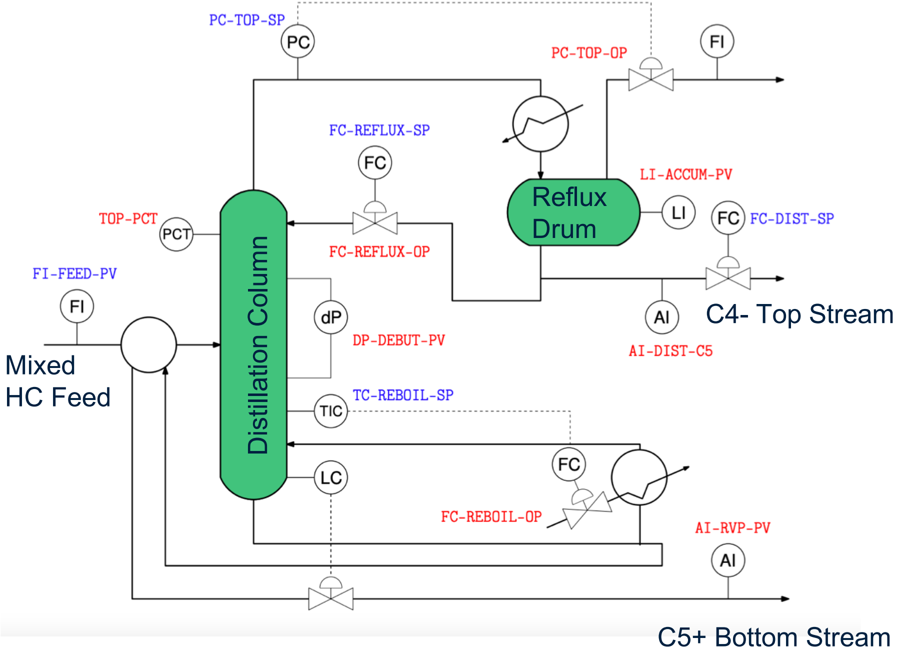
Property 2: All non-collinear 2x2 submatrices are guaranteed to have $\mathrm{RGA} \leq \mathrm{RGA}_{\mathrm{max}}$

- One-pass formula. There is no iteration.

Case study: Debutanizer application

Debutanizer:

Separate C4- from C5+ hydrocarbons



Case study: Debutanizer application

Debutanizer:

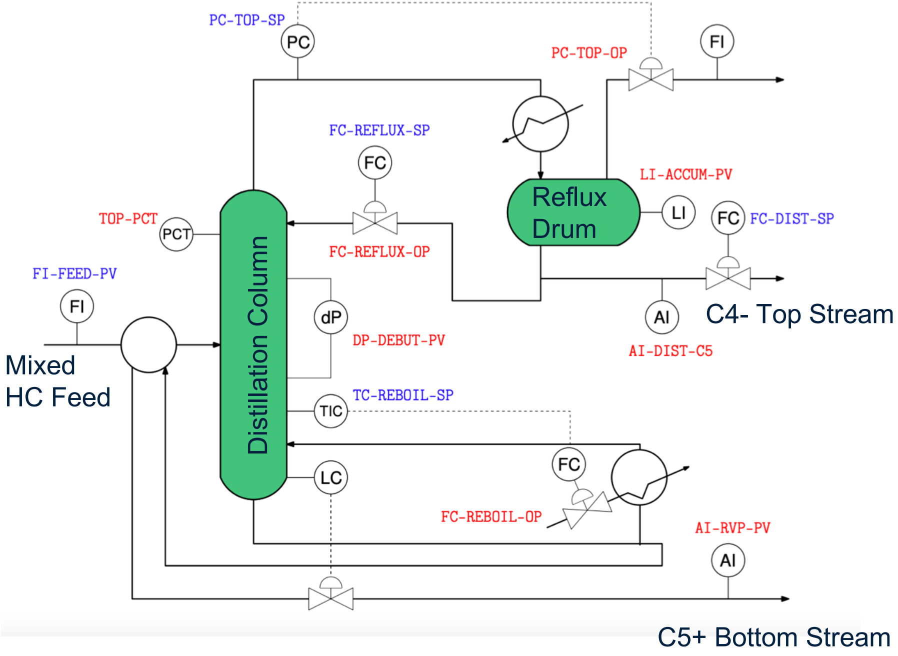
Separate C4- from C5+ hydrocarbons

APC objective:

Use 5 blue MVs to control 8 red CVs.

Product specs

- C5% impurity in top stream
- RVP of bottom stream

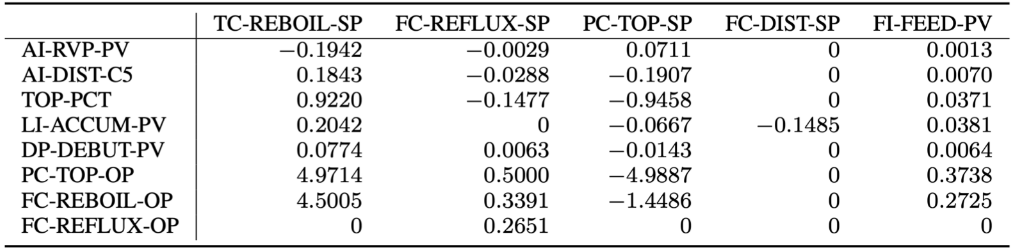


Apply typical move scaling

Raw gain matrix source:

Simulated debutanizer data from

CCI training classes



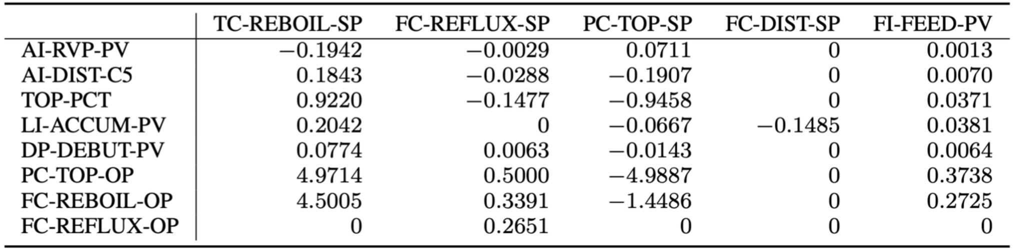
	TC-REBOIL-SP	FC-REFLUX-SP	PC-TOP-SP	FC-DIST-SP	FI-FEED-PV
AI-RVP-PV	−0.1942	−0.0029	0.0711	0	0.0013
AI-DIST-C5	0.1843	−0.0288	−0.1907	0	0.0070
TOP-PCT	0.9220	−0.1477	−0.9458	0	0.0371
LI-ACCUM-PV	0.2042	0	−0.0667	−0.1485	0.0381
DP-DEBUT-PV	0.0774	0.0063	−0.0143	0	0.0064
PC-TOP-OP	4.9714	0.5000	−4.9887	0	0.3738
FC-REBOIL-OP	4.5005	0.3391	−1.4486	0	0.2725
FC-REFLUX-OP	0	0.2651	0	0	0

Raw gain matrix

Apply typical move scaling

Raw gain matrix source:

Simulated debutanizer data from CCI training classes



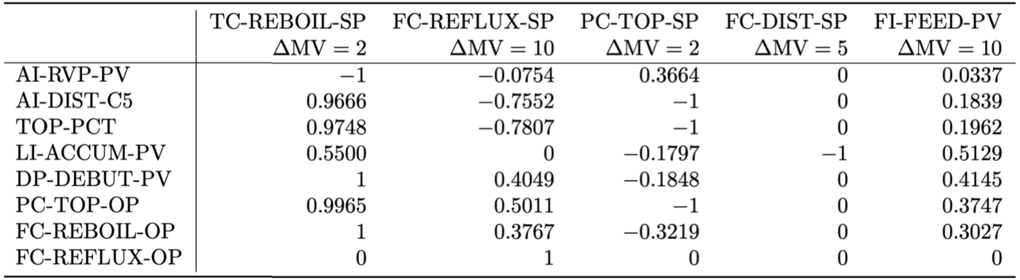
	TC-REBOIL-SP	FC-REFLUX-SP	PC-TOP-SP	FC-DIST-SP	FI-FEED-PV
AI-RVP-PV	−0.1942	−0.0029	0.0711	0	0.0013
AI-DIST-C5	0.1843	−0.0288	−0.1907	0	0.0070
TOP-PCT	0.9220	−0.1477	−0.9458	0	0.0371
LI-ACCUM-PV	0.2042	0	−0.0667	−0.1485	0.0381
DP-DEBUT-PV	0.0774	0.0063	−0.0143	0	0.0064
PC-TOP-OP	4.9714	0.5000	−4.9887	0	0.3738
FC-REBOIL-OP	4.5005	0.3391	−1.4486	0	0.2725
FC-REFLUX-OP	0	0.2651	0	0	0

Raw gain matrix

Typical move scaling:

Typical move scaling

Gains are ‘normalized’ to [-1,1] by considering both MV move sizes and CV responses.



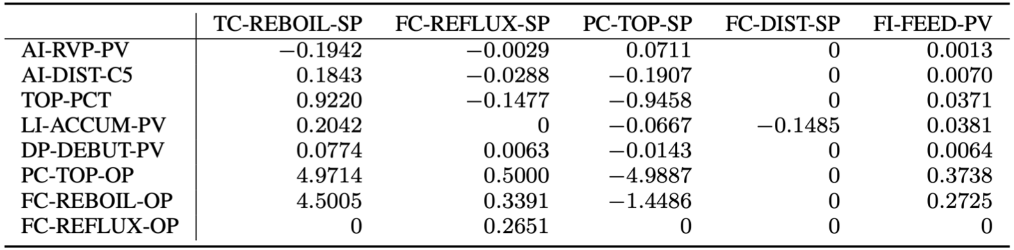
	TC-REBOIL-SP ΔMV $= 2$	FC-REFLUX-SP ΔMV $= 10$	PC-TOP-SP ΔMV $= 2$	FC-DIST-SP ΔMV $= 5$	FI-FEED-PV ΔMV $= 10$
AI-RVP-PV	−1	−0.0754	0.3664	0	0.0337
AI-DIST-C5	0.9666	−0.7552	−1	0	0.1839
TOP-PCT	0.9748	−0.7807	−1	0	0.1962
LI-ACCUM-PV	0.5500	0	−0.1797	−1	0.5129
DP-DEBUT-PV	1	0.4049	−0.1848	0	0.4145
PC-TOP-OP	0.9965	0.5011	−1	0	0.3747
FC-REBOIL-OP	1	0.3767	−0.3219	0	0.3027
FC-REFLUX-OP	0	1	0	0	0

Scaled gain matrix

Apply typical move scaling

Raw gain matrix source:

Simulated debutanizer data from CCI training classes

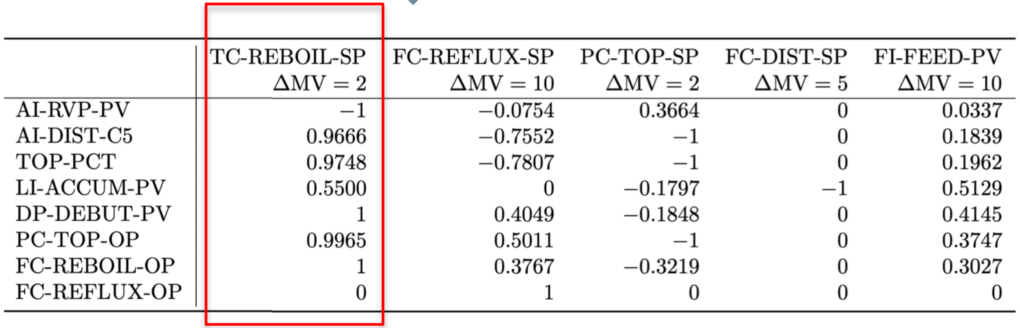


	TC-REBOIL-SP	FC-REFLUX-SP	PC-TOP-SP	FC-DIST-SP	FI-FEED-PV
AI-RVP-PV	−0.1942	−0.0029	0.0711	0	0.0013
AI-DIST-C5	0.1843	−0.0288	−0.1907	0	0.0070
TOP-PCT	0.9220	−0.1477	−0.9458	0	0.0371
LI-ACCUM-PV	0.2042	0	−0.0667	−0.1485	0.0381
DP-DEBUT-PV	0.0774	0.0063	−0.0143	0	0.0064
PC-TOP-OP	4.9714	0.5000	−4.9887	0	0.3738
FC-REBOIL-OP	4.5005	0.3391	−1.4486	0	0.2725
FC-REFLUX-OP	0	0.2651	0	0	0

Raw gain matrix

Typical move scaling:

- Multiply each MV column by the largest MV move size made during plant test



	TC-REBOIL-SP	FC-REFLUX-SP	PC-TOP-SP	FC-DIST-SP	FI-FEED-PV
	ΔMV = 2	ΔMV = 10	ΔMV = 2	ΔMV = 5	ΔMV = 10
AI-RVP-PV	−1	−0.0754	0.3664	0	0.0337
AI-DIST-C5	0.9666	−0.7552	−1	0	0.1839
TOP-PCT	0.9748	−0.7807	−1	0	0.1962
LI-ACCUM-PV	0.5500	0	−0.1797	−1	0.5129
DP-DEBUT-PV	1	0.4049	−0.1848	0	0.4145
PC-TOP-OP	0.9965	0.5011	−1	0	0.3747
FC-REBOIL-OP	1	0.3767	−0.3219	0	0.3027
FC-REFLUX-OP	0	1	0	0	0

Scaled gain matrix

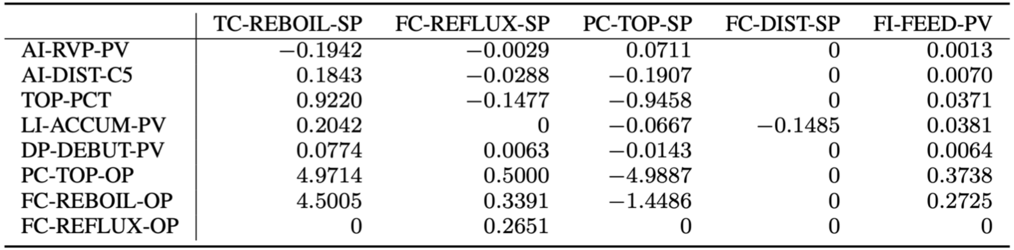
Apply typical move scaling

Raw gain matrix source:

Simulated debutanizer data from CCI training classes

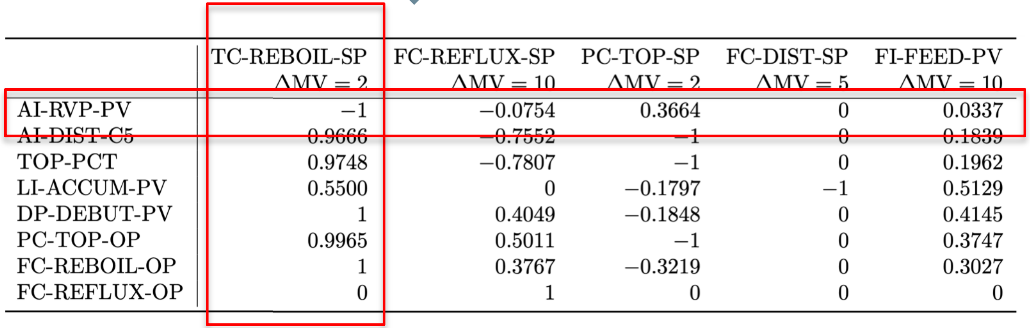
Typical move scaling:

- Multiply each MV column by the largest MV move size made during plant test
- Then, divide each CV row by the maximum abs gain in that row



	TC-REBOIL-SP	FC-REFLUX-SP	PC-TOP-SP	FC-DIST-SP	FI-FEED-PV
AI-RVP-PV	−0.1942	−0.0029	0.0711	0	0.0013
AI-DIST-C5	0.1843	−0.0288	−0.1907	0	0.0070
TOP-PCT	0.9220	−0.1477	−0.9458	0	0.0371
LI-ACCUM-PV	0.2042	0	−0.0667	−0.1485	0.0381
DP-DEBUT-PV	0.0774	0.0063	−0.0143	0	0.0064
PC-TOP-OP	4.9714	0.5000	−4.9887	0	0.3738
FC-REBOIL-OP	4.5005	0.3391	−1.4486	0	0.2725
FC-REFLUX-OP	0	0.2651	0	0	0

Raw gain matrix



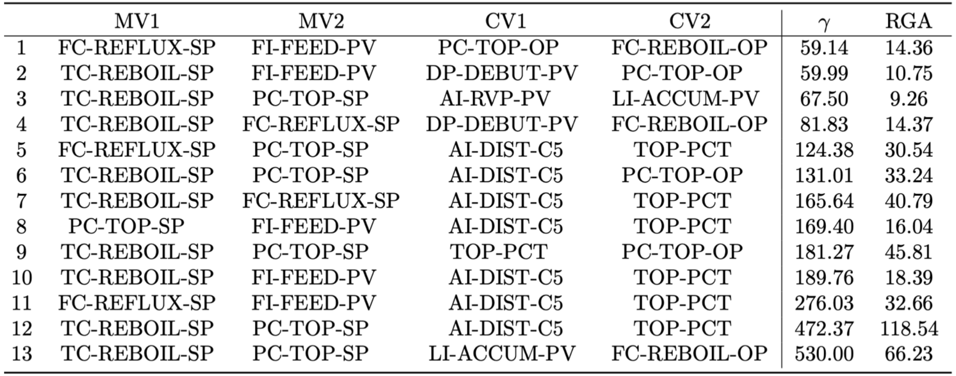
	TC-REBOIL-SP $\Delta MV = 2$	FC-REFLUX-SP $\Delta MV = 10$	PC-TOP-SP $\Delta MV = 2$	FC-DIST-SP $\Delta MV = 5$	FI-FEED-PV $\Delta MV = 10$
AI-RVP-PV	−1	−0.0754	0.3664	0	0.0337
AI-DIST-C5	0.9666	−0.7552	−1	0	0.1839
TOP-PCT	0.9748	−0.7807	−1	0	0.1962
LI-ACCUM-PV	0.5500	0	−0.1797	−1	0.5129
DP-DEBUT-PV	1	0.4049	−0.1848	0	0.4145
PC-TOP-OP	0.9965	0.5011	−1	0	0.3747
FC-REBOIL-OP	1	0.3767	−0.3219	0	0.3027
FC-REFLUX-OP	0	1	0	0	0

Scaled gain matrix

Check 2x2 matrix conditioning using RGA

Found 13 near-collinear 2x2 pairs:

Based on threshold for max RGA = 12



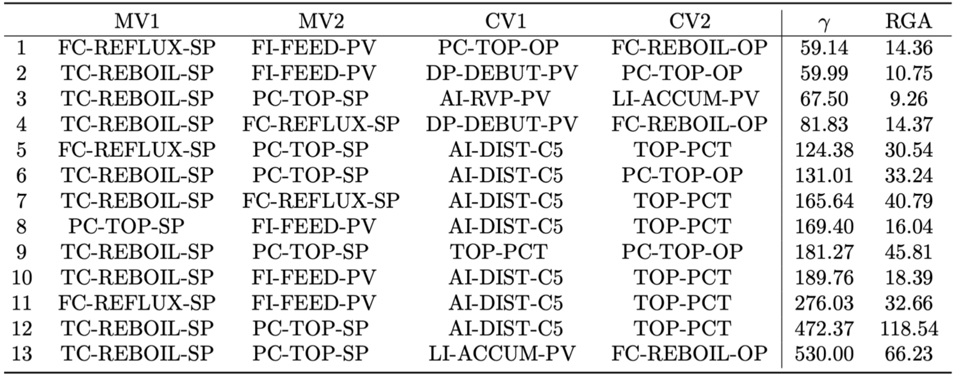
	MV1	MV2	CV1	CV2	γ	RGA
1	FC-REFLUX-SP	FI-FEED-PV	PC-TOP-OP	FC-REBOIL-OP	59.14	14.36
2	TC-REBOIL-SP	FI-FEED-PV	DP-DEBUT-PV	PC-TOP-OP	59.99	10.75
3	TC-REBOIL-SP	PC-TOP-SP	AI-RVP-PV	LI-ACCUM-PV	67.50	9.26
4	TC-REBOIL-SP	FC-REFLUX-SP	DP-DEBUT-PV	FC-REBOIL-OP	81.83	14.37
5	FC-REFLUX-SP	PC-TOP-SP	AI-DIST-C5	TOP-PCT	124.38	30.54
6	TC-REBOIL-SP	PC-TOP-SP	AI-DIST-C5	PC-TOP-OP	131.01	33.24
7	TC-REBOIL-SP	FC-REFLUX-SP	AI-DIST-C5	TOP-PCT	165.64	40.79
8	PC-TOP-SP	FI-FEED-PV	AI-DIST-C5	TOP-PCT	169.40	16.04
9	TC-REBOIL-SP	PC-TOP-SP	TOP-PCT	PC-TOP-OP	181.27	45.81
10	TC-REBOIL-SP	FI-FEED-PV	AI-DIST-C5	TOP-PCT	189.76	18.39
11	FC-REFLUX-SP	FI-FEED-PV	AI-DIST-C5	TOP-PCT	276.03	32.66
12	TC-REBOIL-SP	PC-TOP-SP	AI-DIST-C5	TOP-PCT	472.37	118.54
13	TC-REBOIL-SP	PC-TOP-SP	LI-ACCUM-PV	FC-REBOIL-OP	530.00	66.23

Near-collinear pairs

Check 2x2 matrix conditioning using RGA

Found 13 near-collinear 2x2 pairs:

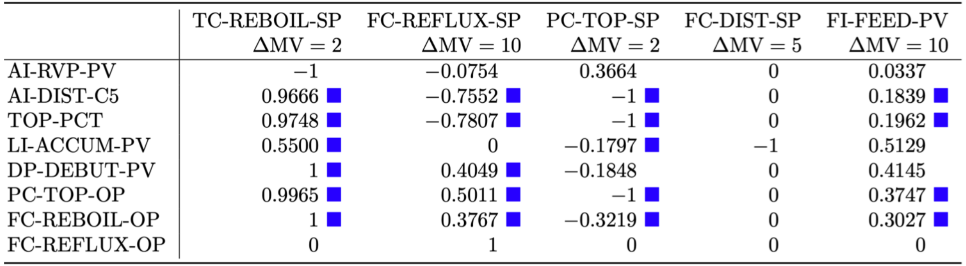
Based on threshold for max RGA = 12



	MV1	MV2	CV1	CV2	γ	RGA
1	FC-REFLUX-SP	FI-FEED-PV	PC-TOP-OP	FC-REBOIL-OP	59.14	14.36
2	TC-REBOIL-SP	FI-FEED-PV	DP-DEBUT-PV	PC-TOP-OP	59.99	10.75
3	TC-REBOIL-SP	PC-TOP-SP	AI-RVP-PV	LI-ACCUM-PV	67.50	9.26
4	TC-REBOIL-SP	FC-REFLUX-SP	DP-DEBUT-PV	FC-REBOIL-OP	81.83	14.37
5	FC-REFLUX-SP	PC-TOP-SP	AI-DIST-C5	TOP-PCT	124.38	30.54
6	TC-REBOIL-SP	PC-TOP-SP	AI-DIST-C5	PC-TOP-OP	131.01	33.24
7	TC-REBOIL-SP	FC-REFLUX-SP	AI-DIST-C5	TOP-PCT	165.64	40.79
8	PC-TOP-SP	FI-FEED-PV	AI-DIST-C5	TOP-PCT	169.40	16.04
9	TC-REBOIL-SP	PC-TOP-SP	TOP-PCT	PC-TOP-OP	181.27	45.81
10	TC-REBOIL-SP	FI-FEED-PV	AI-DIST-C5	TOP-PCT	189.76	18.39
11	FC-REFLUX-SP	FI-FEED-PV	AI-DIST-C5	TOP-PCT	276.03	32.66
12	TC-REBOIL-SP	PC-TOP-SP	AI-DIST-C5	TOP-PCT	472.37	118.54
13	TC-REBOIL-SP	PC-TOP-SP	LI-ACCUM-PV	FC-REBOIL-OP	530.00	66.23

Near-collinear pairs

Mark affected gains in blue squares



	TC-REBOIL-SP $\Delta\mathrm{MV} = 2$	FC-REFLUX-SP $\Delta\mathrm{MV} = 10$	PC-TOP-SP $\Delta\mathrm{MV} = 2$	FC-DIST-SP $\Delta\mathrm{MV} = 5$	FI-FEED-PV $\Delta\mathrm{MV} = 10$
AI-RVP-PV	-1	-0.0754	0.3664	0	0.0337
AI-DIST-C5	0.9666 🟦	-0.7552 🟦	-1 🟦	0	0.1839 🟦
TOP-PCT	0.9748 🟦	-0.7807 🟦	-1 🟦	0	0.1962 🟦
LI-ACCUM-PV	0.5500 🟦	0	-0.1797 🟦	-1	0.5129
DP-DEBUT-PV	1 🟦	0.4049 🟦	-0.1848	0	0.4145
PC-TOP-OP	0.9965 🟦	0.5011 🟦	-1 🟦	0	0.3747 🟦
FC-REBOIL-OP	1 🟦	0.3767 🟦	-0.3219 🟦	0	0.3027 🟦
FC-REFLUX-OP	0	1	0	0	0

Scaled gain matrix

Check 2x2 matrix conditioning using RGA

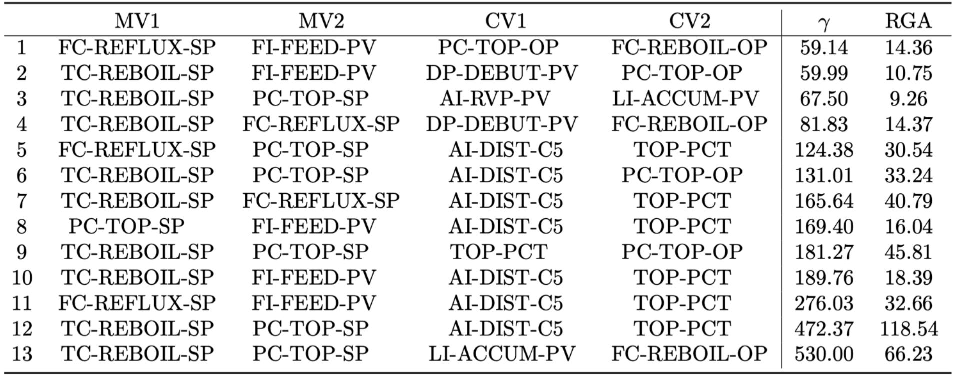
Found 13 near-collinear 2x2 pairs:

Based on threshold for max RGA = 12

Fix near-collinear pairs using engineering judgement and domain/process knowledge:

- Could make them exactly collinear if we can't control both CVs.
- Assume that making them collinear is the correct approach for this debutanizer example.

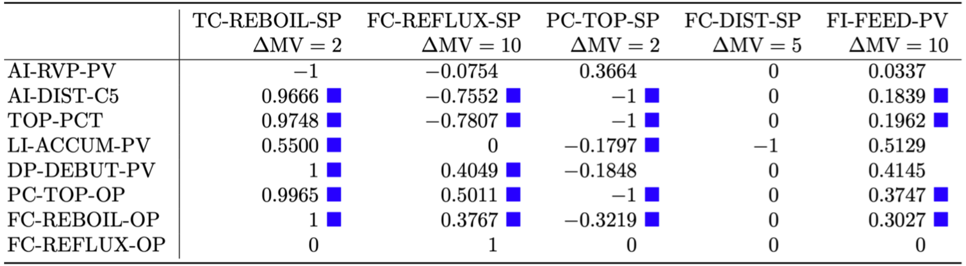
  Different for each process!



	MV1	MV2	CV1	CV2	γ	RGA
1	FC-REFLUX-SP	FI-FEED-PV	PC-TOP-OP	FC-REBOIL-OP	59.14	14.36
2	TC-REBOIL-SP	FI-FEED-PV	DP-DEBUT-PV	PC-TOP-OP	59.99	10.75
3	TC-REBOIL-SP	PC-TOP-SP	AI-RVP-PV	LI-ACCUM-PV	67.50	9.26
4	TC-REBOIL-SP	FC-REFLUX-SP	DP-DEBUT-PV	FC-REBOIL-OP	81.83	14.37
5	FC-REFLUX-SP	PC-TOP-SP	AI-DIST-C5	TOP-PCT	124.38	30.54
6	TC-REBOIL-SP	PC-TOP-SP	AI-DIST-C5	PC-TOP-OP	131.01	33.24
7	TC-REBOIL-SP	FC-REFLUX-SP	AI-DIST-C5	TOP-PCT	165.64	40.79
8	PC-TOP-SP	FI-FEED-PV	AI-DIST-C5	TOP-PCT	169.40	16.04
9	TC-REBOIL-SP	PC-TOP-SP	TOP-PCT	PC-TOP-OP	181.27	45.81
10	TC-REBOIL-SP	FI-FEED-PV	AI-DIST-C5	TOP-PCT	189.76	18.39
11	FC-REFLUX-SP	FI-FEED-PV	AI-DIST-C5	TOP-PCT	276.03	32.66
12	TC-REBOIL-SP	PC-TOP-SP	AI-DIST-C5	TOP-PCT	472.37	118.54
13	TC-REBOIL-SP	PC-TOP-SP	LI-ACCUM-PV	FC-REBOIL-OP	530.00	66.23

Near-collinear pairs

Mark affected gains in blue squares

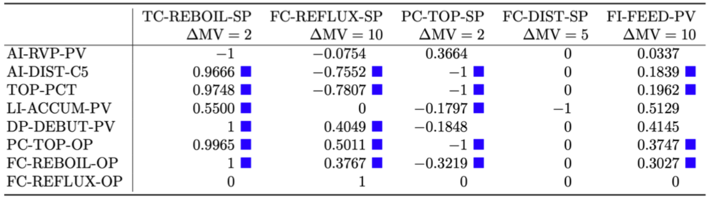


	TC-REBOIL-SP $\Delta MV = 2$	FC-REFLUX-SP $\Delta MV = 10$	PC-TOP-SP $\Delta MV = 2$	FC-DIST-SP $\Delta MV = 5$	FI-FEED-PV $\Delta MV = 10$
AI-RVP-PV	−1	−0.0754	0.3664	0	0.0337
AI-DIST-C5	0.9666 🟦	−0.7552 🟦	−1 🟦	0	0.1839 🟦
TOP-PCT	0.9748 🟦	−0.7807 🟦	−1 🟦	0	0.1962 🟦
LI-ACCUM-PV	0.5500 🟦	0	−0.1797 🟦	−1	0.5129
DP-DEBUT-PV	1 🟦	0.4049 🟦	−0.1848	0	0.4145
PC-TOP-OP	0.9965 🟦	0.5011 🟦	−1 🟦	0	0.3747 🟦
FC-REBOIL-OP	1 🟦	0.3767 🟦	−0.3219 🟦	0	0.3027 🟦
FC-REFLUX-OP	0	1	0	0	0

Scaled gain matrix

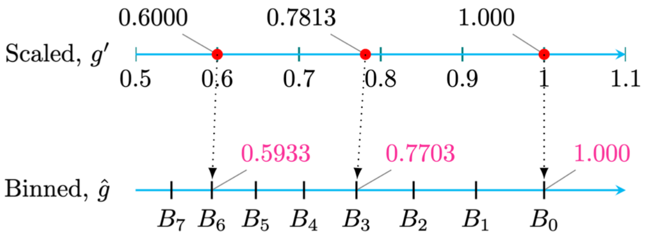
Generate binning grid, then adjust affected gains

Scaled gain matrix



	TC-REBOIL-SP $\Delta MV = 2$	FC-REFLUX-SP $\Delta MV = 10$	PC-TOP-SP $\Delta MV = 2$	FC-DIST-SP $\Delta MV = 5$	FI-FEED-PV $\Delta MV = 10$
AI-RVP-PV	-1	-0.0754	0.3664	0	0.0337
AI-DIST-C5	0.9666 🟦	-0.7552 🟦	-1 🟦	0	0.1839 🟦
TOP-PCT	0.9748 🟦	-0.7807 🟦	-1 🟦	0	0.1962 🟦
LI-ACCUM-PV	0.5500 🟦	0	-0.1797 🟦	-1	0.5129
DP-DEBUT-PV	1 🟦	0.4049 🟦	-0.1848	0	0.4145
PC-TOP-OP	0.9965 🟦	0.5011 🟦	-1 🟦	0	0.3747 🟦
FC-REBOIL-OP	1 🟦	0.3767 🟦	-0.3219 🟦	0	0.3027 🟦
FC-REFLUX-OP	0	1	0	0	0

Binning

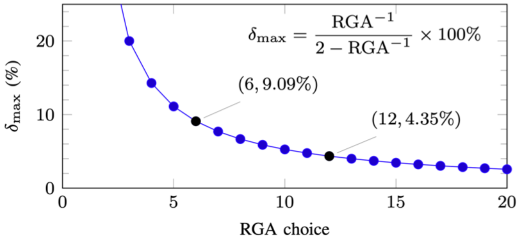


For each affected gain marked in blue, adjust to the closest absolute value bin in the grid
(while preserving signs, e.g. adjust -0.6000 to -0.5933)

Results: Property 1 – gain adjustment % bounded

Binned gain matrix *(gain adjustments in red)*

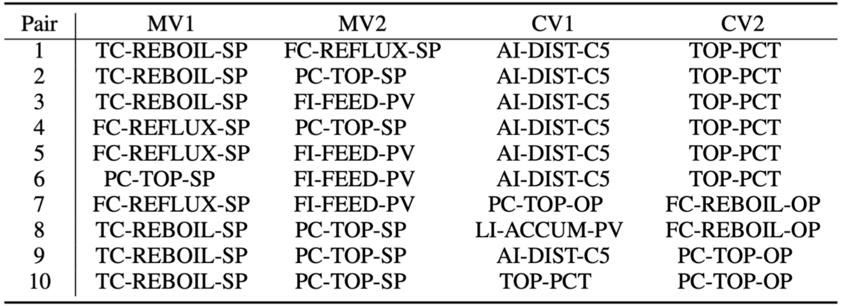
	TC-REBOIL-SP	FC-REFLUX-SP	PC-TOP-SP	FC-DIST-SP	FI-FEED-PV
AI-RVP-PV	−1	−0.0754	0.3664	0	0.0337
AI-DIST-C5	1 (3.46%)	−0.7703 (2.00%)	−1	0	0.1914 (4.08%)
TOP-PCT	1 (2.59%)	−0.7703 (−1.34%)	−1	0	0.1914 (−2.41%)
LI-ACCUM-PV	0.5439 (−1.11%)	0	−0.1755 (−2.37%)	−1	0.5129
DP-DEBUT-PV	1	0.4189 (3.46%)	−0.1848	0	0.4189 (1.06%)
PC-TOP-OP	1 (0.35%)	0.4985 (−0.52%)	−1	0	0.3840 (2.49%)
FC-REBOIL-OP	1	0.3840 (1.93%)	−0.3227 (0.25%)	0	0.2958 (−2.30%)
FC-REFLUX-OP	0	1	0	0	0



Property 1: All gain adjustments are indeed less than 4.35%, for a max RGA requirement of 12.

Results: Property 2 – RGA thresholds satisfied

Created 10 new collinear pairs that were originally near-collinear pairs.



Pair	MV1	MV2	CV1	CV2
1	TC-REBOIL-SP	FC-REFLUX-SP	AI-DIST-C5	TOP-PCT
2	TC-REBOIL-SP	PC-TOP-SP	AI-DIST-C5	TOP-PCT
3	TC-REBOIL-SP	FI-FEED-PV	AI-DIST-C5	TOP-PCT
4	FC-REFLUX-SP	PC-TOP-SP	AI-DIST-C5	TOP-PCT
5	FC-REFLUX-SP	FI-FEED-PV	AI-DIST-C5	TOP-PCT
6	PC-TOP-SP	FI-FEED-PV	AI-DIST-C5	TOP-PCT
7	FC-REFLUX-SP	FI-FEED-PV	PC-TOP-OP	FC-REBOIL-OP
8	TC-REBOIL-SP	PC-TOP-SP	LI-ACCUM-PV	FC-REBOIL-OP
9	TC-REBOIL-SP	PC-TOP-SP	AI-DIST-C5	PC-TOP-OP
10	TC-REBOIL-SP	PC-TOP-SP	TOP-PCT	PC-TOP-OP

Property 2: All non-collinear submatrices are now below RGA = 12.

Binning 'repaired' the gain matrix:

- Adjusted 13 near-collinear pairs into 10 collinear pairs (APC won't try to control those CV simultaneously)
- Adjusted the 3 remaining pairs to $\mathrm{RGA} \leq 12$ (APC won't make large MV moves)

Limitations and future work

- **How can we enforce mass balance in this binning scheme?**
  Certain gains or gain combinations (ratios/sums etc.) must be locked to satisfy process mass balance and other physical constraints, how can we reconcile that during binning?
  Work in progress, in collaboration with Nick Alsop at Borealis AG (Sweden)

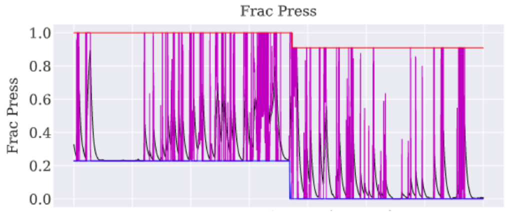
- **How can we design better visualization tools for matrix conditioning?**
  Can we do better than just coloring/marking gains of ill-conditioned pairs in the table?

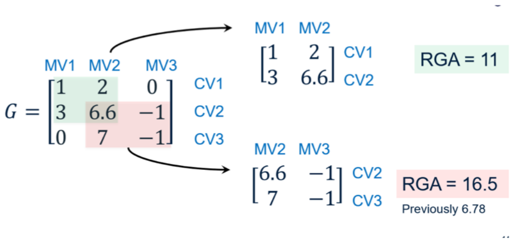
- **How can we extend binning to higher-order interactions?**
  Can we also apply similar binning techniques for higher-order 3x3, 4x4 etc. submatrices?

Takeaways - Details see APCpapers.github.io

- **Near-collinear submatrices can cause the LP optimizer to make undesirable, large MV moves or generate erratic steady-state targets.**

Takeaways - Details see APCpapers.github.io





$$G = \begin{array}{cccc} \text{MV1} & \text{MV2} & \text{MV3} & \\ \left[\begin{matrix} 1 \\ 3 \\ 0 \end{matrix}\right. & \begin{matrix} 2 \\ 6.6 \\ 7 \end{matrix} & \left.\begin{matrix} 0 \\ -1 \\ -1 \end{matrix}\right] & \begin{matrix} \text{CV1} \\ \text{CV2} \\ \text{CV3} \end{matrix} \end{array}$$

$$\begin{array}{ccc} \text{MV1} & \text{MV2} & \\ \left[\begin{matrix} 1 \\ 3 \end{matrix}\right. & \left.\begin{matrix} 2 \\ 6.6 \end{matrix}\right] & \begin{matrix} \text{CV1} \\ \text{CV2} \end{matrix} \end{array}$$

RGA = 11

$$\begin{array}{ccc} \text{MV2} & \text{MV3} & \\ \left[\begin{matrix} 6.6 \\ 7 \end{matrix}\right. & \left.\begin{matrix} -1 \\ -1 \end{matrix}\right] & \begin{matrix} \text{CV2} \\ \text{CV3} \end{matrix} \end{array}$$

RGA = 16.5

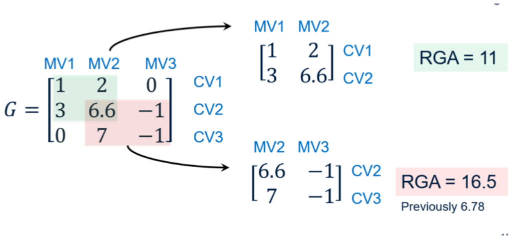
Previously 6.78

- **Near-collinear submatrices can cause the LP optimizer to make undesirable, large MV moves or generate erratic steady-state targets.**

- **Collinearity repair using classical trial-and-error adjustments can be very time-consuming and tedious for large models.**

Takeaways - Details see APCpapers.github.io

- **Near-collinear submatrices can cause the LP optimizer to make undesirable, large MV moves or generate erratic steady-state targets.**

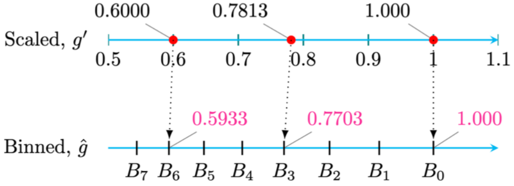


$$G = \begin{array}{cccc} \text{MV1} & \text{MV2} & \text{MV3} & \\ 1 & 2 & 0 & \text{CV1} \\ 3 & 6.6 & -1 & \text{CV2} \\ 0 & 7 & -1 & \text{CV3} \end{array}$$

$$\begin{array}{ccc} \text{MV1} & \text{MV2} & \\ 1 & 2 & \text{CV1} \\ 3 & 6.6 & \text{CV2} \end{array}$$

$$\begin{array}{ccc} \text{MV2} & \text{MV3} & \\ 6.6 & -1 & \text{CV2} \\ 7 & -1 & \text{CV3} \end{array}$$

- **Collinearity repair using classical trial-and-error adjustments can be very time-consuming and tedious for large models.**

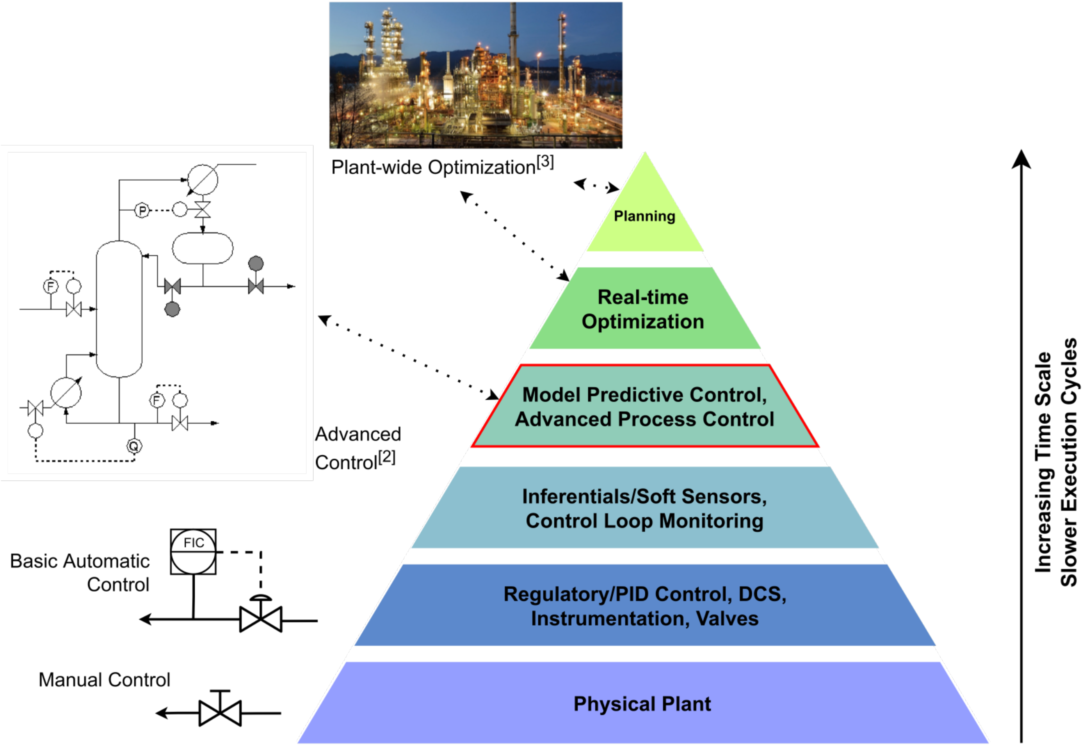


- **We present a binning solution that will condition the matrix in a single pass to a user-defined RGA, with bounded gain % adjustments.**

THE UNIVERSITY OF BRITISH COLUMBIA

APPENDIX A: Additional Slides

Industrial Model Predictive Control (MPC)



Plant-wide Optimization[3]

Advanced Control[2]

Basic Automatic Control

Manual Control

Planning

Real-time Optimization

Model Predictive Control, Advanced Process Control

Inferentials/Soft Sensors, Control Loop Monitoring

Regulatory/PID Control, DCS, Instrumentation, Valves

Physical Plant

Increasing Time Scale
Slower Execution Cycles

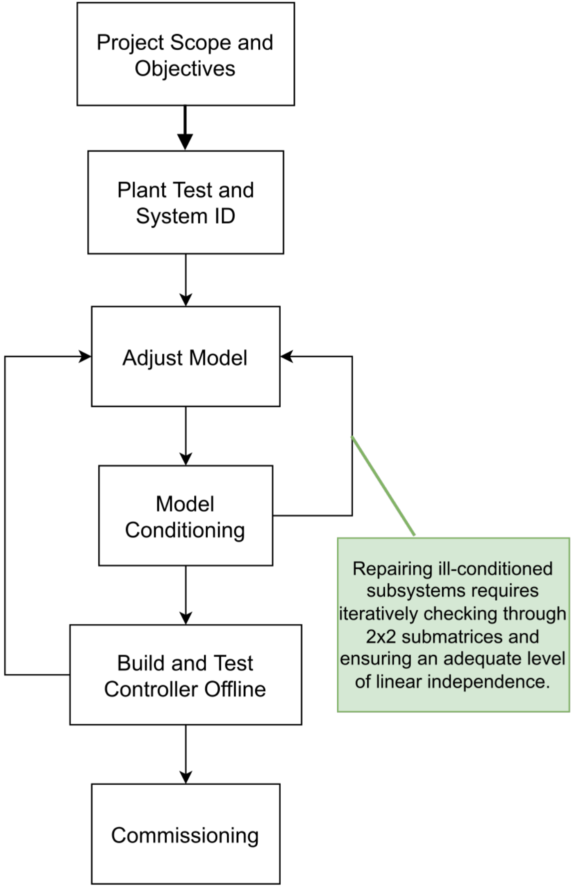
- Base layer includes PIDs, regulatory control, etc.
- Advanced Process Control (APC) sits above base layer and has longer execution cycles
- PIDs have no view of other systems in the plant
- MPC adds predictive capability and economic optimization

Adapted from [1]: Strand, S. (n.d.). *MPC in Statoil*.
[2]: Ponton, J. (2007). *Module 3.1: Control of Distillation Columns*.
[3]: Why Vancouver desperately needs a new oil refinery. (2016, March 3). *Oil Sands Magazine*.

Typical MPC Workflow

- System ID relates inputs to outputs, and is used to obtain the steady-state (SS) gain matrix

- SS gain matrix is adjusted iteratively to meet control objectives using engineering judgment
  - System ID results are not perfect



Repairing ill-conditioned subsystems requires iteratively checking through 2x2 submatrices and ensuring an adequate level of linear independence.

Why is the traditional conditioning approach difficult?

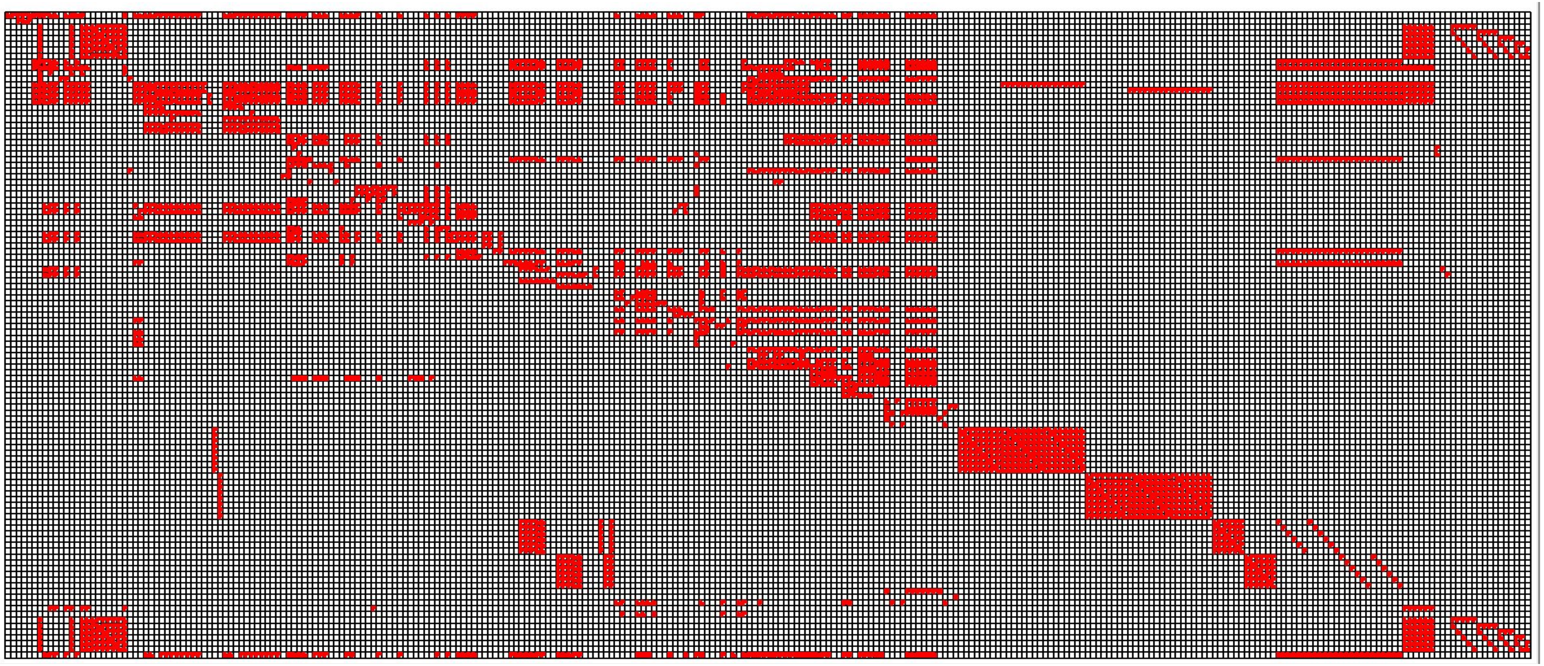


Figure from Control Consulting, Inc.

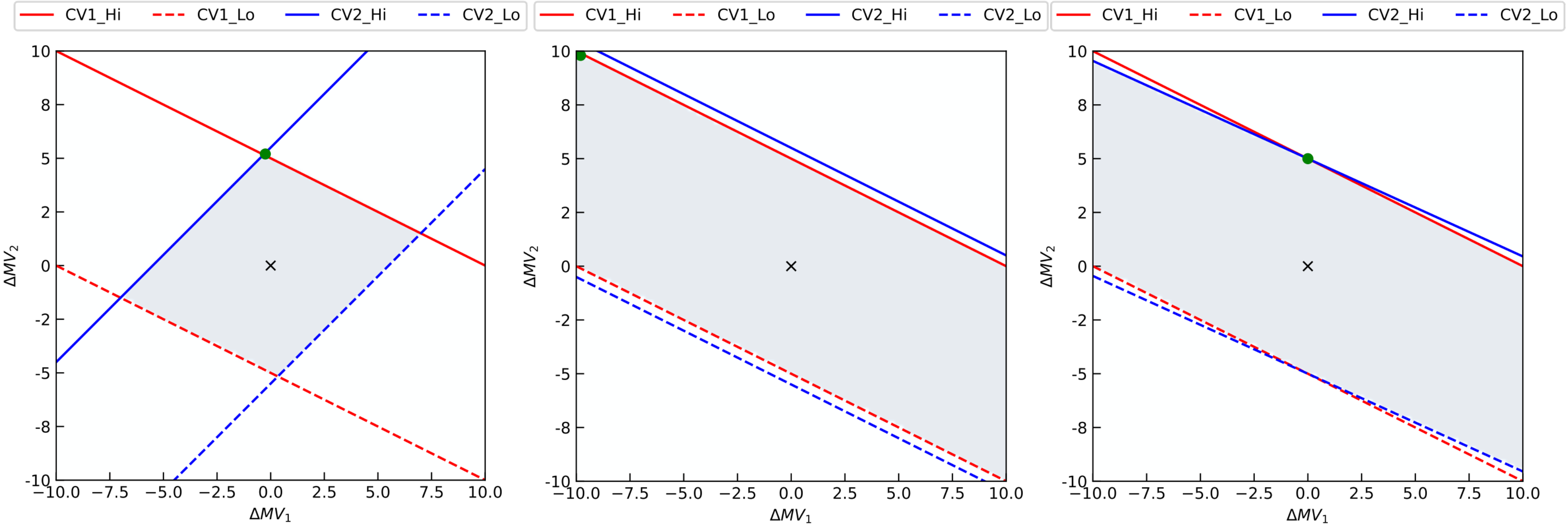
[1]: Hall, R. S., Peterson, T. J., Pottorf, T. S., Punuru, A. R., & Vowell, L. E. (2008). *Method for model gain matrix modification* (Canada Patent No. CA2661478A1).

[2]: Ishikawa, A., Ohshima, M., & Tanigaki, M. (1997). A practical method of removing ill-conditioning in industrial constrained predictive control. *Computers & Chemical Engineering*, *21*, S1093–S1098.

[3]: Zheng, Q., Harmse, M. J., Rasmussen, K. H., & Mcintyre, B. (2014). *Methods and articles for detecting, verifying, and repairing matrix collinearity* (Canada Patent No. CA2519783C).

Classifying 2x2 gain interactions

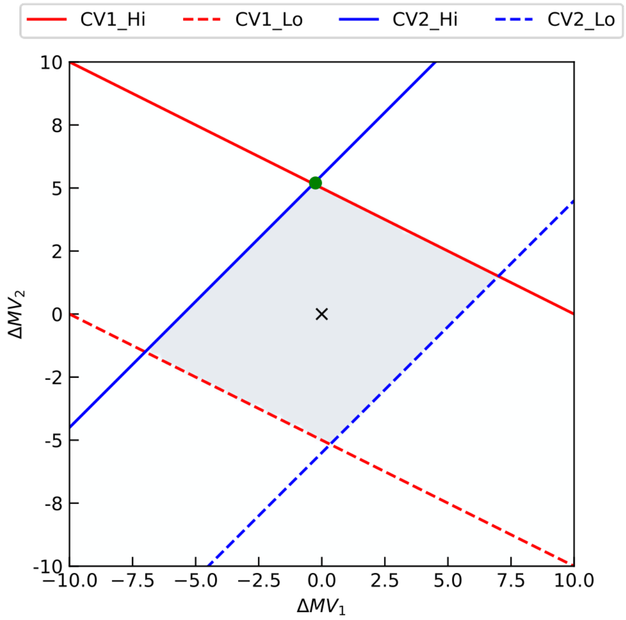
$\text{RGA} < \text{RGA}_T$ (threshold) $\text{RGA} \rightarrow \infty$ $\text{RGA} > \text{RGA}_T$



Properly conditioned **Perfectly collinear** **Ill-conditioned**

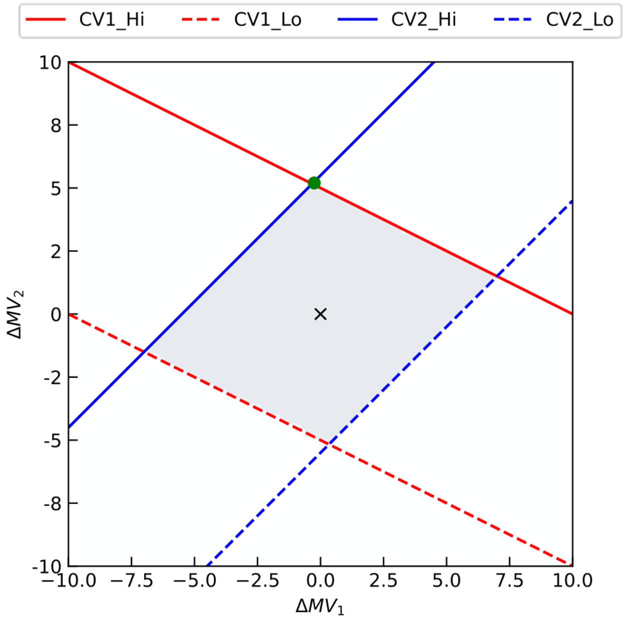
LP solution for change – well-conditioned

ΔMV_1 = -0.25
ΔMV_2 = +5.2



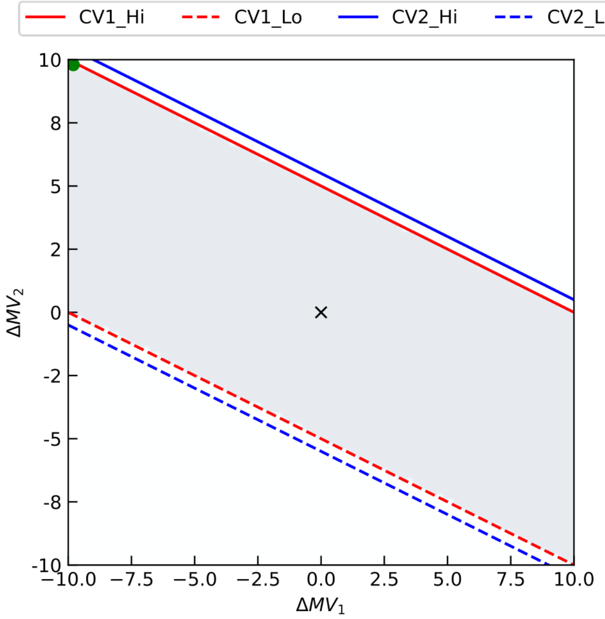
CV$_2$ high limit dropped from 1.1 to 1.0

ΔMV_1 = +0.1
ΔMV_2 = +4.9



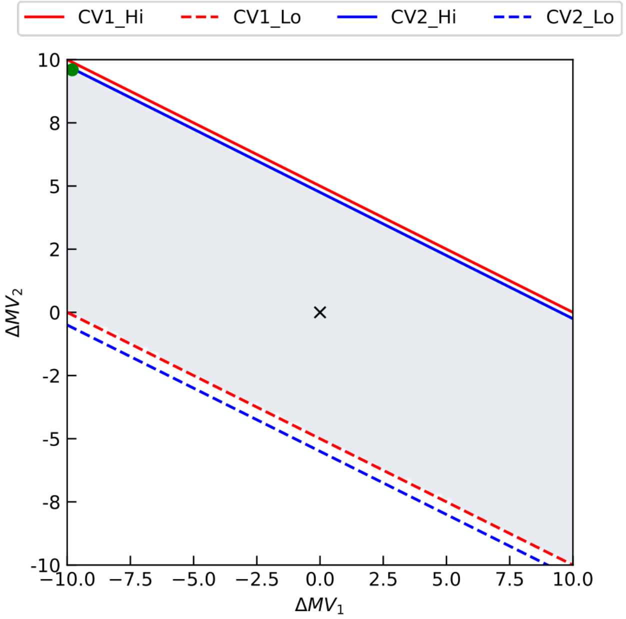
LP solution for change – collinear

$\Delta MV_1 = -10.0$
$\Delta MV_2 = +10.0$



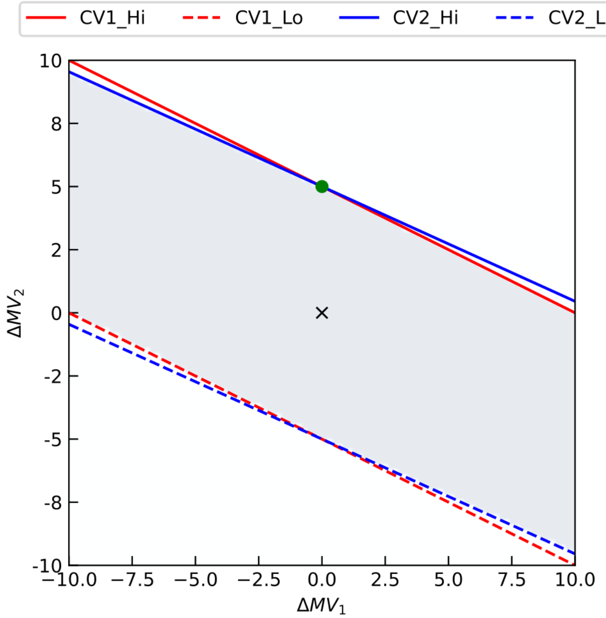
CV_2 high limit dropped from 1.1 to 1.0

$\Delta MV_1 = -10.0$
$\Delta MV_2 = +9.7$



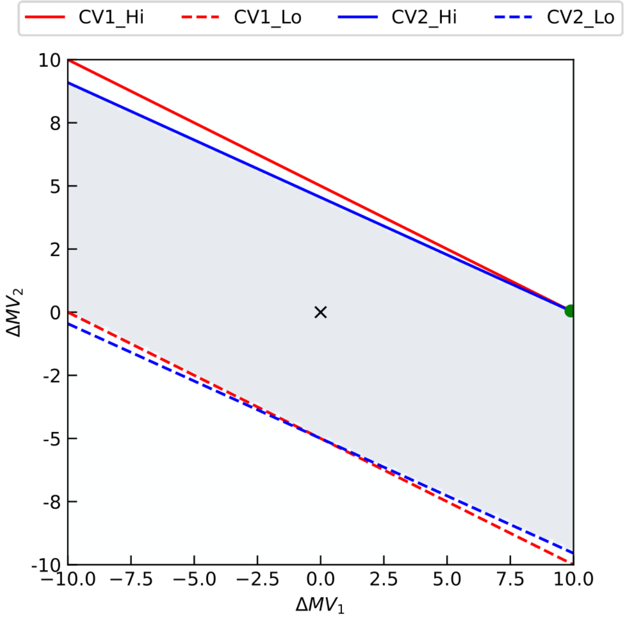
LP solution for change – ill-conditioned

$\Delta MV_1 = 0.0$
$\Delta MV_2 = +5.0$



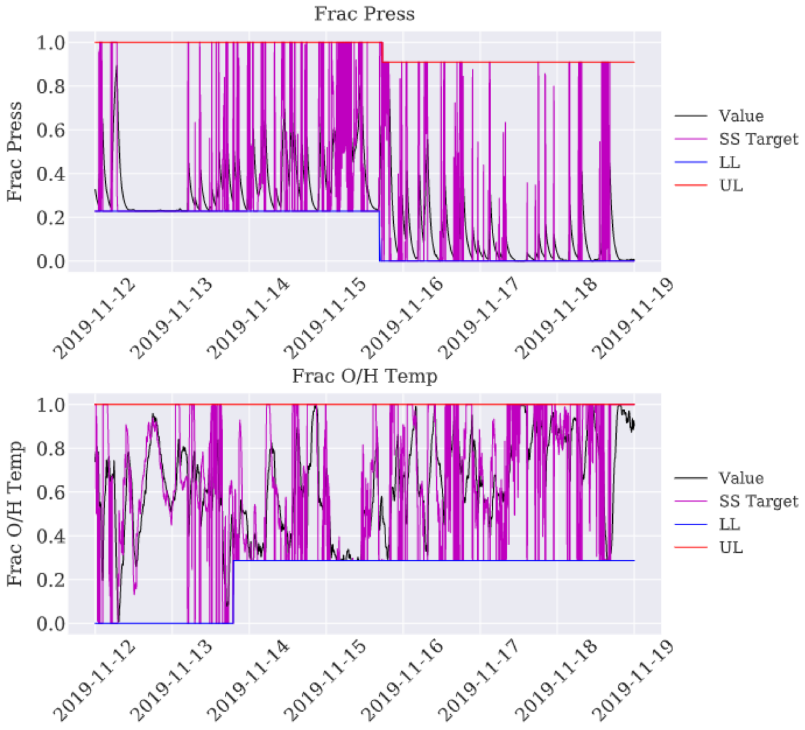
CV$_2$ high limit dropped from 1.1 to 1.0

$\Delta MV_1 = +10.0$
$\Delta MV_2 = +0.05$



Consequences of ill-conditioning

- Moving CV/MV limits with a near-collinearity can cause excessive variable movement
- Difficult to understand when these are caused by near-collinearities specifically



How do we repair ill-conditioned submatrices?

Option 1: force collinearity

- Reduces degrees of freedom
- Only one CV constraint can be satisfied

Option 2: reduce RGA

- Creates a well-conditioned submatrix
- Both CVs can be adequately controlled

Option 3: Zero out gain(s)

- If the gain direction is weak, perhaps control objectives can be achieved without it

Option 4: ignore near-collinearity

- Perhaps large MV moves are fine for the control objectives
- Also possibly useful for unrealistic variable combinations