# Bounded dual simplex using intervals

Damien Massé

LabSTICC - Robex
Université de Bretagne Occidentale
Brest, France

Robex, dec. 2023

# Outline

1. Simplex algorithm.
2. Bounded version with intervals.

# Linear optimisation

We consider the following presentation of the linear optimisation problem, with $\boldsymbol{c} \in \mathbb{R}^n$, $A \in \mathcal{M}_{m,n}(\mathbb{R})$ and $\boldsymbol{b} \in \mathbb{R}^m$:

$$\begin{aligned} S = \max \boldsymbol{c}^T \boldsymbol{x} \\ \text{s.t. } A\boldsymbol{x} \leq \boldsymbol{b} \text{ and } \boldsymbol{x} \in \mathbb{R}^n \end{aligned}$$

This problem can be seen as finding the vertex of a polyhedron maximizing a linear form. Here $n$ if the dimension of the space, and $m$ the number of constraints.

# Dual problem

Each vertex of the polyhedron can be characterized by $N$ constraints which are equalities on it. Hence, the linear optimisation problem can be solved buy finding the "correct" $N$ constraints.
More precisely, let $A_{\mathcal{B}}$ an (invertible) $n \times n$ sub-matrix of $A$ (the *basis*). By posing $\boldsymbol{x} = A_{\mathcal{B}}^{-1}\boldsymbol{y}$, we get:

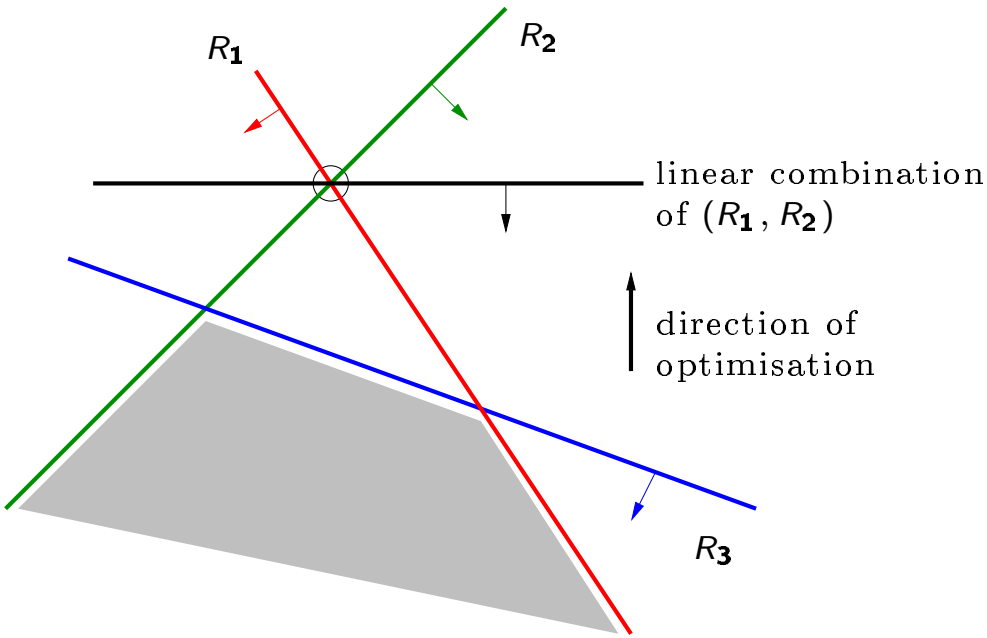
- $\boldsymbol{c}^T\boldsymbol{x} = \boldsymbol{c}^T A_{\mathcal{B}}^{-1}\boldsymbol{y}$
- $A\boldsymbol{x} \leq \boldsymbol{b} \;\Rightarrow\; \boldsymbol{y} \leq \boldsymbol{b}_{\mathcal{B}}$

As a consequence:

$$\boldsymbol{c}^T A_{\mathcal{B}}^{-1} \geq 0 \;\Rightarrow\; S \leq \boldsymbol{c}^T A_{\mathcal{B}}^{-1}\boldsymbol{b}_{\mathcal{B}}$$

The condition $\boldsymbol{c}^T A_{\mathcal{B}}^{-1} \geq 0$ means that $\mathcal{B}$ is *dual-feasible*. We note by $\lambda_{\mathcal{B}} = \boldsymbol{c}^T A_{\mathcal{B}}^{-1}$ the solution of the dual problem.

# Dual feasibility and optimality



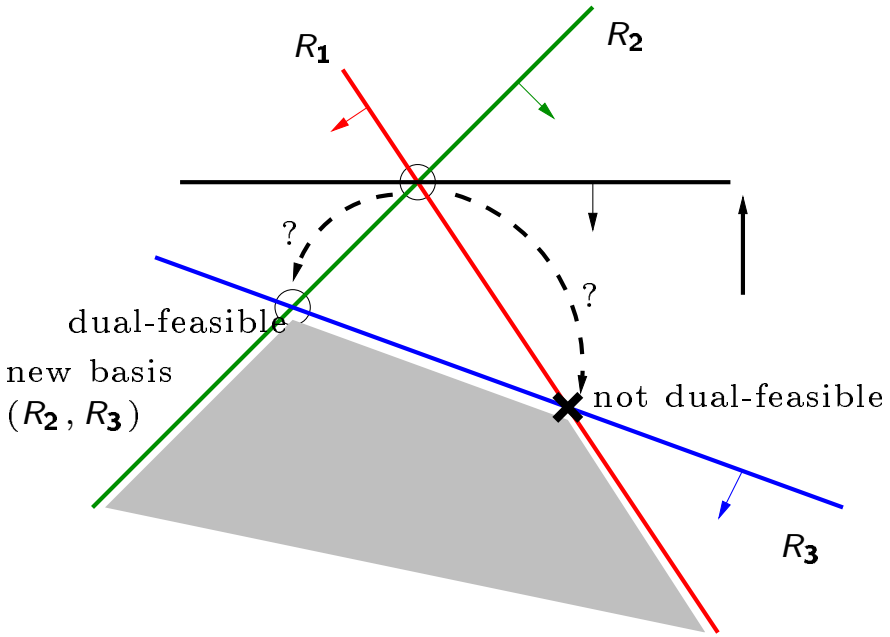

A dual-feasible basis bounds $S$ (but the associated vertex may not be inside the polyhedron).

To prove optimality, we need to check that $\boldsymbol{y} = \boldsymbol{b}_{\mathcal{B}}$ satisfies all the constraints, i.e. $\boldsymbol{x} = A_{\mathcal{B}}^{-1}\boldsymbol{b}_{\mathcal{B}}$ is a valid vertex of the polyhedron:

$$AA_{\mathcal{B}}^{-1}\boldsymbol{b}_{\mathcal{B}} \leq \boldsymbol{b}$$

# Dual simplex algorithm

The dual simplex algorithm starts from a dual-feasible basis and changes the basis (one constraint by one) until reaching an optimal one.



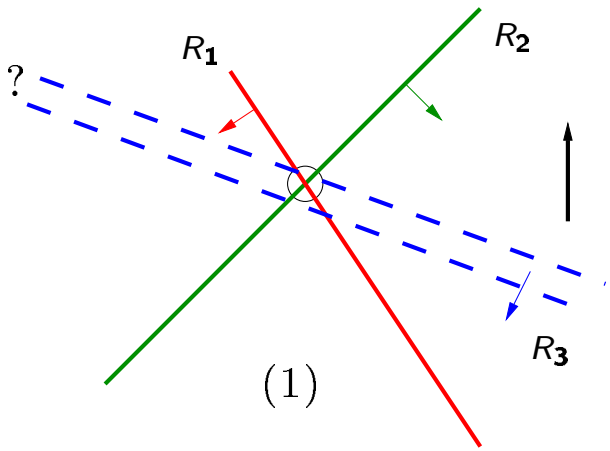

1. Find a unsatisfied constraint $\boldsymbol{R}^T\boldsymbol{x} \leq \boldsymbol{b}_R$ such that $\boldsymbol{R}^T A_{\mathcal{B}}^{-1}\boldsymbol{b}_{\mathcal{B}} > \boldsymbol{b}_R$ (will enter the basis)
2. Find the (one?) direction which keeps the basis dual-feasible *and* enables to satisfy $R$.
3. Iterate.

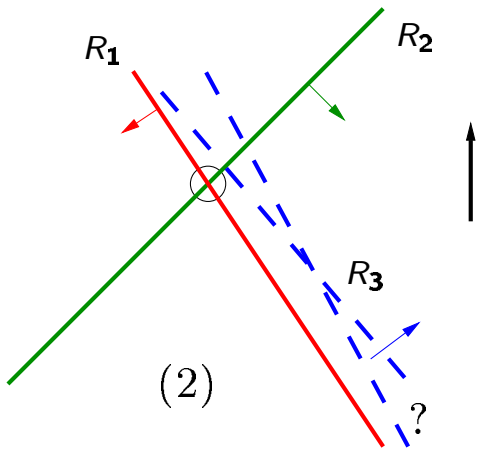
Note: $AA_{\mathcal{B}}^{-1}$ (and the associated vectors) can be updated iteratively or recomputed from $A_{\mathcal{B}}$ (more costly).

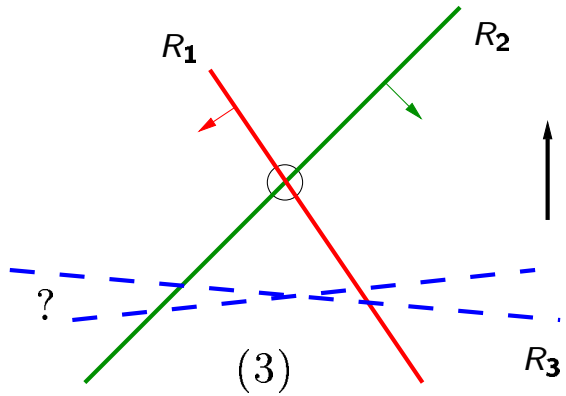
# Sources of errors

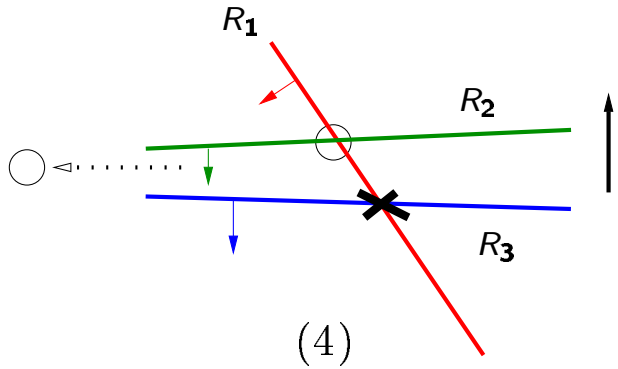
The algorithm uses comparisons to evaluate:

1. If a constraint is satisfied/unsatisfied.
2. If a direction enables to satisfy (or not) an unsatisfied constraint.
3. If a direction keeps the basis dual-feasible.




All these choices can lead to non-optimal basis (or an empty answer).
Furthermore, ill-conditioned basis can produce an unreliable $AA_{\mathcal{B}}^{-1}$ which may propagate (case 4).

# Requirements

We want overapproximations.

1. We need a *dual-feasible* basis (which will give a reliable bound for $S$).
2. With this basis, we need an upper bound of $\lambda_{\mathcal{B}} \boldsymbol{b}_{\mathcal{B}}$ ($\rightarrow$ upper bound for $S$).
3. If the algorithm returns "empty" ($S = -\infty$), the polyhedron *is* empty.

Having a near-optimal basis (all constraints are satisfied or "almost" satisfied) is a preference.
Unfortunaly, cases (2) and (3) may give a non-dual feasible basis. We can "solve" the problem if the constraints are bounded.

# Neumaier's approach

We consider that the polyhedron is inside a bounding box:

$$\boldsymbol{x} \in [B]$$

and we suppose that we have an approximate dual basis and solution $\lambda_{\mathcal{B}} = c^T A_{\mathcal{B}}^{-1}$ (we denote by $\hat{\lambda}_{\mathcal{B}}$ the approximation of the solution).
Let's compute $[r] \subseteq \hat{\lambda}_{\mathcal{B}} A_{\mathcal{B}} - c^T$. Then

$$S \leq ub(\hat{\lambda}_{\mathcal{B}} \boldsymbol{b}_{\mathcal{B}} - [r][B])$$

Note: no indication is given on the dual-feasibility of the basis.

# Alternative approach

If the bounds are given for the current basis constraints:

$$A_{\mathcal{B}}\boldsymbol{x} \in [\boldsymbol{b}_{\mathcal{B}}],$$

then if $A_{\mathcal{B}}$ is invertible, the constraints represent of bounded parallelepiped. As such, it is always "dual-feasible" (even if $c^T A_{\mathcal{B}}^{-1}$ has negative coefficients, in which case we can use the lower bound).
If we can compute an interval $[\lambda_{\mathcal{B}}]$ which includes $\lambda_{\mathcal{B}}$:

1. $S \leq ub([\lambda_{\mathcal{B}}][\boldsymbol{b}_{\mathcal{B}}])$;
2. the signs of $[\lambda_{\mathcal{B}}]$ (if unique for each component) gives the dual-feasible vertex (otherwise, we get a set of potential dual-feasible vertices).

# Adaptating the simplex algorithm

Let's consider the case where all constraints have interval bounds:

$$A\boldsymbol{x} \in [\boldsymbol{b}]$$

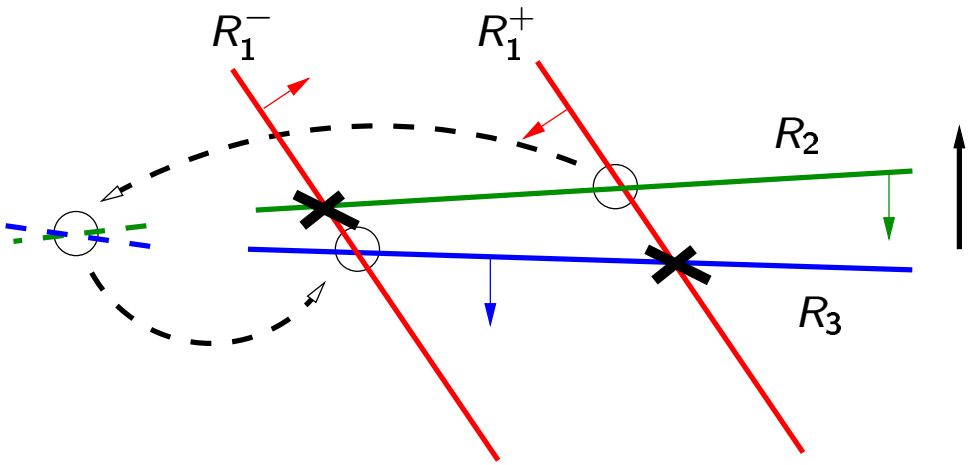
Considering potential bases, a constraint $R^T\boldsymbol{x} \in [b]$ may be:

- in the basis positively $R^T\boldsymbol{x} = \overline{b}$ or negatively $R^T\boldsymbol{x} = \underline{b}$;
- or out of the basis, unsatisfied (on one side) or satisfied (on both sides).

## Notes

1. $A_{\mathcal{B}}$ does not depend on the "sign" of the appartenance in the basis. This "sign" depends on the value of $\lambda$ and changes the selection of the bound ($\overline{b}$ or $\underline{b}$) when computing the "current" vertex.
2. "Switching" the side of a constraint never appears in the usual dual simplex algorithm, but is easy computationnally and could be useful...

# Skipping steps




Successive bases for usual algorithm:

$$(R_1^+, R_2) \rightarrow (R_3, R_2) \rightarrow (R_3, R_1^-)$$

Problem: $(R_3, R_2)$ is ill-conditioned.
Using another "path" to reach $(R_3, R_1^-)$ is problematic as $(R_1^-, R_2)$ and $(R_1^+, R_3)$ are not dual-feasible.
Can we go directly from $(R_1^+, R_2)$ to $(R_3, R_1^-)$? Considering the basis as a set of "active" constraints (positively or negatively), yes.

# Simplex revisited

At each iteration, we keep:

- the current basis $\mathcal{B}$, which represents a parallelepiped which includes the polyhedron; along with $\mathcal{B}$, we keep a approximation $[A_{\mathcal{B}}^{-1}]$ of $A_{\mathcal{B}}^{-1}$;
- the "active" bound $\underline{b}$ or $\overline{b}$ for each constraint in the basis (needed to check the optimality). We denote by $\hat{\boldsymbol{b}}_{\mathcal{B}}$ the vector of active bounds.

The choice of the active bounds must follow the signs of the components of $c^T[A_{\mathcal{B}}^{-1}]$. If both signs appear on a component, we may consider the sign of the midpoint.

# Optimality and emptiness checking

Optimality is reached when each non-basic constraint $R\boldsymbol{x} \in [r]$ is satisfied. Here:

$$[R[A_{\mathcal{B}}^{-1}]\hat{\boldsymbol{b}}_{\mathcal{B}}] \cap [r] \neq \emptyset$$

Emptiness is reached with a non-basis constraint is not satisfied over the paralleliped, i.e.:

$$[R[A_{\mathcal{B}}^{-1}][\boldsymbol{b}_{\mathcal{B}}]] \cap [r] = \emptyset$$

In the following, we consider that $R\boldsymbol{x} \leq \overline{r}$ is not satisfied, but the emptiness condition is not satisfied.

# Changing the basis

## Lemma

Let $P$ a parallelepiped, $H$ an half-space and $\boldsymbol{c}^T$ a linear form to maximize. Let $\boldsymbol{x}$ be an optimal vertex for $\boldsymbol{c}^T$ in $P$. Then if $\boldsymbol{x} \notin H$ and $P \cap H \neq \emptyset$, there exists an optimal vertex for $\boldsymbol{c}$ in $P \cap H$ which belongs to the boundary of $H$.

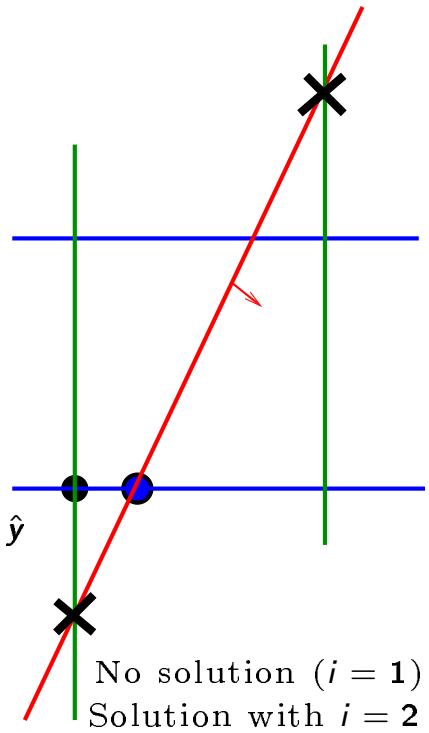
Hence, when a constraint is not satisfied, we look for a point intersecting the boundary of the constraint with an edge of the parallelepiped. This is given by:

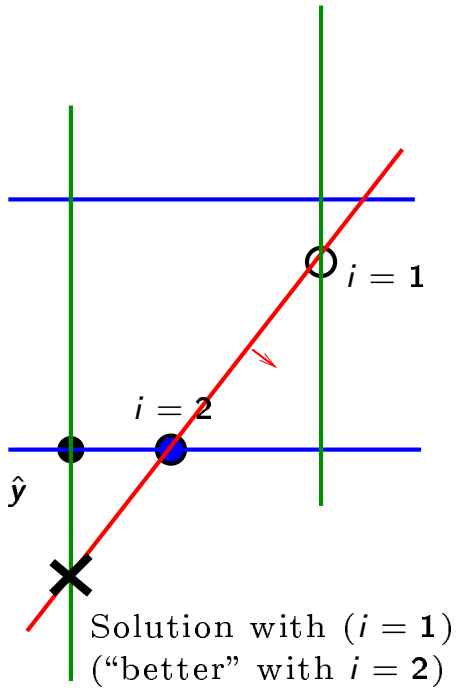
- a new basis (the new constraint enters, another leaves);
- the elements of the current basis which "switch" side.

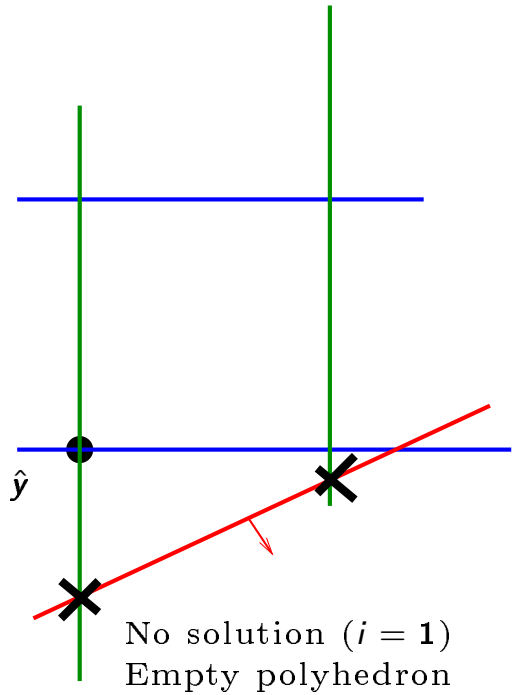
Like the classical simplex algorithm, each potentially leaving constraint is checked one by one.

# Leaving constraint (exact)

We note $R_{\mathcal{B}} = RA_{\mathcal{B}}^{-1}$, $\boldsymbol{c}_{\mathcal{B}}^T = \boldsymbol{c}^T A_{\mathcal{B}}^{-1}$ and $\boldsymbol{y} = A_{\mathcal{B}}\boldsymbol{x}$.
The new vertex $\boldsymbol{y}$ must satisfy $R_{\mathcal{B}}\boldsymbol{y} = \overline{r}$ (initially $R_{\mathcal{B}}\hat{\boldsymbol{y}} > \overline{r}$). We consider an index $i$ such that: $\hat{\boldsymbol{y}}^i = \overline{b_i}$:
If $R_{\mathcal{B}}^i \leq 0$, either there is no solution or a "better" index can be found.



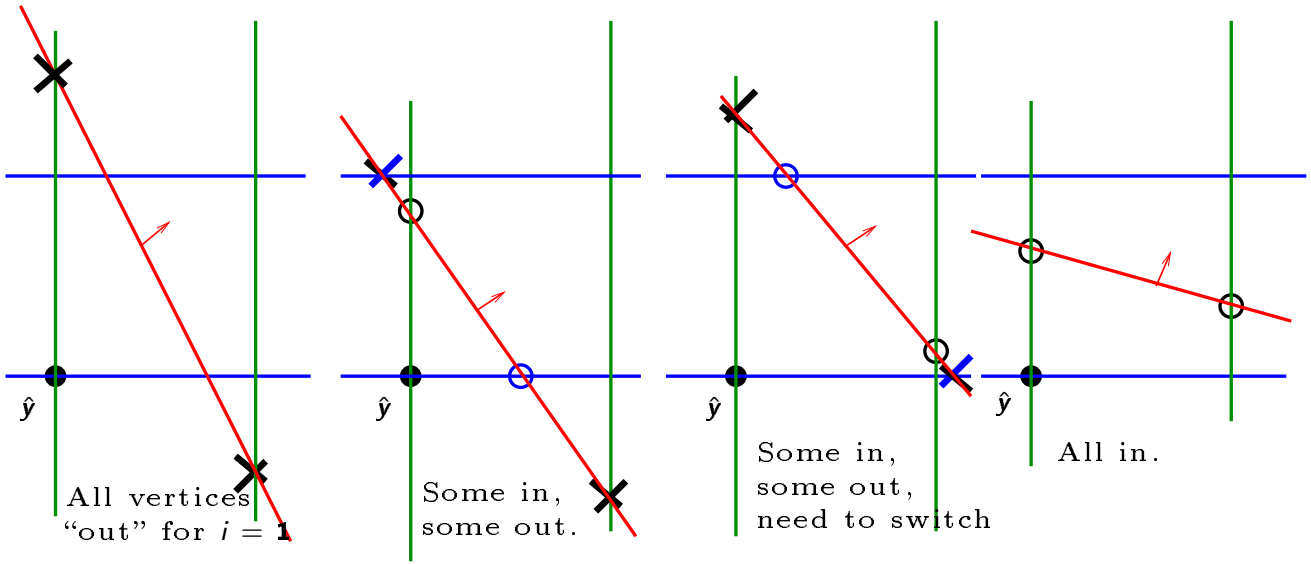
# Leaving contraint (exact) (2)

If $R^i_{\mathcal{B}} \geq 0$:

$$\boldsymbol{y}^i = \frac{\overline{r} - \sum_{j \neq i} R^j_{\mathcal{B}} \boldsymbol{y}^j}{R^i_{\mathcal{B}}}$$

We look for "new" values for $\boldsymbol{y}^j$ such that:

- the objective is maximal;
- the new vertex is still in the paralleliped.



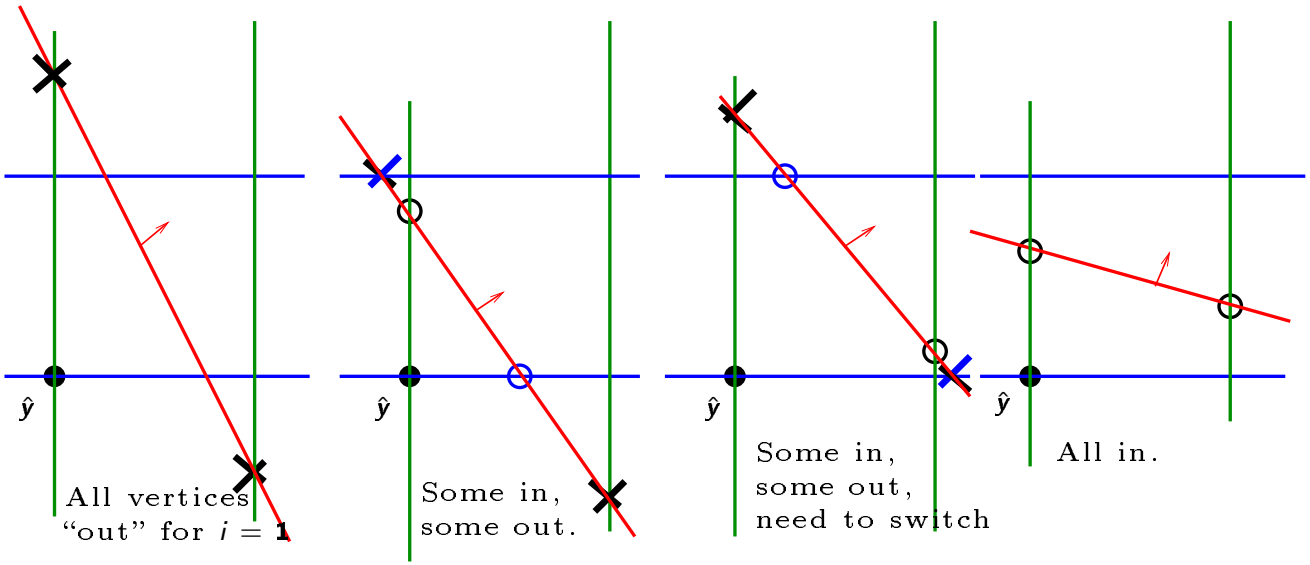
# Leaving contraint (exact) (3)

The expression of $\boldsymbol{c}_{\mathcal{B}}^T \boldsymbol{y}$ is:

$$\boldsymbol{c}_{\mathcal{B}}^T \boldsymbol{y} = \frac{\boldsymbol{c}_{\mathcal{B}}^i}{R_{\mathcal{B}}^i} \overline{r} + \sum_{j \neq i} \left( \boldsymbol{c}_{\mathcal{B}}^j - \frac{\boldsymbol{c}_{\mathcal{B}}^i R_{\mathcal{B}}^j}{R_{\mathcal{B}}^i} \right) \boldsymbol{y}^j$$

The sign of $\boldsymbol{c}_{\mathcal{B}}^j - \frac{\boldsymbol{c}_{\mathcal{B}}^i R_{\mathcal{B}}^j}{R_{\mathcal{B}}^i}$ gives us the new bound for $\boldsymbol{y}^j$ (either $\overline{b}_j$ or $\underline{b}_j$).

If there is only one optimal (strict signs), ok if $\boldsymbol{y}^i$ is still in $[b]_i$, otherwise check another indice. If several possibilities and $\boldsymbol{y}^i$ outside $[b]_i$, we can try to change it (case 3).



# Leaving constraint (interval)

The computation is similar, with some compromises:

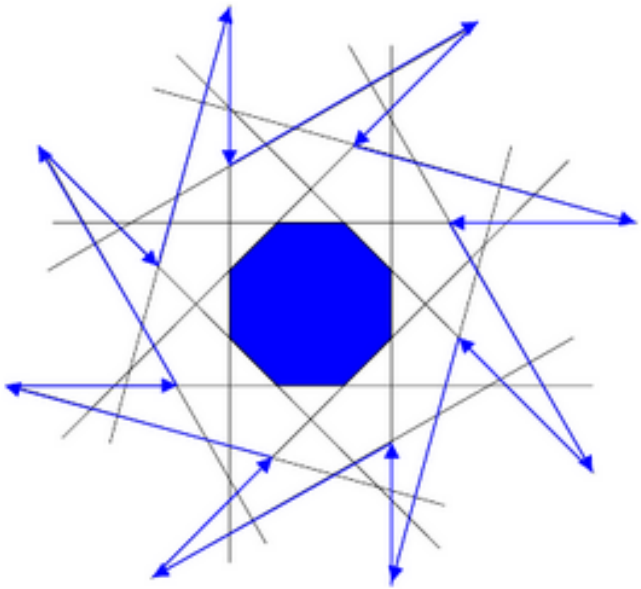
1. we consider $i$ such that $[R^i_{\mathcal{B}}] > 0$ ; if no such $i$ exists, the constraint cannot enter the basis (if no constraint can enter the basis, we stop there);
2. if the sign of $\boldsymbol{c}^j_{\mathcal{B}} - \frac{\boldsymbol{c}^i_{\mathcal{B}} R^j_{\mathcal{B}}}{R^i_{\mathcal{B}}}$ is ambiguous, we treat it as 0, but use the interval $[b_j]$ to bound $\boldsymbol{c}^T_{\mathcal{B}} \boldsymbol{y}$.

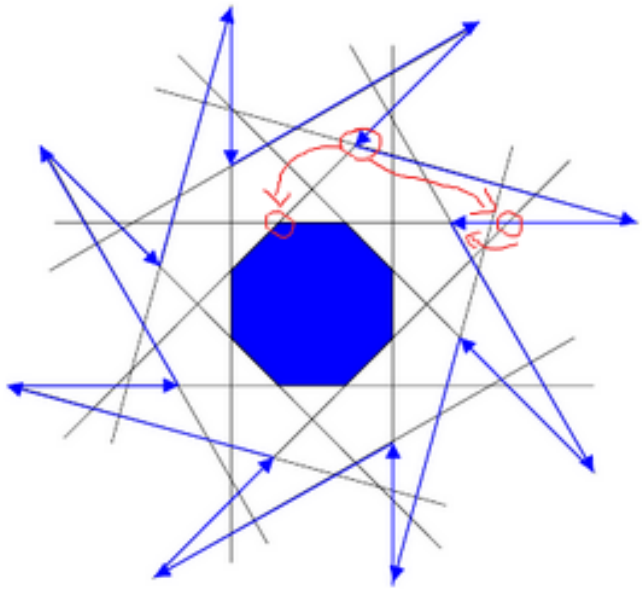
# Cycling

Known results:

- Simplex algorithm can cycle.
- (Inefficient but) simple rules (like Bland's rule) can avoir cycling.

We can adapt Bland's rule to our modified simplex algorithm. Until now, the proof that it cannot cycle is not found.





# Conclusion

Not implemented(!).