Linear Programs

The Simplex Algorithm II

Bashar Dudin

January 3, 2017

EPITA

Where We Stand, What We Face

Working out a (still vague) procedure we've managed to find optimal solutions for simple examples of linear programs. Unfortunately, this heuristic is far from being satisfactory. Given a linear program, here is what we're missing :

- We have no way of testing if L is feasible, i.e. if it has at least a feasible solution
- Assuming L is feasible how can we tackle the case when the basic solution is not feasible?
- How can we check if L is unbounded?
- Does the procedure we worked out even terminate in general?
- If it does terminate with a finite objective value, is it an optimal one?

Where We Stand, What We Face

Working out a (still vague) procedure we've managed to find optimal solutions for simple examples of linear programs. Unfortunately, this heuristic is far from being satisfactory. Given a linear program, here is what we're missing :

- We have no way of testing if L is feasible, i.e. if it has at least a feasible solution
- Assuming L is feasible how can we tackle the case when the basic solution is not feasible?
- How can we check if L is unbounded?
- Does the procedure we worked out even terminate in general?
- If it does terminate with a finite objective value, is it an optimal one?

Pivoting

Pivoting

In order to answer both previous questions we'll need to properly write down involved algorithms. Given the linear program L in standard form

$$\begin{aligned} &\text{maximize} && z = v + \sum_{j=1}^{n} c_j x_j \\ &\text{subject to} \\ & && \forall i \in B, \quad \sum_{j=1}^{n} a_{ij} x_j \leq b_i \\ &\text{with} && \forall j \in N, \quad x_j \geq 0 \end{aligned}$$

we write

- $\boldsymbol{A}$ for the (m, n) matrix of coefficients $(a_{ij})_{i,j}$
- $\boldsymbol{b}$ for the m-tuple of b_is
- $\boldsymbol{c}$ for the n-tuple of c_js.

Pivoting

In order to answer both previous questions we'll need to properly write down involved algorithms. Given the linear program L in standard form

$$\begin{aligned} &\text{maximize} && z = v + \sum_{j=1}^{n} c_j x_j \\ &\text{subject to} \\ & && \forall\, i \in B, \quad \sum_{j=1}^{n} a_{ij} x_j \leq b_i \\ &\text{with} && \forall\, j \in N, \quad x_j \geq 0 \end{aligned}$$

The linear program L in its standard form is determined by the data $(\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c})$.

Pivoting

In order to answer both previous questions we'll need to properly write down involved algorithms. Given the linear program L in standard form

$$\begin{aligned} &\text{maximize} && z = v + \sum_{j=1}^{n} c_j x_j \\ &\text{subject to} \\ & && \forall i \in B, \quad x_i = b_i - \sum_{j=1}^{n} a_{ij} x_j \\ &\text{with} && \forall j \in N \cup B, \quad x_j \geq 0 \end{aligned}$$

The linear program L in its standard form is determined by the data $(\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c})$. In order to recover the corresponding slack form we usually include the data N, B of non-basic and basic sets as well as the constant v. Thus a slack form is assumed to be given by the data $(N, B, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, v)$.

Pivoting

We are given input $(N, B, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, v)$ and two indexes $e \in N$, $\ell \in B$ respectively corresponding to entering and leaving variables to the set of *basic* variables B. Denote $(\widehat{N}, \widehat{B}, \widehat{\boldsymbol{A}}, \widehat{\boldsymbol{b}}, \widehat{\boldsymbol{c}}, \widehat{v})$ expected output. It is a linear program equivalent to $(N, B, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, v)$.

Pivoting

We are given input $(N, B, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, v)$ and two indexes $e \in N$, $\ell \in B$ respectively corresponding to entering and leaving variables to the set of *basic* variables B. Denote $(\widehat{N}, \widehat{B}, \widehat{\boldsymbol{A}}, \widehat{\boldsymbol{b}}, \widehat{\boldsymbol{c}}, \widehat{v})$ expected output. It is a linear program equivalent to $(N, B, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, v)$.

- Express x_e in terms of other variables in equation ℓ

$$\widehat{b}_e = \frac{b_\ell}{a_{\ell e}} \quad \text{and} \quad \widehat{a}_{e\ell} = \frac{1}{a_{\ell e}}$$

for all $j \neq e$

$$\widehat{a}_{ej} = \frac{a_{\ell j}}{a_{\ell e}}$$

Pivoting

We are given input $(N, B, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, v)$ and two indexes $e \in N$, $\ell \in B$ respectively corresponding to entering and leaving variables to the set of *basic* variables B. Denote $(\widehat{N}, \widehat{B}, \widehat{\boldsymbol{A}}, \widehat{\boldsymbol{b}}, \widehat{\boldsymbol{c}}, \widehat{v})$ expected output. It is a linear program equivalent to $(N, B, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, v)$.

- Express x_e in terms of other variables in equation ℓ
- Replace x_e by previously obtained expression in linear constraints

For all $i \in B \setminus \{\ell\}$

$$\widehat{b}_i = b_i - a_{ie}\widehat{b}_e$$

$$\widehat{a}_{i\ell} = -a_{ie}\widehat{a}_{e\ell}$$

for all $j \neq e$

$$\widehat{a}_{ij} = a_{ij} - a_{ie}\widehat{a}_{ej}$$

Pivoting

We are given input $(N, B, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, v)$ and two indexes $e \in N$, $\ell \in B$ respectively corresponding to entering and leaving variables to the set of *basic* variables B. Denote $(\widehat{N}, \widehat{B}, \widehat{\boldsymbol{A}}, \widehat{\boldsymbol{b}}, \widehat{\boldsymbol{c}}, \widehat{v})$ expected output. It is a linear program equivalent to $(N, B, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, v)$.

- Express x_e in terms of other variables in equation ℓ
- Replace x_e by previously obtained expression in linear constraints
- Replace x_e by corresponding expression in the value function

$$\widehat{v} = v + c_e \widehat{b}_e \quad \text{and} \quad \widehat{c}_\ell = -c_e \widehat{a}_{e\ell}$$

for all $j \neq e$

$$\widehat{c}_j = c_j - c_e \widehat{a}_{ej}$$

Pivoting

We are given input $(N, B, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, v)$ and two indexes $e \in N$, $\ell \in B$ respectively corresponding to entering and leaving variables to the set of *basic* variables B. Denote $(\widehat{N}, \widehat{B}, \widehat{\boldsymbol{A}}, \widehat{\boldsymbol{b}}, \widehat{\boldsymbol{c}}, \widehat{v})$ expected output. It is a linear program equivalent to $(N, B, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, v)$.

- Express x_e in terms of other variables in equation ℓ
- Replace x_e by previously obtained expression in linear constraints
- Replace x_e by corresponding expression in the value function
- Update basic and none basic sets of variables.

$$\widehat{N} = \left(N \setminus e\right) \cup \{\ell\}$$

$$\widehat{B} = \left(B \setminus \ell\right) \cup \{e\}$$

Pivot Function

1: **function** PIVOT($N, B, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, v, e, l$)
2: let $\widehat{\boldsymbol{A}}$ be an (m, n) matrix
3: $\widehat{b}_e = \frac{b_\ell}{a_{\ell e}}$
4: **for all** $j \in N \setminus \{e\}$ **do**
5: $\widehat{a}_{ej} = \frac{a_{\ell j}}{a_{\ell e}}$
6: **end for**
7: $\widehat{a}_{e\ell} = \frac{1}{a_{\ell e}}$
8: **for all** $i \in B \setminus \{\ell\}$ **do**
9: $\widehat{b}_i = b_i - a_{ie}\widehat{b}_e$
10: **for all** $ji \in N \setminus \{e\}$ **do**
11: $\widehat{a}_{ij} = a_{ij} - a_{ie}\widehat{a}_{ej}$
12: **end for**
13: $\widehat{a_{i\ell}} = -a_{ie}\widehat{a}_{e\ell}$
14: **end for**
15: $\widehat{v} = v + c_e\widehat{b}_e$
16: **for all** $j \in N \setminus \{e\}$ **do**
17: $\widehat{c}_j = c_j - c_e\widehat{a}_{ej}$
18: **end for**
19: $\widehat{c}_\ell = -c_e\widehat{a}_{e\ell}$
20: $\widehat{N} = \left(N \setminus e\right) \cup \{\ell\}$
21: $\widehat{B} = \left(B \setminus \ell\right) \cup \{e\}$
22: **return** $(\widehat{N}, \widehat{B}, \widehat{\boldsymbol{A}}, \widehat{\boldsymbol{b}}, \widehat{\boldsymbol{c}}, \widehat{v})$
23: **end function**

Pivot Function

Remark : The Pivot Function has better not have any entries e, ℓ such that $a_{e\ell}$. We shall ensure this is never the case when Pivot is used.

The *basic solution* of the output $(\widehat{N}, \widehat{B}, \widehat{\boldsymbol{A}}, \widehat{\boldsymbol{b}}, \widehat{\boldsymbol{c}}, \widehat{v})$ is given by

$$\forall i \in \widehat{N}, \quad x_i = 0$$
$$\forall i \in \widehat{B}, \quad x_i = \widehat{b}_i$$

Replacing each $\widehat{b}_i$ by its corresponding expression in terms of coefficients of $(N, B, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, v)$, we get

$$\forall i \in \widehat{N}, \quad x_i = 0$$
$$\forall i \in \widehat{B} \setminus \{e\}, \quad x_i = b_i - a_{ie}\frac{b_\ell}{a_{\ell e}}$$
$$x_e = \frac{b_\ell}{a_{\ell e}}$$

Facing Unboundedness

Testing Boundedness

Let L be the following linear program in standard form

$$\begin{aligned}
&\text{maximize} && z = v + \sum_{j=1}^{n} c_j x_j \\
&\text{subject to} \\
& && \forall i \in B, \quad \sum_{j=1}^{n} a_{ij} x_j \leq b_i \\
&\text{with} && \forall j \in N, \quad x_j \geq 0
\end{aligned}$$

Testing Boundedness

Let L be the following linear program in standard form

$$\begin{aligned} &\text{maximize} && z = v + \sum_{j=1}^{n} c_j x_j \\ &\text{subject to} \\ & && \forall i \in B, \quad \sum_{j=1}^{n} a_{ij} x_j \leq b_i \\ &\text{with} && \forall j \in N, \quad x_j \geq 0 \end{aligned}$$

Fact

If there was an index j such that $c_j > 0$ and all coefficients of x_j in the linear constraints were non-positive then L is unbounded.

Testing Boundedness

Using the previous fact we can test, each time we call Pivot, that the input linear program doesn't satisfy the previous property :

> *There is an index j such that $c_j > 0$ and all coefficients of x_j in the linear constraints are non-positive.*

This is a necessary healthy check, but there is yet no garantee that this condition is fullfilled when the linear program we work with is unbounded. We will not have the needed machinery to properly answer this question untill we introduce some duality. For the time being we'll have to accept that the naive check at hand will be enough.

The Simplex Algorithm : Second Try

The Simplex Algorithm *Restricted*

Input: $(N, B, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, v)$ a linear program having feasible basic solution

Output: a basic feasible solution having *a priori* a maximal objetive value

function SIMPLEX($N, B, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, v$)
  Let ∂ and x be zero vectors of length n
  while $\exists$ index $j \in N$ such that $c_j > 0$ **do**
    choose $e \in N$ for which $c_e > 0$
    for all index $i \in B$ **do**
      if $a_{ie} > 0$ **then**
        $\partial_i = \frac{b_i}{a_{ie}}$
      else
        $\partial_i = \infty$
      end if
    end for
    choose $\ell \in B$ that minimizes ∂_i
    if $\partial_\ell = \infty$ **then**
      raise exception *Unbounded*
    else
      $(N, B, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, v) =$ PIVOT($N, B, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, v, e, \ell$)
    end if
  end while
  for all $i \in \{1, \ldots, n\}$ **do**
    if $i \in B$ **then**
      $x_i = b_i$
    end if
  end for
  return $(x_1, \ldots, x_n)$
end function

The Simplex Algorithm : First Validity Checks

There are three things we need to check in order to temporarily accept the previous version of the simplex algorithm

1. after each call to PIVOT the linear program we get has feasible basic solution
2. the objective value does not decrease while looping

and a a last point to *understand* :

3. what does it mean for SIMPLEX not to terminate? how can we deal with it?

The Simplex Algorithm : First Validity Checks

There are three things we need to check in order to temporarily accept the previous version of the simplex algorithm

1. after each call to PIVOT the linear program we get has feasible basic solution
2. the objective value does not decrease while looping **(We choose it that way!)**

and a a last point to *understand* :

3. what does it mean for SIMPLEX not to terminate? how can we deal with it?

The Simplex Algorithm : First Validity Checks

There are three things we need to check in order to temporarily accept the previous version of the simplex algorithm

1. after each call to PIVOT the linear program we get has feasible basic solution
2. the objective value does not decrease while looping **(We choose it that way!)**

and a a last point to *understand* :

3. what does it mean for SIMPLEX not to terminate? how can we deal with it?

Feasability of the basic solution after each call for PIVOT

Recall we are given as input the program $(N, B, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{c}, v)$ whose basic solution is feasible, i.e. all b_is are non-negative. We show that the output $(\widehat{N}, \widehat{B}, \widehat{\boldsymbol{A}}, \widehat{\boldsymbol{b}}, \widehat{\boldsymbol{c}}, \widehat{v})$ of PIVOT (with entering and leaving indexes e and ℓ) has feasible basic solution as well. Since we assume PIVOT is called, the coefficient $a_{\ell e} \geq 0$.

Feasability of the basic solution after each call for PIVOT

Recall we are given as input the program $(N, B, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{c}, v)$ whose basic solution is feasible, i.e. all b_is are non-negative. We show that the output $(\widehat{N}, \widehat{B}, \widehat{\boldsymbol{A}}, \widehat{\boldsymbol{b}}, \widehat{\boldsymbol{c}}, \widehat{v})$ of PIVOT (with entering and leaving indexes e and ℓ) has feasible basic solution as well. Since we assume PIVOT is called, the coefficient $a_{\ell e} \geq 0$.

We've already seen that

$$\widehat{b}_e = b_\ell / a_{\ell e}$$

and for all $i \in B \setminus \{\ell\}$

$$\widehat{b}_i = b_i - a_{ie}\widehat{b}_e.$$

Since b_ℓ and $a_{\ell e}$ are non-negative $\widehat{b}_e$ is non-negative as well. In case $a_{ie} \leq 0$, then $\widehat{b}_e$ is positive because all other involved quantities are positive.

Feasability of the basic solution after each call for PIVOT

Recall we are given as input the program $(N, B, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{c}, v)$ whose basic solution is feasible, i.e. all b_is are non-negative. We show that the output $(\widehat{N}, \widehat{B}, \widehat{\boldsymbol{A}}, \widehat{\boldsymbol{b}}, \widehat{\boldsymbol{c}}, \widehat{v})$ of PIVOT (with entering and leaving indexes e and ℓ) has feasible basic solution as well. Since we assume PIVOT is called, the coefficient $a_{\ell e} \geq 0$.

We've already seen that

$$\widehat{b}_e = b_\ell / a_{\ell e}$$

and for all $i \in B \setminus \{\ell\}$

$$\widehat{b}_i = b_i - a_{ie}\widehat{b}_e.$$

Since b_ℓ and $a_{\ell e}$ are non-negative $\widehat{b}_e$ is non-negative as well. In case $a_{ie} \leq 0$, then $\widehat{b}_e$ is positive because all other involved quantities are positive. When $a_{ie} > 0$ notice that SIMPLEX choses ℓ in such a way that, for all $i \in B$ with $a_{ie} > 0$,

$$b_\ell / a_{\ell e} \leq b_i / a_{ie}.$$

Thus, for all $i \in B$ with $a_{ie} > 0$,

$$\widehat{b}_i = b_i - a_{ie} b_\ell / a_{\ell e}$$

Feasability of the basic solution after each call for PIVOT

Recall we are given as input the program $(N, B, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{c}, v)$ whose basic solution is feasible, i.e. all b_is are non-negative. We show that the output $(\widehat{N}, \widehat{B}, \widehat{\boldsymbol{A}}, \widehat{\boldsymbol{b}}, \widehat{\boldsymbol{c}}, \widehat{v})$ of PIVOT (with entering and leaving indexes e and ℓ) has feasible basic solution as well. Since we assume PIVOT is called, the coefficient $a_{\ell e} \geq 0$.

We've already seen that

$$\widehat{b}_e = b_\ell / a_{\ell e}$$

and for all $i \in B \setminus \{\ell\}$

$$\widehat{b}_i = b_i - a_{ie}\widehat{b}_e.$$

Since b_ℓ and $a_{\ell e}$ are non-negative $\widehat{b}_e$ is non-negative as well. In case $a_{ie} \leq 0$, then $\widehat{b}_e$ is positive because all other involved quantities are positive. When $a_{ie} > 0$ notice that SIMPLEX choses ℓ in such a way that, for all $i \in B$ with $a_{ie} > 0$,

$$b_\ell / a_{\ell e} \leq b_i / a_{ie}.$$

Thus, for all $i \in B$ with $a_{ie} > 0$,

$$\widehat{b}_i \geq b_i - a_{ie} b_i / a_{ie} = 0$$

Cycling

Each time we step into the `while` loop of the SIMPLEX algorithm we either *increase (or keep constant)* the objective value or discover the linear program is *unbounded*. A priori, SIMPLEX might run on indefinitely among different slack forms without ever increasing the objective value. We are going to make clear that such behaviour can be either detected or avoided.

Proposition C

Let L be a linear program given by the data $(N, B, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, v)$ where $\boldsymbol{A}$ is an (m, n) matrix then if SIMPLEX runs more than $\binom{n+m}{m}$ iterations it does cycle, i.e. it goes indefinitely among the same finite set of slack forms with same given objective value.

Remark : This means that whenever SIMPLEX runs more than $\binom{n+m}{m}$ times then one can return the current objective value and basic solution.

Cycling

The proof of the above proposition is based on two facts :

- SIMPLEX is deterministic, if we ever get back to a previously obtained slack form then we are going to go back through all the slack forms starting with this one, once again.
- There is only a finite set of possible slack forms for the same given linear program.

The latter point is the only point we need to make clear.

Cycling

Lemma *B*

Let L be a linear program given in standard form by $(\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c})$. A slack form of L is determined by the choice of a set B of basic variables.

Cycling

Lemma *B*

Let L be a linear program given in standard form by $(\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c})$. A slack form of L is determined by the choice of a set B of basic variables.

Proof : Assume given two slack forms of L with same basic sets (and thus same non-basic set) : $(N, B, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, v)$ and $(N, B, \boldsymbol{A}', \boldsymbol{b}', \boldsymbol{c}', v')$.

Lemma *B*

Let L be a linear program given in standard form by $(\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c})$. A slack form of L is determined by the choice of a set B of basic variables.

Proof : Assume given two slack forms of L with same basic sets (and thus same non-basic set) : $(N, B, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, v)$ and $(N, B, \boldsymbol{A}', \boldsymbol{b}', \boldsymbol{c}', v')$. Forgetting about non-negativtiy constraints, substracting the linear constraints, we get the relations :

$$0 = (v - v') + \sum_{i \in N} (c_j - c'_j) x_j$$

and for each $i \in B$

$$0 = (b_i - b'_i) - \sum_{j \in N} (a_{ij} - a'_{ij}) x_j.$$

Lemma *B*

Let L be a linear program given in standard form by $(\boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c})$. A slack form of L is determined by the choice of a set B of basic variables.

Proof : Assume given two slack forms of L with same basic sets (and thus same non-basic set) : $(N, B, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, v)$ and $(N, B, \boldsymbol{A}', \boldsymbol{b}', \boldsymbol{c}', v')$. Forgetting about non-negativtiy constraints, substracting the linear constraints, we get the relations :

$$0 = (v - v') + \sum_{i \in N} (c_j - c'_j) x_j$$

and for each $i \in B$

$$0 = (b_i - b'_i) - \sum_{j \in N} (a_{ij} - a'_{ij}) x_j.$$

It is now a simple exercise to prove that for each $i \in B$ and $j \in N$, $v = v'$, $b_i = b'_i$, $c_j = c'_j$ and $a_{ij} = a'_{ij}$. ■

Cycling

Proposition C

Let L be a linear program given by the data $(N, B, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, v)$ where $\boldsymbol{A}$ is an (m, n) matrix then if SIMPLEX runs more than $\binom{n+m}{n}$ iterations it does cycle, i.e. it goes indefinitely among the same finite set of slack forms with same given objective value.

Proof : By lemma B, there are at most as many different slack forms as the number of possible choices of basic sets of variables. There are $\binom{m+n}{m}$ such choices. Thus if SIMPLEX runs more iterations than $\binom{m+n}{m}$ then we already obtained twice the same slack form. ■

Remark : Adding a counter to SIMPLEX insures SIMPLEX terminates.

In practice we use a different solution, called ***Bland's rule*** : each time we choose an index at lines 5 and 12 we choose the smallest possibile indices.

Simplex Algorithm *Restricted* | No Cycling version

Input: $(N, B, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{c}, v)$ a linear program having feasible basic solution

Output: a basic feasible solution having *a priori* a maximal objetive value

```
1:  function SIMPLEX(N, B, A, b, c, v)
2:      Let ∂ and x be zero vectors of length n
3:      while ∃ index j ∈ N such that c_j > 0 do
4:          choose smallest e ∈ N for which c_e > 0
5:          for all index i ∈ B do
6:              if a_ie > 0 then
7:                  ∂_i = b_i / a_ie
8:              else
9:                  ∂_i = ∞
10:             end if
11:         end for
12:         choose smallest ℓ ∈ B minimizing ∂_i
13:         if ∂_ℓ = ∞ then
14:             raise exception Unbounded
15:         else
16:             (N, B, A, b, c, v) =
        PIVOT(N, B, A, b, c, v, e, ℓ)
17:         end if
18:     end while
19:     for all i ∈ {1, ..., n} do
20:         if i ∈ B then
21:             x_i = b_i
22:         end if
23:     end for
24:     return (x_1, ..., x_n)
25: end function
```

That’s it for today !