Some past and future robustness improvements for wells and well groups

Stein Krogstad, SINTEF Digital

Background/Outline

My talk at OPM Summit 2024

We're not done yet …

- Will not say that much about wells.
  - Many small improvements since last that make a big difference (Tor Harald, Kai, ++)
- Will talk about a small *side-project* on improving *balancing* of well-group-trees
  - Interesting problem and can show colorful trees

Why do well-equations fail to converge?

Most common causes in my opinion:

1. There is no solution
   - THP-limit too high for the well to produce
   - Some rate-limit is too low for the well to produce
2. Well primary-variables are in bad shape
3. Some complex multi-segment well equations are just difficult to solve

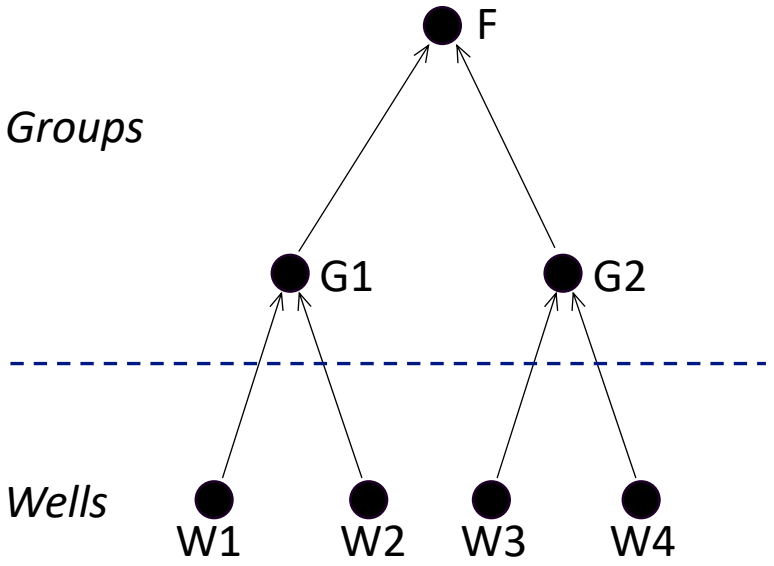
Not too big concern: we treat this reasonably well by subsequent operability-check that will close a non-operable well

This we don't handle very well. Culprit is often an un-balanced group tree that gives unreasonable target rates.

We might see gains from improved initialization, but certainly we might end up in a bad state due to the above.

Convergence issues for wells with a reasonable control is not that frequent anymore. But there is still room from improvement (e.g., line-search for Newton update)

Group control for production wells



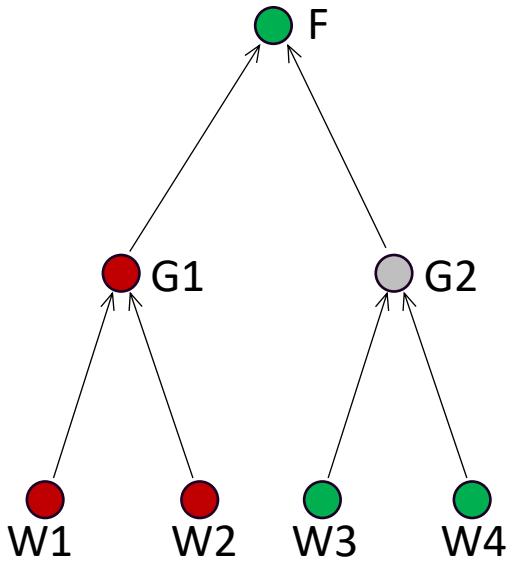
Each group has:

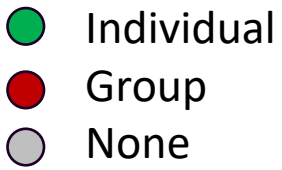
- *Preferred* mode of control (e.g., OIL)
- Upper limit (target) for preferred mode and possibly for other modes
- *Guide-rate* for parent's preferred mode of control (if available)

Each well has:

- Upper rate-limits/lower pressure limits for any number of modes
- *Guide-rate* for parent's preferred mode of control (if available)

Group control for production wells



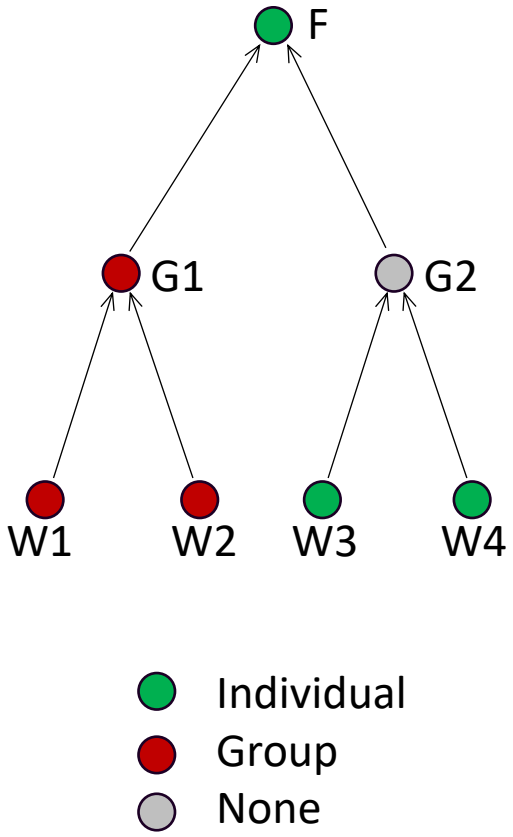


We'll call the tree *balanced* if

- Each group's rates equals the sum of its children's rates
- Each well/group produces less or equal to all limits
- Each well/group produces less or equal to assigned rate from parent
- Each node falls into 1 out of three categories:

1. *Individual* – well/group produces at one of its limits
2. *Group* – group/well produces its assigned rate from parent (of preferred mode)
3. *None* – group rate is constrained by children (all *individual* or *none*)

Group control for production wells



How do we obtain a balanced tree?

- It's the solution of an optimization problem:

$$\max_{\boldsymbol{r}} f(\boldsymbol{r})$$
subject to
$$g(\boldsymbol{r}) \leq \mathbf{0}$$
$$h(\boldsymbol{r}) = \mathbf{0}$$

$\boldsymbol{r}$: all well/group rates
f : field rate of preferred mode
g : limits and group target inequalities
h : rate sum equalities

- We can't attack this optimization problem fully coupled with entire reservoir model
- By fixing the well-fractions and utilizing evaluated IPR
  - Problem becomes linear within each green/red/gray – configuration (piecewise linear overall)
  - Pressure limits for wells can be translated to rate-limits
  - Overall optimization problems resembles a linear programming problem, but not exactly due to the guide-rate business (piecewise linear inequality constraints).

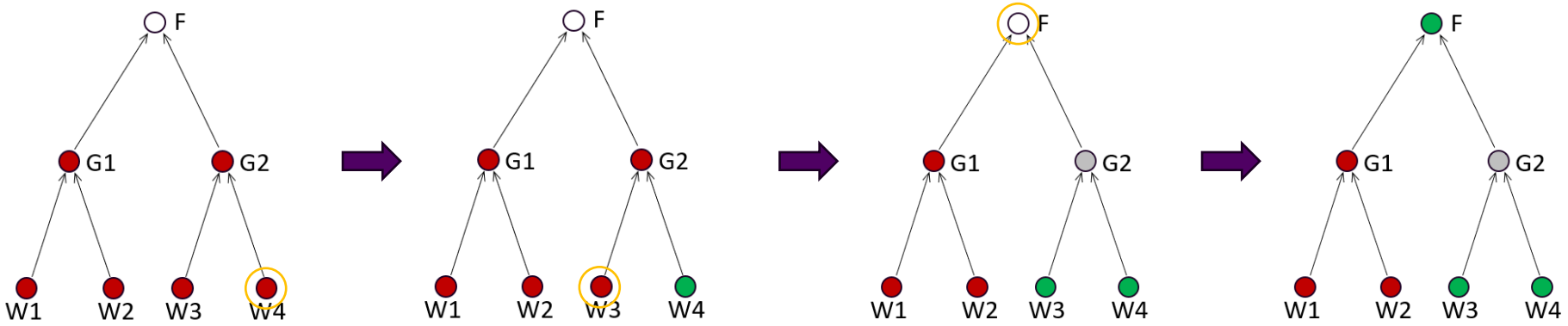
Group control for production wells

One solution approach for optimization problem:

1. Start with all rates equal zero (top node un-categorized, all other as group)
2. Parametrize node-rates in terms of top-node preferred mode rate $q_{1,m}$ (based current config). Each component rate $q_{i,c}$ at node i is a linear function of $q_{1,m}$, i.e., $q_{1,c} = a_{1,c}q_{1,m} + b_{1,c}$ found by traversing down and up the tree
3. Find the node that first hits a limit as top-node rate increases
   - Update node to individual-category
   - Potentially update parent(s) to none-category
4. Update rates for all influenced nodes
5. If node found in 3. is the top-node or top-node has been set to none-category we're done, otherwise go to 2.

Concern:

- Complexity of algorithm is $O(\#\text{nodes}^2)$ and we need to run it lots of times
- Need a more efficient way to balance a tree that is *almost* balanced



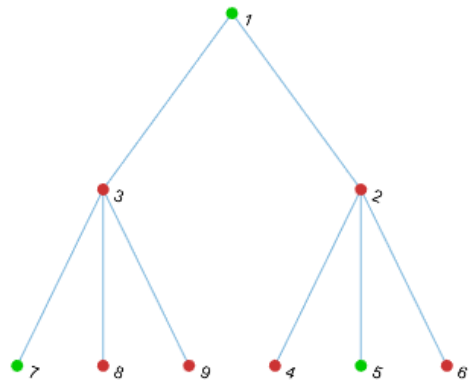
Group control for production wells

Given an un-balanced tree, we do a pre-balancing step:

1. Sum rates and cap at limits for *individual* nodes from well and up
2. Distribute group targets and cap *group* nodes from top and down
3. Repeat 1-2 until fixed

- Result is a balanced tree except from top node
- Always converges (< #node iterations)
- In best case, gives a balanced tree
- In worst case, converges to all rates equal zero

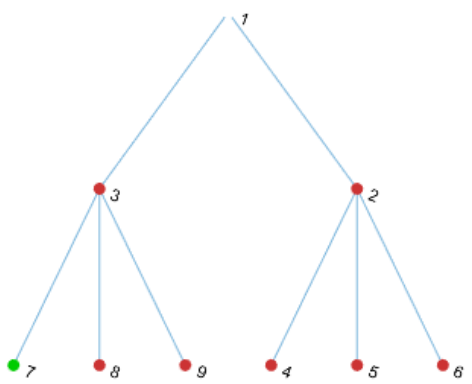
Example:



Balanced tree with field oil target 1e4

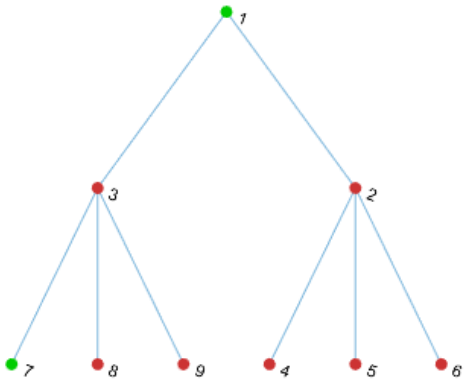
field oil target reduced to 8e3

FOPR = 7.9e3 < 8e3



Pre-balancing (2 its)

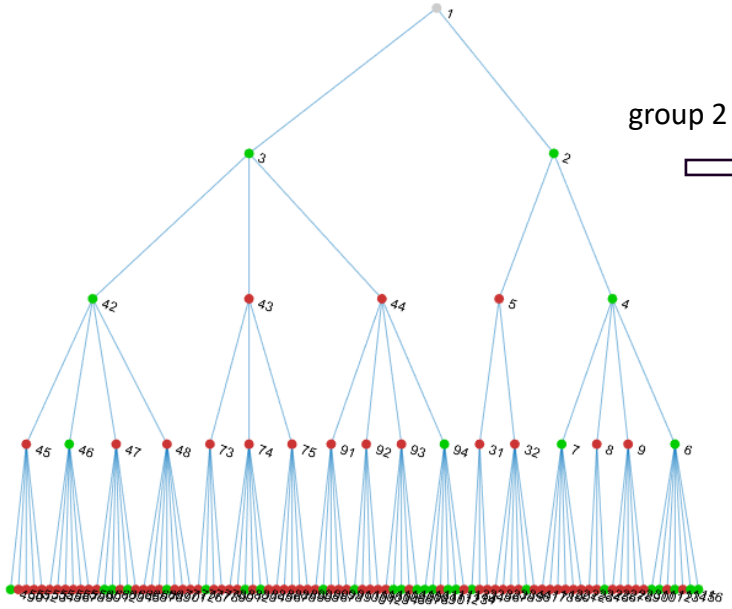
FOPR = 8e3



Balancing (1 it)

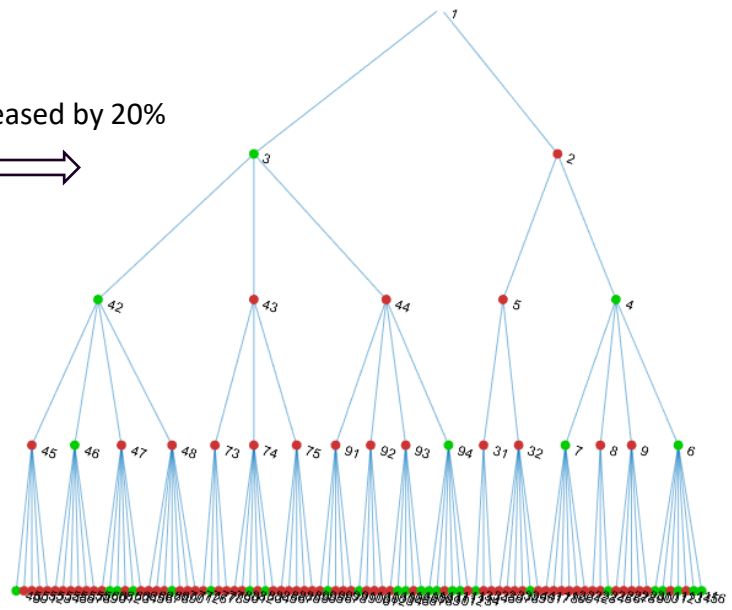
Group control for production wells

Example:

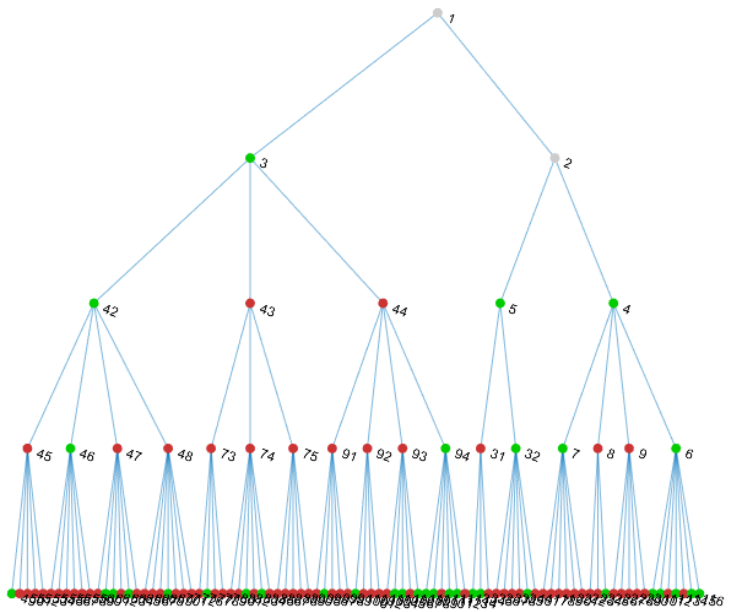


Balanced tree – field production constrained by group 2 and group 3 limits

group 2 capacity increased by 20%



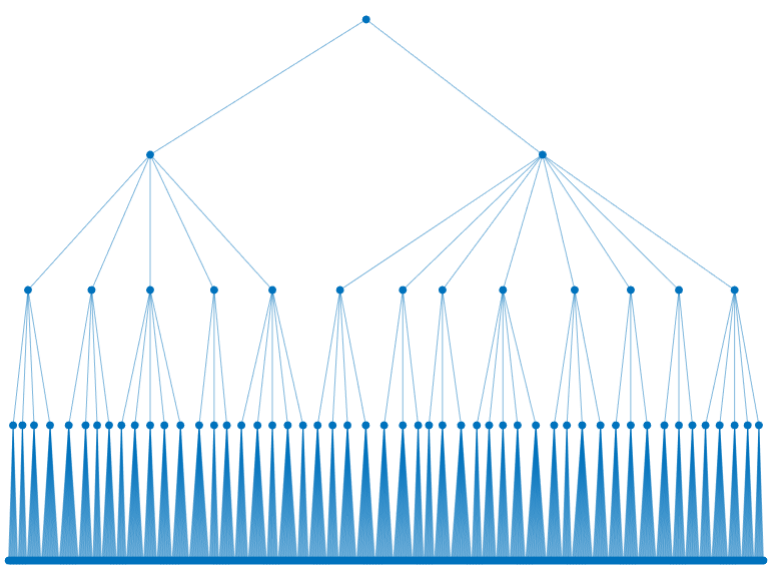
Pre-balancing (**2** its)



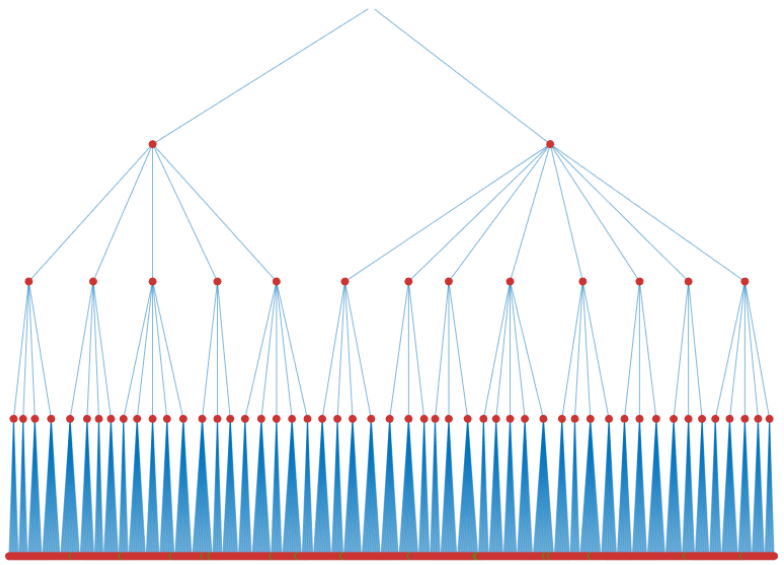
Balancing (**3** its) – group 2 production constrained by group 4 and group 5 limits

Group control for production wells

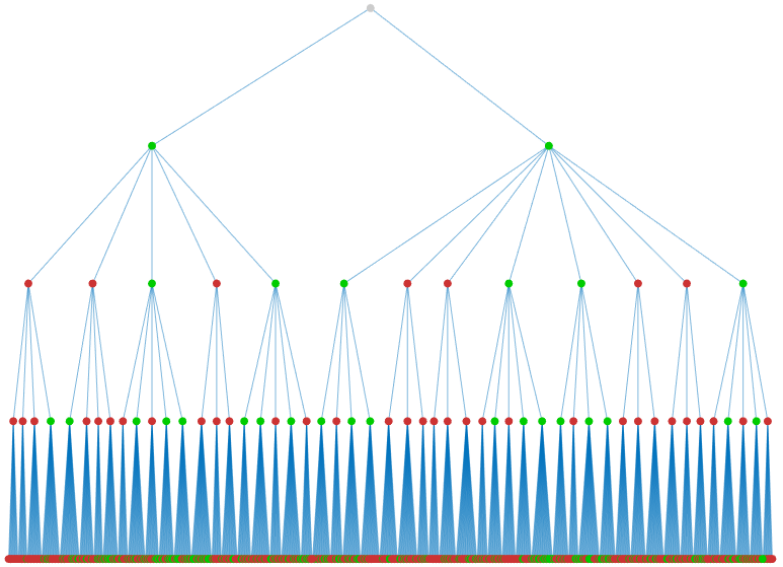
Example:



Un-balanced tree (1747 nodes)– random rates



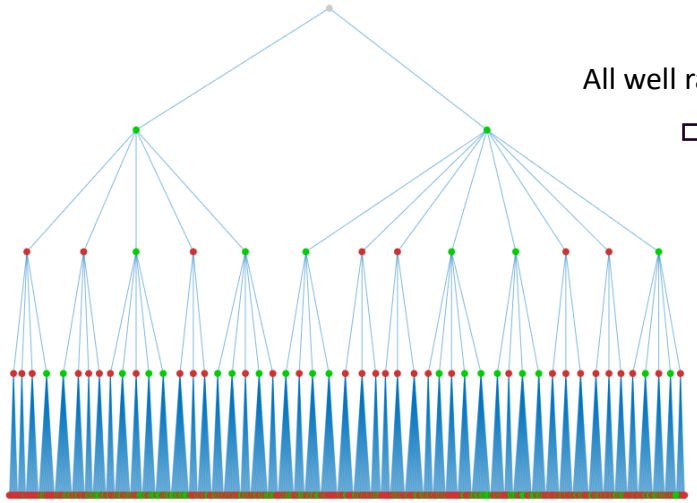
Pre-balancing **5** its – not that helpful



Balancing **300** its (number of green nodes 317)

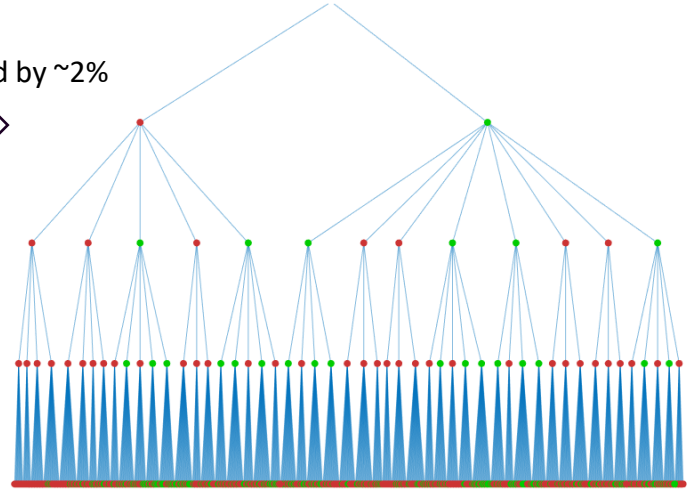
Group control for production wells

Example:

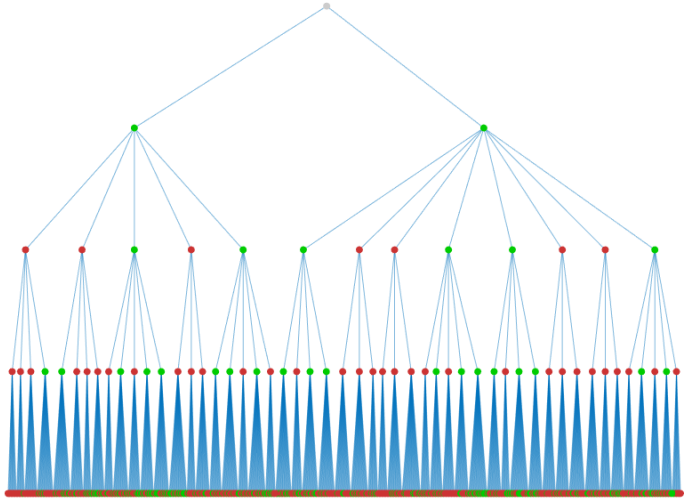


Balanced tree

All well rates fractions changed by ~2%



Pre-balancing - **14** its



Balancing **17** its

Concuding remarks

- Still a lot of room for improvements for wells, groups and network
- Well convergence issues often due to *bad* targets from groups.
  - Highly likely there is a lot to gain from making sure group-trees are balanced before trying to solve well-equations (better convergence + less wasted computations)
- Presented algorithm quite simple in essence, but was (at least for me) rather tricky to get working properly
  - Prototyped outside OPM, will first attempt to incorporate it with some *agentic* help just for testing