Polyhedral Algorithms
Part II: Optimization and Pivoting Algorithms

Jesús A. De Loera, UC Davis

June 24, 2011

How to compute the dimension?

- Recall: A polyhedron P has dimension k if the maximum number of affinely independent points in P is $k+1$ $(i.e.\dim(P) = k+1)$. How do we compute dimension?

How to compute the dimension?

- Recall: A polyhedron P has dimension k if the maximum number of affinely independent points in P is $k+1$ $(i.e.\dim(P)=k+1)$. How do we compute dimension?
- An inequality $\mathbf{a}'\mathbf{x} \geq \mathbf{b}'$ from $A\mathbf{x} \leq \mathbf{b}$ is an implicit equality if $\mathbf{a}'\mathbf{x} = \mathbf{b}'$ for all $\{\mathbf{x} \mid A\mathbf{x} \leq \mathbf{b}\}$.

How to compute the dimension?

- Recall: A polyhedron P has dimension k if the maximum number of affinely independent points in P is $k+1$ ($i.e. \dim(P) = k+1$). How do we compute dimension?
- An inequality $\mathbf{a}'\mathbf{x} \geq \mathbf{b}'$ from $A\mathbf{x} \leq \mathbf{b}$ is an implicit equality if $\mathbf{a}'\mathbf{x} = \mathbf{b}'$ for all $\{\mathbf{x} \mid A\mathbf{x} \leq \mathbf{b}\}$.
- **Theorem** If $P = \{\mathbf{x} \mid A\mathbf{x} \leq \mathbf{b}\} \neq \emptyset$ and $P \subset \mathbb{R}^n$ then $\dim(P) = n - \text{rank}(A^=, \mathbf{b}^=)$ where $A^=\mathbf{x} \leq \mathbf{b}^=$ is the set of implicit equalities.

How to compute the dimension?

- Recall: A polyhedron P has dimension k if the maximum number of affinely independent points in P is $k+1$ $(i.e.\dim(P) = k+1)$. How do we compute dimension?
- An inequality $\mathbf{a}'\mathbf{x} \geq \mathbf{b}'$ from $A\mathbf{x} \leq \mathbf{b}$ is an implicit equality if $\mathbf{a}'\mathbf{x} = \mathbf{b}'$ for all $\{\mathbf{x} \mid A\mathbf{x} \leq \mathbf{b}\}$.
- **Theorem** If $P = \{\mathbf{x} \mid A\mathbf{x} \leq \mathbf{b}\} \neq \emptyset$ and $P \subset \mathbb{R}^n$ then $\dim(P) = n - \text{rank}(A^=, \mathbf{b}^=)$ where $A^=\mathbf{x} \leq \mathbf{b}^=$ is the set of implicit equalities.

Example

What is the dimension of the following polytope?

(1) $x_1 + x_2 + x_3 \geq 2$

(2) $x_1 + x_2 \leq 1$

(3) $x_3 \leq 1$

(4) $x_1 \leq \frac{1}{2}$

(5) $x_1, x_2, x_3 \geq 0$

Example

What is the dimension of the following polytope?

(1) $x_1 + x_2 + x_3 \geq 2$

(2) $x_1 + x_2 \leq 1$

(3) $x_3 \leq 1$

(4) $x_1 \leq \frac{1}{2}$

(5) $x_1, x_2, x_3 \geq 0$

We have $x_1 + x_2 + x_3 = 2$ (one implied equality)
$\Rightarrow \dim(P) = n - \text{rank}(A^=, b^=)$.

How to compute FACES? Eliminate REDUNDACY, FACETS

- Given a polyhedron $P = \{\mathbf{x} \mid A\mathbf{x} \leq \mathbf{b}\}$ the inequality $\mathbf{a}'\mathbf{x} \leq \mathbf{b}'$ is valid for P if it is satisfied by all points in P.

How to compute FACES? Eliminate REDUNDACY, FACETS

- Given a polyhedron $P = \{\mathbf{x} \mid A\mathbf{x} \leq \mathbf{b}\}$ the inequality $\mathbf{a}'\mathbf{x} \leq \mathbf{b}'$ is valid for P if it is satisfied by all points in P.
- Let $\mathbf{v}'\mathbf{x} \leq \mathbf{g}$ be a valid inequality for P and let $F = \{\mathbf{x} \in P \mid \mathbf{v}'\mathbf{x} = \mathbf{g}\}$. Then F is a face of P. A face is proper is $F \neq \emptyset, P$.

How to compute FACES? Eliminate REDUNDACY, FACETS

- Given a polyhedron $P = \{\mathbf{x} \mid A\mathbf{x} \leq \mathbf{b}\}$ the inequality $\mathbf{a}'\mathbf{x} \leq \mathbf{b}'$ is valid for P if it is satisfied by all points in P.
- Let $\mathbf{v}'\mathbf{x} \leq \mathbf{g}$ be a valid inequality for P and let $F = \{\mathbf{x} \in P \mid \mathbf{v}'\mathbf{x} = \mathbf{g}\}$. Then F is a face of P. A face is proper is $F \neq \emptyset, P$.
- A face of P represented by $\mathbf{v}'\mathbf{x} \geq \mathbf{g}$ is a facet if $\dim(F) = \dim(P) - 1$. This is a facet defining inequality.

How to compute FACES? Eliminate REDUNDACY, FACETS

- Given a polyhedron $P = \{\mathbf{x} \mid A\mathbf{x} \leq \mathbf{b}\}$ the inequality $\mathbf{a}'\mathbf{x} \leq \mathbf{b}'$ is valid for P if it is satisfied by all points in P.
- Let $\mathbf{v}'\mathbf{x} \leq \mathbf{g}$ be a valid inequality for P and let $F = \{\mathbf{x} \in P \mid \mathbf{v}'\mathbf{x} = \mathbf{g}\}$. Then F is a face of P. A face is proper is $F \neq \emptyset, P$.
- A face of P represented by $\mathbf{v}'\mathbf{x} \geq \mathbf{g}$ is a facet if $\dim(F) = \dim(P) - 1$. This is a facet defining inequality.
- **Theorem** For each facet F of P, at least one inequality representing F is necessary in any description of P. If an inequality represents a face of smaller dimension that $\dim(P) - 1$, then it can be dropped (IRREDUNDANT SYSTEM).

How to compute FACES? LOW DIMENSION

- **Theorem** Let $P = \{x : Ax \leq b\}$. Then a nonempty subset F of P is a face of P if and only if F is represented as the set of solutions to an inequality system obtained from $Ax \leq b$ by setting some of the inequalities to equalities in an irredundant system of P.

How to compute FACES? LOW DIMENSION

- **Theorem** Let $P = \{x : Ax \leq b\}$. Then a nonempty subset F of P is a face of P if and only if F is represented as the set of solutions to an inequality system obtained from $Ax \leq b$ by setting some of the inequalities to equalities in an irredundant system of P.
- **Corollary** Every minimal nonempty face of P is an affine subspace of form $\{x : A_1 x = b_1\}$ where $A_1 x = b_1$ is a subsystem of $Ax = b$.

How to compute FACES? LOW DIMENSION

- **Theorem** Let $P = \{x : Ax \leq b\}$. Then a nonempty subset F of P is a face of P if and only if F is represented as the set of solutions to an inequality system obtained from $Ax \leq b$ by setting some of the inequalities to equalities in an irredundant system of P.
- **Corollary** Every minimal nonempty face of P is an affine subspace of form $\{x : A_1 x = b_1\}$ where $A_1 x = b_1$ is a subsystem of $Ax = b$.
- **Corollary** There are finitely many faces!

Optimization=Feasibility

- We have been looking at the problem: Is there a point x such that $Ax \leq b$?. This is the Feasibility problem.
- There is another problem, the Optimization problem: Maximize/Minimize linear functional $c^T x$ subject to $Ax \leq b$.

Optimization=Feasibility

- We have been looking at the problem: Is there a point x such that $Ax \leq b$?. This is the Feasibility problem.
- There is another problem, the Optimization problem: Maximize/Minimize linear functional $c^T x$ subject to $Ax \leq b$.
- Surprise: From the point of view theory if you know how to solve one problem, you know how to solve the other?

Optimization=Feasibility

- We have been looking at the problem: Is there a point x such that $Ax \leq b$?. This is the Feasibility problem.
- There is another problem, the Optimization problem: Maximize/Minimize linear functional $c^T x$ subject to $Ax \leq b$.
- Surprise: From the point of view theory if you know how to solve one problem, you know how to solve the other? Of course, in practice they may perform differently, but I do not have time to make the distinction!!
- Recall Farkas:$\{\mathbf{x} \mid A\mathbf{x} \leq \mathbf{b}\}$ is non-empty $\Leftrightarrow$ there is no solution $\{\mathbf{y} \mid \mathbf{y} \geq 0,\ \mathbf{y}^T A = 0,\ \mathbf{y}\mathbf{b} < 0\}$. It has **OPTIMIZATION VERSION TOO!!**

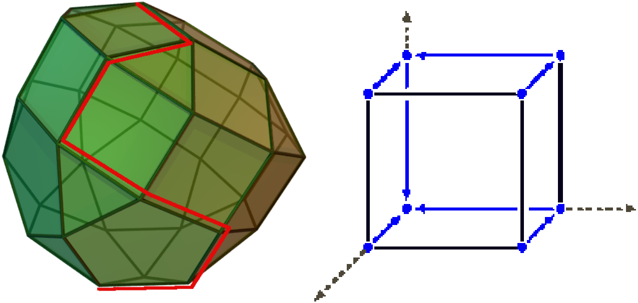
Optimization=Feasibility

- We have been looking at the problem: Is there a point x such that $Ax \leq b$?. This is the Feasibility problem.
- There is another problem, the Optimization problem: Maximize/Minimize linear functional $c^T x$ subject to $Ax \leq b$.
- Surprise: From the point of view theory if you know how to solve one problem, you know how to solve the other? Of course, in practice they may perform differently, but I do not have time to make the distinction!!
- Recall Farkas:$\{\mathbf{x} \mid A\mathbf{x} \leq \mathbf{b}\}$ is non-empty $\Leftrightarrow$ there is no solution $\{\mathbf{y} \mid \mathbf{y} \geq 0,\ \mathbf{y}^T A = 0,\ \mathbf{yb} < 0\}$. It has **OPTIMIZATION VERSION TOO!!**
- **Theorem**: If a finite optimum for $\max\{\mathbf{cx} \mid A\mathbf{x} \leq \mathbf{b}\}$ exists then $\min\{\mathbf{yb} \mid \mathbf{y} \geq 0,\ \mathbf{y}A = \mathbf{c}\}$ has a finite optimum too!!

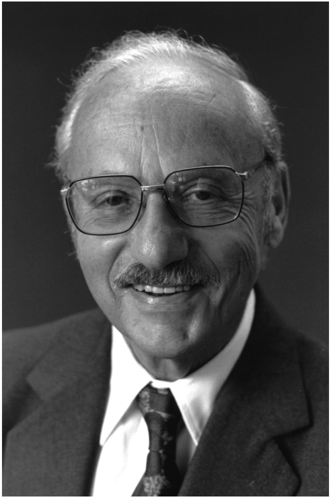
Optimization=Feasibility

- We have been looking at the problem: Is there a point x such that $Ax \leq b$?. This is the Feasibility problem.
- There is another problem, the Optimization problem: Maximize/Minimize linear functional $c^T x$ subject to $Ax \leq b$.
- Surprise: From the point of view theory if you know how to solve one problem, you know how to solve the other? Of course, in practice they may perform differently, but I do not have time to make the distinction!!
- Recall Farkas:$\{\mathbf{x} \mid A\mathbf{x} \leq \mathbf{b}\}$ is non-empty $\Leftrightarrow$ there is no solution $\{\mathbf{y} \mid \mathbf{y} \geq 0,\ \mathbf{y}^T A = 0,\ \mathbf{yb} < 0\}$. It has **OPTIMIZATION VERSION TOO!!**
- **Theorem**: If a finite optimum for $\max\{\mathbf{cx} \mid A\mathbf{x} \leq \mathbf{b}\}$ exists then $\min\{\mathbf{yb} \mid \mathbf{y} \geq 0,\ \mathbf{y}A = \mathbf{c}\}$ has a finite optimum too!! Optimum is at a vector $\mathbf{y} \geq 0$ whose positive components correspond to the linearly independent rows of A!!!

Basic Idea: Search or traverse the graph of a polytope OR a hyperplane arrangement by pivoting operations that move us from one vertex to the next. That way we can generate them all.



The Simplex Method



Is there any solution of $Ax \geq b$?

- We again want to solve the system of inequalities $Ax \geq b$. But we want to make it look more like what you are used to in linear algebra.

Is there any solution of $Ax \geq b$?

- We again want to solve the system of inequalities $Ax \geq b$. But we want to make it look more like what you are used to in linear algebra.
- **lemma** Given any system of inequalities Ax $\leq$ b, Cx $\geq$ d, then it can be transformed into a new system of the form

  $$D\bar{x} = f, \bar{x} \geq 0$$

  with the property that one system has a solution $\Leftrightarrow$ the other system has a solution.

Is there any solution of $Ax \geq b$?

- We again want to solve the system of inequalities $Ax \geq b$. But we want to make it look more like what you are used to in linear algebra.
- **lemma** Given any system of inequalities Ax $\leq$ b, Cx $\geq$ d, then it can be transformed into a new system of the form

$$D\bar{x} = f, \bar{x} \geq 0$$

with the property that one system has a solution $\Leftrightarrow$ the other system has a solution.
proof The inequality $\sum_{j=1}^{n} a_{ij}x_j \leq b_i$ can be turned into an equation:
Add the variable $s_i \rightarrow \sum_{j=1}^{n} a_{ij}x_j + s_i = b_i$ with $s_i \geq 0$

Similarly, $\sum_{j=1}^{n} c_{ij}x_j \geq d_i \rightarrow \sum_{j=1}^{n} c_{ij}x_j - t_i = d_i$
Finally, note that a variable x_i unrestricted can be replaced by

Example: From Inequalities to Equations

Solve the system of inequalities:

7x + 3y - 20z $\leq$ -2

4x - 3y + 9z $\leq$ 3

-x + 2y - z $\geq$ 4

11x - 2y + 2z $\geq$ 11

Using the previous lemma, we can now modify the system:

$$7x^+ - 7x^- + 3y^+ - 3y^- + 20z^+ - 20z^- + s_1 = -2$$
$$4x^+ - 4x^- - 3y^+ + 3y^- + 9z^+ - 9z^- + s_2 = 3$$
$$-x^+ + x^- + 2y^+ - 2y^- - z^+ + z^- - t_1 = 4$$
$$11x^+ - 11x^- - 2y^+ + 2y^- - 2z^+ + 2z^- - t_2 = 11$$

where $x^{\pm}, y^{\pm}, z^{\pm}, t_1, t_2, S_1, s_2 \geq 0$

but how can solve it???

The Simplex Method "Expresso" version

- We will use a simple easy-to-understand version of the SIMPLEX method.

The Simplex Method "Expresso" version

- We will use a simple easy-to-understand version of the SIMPLEX method.
- The key idea was introduced by Robert Bland (1970's) and developed in this form by Avis and Kaluzny.

The Simplex Method "Expresso" version

- We will use a simple easy-to-understand version of the SIMPLEX method.
- The key idea was introduced by Robert Bland (1970's) and developed in this form by Avis and Kaluzny.
- **Algorithm:** B-Rule Algorithm (aka Simplex method)

The Simplex Method "Expresso" version

- We will use a simple easy-to-understand version of the SIMPLEX method.
- The key idea was introduced by Robert Bland (1970's) and developed in this form by Avis and Kaluzny.
- **Algorithm:** B-Rule Algorithm (aka Simplex method)
- **input:** $A \in Q^{m \times n}$ of full row rank and $b \in Q^m$.
- **output:** Either a nonnegative vector x with $Ax = b$ or a vector y certifying infeasibility.
- **Step 1**: Find an invertible $m \times m$ submatrix B of A. Rewrite the system $Ax = b$ leaving the variables associated to B in the left
- **Step 2:** Set all the non-basic variables to zero. Find the *smallest index* of a basic variable with negative solution

Else, select the equation corresponding to that basic variable continue to Step 3.

- **Step 3:** Find the non-basic variable in the equation chosen in Step 2, that has smallest index and a positive coefficient.
- If there is none, then the problem is infeasible, stop!
  Else, solve this equation for the non-basic variable and substitute the result in all other equations.
  This variable becomes now basic, the former basic variable becomes non-basic. Go to Step 2.

NOTE: This last switch of variables is called a PIVOT.
NOTE: The simplex algorithm in general will have different PIVOT RULE to choose which variable leaves which variable enters the set of basic variables.

Example 1

Solve the next system for $x_i \geq 0$, $i = 1, 2, ..., 7$.

$$\begin{aligned}
2x_1 + x_2 + 3x_3 + x_4 + x_5 &= 8 \\
2x_1 + 3x_2 + 4x_4 + x_6 &= 12 \\
3x_1 + 2x_2 + 2x_3 + x_7 &= 18.
\end{aligned}$$

- **Step 1 of the *B*-Rule Algorithm:** find a basis in the matrix A,.
  We choose the easiest basis, which is given by the 5th, 6th and 7th columns of A.
  Denote the basis by $B = \{5, 6, 7\}$ and the set of the remaining vectors by $NB = \{1, 2, 3, 4\}$.
- Next we solve the equation $Ax = b$ for the basic variables $X_B = \{x_5, x_6, x_7\}$.

$$X_B = B^{-1}b - B^{-1}CX_{NB} = \begin{bmatrix} 8 \\ 12 \\ 18 \end{bmatrix} - \begin{bmatrix} 2 & 1 & 3 & 1 \\ 2 & 3 & 0 & 4 \\ 3 & 2 & 2 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

Example1 Continued

- **Step 2 in the B-Rule Algorithm:** Set all non-basic variables equal to zero. We obtain the non-negative values $x_5 = 8$, $x_6 = 12$, $x_7 = 18$.
- Therefore, this problem is feasible, and its solution is given by $x_1 = x_2 = x_3 = x_4 = 0$, $x_5 = 8$, $x_6 = 12$ and $x_7 = 18$.
- Suppose we choose a different basis from the above, say $B' = (1, 4, 7)$, and solve the problem keeping this election. It is not difficult to obtain solution $x_1 = 10/3$, $x_2 = x_3 = 0$, $x_4 = 10/3$, $x_5 = x_6 = 0$, $x_7 = 8$.
- We observe that the two solutions are completely different. In general, the solution always depends on the election of the basis.

Example 2

Next we solve the system $Ax = b$ for $x_i \geq 0$, $i = 1, 2, \ldots, 6.$, where A and b are given by

$$A = \begin{bmatrix} -1 & -2 & 1 & 1 & 0 & 0 \\ 1 & -3 & -1 & 0 & 1 & 0 \\ -1 & -2 & 2 & 0 & 0 & 1 \end{bmatrix} \qquad b = \begin{bmatrix} -1 \\ 2 \\ -2 \end{bmatrix}$$

- **Step 1** Choose a basis from A, say $B = \{4, 5, 6\}$. Next we solve the system for the basic variables $X_B = \{x_4, x_5, x_6\}$.

$$\begin{aligned} x_4 &= -1 + x_1 + 2x_2 - x_3 \\ x_5 &= 2 - x_1 + 3x_2 + x_3 \\ x_6 &= -2 + x_1 + 2x_2 - 2x_3 \end{aligned}$$

- **Step 2:** Setting all non-basic variables equal to zero, we get $x_4 = -1$, $x_5 = 2$ and $x_6 = -2$. Note that x_4 and x_6 are basic variables with negative solution. Choose that one with

Example 2, Continued

- Choose the equation that corresponds to x_4 in the equation above.
- Next, we must find the non-basic variable in the equation that has smallest index and a positive coefficient, in this case x_1 is such a variable.
- **Step 3** Solve the first equation for that non-basic variable, taking from now x_1 as basic variable and go back to Step 2 of the algorithm.

$$\begin{aligned}
x_1 &= 1 - 2x_2 + x_3 + x_4 && = 1 - 2x_2 + x_3 + x_4 \\
x_5 &= 2 - (1 - 2x_2 + x_3 + x_4) + 3x_2 + x_3 && = 1 + 5x_2 - x_4 \\
x_6 &= -2 + (1 - 2x_2 + x_3 + x_4) + 2x_2 - 2x_3 && = -1 - x_3 + x_4
\end{aligned}$$

- Set non-basic variables equal to zero, we obtain $x_1 = 1$, $x_5 = 1$ and $x_6 = -1$. We see x_6 has negative solution so we

Example 2, Continued

- The non-basic variable selected is x_4. Solve that equation for x_4 and rewrite the system as follow.

$$
\begin{aligned}
x_1 &= 1 - 2x_2 + x_3 + (1 + x_3 + x_6) &&= 2 - 2x_2 + 2x_3 + x_6 \\
x_4 &= 1 + x_3 + x_6 &&= 1 + x_3 + x_6 \\
x_5 &= 1 + 5x_2 - (1 + x_3 + x_6) &&= 0 + 5x_2 - x_3 - x_6
\end{aligned}
$$

- Back again to Step 2: now with x_1, x_4 and x_5 as basic variables, we set all non-basic variables equal to zero obtaining non-negative solutions for the basic variables.
- So we have found that one solution to the problem is $x1 = 2$, $x_2 = x_3 = 0$ $x_4 = 1$ and $x_5 = x_6 = 0$.

Example 3

Solve the system $Ax = b$ for $x_i \geq 0$, $i = 1, \ldots 6$, where A and b are given as follow.

$$A = \begin{bmatrix} -1 & 2 & 1 & 1 & 0 & 0 \\ 3 & -2 & 1 & 0 & 1 & 0 \\ -1 & -6 & 23 & 0 & 0 & 1 \end{bmatrix} \qquad b = \begin{bmatrix} 3 \\ -17 \\ 19 \end{bmatrix}$$

First choose a basis from A, say $B = \{4, 5, 6\}$, and solve the system for the basic variables x_4, x_5 and x_6.
We obtain...

Example 3, Continued

$$\begin{aligned} x_4 &= 3 + x_1 - 2x_2 - x_3 \\ x_5 &= -17 - 3x_1 + 2x_2 - x_3 \\ x_6 &= 19 + x_1 + 6x_2 + 23x_3 \end{aligned}$$

- Set all non-basic variables equal to zero. We obtain $x_4 = 3$, $x_5 = -17$ and $x_6 = 19$. Since x_5 has negative solution we have to find the non-basic variable in the second equation that has smallest index and a positive coefficient.
- The variable selected for the pivot is x_2.

Example 3, Continued

Solve that equation for x_2 and after substitute the variable x_5 by the variable x_2 in the basis.

$$\begin{aligned}
x_2 &= 17/2 + 3/2x_1 + 1/2x_3 + 1/2x_5 \\
x_4 &= 3 + x_1 - 2(17/2 + 3/2x_1 + 1/2x_3 + 1/2x_5) - x_3 \\
x_5 &= 19 + x_1 + 6(17/2 + 3/2x_1 + 1/2x_3 + 1/2x_5) + 23x_3
\end{aligned}$$

then

$$\begin{aligned}
x_2 &= 17/2 + 3/2x_1 + 1/2x_3 + 1/2x_5 \\
x_4 &= -14 - 2x_1 - 2x_3 - x_5 \\
x_5 &= 70 + 10x_1 + 26x_3 + 3x_5
\end{aligned}$$

- We are in step two again. Set all non-basic variables equal to zero. The only solution negative is $x_4 = -14$, so we must choose the corresponding equation to x_4.

The algorithm works!!

Note that by construction of the algorithm gives the desired answer IF the algorithm ever terminates!!!

Lemma If x_n is the last variable, during the B-rule iterations, x_n cannot enter the basic variables and then leave OR leave and then enter.

The algorithm works!!

Note that by construction of the algorithm gives the desired answer IF the algorithm ever terminates!!!

Lemma If x_n is the last variable, during the B-rule iterations, x_n cannot enter the basic variables and then leave OR leave and then enter.

(proof of lemma)

- When x_n is chosen to enter the basic variables among the equations of the dictionary one finds

$$x_i = b_i + \sum_{j \notin B} a_{ij} x_j + a_{in} x_n$$

  where $a_{ij} \leq 0$, $b_i < 0$ and $a_{in} > 0$.

The algorithm works!!

Note that by construction of the algorithm gives the desired answer IF the algorithm ever terminates!!!

Lemma If x_n is the last variable, during the B-rule iterations, x_n cannot enter the basic variables and then leave OR leave and then enter.

(proof of lemma)

- When x_n is chosen to enter the basic variables among the equations of the dictionary one finds

$$x_i = b_i + \sum_{j \notin B} a_{ij} x_j + a_{in} x_n$$

  where $a_{ij} \leq 0$, $b_i < 0$ and $a_{in} > 0$.
- Therefore, any solution of the whole system with $x_l \geq 0$ for $l \neq n$ must necessarily have $x_n > 0$, otherwise a contradiction occurs!

- When x_n is chosen to leave the basic variables we have

$$x_i = b'_i + \sum_{j \in N} a'_{ij} x_j \quad (i \in B \setminus \{n\})$$

$$x_n = b'_n + \sum_{j \in N} a'_{nj} x_j$$

- When x_n is chosen to leave the basic variables we have

$$x_i = b'_i + \sum_{j \in N} a'_{ij} x_j \quad (i \in B \setminus \{n\})$$

$$x_n = b'_n + \sum_{j \in N} a'_{nj} x_j$$

- This shows $b'_n < 0$ while $b'_i \geq 0$ for all others indices. Setting the non-basic variables to zero shows there is a solution with $x_1, \ldots, x_{n-1} \geq 0$ but $x_n < 0$.

- When x_n is chosen to leave the basic variables we have

$$x_i = b_i' + \sum_{j \in N} a_{ij}' x_j \quad (i \in B \setminus \{n\})$$

$$x_n = b_n' + \sum_{j \in N} a_{nj}' x_j$$

- This shows $b_n' < 0$ while $b_i' \geq 0$ for all others indices. Setting the non-basic variables to zero shows there is a solution with $x_1, \ldots, x_{n-1} \geq 0$ but $x_n < 0$.
- Thus leaving and entering or entering and leaving are incompatible situations.

Theorem The B-Rule Algorithm terminates
(proof of Theorem): By contradiction.

- Suppose there is a matrix A and a vector b for which the algorithm does not terminate. Let us assume that A is an example with smallest number of rows plus columns.

Theorem The B-Rule Algorithm terminates
(proof of Theorem): By contradiction.

- Suppose there is a matrix A and a vector b for which the algorithm does not terminate. Let us assume that A is an example with smallest number of rows plus columns.
- Since there is a finite number of bases, in fact no more than $\binom{n}{m}$, then if the algorithm does not terminate one can find a cycle of bases in the iterations. One starts at one basis B_1, then moves to $B_2, B_3, \dots,$ and after say k iterations one returns to B_1.

Theorem The B-Rule Algorithm terminates
(proof of Theorem): By contradiction.

- Suppose there is a matrix A and a vector b for which the algorithm does not terminate. Let us assume that A is an example with smallest number of rows plus columns.
- Since there is a finite number of bases, in fact no more than $\binom{n}{m}$, then if the algorithm does not terminate one can find a cycle of bases in the iterations. One starts at one basis B_1, then moves to $B_2, B_3, \ldots$, and after say k iterations one returns to B_1.
- By the lemma, during this cycle of bases, the last variable x_n is either in all B_i or in none of them.

Theorem The B-Rule Algorithm terminates
(proof of Theorem): By contradiction.

- Suppose there is a matrix A and a vector b for which the algorithm does not terminate. Let us assume that A is an example with smallest number of rows plus columns.
- Since there is a finite number of bases, in fact no more than $\binom{n}{m}$, then if the algorithm does not terminate one can find a cycle of bases in the iterations. One starts at one basis B_1, then moves to $B_2, B_3, \ldots,$ and after say k iterations one returns to B_1.
- By the lemma, during this cycle of bases, the last variable x_n is either in all B_i or in none of them.
- If x_n is a basic variable discard the associated equation, it did not affect the choice of variables entering or leaving the basis. We have a smaller counterexample (fewer rows!)

Theorem The B-Rule Algorithm terminates
(proof of Theorem): By contradiction.

- Suppose there is a matrix A and a vector b for which the algorithm does not terminate. Let us assume that A is an example with smallest number of rows plus columns.
- Since there is a finite number of bases, in fact no more than $\binom{n}{m}$, then if the algorithm does not terminate one can find a cycle of bases in the iterations. One starts at one basis B_1, then moves to $B_2, B_3, \ldots$, and after say k iterations one returns to B_1.
- By the lemma, during this cycle of bases, the last variable x_n is either in all B_i or in none of them.
- If x_n is a basic variable discard the associated equation, it did not affect the choice of variables entering or leaving the basis. We have a smaller counterexample (fewer rows!)
- If x_n is always non-basic then delete x_n. A counterexample

A word about Avis-Fukuda Reverse-Search

- This is a very general procedure to LIST combinatorial objects connected by PIVOTS (=directed edges) inside a graph.

A word about Avis-Fukuda Reverse-Search

- This is a very general procedure to LIST combinatorial objects connected by PIVOTS (=directed edges) inside a graph.
- Examples of such situations are the graphs of polytopes (with an objective function to orient the edges), hyperplane arrangements.

A word about Avis-Fukuda Reverse-Search

- This is a very general procedure to LIST combinatorial objects connected by PIVOTS (=directed edges) inside a graph.
- Examples of such situations are the graphs of polytopes (with an objective function to orient the edges), hyperplane arrangements.
- It is called the reverse-search method because it reverses the simplex method: It goes from the optimal vertex in unique paths toward all possible other vertices, recovering a spanning tree of the polytope.

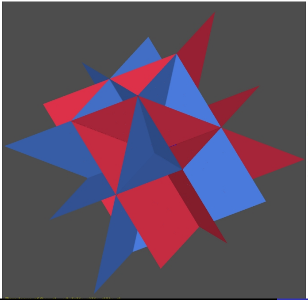
A word about Avis-Fukuda Reverse-Search

- This is a very general procedure to LIST combinatorial objects connected by PIVOTS (=directed edges) inside a graph.
- Examples of such situations are the graphs of polytopes (with an objective function to orient the edges), hyperplane arrangements.
- It is called the reverse-search method because it reverses the simplex method: It goes from the optimal vertex in unique paths toward all possible other vertices, recovering a spanning tree of the polytope.
- For simple polytopes reverse-search is an output-sensitive algorithm.

Application 1: Hyperplane arrangements

Central arrangements=Zonotopes

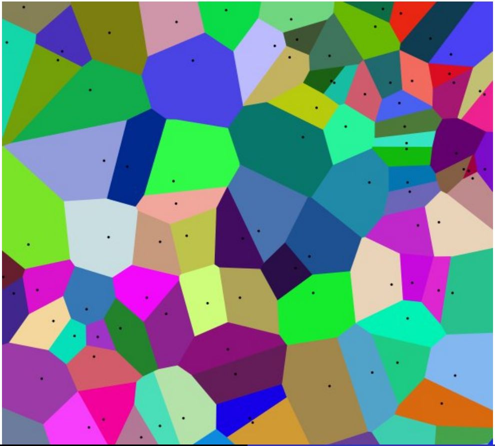
- Given a central arrangement of hyperplanes represented by a $d \times m$ matrix A, i.e, $h_i = x : A_i x = 0$ Look at the cut section of the arrangement with the unit $(d-1)$-sphere S^{d-1}
- Each hyperplane cuts a $(d-2)$-sphere. Each is an arrangement of spheres (or great-circles), giving a spherical polytope!!!



- **Theorem** The face lattice of this spherical polytope is the dual to the zonotope generated as the Minkowski sum of the line segments $[-A_i, A_i]$.

- **Theorem** The face lattice of this spherical polytope is the dual to the zonotope generated as the Minkowski sum of the line segments $[-A_i, A_i]$.
- **Theorem:** For a hyperplane arrangement with in $\mathbb{R}^d$ and m hyperplanes, there is an efficient implementation of Reverse Search that lists all the regions of an arrangement (equivalently the vertices of a zonotope) with time complexity $O(md(\#\textbf{regions}))$ and space complexity $O(md)$. Similarly the vertices of the arrangement can be listed efficiently

Application 2: Voronoi Diagrams



- Question: How to properly assign regions of vigilance to AAA repair shops or Fire stations??

- Question: How to properly assign regions of vigilance to AAA repair shops or Fire stations??
- Given a finite set of point in space $\mathbb{R}^d$. The Voronoi diagram of S is a decomposition of space into regions associated to each of the points $p \in S$:

$$\mathbf{near}(p) = \{x \in \mathbb{R}^d : dist(x,p) \leq dist(x,q) \text{ for all } q \in S - p\}$$

- Each region is a polyhedral cell, a Voronoi cell. Finding those cells has many applications.

- Question: How to properly assign regions of vigilance to AAA repair shops or Fire stations??
- Given a finite set of point in space $\mathbb{R}^d$. The Voronoi diagram of S is a decomposition of space into regions associated to each of the points $p \in S$:

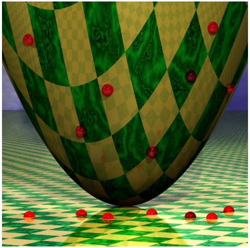
$$\mathbf{near}(p) = \{x \in \mathbb{R}^d : dist(x, p) \leq dist(x, q) \text{ for all } q \in S - p\}$$

- Each region is a polyhedral cell, a Voronoi cell. Finding those cells has many applications.
- IDEA: Voronoi cells can be computed are the projections of the facets of certain nice polytope!!!

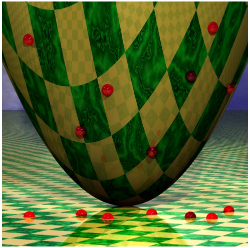
- Question: How to properly assign regions of vigilance to AAA repair shops or Fire stations??
- Given a finite set of point in space $\mathbb{R}^d$. The Voronoi diagram of S is a decomposition of space into regions associated to each of the points $p \in S$:

$$\mathbf{near}(p) = \{x \in \mathbb{R}^d : dist(x,p) \leq dist(x,q) \text{ for all } q \in S - p\}$$

- Each region is a polyhedral cell, a Voronoi cell. Finding those cells has many applications.
- IDEA: Voronoi cells can be computed are the projections of the facets of certain nice polytope!!!
- **HOW?** Lift the points of S to the paraboloid $x_{d+1} = x_1^2 + x_2^2 + \cdots + x_d^2$. The consider the polyhedron whose inequalities are precisely the tangent planes at the lifted points.



- Replace each equation with inequality $\geq$ for each $p \in S$ to obtain a polyhedron P_S given by inequalities $\sum_{j=1}^{d} p_j^2 - 2p_jx_j + x_{d+1} \geq 0$



- Replace each equation with inequality $\geq$ for each $p \in S$ to obtain a polyhedron P_S given by inequalities $\sum_{j=1}^{d} p_j^2 - 2p_jx_j + x_{d+1} \geq 0$
- **Theorem** The Voronoi diagram of S is the orthogonal projection of each facet of P_S back into the original space.

Thank you