

Part II

Linear Programming

Brewery Problem

Brewery brews ale and beer.

- ▶ Production limited by supply of corn, hops and barley malt
- ▶ Recipes for ale and beer require different amounts of resources

	<i>Corn (kg)</i>	<i>Hops (kg)</i>	<i>Malt (kg)</i>	<i>Profit (€)</i>
ale (barrel)	5	4	35	13
beer (barrel)	15	4	20	23
supply	480	160	1190	

Brewery Problem

Brewery brews ale and beer.

- ▶ Production limited by supply of corn, hops and barley malt
- ▶ Recipes for ale and beer require different amounts of resources

	<i>Corn (kg)</i>	<i>Hops (kg)</i>	<i>Malt (kg)</i>	<i>Profit (€)</i>
ale (barrel)	5	4	35	13
beer (barrel)	15	4	20	23
supply	480	160	1190	

Brewery Problem

Brewery brews ale and beer.

- ▶ Production limited by supply of corn, hops and barley malt
- ▶ Recipes for ale and beer require different amounts of resources

	<i>Corn (kg)</i>	<i>Hops (kg)</i>	<i>Malt (kg)</i>	<i>Profit (€)</i>
ale (barrel)	5	4	35	13
beer (barrel)	15	4	20	23
supply	480	160	1190	

Brewery Problem

	<i>Corn</i> <i>(kg)</i>	<i>Hops</i> <i>(kg)</i>	<i>Malt</i> <i>(kg)</i>	<i>Profit</i> <i>(€)</i>
ale (barrel)	5	4	35	13
beer (barrel)	15	4	20	23
supply	480	160	1190	

How can brewer maximize profits?

Brewery Problem

	<i>Corn</i> (kg)	<i>Hops</i> (kg)	<i>Malt</i> (kg)	<i>Profit</i> (€)
ale (barrel)	5	4	35	13
beer (barrel)	15	4	20	23
supply	480	160	1190	

How can brewer maximize profits?

- ▶ only brew ale: 34 barrels of ale $\Rightarrow 442 \text{ €}$
- ▶ only brew beer: 32 barrels of beer $\Rightarrow 736 \text{ €}$
- ▶ 7.5 barrels ale, 29.5 barrels beer $\Rightarrow 775 \text{ €}$
- ▶ 12 barrels ale, 28 barrels beer $\Rightarrow 800 \text{ €}$

Brewery Problem

	<i>Corn</i> (kg)	<i>Hops</i> (kg)	<i>Malt</i> (kg)	<i>Profit</i> (€)
ale (barrel)	5	4	35	13
beer (barrel)	15	4	20	23
supply	480	160	1190	

How can brewer maximize profits?

- ▶ only brew ale: 34 barrels of ale $\Rightarrow$ 442 €
- ▶ only brew beer: 32 barrels of beer $\Rightarrow$ 736 €
- ▶ 7.5 barrels ale, 29.5 barrels beer $\Rightarrow$ 775 €
- ▶ 12 barrels ale, 28 barrels beer $\Rightarrow$ 800 €

Brewery Problem

	<i>Corn</i> (kg)	<i>Hops</i> (kg)	<i>Malt</i> (kg)	<i>Profit</i> (€)
ale (barrel)	5	4	35	13
beer (barrel)	15	4	20	23
supply	480	160	1190	

How can brewer maximize profits?

- ▶ only brew ale: 34 barrels of ale $\Rightarrow$ 442 €
- ▶ only brew beer: 32 barrels of beer $\Rightarrow$ 736 €
- ▶ 7.5 barrels ale, 29.5 barrels beer $\Rightarrow$ 775 €
- ▶ 12 barrels ale, 28 barrels beer $\Rightarrow$ 800 €

Brewery Problem

	<i>Corn</i> (kg)	<i>Hops</i> (kg)	<i>Malt</i> (kg)	<i>Profit</i> (€)
ale (barrel)	5	4	35	13
beer (barrel)	15	4	20	23
supply	480	160	1190	

How can brewer maximize profits?

- ▶ only brew ale: 34 barrels of ale $\Rightarrow$ 442 €
- ▶ only brew beer: 32 barrels of beer $\Rightarrow$ 736 €
- ▶ 7.5 barrels ale, 29.5 barrels beer $\Rightarrow$ 775 €
- ▶ 12 barrels ale, 28 barrels beer $\Rightarrow$ 800 €

Brewery Problem

	<i>Corn</i> (kg)	<i>Hops</i> (kg)	<i>Malt</i> (kg)	<i>Profit</i> (€)
ale (barrel)	5	4	35	13
beer (barrel)	15	4	20	23
supply	480	160	1190	

How can brewer maximize profits?

- ▶ only brew ale: 34 barrels of ale $\Rightarrow$ 442 €
- ▶ only brew beer: 32 barrels of beer $\Rightarrow$ 736 €
- ▶ 7.5 barrels ale, 29.5 barrels beer $\Rightarrow$ 776 €
- ▶ 12 barrels ale, 28 barrels beer $\Rightarrow$ 800 €

Brewery Problem

	<i>Corn</i> (kg)	<i>Hops</i> (kg)	<i>Malt</i> (kg)	<i>Profit</i> (€)
ale (barrel)	5	4	35	13
beer (barrel)	15	4	20	23
supply	480	160	1190	

How can brewer maximize profits?

- ▶ only brew ale: 34 barrels of ale $\Rightarrow$ 442 €
- ▶ only brew beer: 32 barrels of beer $\Rightarrow$ 736 €
- ▶ 7.5 barrels ale, 29.5 barrels beer $\Rightarrow$ 776 €
- ▶ 12 barrels ale, 28 barrels beer $\Rightarrow$ 800 €

Brewery Problem

	<i>Corn</i> (kg)	<i>Hops</i> (kg)	<i>Malt</i> (kg)	<i>Profit</i> (€)
ale (barrel)	5	4	35	13
beer (barrel)	15	4	20	23
supply	480	160	1190	

How can brewer maximize profits?

- ▶ only brew ale: 34 barrels of ale $\Rightarrow$ 442 €
- ▶ only brew beer: 32 barrels of beer $\Rightarrow$ 736 €
- ▶ 7.5 barrels ale, 29.5 barrels beer $\Rightarrow$ 776 €
- ▶ 12 barrels ale, 28 barrels beer $\Rightarrow$ 800 €

Brewery Problem

	<i>Corn</i> (kg)	<i>Hops</i> (kg)	<i>Malt</i> (kg)	<i>Profit</i> (€)
ale (barrel)	5	4	35	13
beer (barrel)	15	4	20	23
supply	480	160	1190	

How can brewer maximize profits?

- ▶ only brew ale: 34 barrels of ale $\Rightarrow$ 442 €
- ▶ only brew beer: 32 barrels of beer $\Rightarrow$ 736 €
- ▶ 7.5 barrels ale, 29.5 barrels beer $\Rightarrow$ 776 €
- ▶ 12 barrels ale, 28 barrels beer $\Rightarrow$ 800 €

Brewery Problem

Linear Program

- Introduce variables a and b that define how much ale and stout to produce.
- Choose the variables in such a way that the profit is maximized.
- Make sure that no constraints (due to limited supply) are violated.

$$\begin{array}{ll}\max & 13a + 23b \\ \text{s.t.} & 5a + 15b \leq 480 \\ & 4a + 4b \leq 160 \\ & 35a + 20b \leq 1190 \\ & a, b \geq 0\end{array}$$

Brewery Problem

Linear Program

- ▶ Introduce variables a and b that define how much ale and beer to produce.
- ▶ Choose the variables in such a way that the objective function (profit) is maximized.
- ▶ Make sure that no constraints (due to limited supply) are violated.

$$\begin{array}{ll} \max & 13a + 23b \\ \text{s.t.} & 5a + 15b \leq 480 \\ & 4a + 4b \leq 160 \\ & 35a + 20b \leq 1190 \\ & a, b \geq 0 \end{array}$$

Brewery Problem

Linear Program

- ▶ Introduce variables a and b that define how much ale and beer to produce.
- ▶ Choose the variables in such a way that the objective function (profit) is maximized.
- ▶ Make sure that no constraints (due to limited supply) are violated.

$$\begin{array}{ll} \max & 13a + 23b \\ \text{s.t.} & 5a + 15b \leq 480 \\ & 4a + 4b \leq 160 \\ & 35a + 20b \leq 1190 \\ & a, b \geq 0 \end{array}$$

Brewery Problem

Linear Program

- ▶ Introduce variables a and b that define how much ale and beer to produce.
- ▶ Choose the variables in such a way that the objective function (profit) is maximized.
- ▶ Make sure that no constraints (due to limited supply) are violated.

$$\begin{array}{ll} \max & 13a + 23b \\ \text{s.t.} & 5a + 15b \leq 480 \\ & 4a + 4b \leq 160 \\ & 35a + 20b \leq 1190 \\ & a, b \geq 0 \end{array}$$

Brewery Problem

Linear Program

- ▶ Introduce variables a and b that define how much ale and beer to produce.
- ▶ Choose the variables in such a way that the objective function (profit) is maximized.
- ▶ Make sure that no constraints (due to limited supply) are violated.

$$\begin{array}{llll} \max & 13a & + & 23b \\ \text{s.t.} & 5a & + & 15b \leq 480 \\ & 4a & + & 4b \leq 160 \\ & 35a & + & 20b \leq 1190 \\ & & & a, b \geq 0 \end{array}$$

Standard Form LPs

LP in standard form:

maximize $c^T x$
subject to $Ax = b$
nonnegative variables x
minimize linear objective function subject to linear constraints

$$\begin{aligned} \max \quad & c_1 x_1 + c_2 x_2 + \dots + c_n x_n \\ \text{s.t.} \quad & a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1 \\ & a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n = b_2 \\ & \vdots \\ & a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n = b_m \\ & x_1 \geq 0, x_2 \geq 0, \dots, x_n \geq 0 \end{aligned}$$

$$\begin{aligned} \min \quad & c_1 x_1 + c_2 x_2 + \dots + c_n x_n \\ \text{s.t.} \quad & a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1 \\ & a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n = b_2 \\ & \vdots \\ & a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n = b_m \\ & x_1 \geq 0, x_2 \geq 0, \dots, x_n \geq 0 \end{aligned}$$

Standard Form LPs

LP in standard form:

- ▶ input: numbers a_{ij} , c_j , b_i
- ▶ output: numbers x_j
- ▶ n = #decision variables, m = #constraints
- ▶ maximize linear objective function subject to linear (in)equalities

$$\begin{aligned} \max \quad & \sum_{j=1}^n c_j x_j \\ \text{s.t.} \quad & \sum_{j=1}^n a_{ij} x_j = b_i \quad 1 \leq i \leq m \\ & x_j \geq 0 \quad 1 \leq j \leq n \end{aligned}$$

$$\begin{aligned} \max \quad & c^T x \\ \text{s.t.} \quad & Ax = b \\ & x \geq 0 \end{aligned}$$

Standard Form LPs

LP in standard form:

- ▶ input: numbers a_{ij} , c_j , b_i
- ▶ output: numbers x_j
- ▶ n = #decision variables, m = #constraints
- ▶ maximize linear objective function subject to linear (in)equalities

$$\begin{aligned} \max \quad & c_1 x_1 + c_2 x_2 + \dots + c_n x_n \\ \text{s.t.} \quad & a_{11} x_1 + a_{12} x_2 + \dots + a_{1n} x_n = b_1 \\ & a_{21} x_1 + a_{22} x_2 + \dots + a_{2n} x_n = b_2 \\ & \vdots \\ & a_{m1} x_1 + a_{m2} x_2 + \dots + a_{mn} x_n = b_m \\ & x_1, x_2, \dots, x_n \geq 0 \end{aligned}$$

$$\begin{aligned} \max \quad & c^T x \\ \text{s.t.} \quad & A x = b \\ & x \geq 0 \end{aligned}$$

Standard Form LPs

LP in standard form:

- ▶ input: numbers a_{ij} , c_j , b_i
- ▶ output: numbers x_j
- ▶ n = #decision variables, m = #constraints
- ▶ maximize linear objective function subject to linear (in)equalities

$$\begin{aligned} \max \quad & c_1 x_1 + c_2 x_2 + \dots + c_n x_n \\ \text{s.t.} \quad & a_{11} x_1 + a_{12} x_2 + \dots + a_{1n} x_n = b_1 \\ & a_{21} x_1 + a_{22} x_2 + \dots + a_{2n} x_n \leq b_2 \\ & \vdots \\ & a_{m1} x_1 + a_{m2} x_2 + \dots + a_{mn} x_n \leq b_m \\ & x_1, x_2, \dots, x_n \geq 0 \end{aligned}$$

$$\begin{aligned} \max \quad & c_1 x_1 + c_2 x_2 + \dots + c_n x_n \\ \text{s.t.} \quad & a_{11} x_1 + a_{12} x_2 + \dots + a_{1n} x_n = b_1 \\ & a_{21} x_1 + a_{22} x_2 + \dots + a_{2n} x_n \leq b_2 \\ & \vdots \\ & a_{m1} x_1 + a_{m2} x_2 + \dots + a_{mn} x_n \leq b_m \\ & x_1, x_2, \dots, x_n \geq 0 \end{aligned}$$

Standard Form LPs

LP in standard form:

- ▶ input: numbers a_{ij} , c_j , b_i
- ▶ output: numbers x_j
- ▶ n = #decision variables, m = #constraints
- ▶ maximize linear objective function subject to linear (in)equalities

$$\begin{array}{ll}\max & \sum_{j=1}^n c_j x_j \\ \text{s.t.} & \sum_{j=1}^n a_{ij} x_j = b_i \quad 1 \leq i \leq m \\ & x_j \geq 0 \quad 1 \leq j \leq n\end{array}$$

$$\begin{array}{ll}\max & c^T x \\ \text{s.t.} & Ax = b \\ & x \geq 0\end{array}$$

Standard Form LPs

LP in standard form:

- ▶ input: numbers a_{ij} , c_j , b_i
- ▶ output: numbers x_j
- ▶ n = #decision variables, m = #constraints
- ▶ maximize linear objective function subject to linear (in)equalities

$$\begin{array}{ll}\max & \sum_{j=1}^n c_j x_j \\ \text{s.t.} & \sum_{j=1}^n a_{ij} x_j = b_i \quad 1 \leq i \leq m \\ & x_j \geq 0 \quad 1 \leq j \leq n\end{array}$$

$$\begin{array}{ll}\max & c^T x \\ \text{s.t.} & Ax = b \\ & x \geq 0\end{array}$$

Standard Form LPs

LP in standard form:

- ▶ input: numbers a_{ij} , c_j , b_i
- ▶ output: numbers x_j
- ▶ n = #decision variables, m = #constraints
- ▶ maximize linear objective function subject to linear (in)equalities

$$\begin{array}{ll}\max & \sum_{j=1}^n c_j x_j \\ \text{s.t.} & \sum_{j=1}^n a_{ij} x_j = b_i \quad 1 \leq i \leq m \\ & x_j \geq 0 \quad 1 \leq j \leq n\end{array}$$

$$\begin{array}{ll}\max & c^T x \\ \text{s.t.} & Ax = b \\ & x \geq 0\end{array}$$

Standard Form LPs

Original LP

$$\begin{array}{ll}\max & 13a + 23b \\ \text{s.t.} & 5a + 15b \leq 480 \\ & 4a + 4b \leq 160 \\ & 35a + 20b \leq 1190 \\ & a, b \geq 0\end{array}$$

Standard Form

Add a **slack variable** to every constraint.

$$\begin{array}{llllllll}\max & 13a & + & 23b & & & & \\ \text{s.t.} & 5a & + & 15b & + & s_c & & = 480 \\ & 4a & + & 4b & & & + & s_h = 160 \\ & 35a & + & 20b & & & + & s_m = 1190 \\ & a & , & b & , & s_c & , & s_h , s_m \geq 0\end{array}$$

Standard Form LPs

Original LP

$$\begin{array}{ll}\max & 13a + 23b \\ \text{s.t.} & 5a + 15b \leq 480 \\ & 4a + 4b \leq 160 \\ & 35a + 20b \leq 1190 \\ & a, b \geq 0\end{array}$$

Standard Form

Add a **slack variable** to every constraint.

$$\begin{array}{llllllll}\max & 13a & + & 23b & & & & \\ \text{s.t.} & 5a & + & 15b & + & s_c & & = 480 \\ & 4a & + & 4b & & & + & s_h = 160 \\ & 35a & + & 20b & & & + & s_m = 1190 \\ & a & , & b & , & s_c & , & s_h , s_m \geq 0\end{array}$$

Standard Form LPs

There are different standard forms:

standard form

$$\begin{array}{ll}\max & c^T x \\ \text{s.t.} & Ax = b \\ & x \geq 0\end{array}$$

$$\begin{array}{ll}\min & c^T x \\ \text{s.t.} & Ax = b \\ & x \geq 0\end{array}$$

standard
maximization form

$$\begin{array}{ll}\max & c^T x \\ \text{s.t.} & Ax \leq b \\ & x \geq 0\end{array}$$

standard
minimization form

$$\begin{array}{ll}\min & c^T x \\ \text{s.t.} & Ax \geq b \\ & x \geq 0\end{array}$$

Standard Form LPs

There are different standard forms:

standard form

$$\begin{array}{ll}\max & c^T x \\ \text{s.t.} & Ax = b \\ & x \geq 0\end{array}$$

$$\begin{array}{ll}\min & c^T x \\ \text{s.t.} & Ax = b \\ & x \geq 0\end{array}$$

standard
maximization form

$$\begin{array}{ll}\max & c^T x \\ \text{s.t.} & Ax \leq b \\ & x \geq 0\end{array}$$

standard
minimization form

$$\begin{array}{ll}\min & c^T x \\ \text{s.t.} & Ax \geq b \\ & x \geq 0\end{array}$$

Standard Form LPs

There are different standard forms:

standard form

$$\begin{array}{ll}\max & c^T x \\ \text{s.t.} & Ax = b \\ & x \geq 0\end{array}$$

$$\begin{array}{ll}\min & c^T x \\ \text{s.t.} & Ax = b \\ & x \geq 0\end{array}$$

standard
maximization form

$$\begin{array}{ll}\max & c^T x \\ \text{s.t.} & Ax \leq b \\ & x \geq 0\end{array}$$

standard
minimization form

$$\begin{array}{ll}\min & c^T x \\ \text{s.t.} & Ax \geq b \\ & x \geq 0\end{array}$$

Standard Form LPs

There are different standard forms:

standard form

$$\begin{array}{ll}\max & c^T x \\ \text{s.t.} & Ax = b \\ & x \geq 0\end{array}$$

$$\begin{array}{ll}\min & c^T x \\ \text{s.t.} & Ax = b \\ & x \geq 0\end{array}$$

standard
maximization form

$$\begin{array}{ll}\max & c^T x \\ \text{s.t.} & Ax \leq b \\ & x \geq 0\end{array}$$

standard
minimization form

$$\begin{array}{ll}\min & c^T x \\ \text{s.t.} & Ax \geq b \\ & x \geq 0\end{array}$$

Standard Form LPs

It is easy to transform variants of LPs into (any) standard form:

1. Maximize z

2. Minimize z

3. No objective function

4. No constraints

5. Equality constraints

6. Inequality constraints

7. Free variables

8. Nonlinear constraints

9. Nonlinear objective function

10. Nonlinear constraints and objective function

Standard Form LPs

It is easy to transform variants of LPs into (any) standard form:

- ▶ **less or equal to equality:**

$$a - 3b + 5c \leq 12 \Rightarrow \begin{aligned} a - 3b + 5c + s &= 12 \\ s &\geq 0 \end{aligned}$$

- ▶ greater or equal to equality:

$$\begin{aligned} a - 3b + 5c - s &= 12 \\ s &\geq 0 \end{aligned}$$

- ▶ min to max:

$$\min z \Rightarrow \max -z$$

Standard Form LPs

It is easy to transform variants of LPs into (any) standard form:

- ▶ **less or equal to equality:**

$$a - 3b + 5c \leq 12 \Rightarrow \begin{aligned} a - 3b + 5c + s &= 12 \\ s &\geq 0 \end{aligned}$$

- ▶ greater or equal to equality:

- ▶ min to max:

Standard Form LPs

It is easy to transform variants of LPs into (any) standard form:

- ▶ **less or equal to equality:**

$$a - 3b + 5c \leq 12 \Rightarrow \begin{array}{l} a - 3b + 5c + s = 12 \\ s \geq 0 \end{array}$$

- ▶ **greater or equal to equality:**

$$a - 3b + 5c \geq 12 \Rightarrow \begin{array}{l} a - 3b + 5c - s = 12 \\ s \geq 0 \end{array}$$

- ▶ **min to max:**

Standard Form LPs

It is easy to transform variants of LPs into (any) standard form:

- ▶ **less or equal to equality:**

$$a - 3b + 5c \leq 12 \Rightarrow \begin{array}{l} a - 3b + 5c + s = 12 \\ s \geq 0 \end{array}$$

- ▶ **greater or equal to equality:**

$$a - 3b + 5c \geq 12 \Rightarrow \begin{array}{l} a - 3b + 5c - s = 12 \\ s \geq 0 \end{array}$$

- ▶ **min to max:**

Standard Form LPs

It is easy to transform variants of LPs into (any) standard form:

- ▶ **less or equal to equality:**

$$a - 3b + 5c \leq 12 \Rightarrow \begin{aligned} a - 3b + 5c + s &= 12 \\ s &\geq 0 \end{aligned}$$

- ▶ **greater or equal to equality:**

$$a - 3b + 5c \geq 12 \Rightarrow \begin{aligned} a - 3b + 5c - s &= 12 \\ s &\geq 0 \end{aligned}$$

- ▶ **min to max:**

$$\min a - 3b + 5c \Rightarrow \max -a + 3b - 5c$$

Standard Form LPs

It is easy to transform variants of LPs into (any) standard form:

- ▶ **less or equal to equality:**

$$a - 3b + 5c \leq 12 \Rightarrow \begin{array}{l} a - 3b + 5c + s = 12 \\ s \geq 0 \end{array}$$

- ▶ **greater or equal to equality:**

$$a - 3b + 5c \geq 12 \Rightarrow \begin{array}{l} a - 3b + 5c - s = 12 \\ s \geq 0 \end{array}$$

- ▶ **min to max:**

$$\min a - 3b + 5c \Rightarrow \max -a + 3b - 5c$$

Standard Form LPs

It is easy to transform variants of LPs into (any) standard form:

- ▶ **equality to less or equal:**

$$a - 3b + 5c = 12 \Rightarrow \begin{array}{l} a - 3b + 5c \leq 12 \\ -a + 3b - 5c \leq -12 \end{array}$$

- ▶ equality to greater or equal:

- ▶ unrestricted to nonnegative:

Standard Form LPs

It is easy to transform variants of LPs into (any) standard form:

- ▶ **equality to less or equal:**

$$a - 3b + 5c = 12 \Rightarrow \begin{array}{l} a - 3b + 5c \leq 12 \\ -a + 3b - 5c \leq -12 \end{array}$$

- ▶ equality to greater or equal:

- ▶ unrestricted to nonnegative:

Standard Form LPs

It is easy to transform variants of LPs into (any) standard form:

► **equality to less or equal:**

$$a - 3b + 5c = 12 \Rightarrow \begin{aligned} a - 3b + 5c &\leq 12 \\ -a + 3b - 5c &\leq -12 \end{aligned}$$

► **equality to greater or equal:**

$$a - 3b + 5c = 12 \Rightarrow \begin{aligned} a - 3b + 5c &\geq 12 \\ -a + 3b - 5c &\geq -12 \end{aligned}$$

► **unrestricted to nonnegative:**

Standard Form LPs

It is easy to transform variants of LPs into (any) standard form:

► **equality to less or equal:**

$$a - 3b + 5c = 12 \Rightarrow \begin{aligned} a - 3b + 5c &\leq 12 \\ -a + 3b - 5c &\leq -12 \end{aligned}$$

► **equality to greater or equal:**

$$a - 3b + 5c = 12 \Rightarrow \begin{aligned} a - 3b + 5c &\geq 12 \\ -a + 3b - 5c &\geq -12 \end{aligned}$$

► **unrestricted to nonnegative:**

Standard Form LPs

It is easy to transform variants of LPs into (any) standard form:

► **equality to less or equal:**

$$a - 3b + 5c = 12 \Rightarrow \begin{aligned} a - 3b + 5c &\leq 12 \\ -a + 3b - 5c &\leq -12 \end{aligned}$$

► **equality to greater or equal:**

$$a - 3b + 5c = 12 \Rightarrow \begin{aligned} a - 3b + 5c &\geq 12 \\ -a + 3b - 5c &\geq -12 \end{aligned}$$

► **unrestricted to nonnegative:**

$$x \text{ unrestricted} \Rightarrow x = x^+ - x^-, x^+ \geq 0, x^- \geq 0$$

Standard Form LPs

It is easy to transform variants of LPs into (any) standard form:

► **equality to less or equal:**

$$a - 3b + 5c = 12 \Rightarrow \begin{aligned} a - 3b + 5c &\leq 12 \\ -a + 3b - 5c &\leq -12 \end{aligned}$$

► **equality to greater or equal:**

$$a - 3b + 5c = 12 \Rightarrow \begin{aligned} a - 3b + 5c &\geq 12 \\ -a + 3b - 5c &\geq -12 \end{aligned}$$

► **unrestricted to nonnegative:**

$$x \text{ unrestricted} \Rightarrow x = x^+ - x^-, x^+ \geq 0, x^- \geq 0$$

Observations:

- ▶ a linear program does not contain x^2 , $\cos(x)$, etc.
- ▶ transformations between standard forms can be done efficiently and only change the size of the LP by a small constant factor
- ▶ for the standard minimization or maximization LPs we could include the nonnegativity constraints into the set of ordinary constraints; this is of course not possible for the standard form

Observations:

- ▶ a linear program does not contain x^2 , $\cos(x)$, etc.
- ▶ transformations between standard forms can be done efficiently and only change the size of the LP by a small constant factor
- ▶ for the standard minimization or maximization LPs we could include the nonnegativity constraints into the set of ordinary constraints; this is of course not possible for the standard form

Observations:

- ▶ a linear program does not contain x^2 , $\cos(x)$, etc.
- ▶ transformations between standard forms can be done efficiently and only change the size of the LP by a small constant factor
- ▶ for the standard minimization or maximization LPs we could include the nonnegativity constraints into the set of ordinary constraints; this is of course not possible for the standard form

Fundamental Questions

Definition 1 (Linear Programming Problem (LP))

Let $A \in \mathbb{Q}^{m \times n}$, $b \in \mathbb{Q}^m$, $c \in \mathbb{Q}^n$, $\alpha \in \mathbb{Q}$. Does there exist $x \in \mathbb{Q}^n$
s.t. $Ax = b$, $x \geq 0$, $c^T x \geq \alpha$?

Questions:

1. Is LP in NP?

2. Is LP in co-NP?

3. Is LP in P?

Input size:

- ▶ n number of variables, m constraints, L number of bits to encode the input

Fundamental Questions

Definition 1 (Linear Programming Problem (LP))

Let $A \in \mathbb{Q}^{m \times n}$, $b \in \mathbb{Q}^m$, $c \in \mathbb{Q}^n$, $\alpha \in \mathbb{Q}$. Does there exist $x \in \mathbb{Q}^n$
s.t. $Ax = b$, $x \geq 0$, $c^T x \geq \alpha$?

Questions:

Input size:

- ▶ n number of variables, m constraints, L number of bits to encode the input

Fundamental Questions

Definition 1 (Linear Programming Problem (LP))

Let $A \in \mathbb{Q}^{m \times n}$, $b \in \mathbb{Q}^m$, $c \in \mathbb{Q}^n$, $\alpha \in \mathbb{Q}$. Does there exist $x \in \mathbb{Q}^n$
s.t. $Ax = b$, $x \geq 0$, $c^T x \geq \alpha$?

Questions:

- ▶ Is LP in NP?
- ▶ Is LP in co-NP?
- ▶ Is LP in P?

Input size:

- ▶ n number of variables, m constraints, L number of bits to encode the input

Fundamental Questions

Definition 1 (Linear Programming Problem (LP))

Let $A \in \mathbb{Q}^{m \times n}$, $b \in \mathbb{Q}^m$, $c \in \mathbb{Q}^n$, $\alpha \in \mathbb{Q}$. Does there exist $x \in \mathbb{Q}^n$
s.t. $Ax = b$, $x \geq 0$, $c^T x \geq \alpha$?

Questions:

- ▶ Is LP in NP?
- ▶ Is LP in co-NP?
- ▶ Is LP in P?

Input size:

- ▶ n number of variables, m constraints, L number of bits to encode the input

Fundamental Questions

Definition 1 (Linear Programming Problem (LP))

Let $A \in \mathbb{Q}^{m \times n}$, $b \in \mathbb{Q}^m$, $c \in \mathbb{Q}^n$, $\alpha \in \mathbb{Q}$. Does there exist $x \in \mathbb{Q}^n$
s.t. $Ax = b$, $x \geq 0$, $c^T x \geq \alpha$?

Questions:

- ▶ Is LP in NP?
- ▶ Is LP in co-NP?
- ▶ Is LP in P?

Input size:

- ▶ n number of variables, m constraints, L number of bits to encode the input

Fundamental Questions

Definition 1 (Linear Programming Problem (LP))

Let $A \in \mathbb{Q}^{m \times n}$, $b \in \mathbb{Q}^m$, $c \in \mathbb{Q}^n$, $\alpha \in \mathbb{Q}$. Does there exist $x \in \mathbb{Q}^n$
s.t. $Ax = b$, $x \geq 0$, $c^T x \geq \alpha$?

Questions:

- ▶ Is LP in NP?
- ▶ Is LP in co-NP?
- ▶ Is LP in P?

Input size:

- ▶ n number of variables, m constraints, L number of bits to encode the input

Fundamental Questions

Definition 1 (Linear Programming Problem (LP))

Let $A \in \mathbb{Q}^{m \times n}$, $b \in \mathbb{Q}^m$, $c \in \mathbb{Q}^n$, $\alpha \in \mathbb{Q}$. Does there exist $x \in \mathbb{Q}^n$ s.t. $Ax = b$, $x \geq 0$, $c^T x \geq \alpha$?

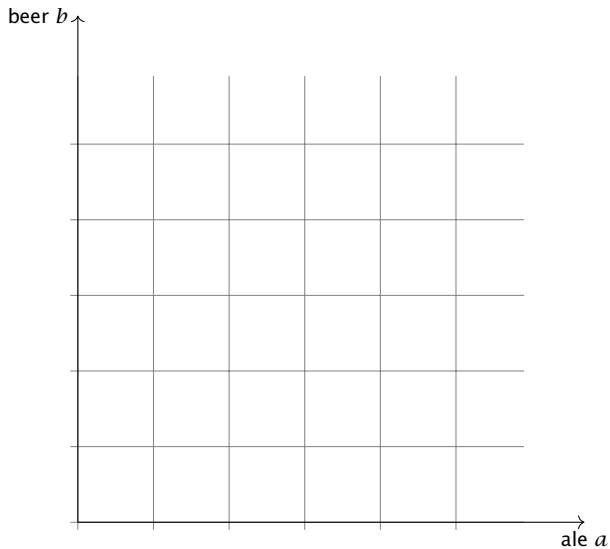
Questions:

- ▶ Is LP in NP?
- ▶ Is LP in co-NP?
- ▶ Is LP in P?

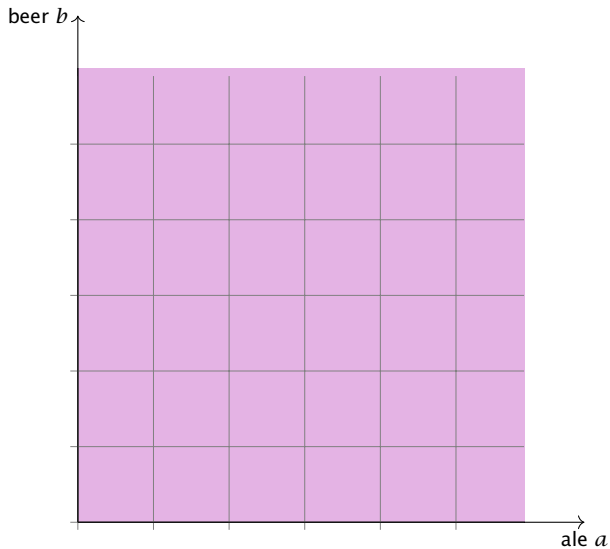
Input size:

- ▶ n number of variables, m constraints, L number of bits to encode the input

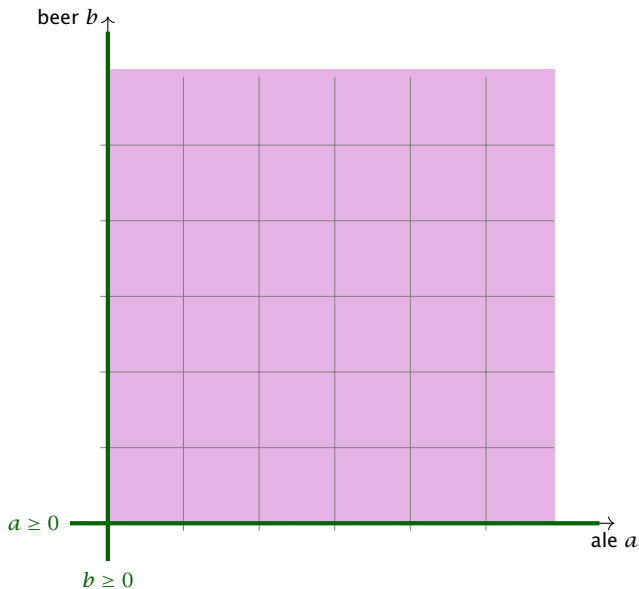
Geometry of Linear Programming



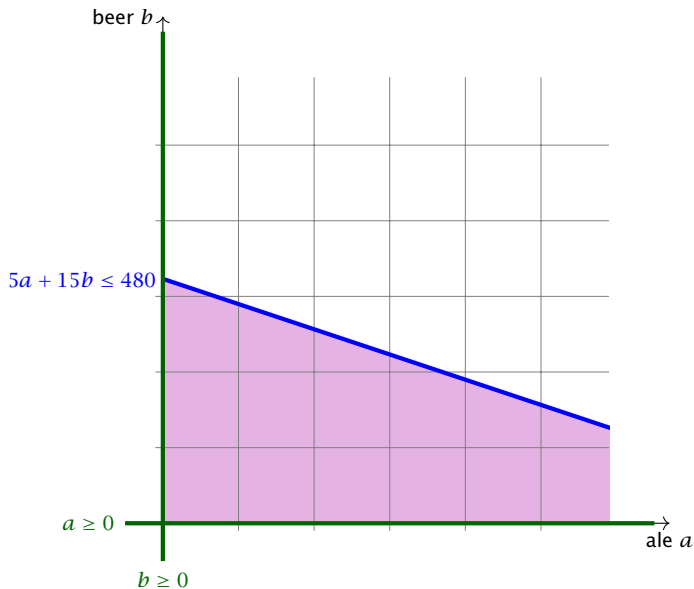
Geometry of Linear Programming



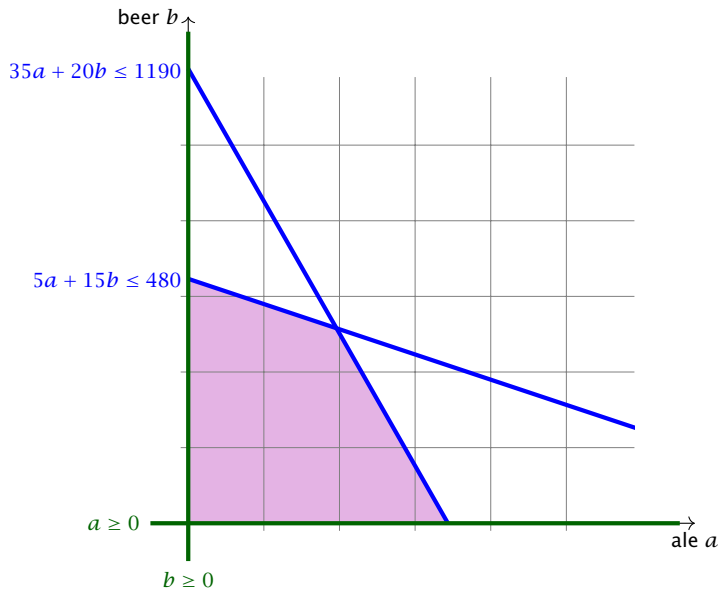
Geometry of Linear Programming



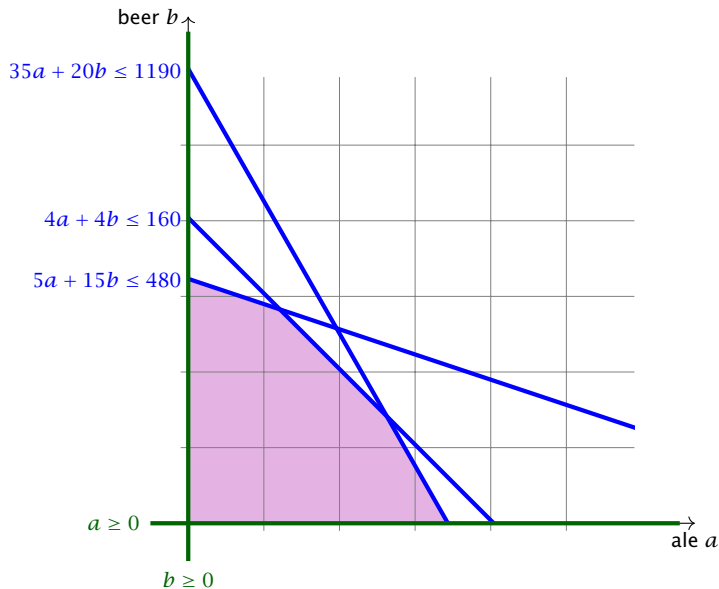
Geometry of Linear Programming



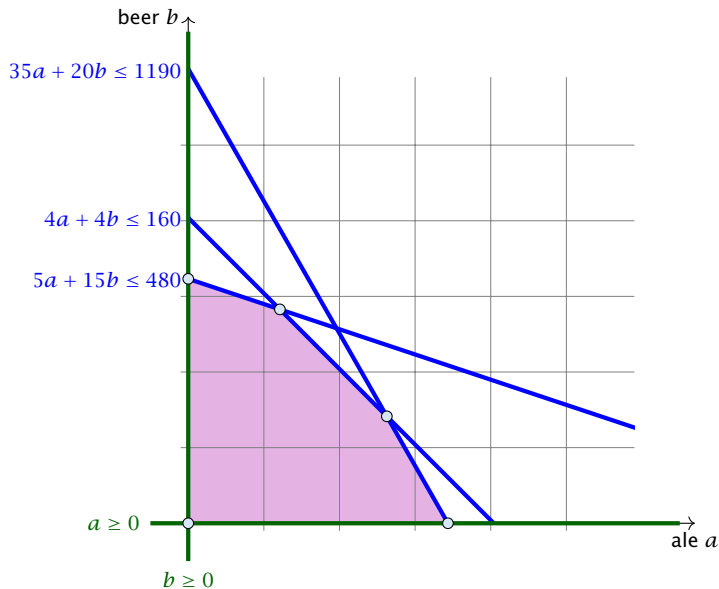
Geometry of Linear Programming



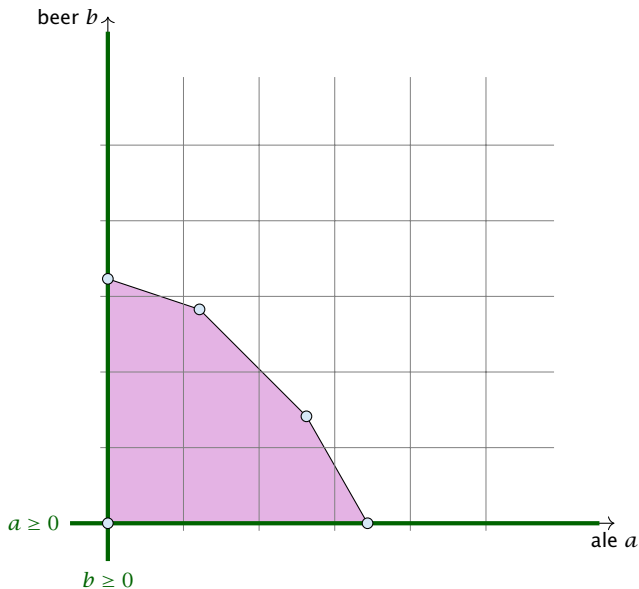
Geometry of Linear Programming



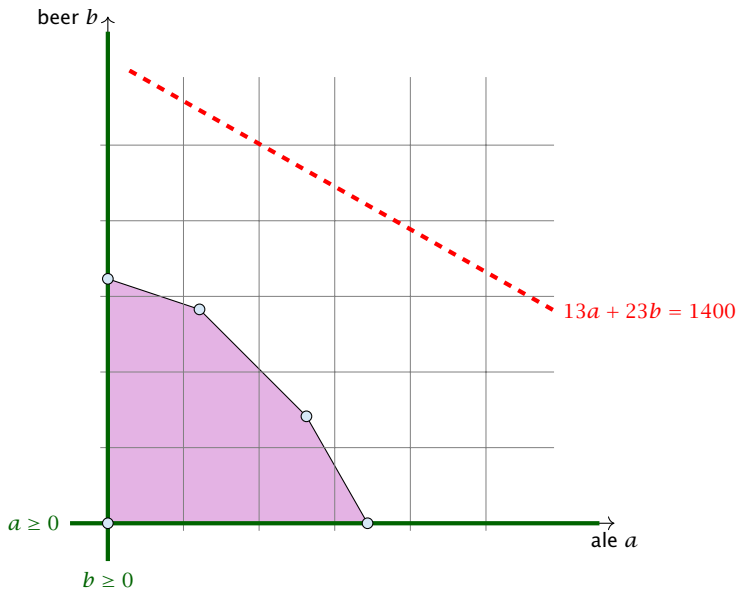
Geometry of Linear Programming



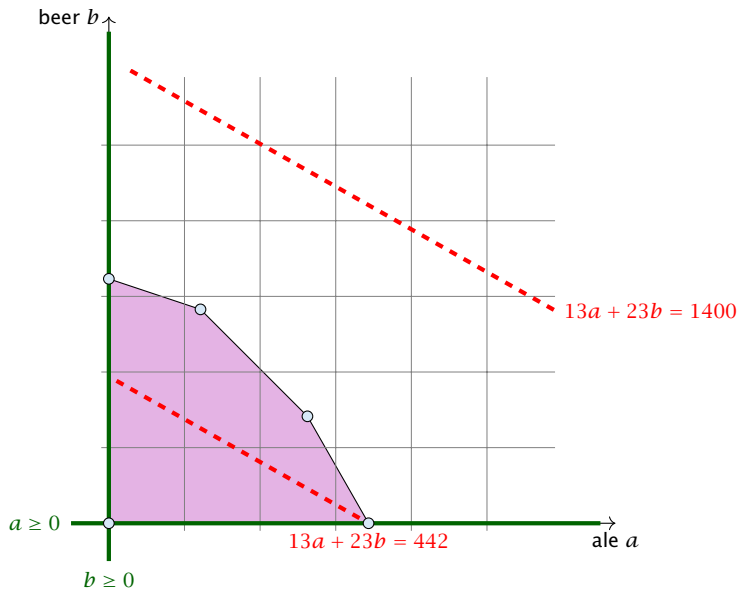
Geometry of Linear Programming



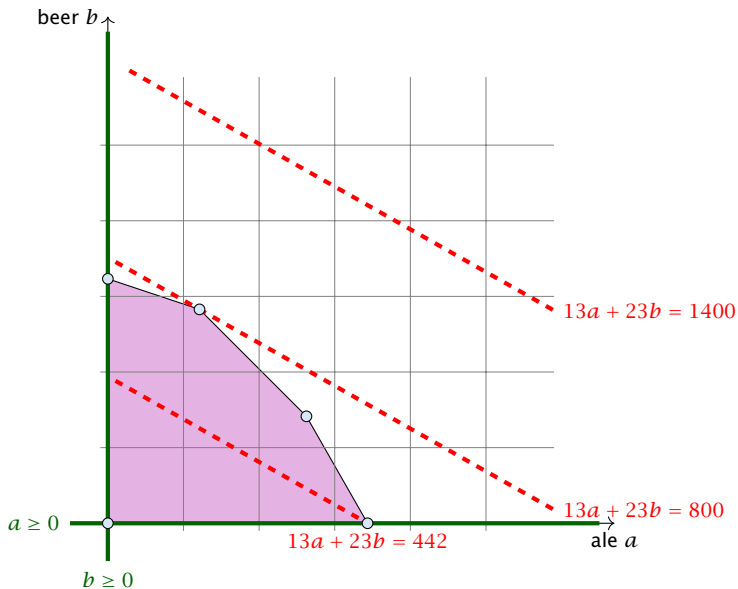
Geometry of Linear Programming



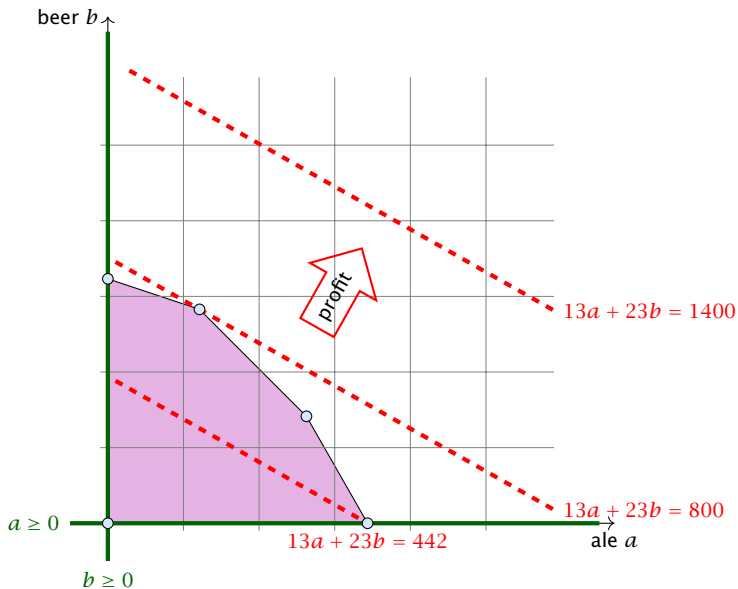
Geometry of Linear Programming



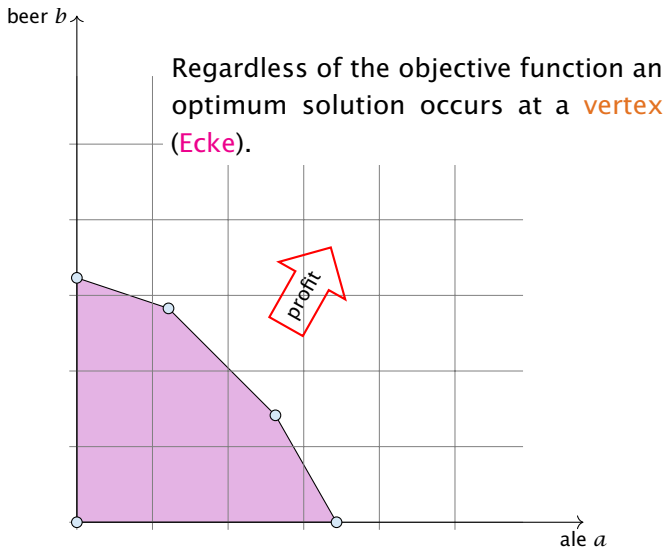
Geometry of Linear Programming



Geometry of Linear Programming



Geometry of Linear Programming



Definitions

Let for a Linear Program in standard form

$$P = \{x \mid Ax = b, x \geq 0\}.$$

Definitions

Let for a Linear Program in standard form

$$P = \{x \mid Ax = b, x \geq 0\}.$$

- ▶ P is called the **feasible region** (**Lösungsraum**) of the LP.
- ▶ A point $x \in P$ is called a **feasible point** (**gültige Lösung**).
- ▶ If $P \neq \emptyset$ then the LP is called **feasible** (**erfüllbar**). Otherwise, it is called **infeasible** (**unerfüllbar**).
- ▶ An LP is **bounded** (**beschränkt**) if it is feasible and
if $c^T x \leq M$ for all $x \in P$ (for maximization problems)
if $c^T x \geq m$ for all $x \in P$ (for minimization problems).

Definitions

Let for a Linear Program in standard form

$$P = \{x \mid Ax = b, x \geq 0\}.$$

- ▶ P is called the **feasible region** (**Lösungsraum**) of the LP.
- ▶ A point $x \in P$ is called a **feasible point** (**gültige Lösung**).
- ▶ If $P \neq \emptyset$ then the LP is called **feasible** (erfüllbar). Otherwise, it is called **infeasible** (unerfüllbar).
- ▶ An LP is **bounded** (beschränkt) if it is feasible and
if $\exists \alpha \in \mathbb{R}$ s.t. $\forall x \in P$ the maximization problem
 $\max \{c^T x \mid x \in P\}$ has an optimal solution x^* with $c^T x^* = \alpha$
for all $c \in \mathbb{R}^n$ (the minimization problem).

Definitions

Let for a Linear Program in standard form

$$P = \{x \mid Ax = b, x \geq 0\}.$$

- ▶ P is called the **feasible region** (**Lösungsraum**) of the LP.
- ▶ A point $x \in P$ is called a **feasible point** (**gültige Lösung**).
- ▶ If $P \neq \emptyset$ then the LP is called **feasible** (**erfüllbar**). Otherwise, it is called **infeasible** (**unerfüllbar**).
- ▶ An LP is **bounded** (**beschränkt**) if it is feasible and
for all $c \in \mathbb{R}^n$ the maximization problem
for all $c \in \mathbb{R}^n$ the minimization problem

Definitions

Let for a Linear Program in standard form

$$P = \{x \mid Ax = b, x \geq 0\}.$$

- ▶ P is called the **feasible region** (**Lösungsraum**) of the LP.
- ▶ A point $x \in P$ is called a **feasible point** (**gültige Lösung**).
- ▶ If $P \neq \emptyset$ then the LP is called **feasible** (**erfüllbar**). Otherwise, it is called **infeasible** (**unerfüllbar**).
- ▶ An LP is **bounded** (**beschränkt**) if it is feasible and
for all $c \in \mathbb{R}^n$ the optimization problem
for all $c \in \mathbb{R}^n$ the minimization problem

Definitions

Let for a Linear Program in standard form

$$P = \{x \mid Ax = b, x \geq 0\}.$$

- ▶ P is called the **feasible region** (**Lösungsraum**) of the LP.
- ▶ A point $x \in P$ is called a **feasible point** (**gültige Lösung**).
- ▶ If $P \neq \emptyset$ then the LP is called **feasible** (**erfüllbar**). Otherwise, it is called **infeasible** (**unerfüllbar**).
- ▶ An LP is **bounded** (**beschränkt**) if it is feasible and
 - ▶ $c^T x < \infty$ for all $x \in P$ (for maximization problems)
 - ▶ $c^T x > -\infty$ for all $x \in P$ (for minimization problems)

Definitions

Let for a Linear Program in standard form

$$P = \{x \mid Ax = b, x \geq 0\}.$$

- ▶ P is called the **feasible region** (**Lösungsraum**) of the LP.
- ▶ A point $x \in P$ is called a **feasible point** (**gültige Lösung**).
- ▶ If $P \neq \emptyset$ then the LP is called **feasible** (**erfüllbar**). Otherwise, it is called **infeasible** (**unerfüllbar**).
- ▶ An LP is **bounded** (**beschränkt**) if it is feasible and
 - ▶ $c^T x < \infty$ for all $x \in P$ (for maximization problems)
 - ▶ $c^T x > -\infty$ for all $x \in P$ (for minimization problems)

Definitions

Let for a Linear Program in standard form

$$P = \{x \mid Ax = b, x \geq 0\}.$$

- ▶ P is called the **feasible region** (**Lösungsraum**) of the LP.
- ▶ A point $x \in P$ is called a **feasible point** (**gültige Lösung**).
- ▶ If $P \neq \emptyset$ then the LP is called **feasible** (**erfüllbar**). Otherwise, it is called **infeasible** (**unerfüllbar**).
- ▶ An LP is **bounded** (**beschränkt**) if it is feasible and
 - ▶ $c^T x < \infty$ for all $x \in P$ (for maximization problems)
 - ▶ $c^T x > -\infty$ for all $x \in P$ (for minimization problems)

Definition 2

Given vectors/points $x_1, \dots, x_k \in \mathbb{R}^n$, $\sum \lambda_i x_i$ is called

- ▶ **linear combination** if $\lambda_i \in \mathbb{R}$.
- ▶ **affine combination** if $\lambda_i \in \mathbb{R}$ and $\sum_i \lambda_i = 1$.
- ▶ **convex combination** if $\lambda_i \in \mathbb{R}$ and $\sum_i \lambda_i = 1$ and $\lambda_i \geq 0$.
- ▶ **conic combination** if $\lambda_i \in \mathbb{R}$ and $\lambda_i \geq 0$.

Note that a combination involves only finitely many vectors.

Definition 3

A set $X \subseteq \mathbb{R}^n$ is called

- ▶ a **linear subspace** if it is closed under linear combinations.
- ▶ an **affine subspace** if it is closed under affine combinations.
- ▶ **convex** if it is closed under convex combinations.
- ▶ a **convex cone** if it is closed under conic combinations.

Note that an affine subspace is **not** a vector space

Definition 4

Given a set $X \subseteq \mathbb{R}^n$.

- ▶ $\text{span}(X)$ is the set of all linear combinations of X
(linear hull, span)
- ▶ $\text{aff}(X)$ is the set of all affine combinations of X
(affine hull)
- ▶ $\text{conv}(X)$ is the set of all convex combinations of X
(convex hull)
- ▶ $\text{cone}(X)$ is the set of all conic combinations of X
(conic hull)

Definition 5

A function $f: \mathbb{R}^n \rightarrow \mathbb{R}$ is **convex** if for $x, y \in \mathbb{R}^n$ and $\lambda \in [0, 1]$ we have

$$f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y)$$

Lemma 6

If $P \subseteq \mathbb{R}^n$, and $f: \mathbb{R}^n \rightarrow \mathbb{R}$ convex then also

$$Q = \{x \in P \mid f(x) \leq t\}$$

Definition 5

A function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is **convex** if for $x, y \in \mathbb{R}^n$ and $\lambda \in [0, 1]$ we have

$$f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda)f(y)$$

Lemma 6

If $P \subseteq \mathbb{R}^n$, and $f : \mathbb{R}^n \rightarrow \mathbb{R}$ convex then also

$$Q = \{x \in P \mid f(x) \leq t\}$$

Dimensions

Definition 7

The **dimension** $\dim(A)$ of an affine subspace $A \subseteq \mathbb{R}^n$ is the dimension of the vector space $\{x - a \mid x \in A\}$, where $a \in A$.

Definition 8

The **dimension** $\dim(X)$ of a convex set $X \subseteq \mathbb{R}^n$ is the dimension of its affine hull $\text{aff}(X)$.

Definition 9

A set $H \subseteq \mathbb{R}^n$ is a **hyperplane** if $H = \{x \mid a^T x = b\}$, for $a \neq 0$.

Definition 10

A set $H' \subseteq \mathbb{R}^n$ is a (closed) **halfspace** if $H = \{x \mid a^T x \leq b\}$, for $a \neq 0$.

Definition 9

A set $H \subseteq \mathbb{R}^n$ is a **hyperplane** if $H = \{x \mid a^T x = b\}$, for $a \neq 0$.

Definition 10

A set $H' \subseteq \mathbb{R}^n$ is a (closed) **halfspace** if $H = \{x \mid a^T x \leq b\}$, for $a \neq 0$.

Definitions

Definition 11

A **polytop** is a set $P \subseteq \mathbb{R}^n$ that is the convex hull of a **finite** set of points, i.e., $P = \text{conv}(X)$ where $|X| = c$.

Definitions

Definition 12

A **polyhedron** is a set $P \subseteq \mathbb{R}^n$ that can be represented as the intersection of **finitely** many half-spaces $\{H(a_1, b_1), \dots, H(a_m, b_m)\}$, where

$$H(a_i, b_i) = \{x \in \mathbb{R}^n \mid a_i x \leq b_i\} \quad .$$

Definition 13

A polyhedron P is **bounded** if there exists B s.t. $\|x\|_2 \leq B$ for all $x \in P$.

Definitions

Definition 12

A **polyhedron** is a set $P \subseteq \mathbb{R}^n$ that can be represented as the intersection of **finitely** many half-spaces $\{H(a_1, b_1), \dots, H(a_m, b_m)\}$, where

$$H(a_i, b_i) = \{x \in \mathbb{R}^n \mid a_i x \leq b_i\} \quad .$$

Definition 13

A polyhedron P is **bounded** if there exists B s.t. $\|x\|_2 \leq B$ for all $x \in P$.

Theorem 14

P is a bounded polyhedron iff P is a polytop.

Definition 15

Let $P \subseteq \mathbb{R}^n$, $a \in \mathbb{R}^n$ and $b \in \mathbb{R}$. The hyperplane

$$H(a, b) = \{x \in \mathbb{R}^n \mid a^T x = b\}$$

is a **supporting hyperplane** of P if $\max\{a^T x \mid x \in P\} = b$.

Definition 16

Let $P \subseteq \mathbb{R}^n$. F is a **face** of P if $F = P$ or $F = P \cap H$ for some supporting hyperplane H .

Definition 17

Let $P \subseteq \mathbb{R}^n$.

- ▶ a face v is a **vertex** of P if $\{v\}$ is a face of P .
- ▶ a face e is an **edge** of P if e is a face and $\dim(e) = 1$.
- ▶ a face F is a **facet** of P if F is a face and $\dim(F) = \dim(P) - 1$.

Definition 15

Let $P \subseteq \mathbb{R}^n$, $a \in \mathbb{R}^n$ and $b \in \mathbb{R}$. The hyperplane

$$H(a, b) = \{x \in \mathbb{R}^n \mid a^T x = b\}$$

is a **supporting hyperplane** of P if $\max\{a^T x \mid x \in P\} = b$.

Definition 16

Let $P \subseteq \mathbb{R}^n$. F is a **face** of P if $F = P$ or $F = P \cap H$ for some supporting hyperplane H .

Definition 17

Let $P \subseteq \mathbb{R}^n$.

- ▶ a face v is a **vertex** of P if $\{v\}$ is a face of P .
- ▶ a face e is an **edge** of P if e is a face and $\dim(e) = 1$.
- ▶ a face F is a **facet** of P if F is a face and $\dim(F) = \dim(P) - 1$.

Definition 15

Let $P \subseteq \mathbb{R}^n$, $a \in \mathbb{R}^n$ and $b \in \mathbb{R}$. The hyperplane

$$H(a, b) = \{x \in \mathbb{R}^n \mid a^T x = b\}$$

is a **supporting hyperplane** of P if $\max\{a^T x \mid x \in P\} = b$.

Definition 16

Let $P \subseteq \mathbb{R}^n$. F is a **face** of P if $F = P$ or $F = P \cap H$ for some supporting hyperplane H .

Definition 17

Let $P \subseteq \mathbb{R}^n$.

- ▶ a face v is a **vertex** of P if $\{v\}$ is a face of P .
- ▶ a face e is an **edge** of P if e is a face and $\dim(e) = 1$.
- ▶ a face F is a **facet** of P if F is a face and $\dim(F) = \dim(P) - 1$.

Equivalent definition for vertex:

Definition 18

Given polyhedron P . A point $x \in P$ is a **vertex** if $\exists c \in \mathbb{R}^n$ such that $c^T y < c^T x$, for all $y \in P$, $y \neq x$.

Definition 19

Given polyhedron P . A point $x \in P$ is an **extreme point** if $\nexists a, b \neq x$, $a, b \in P$, with $\lambda a + (1 - \lambda)b = x$ for $\lambda \in [0, 1]$.

Lemma 20

A vertex is also an extreme point.

Equivalent definition for vertex:

Definition 18

Given polyhedron P . A point $x \in P$ is a **vertex** if $\exists c \in \mathbb{R}^n$ such that $c^T y < c^T x$, for all $y \in P$, $y \neq x$.

Definition 19

Given polyhedron P . A point $x \in P$ is an **extreme point** if $\nexists a, b \neq x$, $a, b \in P$, with $\lambda a + (1 - \lambda)b = x$ for $\lambda \in [0, 1]$.

Lemma 20

A vertex is also an extreme point.

Observation

The feasible region of an LP is a Polyhedron.

Theorem 21

If there exists an optimal solution to an LP (in standard form) then there exists an optimum solution that is an extreme point.

Proof

- ▶ suppose x is optimal solution that is not extreme point
- ▶ there exists direction $d \neq 0$ such that $x \pm d \in P$
- ▶ $Ad = 0$ because $A(x \pm d) = b$
- ▶ Wlog. assume $c^T d \geq 0$ (by taking either d or $-d$)
- ▶ Consider $x + \lambda d, \lambda > 0$

Theorem 21

If there exists an optimal solution to an LP (in standard form) then there exists an optimum solution that is an extreme point.

Proof

- ▶ suppose x is optimal solution that is not extreme point
- ▶ there exists direction $d \neq 0$ such that $x \pm d \in P$
 - ▶ $Ad = 0$ because $A(x \pm d) = b$
 - ▶ Wlog. assume $c^T d \geq 0$ (by taking either d or $-d$)
 - ▶ Consider $x + \lambda d, \lambda > 0$

Theorem 21

If there exists an optimal solution to an LP (in standard form) then there exists an optimum solution that is an extreme point.

Proof

- ▶ suppose x is optimal solution that is not extreme point
- ▶ there exists direction $d \neq 0$ such that $x \pm d \in P$
- ▶ $Ad = 0$ because $A(x \pm d) = b$
- ▶ Wlog. assume $c^T d \geq 0$ (by taking either d or $-d$)
- ▶ Consider $x + \lambda d, \lambda > 0$

Theorem 21

If there exists an optimal solution to an LP (in standard form) then there exists an optimum solution that is an extreme point.

Proof

- ▶ suppose x is optimal solution that is not extreme point
- ▶ there exists direction $d \neq 0$ such that $x \pm d \in P$
- ▶ $Ad = 0$ because $A(x \pm d) = b$
- ▶ Wlog. assume $c^T d \geq 0$ (by taking either d or $-d$)
- ▶ Consider $x + \lambda d, \lambda > 0$

Theorem 21

If there exists an optimal solution to an LP (in standard form) then there exists an optimum solution that is an extreme point.

Proof

- ▶ suppose x is optimal solution that is not extreme point
- ▶ there exists direction $d \neq 0$ such that $x \pm d \in P$
- ▶ $Ad = 0$ because $A(x \pm d) = b$
- ▶ Wlog. assume $c^T d \geq 0$ (by taking either d or $-d$)
- ▶ Consider $x + \lambda d, \lambda > 0$

Case 1. $[\exists j \text{ s.t. } d_j < 0]$

• Increase λ to λ^* until first component of $d + \lambda c$ hits 0.
• $d + \lambda^* c$ is feasible. Since $-c^T d = -c^T(d + \lambda^* c) \geq 0$ and $-c^T d > 0$,
• $d + \lambda^* c$ has the more zero-component $\lambda_j = 0$ for some j ($d_j < 0$ and $c_j > 0$).
• (Another λ gives $d + \lambda c$ with $c^T d + \lambda c^T c > 0$.)

Case 2. $[d_j \geq 0 \text{ for all } j \text{ and } c^T d > 0]$

• $d + \lambda c$ is feasible for all $\lambda \geq 0$ since $d_j \geq 0$ and $c_j \geq 0$ and
• $-c^T d > 0$ means $c^T d < 0$.
• As $\lambda \rightarrow \infty$, $c^T(d + \lambda c) \rightarrow \infty$ as $c^T c > 0$.

Case 1. $[\exists j \text{ s.t. } d_j < 0]$

Consider $y = 0$ until first component of d that is < 0 . This y is feasible. Since $x = 0$ and $y = 0$ is feasible, x has the more zero-component (up to j for case 1, and $j+1$ for case 2).

For case 1, $d_j < 0$. For case 2, $d_j \geq 0$ for all j .

Case 2. $[d_j \geq 0 \text{ for all } j \text{ and } c^T d > 0]$

$x = 0$ is feasible for all $\lambda > 0$ since $x = 0$ and $y = 0$ is feasible. For $\lambda > 0$, $x = \lambda d$ is feasible since $d_j \geq 0$ for all j . For $\lambda > 0$, $c^T x = \lambda c^T d > 0$.

Case 1. [$\exists j$ s.t. $d_j < 0$]

- ▶ increase λ to λ' until first component of $x + \lambda d$ hits 0
- ▶ $x + \lambda' d$ is feasible. Since $A(x + \lambda' d) = b$ and $x + \lambda' d \geq 0$
- ▶ $x + \lambda' d$ has one more zero-component ($d_k = 0$ for $x_k = 0$ as $x \pm d \in P$)
- ▶ $c^T x' = c^T (x + \lambda' d) = c^T x + \lambda' c^T d \geq c^T x$

Case 2. [$d_j \geq 0$ for all j and $c^T d > 0$]

- ▶ $x + \lambda d$ is feasible for all $\lambda \geq 0$ since $A(x + \lambda d) = b$ and $x + \lambda d \geq 0$
- ▶ $c^T x + \lambda c^T d > c^T x$ for all $\lambda > 0$
- ▶ P is unbounded

Case 1. $[\exists j \text{ s.t. } d_j < 0]$

- ▶ increase λ to λ' until first component of $x + \lambda d$ hits 0
- ▶ $x + \lambda' d$ is feasible. Since $A(x + \lambda' d) = b$ and $x + \lambda' d \geq 0$
- ▶ $x + \lambda' d$ has one more zero-component ($d_k = 0$ for $x_k = 0$ as $x \pm d \in P$)
- ▶ $c^T x' = c^T (x + \lambda' d) = c^T x + \lambda' c^T d \geq c^T x$

Case 2. $[d_j \geq 0 \text{ for all } j \text{ and } c^T d > 0]$

- ▶ $x + \lambda d$ is feasible for all $\lambda \geq 0$ since $x + \lambda d \geq 0$ and $A(x + \lambda d) = b$
- ▶ $c^T x' = c^T (x + \lambda d) = c^T x + \lambda c^T d > c^T x$

Case 1. $[\exists j \text{ s.t. } d_j < 0]$

- ▶ increase λ to λ' until first component of $x + \lambda d$ hits 0
- ▶ $x + \lambda' d$ is feasible. Since $A(x + \lambda' d) = b$ and $x + \lambda' d \geq 0$
- ▶ $x + \lambda' d$ has one more zero-component ($d_k = 0$ for $x_k = 0$ as $x \pm d \in P$)
- ▶ $c^T x' = c^T (x + \lambda' d) = c^T x + \lambda' c^T d \geq c^T x$

Case 2. $[d_j \geq 0 \text{ for all } j \text{ and } c^T d > 0]$

- ▶ $x + \lambda d$ is feasible for all $\lambda \geq 0$ since $x + \lambda d \geq 0$ and $A(x + \lambda d) = b$
- ▶ $c^T x + \lambda c^T d > c^T x$ for all $\lambda > 0$
- ▶ P is unbounded

Case 1. $[\exists j \text{ s.t. } d_j < 0]$

- ▶ increase λ to λ' until first component of $x + \lambda d$ hits 0
- ▶ $x + \lambda' d$ is feasible. Since $A(x + \lambda' d) = b$ and $x + \lambda' d \geq 0$
- ▶ $x + \lambda' d$ has one more zero-component ($d_k = 0$ for $x_k = 0$ as $x \pm d \in P$)
- ▶ $c^T x' = c^T (x + \lambda' d) = c^T x + \lambda' c^T d \geq c^T x$

Case 2. $[d_j \geq 0 \text{ for all } j \text{ and } c^T d > 0]$

Convex Sets

Case 1. $[\exists j \text{ s.t. } d_j < 0]$

- ▶ increase λ to λ' until first component of $x + \lambda d$ hits 0
- ▶ $x + \lambda' d$ is feasible. Since $A(x + \lambda' d) = b$ and $x + \lambda' d \geq 0$
- ▶ $x + \lambda' d$ has one more zero-component ($d_k = 0$ for $x_k = 0$ as $x \pm d \in P$)
- ▶ $c^T x' = c^T (x + \lambda' d) = c^T x + \lambda' c^T d \geq c^T x$

Case 2. $[d_j \geq 0 \text{ for all } j \text{ and } c^T d > 0]$

Case 1. $[\exists j \text{ s.t. } d_j < 0]$

- ▶ increase λ to λ' until first component of $x + \lambda d$ hits 0
- ▶ $x + \lambda' d$ is feasible. Since $A(x + \lambda' d) = b$ and $x + \lambda' d \geq 0$
- ▶ $x + \lambda' d$ has one more zero-component ($d_k = 0$ for $x_k = 0$ as $x \pm d \in P$)
- ▶ $c^T x' = c^T (x + \lambda' d) = c^T x + \lambda' c^T d \geq c^T x$

Case 2. $[d_j \geq 0 \text{ for all } j \text{ and } c^T d > 0]$

- ▶ $x + \lambda d$ is feasible for all $\lambda \geq 0$ since $A(x + \lambda d) = b$ and $x + \lambda d \geq x \geq 0$
- ▶ as $\lambda \rightarrow \infty$, $c^T (x + \lambda d) \rightarrow \infty$ as $c^T d > 0$

Convex Sets

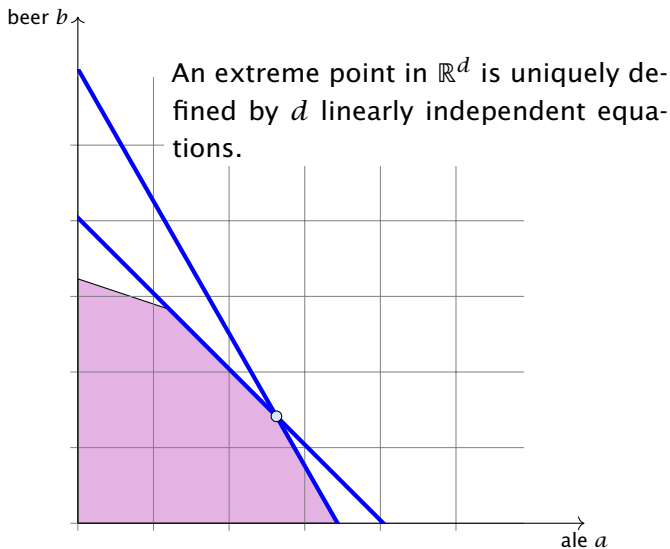
Case 1. $[\exists j \text{ s.t. } d_j < 0]$

- ▶ increase λ to λ' until first component of $x + \lambda d$ hits 0
- ▶ $x + \lambda' d$ is feasible. Since $A(x + \lambda' d) = b$ and $x + \lambda' d \geq 0$
- ▶ $x + \lambda' d$ has one more zero-component ($d_k = 0$ for $x_k = 0$ as $x \pm d \in P$)
- ▶ $c^T x' = c^T (x + \lambda' d) = c^T x + \lambda' c^T d \geq c^T x$

Case 2. $[d_j \geq 0 \text{ for all } j \text{ and } c^T d > 0]$

- ▶ $x + \lambda d$ is feasible for all $\lambda \geq 0$ since $A(x + \lambda d) = b$ and $x + \lambda d \geq x \geq 0$
- ▶ as $\lambda \rightarrow \infty$, $c^T (x + \lambda d) \rightarrow \infty$ as $c^T d > 0$

Algebraic View



Notation

Suppose $B \subseteq \{1 \dots n\}$ is a set of column-indices. Define A_B as the subset of columns of A indexed by B .

Theorem 22

Let $P = \{x \mid Ax = b, x \geq 0\}$. For $x \in P$, define $B = \{j \mid x_j > 0\}$. Then x is extreme point iff A_B has linearly independent columns.

Notation

Suppose $B \subseteq \{1 \dots n\}$ is a set of column-indices. Define A_B as the subset of columns of A indexed by B .

Theorem 22

Let $P = \{x \mid Ax = b, x \geq 0\}$. For $x \in P$, define $B = \{j \mid x_j > 0\}$.
Then x is extreme point **iff** A_B has linearly independent columns.

Theorem 22

Let $P = \{x \mid Ax = b, x \geq 0\}$. For $x \in P$, define $B = \{j \mid x_j > 0\}$.
Then x is extreme point **iff** A_B has linearly independent columns.

Proof ($\Leftarrow$)

Theorem 22

Let $P = \{x \mid Ax = b, x \geq 0\}$. For $x \in P$, define $B = \{j \mid x_j > 0\}$.
Then x is extreme point **iff** A_B has linearly independent columns.

Proof ($\Leftarrow$)

- ▶ assume x is not extreme point
- ▶ there exists direction d s.t. $x \pm d \in P$
- ▶ $Ad = 0$ because $A(x \pm d) = b$
- ▶ define $B' = \{j \mid d_j \neq 0\}$
- ▶ $A_{B'}$ has linearly dependent columns as $Ad = 0$
- ▶ $d_j = 0$ for all j with $x_j = 0$ as $x \pm d \geq 0$
- ▶ Hence, $B' \subseteq B$, $A_{B'}$ is sub-matrix of A_B

Theorem 22

Let $P = \{x \mid Ax = b, x \geq 0\}$. For $x \in P$, define $B = \{j \mid x_j > 0\}$.
Then x is extreme point **iff** A_B has linearly independent columns.

Proof ($\Leftarrow$)

- ▶ assume x is not extreme point
- ▶ there exists direction d s.t. $x \pm d \in P$
 - ▶ $Ad = 0$ because $A(x \pm d) = b$
 - ▶ define $B' = \{j \mid d_j \neq 0\}$
 - ▶ $A_{B'}$ has linearly dependent columns as $Ad = 0$
 - ▶ $d_j = 0$ for all j with $x_j = 0$ as $x \pm d \geq 0$
 - ▶ Hence, $B' \subseteq B$, $A_{B'}$ is sub-matrix of A_B

Theorem 22

Let $P = \{x \mid Ax = b, x \geq 0\}$. For $x \in P$, define $B = \{j \mid x_j > 0\}$.
Then x is extreme point **iff** A_B has linearly independent columns.

Proof ($\Leftarrow$)

- ▶ assume x is not extreme point
- ▶ there exists direction d s.t. $x \pm d \in P$
- ▶ $Ad = 0$ because $A(x \pm d) = b$
- ▶ define $B' = \{j \mid d_j \neq 0\}$
- ▶ $A_{B'}$ has linearly dependent columns as $Ad = 0$
- ▶ $d_j = 0$ for all j with $x_j = 0$ as $x \pm d \geq 0$
- ▶ Hence, $B' \subseteq B$, $A_{B'}$ is sub-matrix of A_B

Theorem 22

Let $P = \{x \mid Ax = b, x \geq 0\}$. For $x \in P$, define $B = \{j \mid x_j > 0\}$.
Then x is extreme point **iff** A_B has linearly independent columns.

Proof ($\Leftarrow$)

- ▶ assume x is not extreme point
- ▶ there exists direction d s.t. $x \pm d \in P$
- ▶ $Ad = 0$ because $A(x \pm d) = b$
- ▶ define $B' = \{j \mid d_j \neq 0\}$
- ▶ $A_{B'}$ has linearly dependent columns as $Ad = 0$
- ▶ $d_j = 0$ for all j with $x_j = 0$ as $x \pm d \geq 0$
- ▶ Hence, $B' \subseteq B$, $A_{B'}$ is sub-matrix of A_B

Theorem 22

Let $P = \{x \mid Ax = b, x \geq 0\}$. For $x \in P$, define $B = \{j \mid x_j > 0\}$.
Then x is extreme point **iff** A_B has linearly independent columns.

Proof ($\Leftarrow$)

- ▶ assume x is not extreme point
- ▶ there exists direction d s.t. $x \pm d \in P$
- ▶ $Ad = 0$ because $A(x \pm d) = b$
- ▶ define $B' = \{j \mid d_j \neq 0\}$
- ▶ $A_{B'}$ has linearly dependent columns as $Ad = 0$
- ▶ $d_j = 0$ for all j with $x_j = 0$ as $x \pm d \geq 0$
- ▶ Hence, $B' \subseteq B$, $A_{B'}$ is sub-matrix of A_B

Theorem 22

Let $P = \{x \mid Ax = b, x \geq 0\}$. For $x \in P$, define $B = \{j \mid x_j > 0\}$.
Then x is extreme point **iff** A_B has linearly independent columns.

Proof ($\Leftarrow$)

- ▶ assume x is not extreme point
- ▶ there exists direction d s.t. $x \pm d \in P$
- ▶ $Ad = 0$ because $A(x \pm d) = b$
- ▶ define $B' = \{j \mid d_j \neq 0\}$
- ▶ $A_{B'}$ has linearly dependent columns as $Ad = 0$
- ▶ $d_j = 0$ for all j with $x_j = 0$ as $x \pm d \geq 0$
- ▶ Hence, $B' \subseteq B$, $A_{B'}$ is sub-matrix of A_B

Theorem 22

Let $P = \{x \mid Ax = b, x \geq 0\}$. For $x \in P$, define $B = \{j \mid x_j > 0\}$.
Then x is extreme point **iff** A_B has linearly independent columns.

Proof ($\Leftarrow$)

- ▶ assume x is not extreme point
- ▶ there exists direction d s.t. $x \pm d \in P$
- ▶ $Ad = 0$ because $A(x \pm d) = b$
- ▶ define $B' = \{j \mid d_j \neq 0\}$
- ▶ $A_{B'}$ has linearly dependent columns as $Ad = 0$
- ▶ $d_j = 0$ for all j with $x_j = 0$ as $x \pm d \geq 0$
- ▶ Hence, $B' \subseteq B$, $A_{B'}$ is sub-matrix of A_B

Theorem 22

Let $P = \{x \mid Ax = b, x \geq 0\}$. For $x \in P$, define $B = \{j \mid x_j > 0\}$.
Then x is extreme point **iff** A_B has linearly independent columns.

Proof ($\Rightarrow$)

Theorem 22

Let $P = \{x \mid Ax = b, x \geq 0\}$. For $x \in P$, define $B = \{j \mid x_j > 0\}$.
Then x is extreme point **iff** A_B has linearly independent columns.

Proof ($\Rightarrow$)

- ▶ assume A_B has linearly dependent columns
- ▶ there exists $d \neq 0$ such that $A_B d = 0$
- ▶ extend d to $\mathbb{R}^n$ by adding 0-components
- ▶ now, $Ad = 0$ and $d_j = 0$ whenever $x_j = 0$
- ▶ for sufficiently small λ we have $x \pm \lambda d \in P$
- ▶ hence, x is not extreme point

Theorem 22

Let $P = \{x \mid Ax = b, x \geq 0\}$. For $x \in P$, define $B = \{j \mid x_j > 0\}$.
Then x is extreme point **iff** A_B has linearly independent columns.

Proof ($\Rightarrow$)

- ▶ assume A_B has linearly dependent columns
- ▶ there exists $d \neq 0$ such that $A_B d = 0$
 - ▶ extend d to $\mathbb{R}^n$ by adding 0-components
 - ▶ now, $Ad = 0$ and $d_j = 0$ whenever $x_j = 0$
 - ▶ for sufficiently small λ we have $x \pm \lambda d \in P$
 - ▶ hence, x is not extreme point

Theorem 22

Let $P = \{x \mid Ax = b, x \geq 0\}$. For $x \in P$, define $B = \{j \mid x_j > 0\}$.
Then x is extreme point **iff** A_B has linearly independent columns.

Proof ($\Rightarrow$)

- ▶ assume A_B has linearly dependent columns
- ▶ there exists $d \neq 0$ such that $A_B d = 0$
- ▶ extend d to $\mathbb{R}^n$ by adding 0-components
- ▶ now, $Ad = 0$ and $d_j = 0$ whenever $x_j = 0$
- ▶ for sufficiently small λ we have $x \pm \lambda d \in P$
- ▶ hence, x is not extreme point

Theorem 22

Let $P = \{x \mid Ax = b, x \geq 0\}$. For $x \in P$, define $B = \{j \mid x_j > 0\}$.
Then x is extreme point **iff** A_B has linearly independent columns.

Proof ($\Rightarrow$)

- ▶ assume A_B has linearly dependent columns
- ▶ there exists $d \neq 0$ such that $A_B d = 0$
- ▶ extend d to $\mathbb{R}^n$ by adding 0-components
- ▶ now, $Ad = 0$ and $d_j = 0$ whenever $x_j = 0$
- ▶ for sufficiently small λ we have $x \pm \lambda d \in P$
- ▶ hence, x is not extreme point

Theorem 22

Let $P = \{x \mid Ax = b, x \geq 0\}$. For $x \in P$, define $B = \{j \mid x_j > 0\}$.
Then x is extreme point **iff** A_B has linearly independent columns.

Proof ($\Rightarrow$)

- ▶ assume A_B has linearly dependent columns
- ▶ there exists $d \neq 0$ such that $A_B d = 0$
- ▶ extend d to $\mathbb{R}^n$ by adding 0-components
- ▶ now, $Ad = 0$ and $d_j = 0$ whenever $x_j = 0$
- ▶ for sufficiently small λ we have $x \pm \lambda d \in P$
- ▶ hence, x is not extreme point

Theorem 22

Let $P = \{x \mid Ax = b, x \geq 0\}$. For $x \in P$, define $B = \{j \mid x_j > 0\}$.
Then x is extreme point **iff** A_B has linearly independent columns.

Proof ($\Rightarrow$)

- ▶ assume A_B has linearly dependent columns
- ▶ there exists $d \neq 0$ such that $A_B d = 0$
- ▶ extend d to $\mathbb{R}^n$ by adding 0-components
- ▶ now, $Ad = 0$ and $d_j = 0$ whenever $x_j = 0$
- ▶ for sufficiently small λ we have $x \pm \lambda d \in P$
- ▶ hence, x is not extreme point

Theorem 23

Let $P = \{x \mid Ax = b, x \geq 0\}$. For $x \in P$, define $B = \{j \mid x_j > 0\}$. If A_B has linearly independent columns then x is a vertex of P .

- ▶ define $c_j = \begin{cases} 0 & j \in B \\ -1 & j \notin B \end{cases}$
- ▶ then $c^T x = 0$ and $c^T y \leq 0$ for $y \in P$
- ▶ assume $c^T y = 0$; then $y_j = 0$ for all $j \notin B$
- ▶ $b = Ay = A_B y_B = Ax = A_B x_B$ gives that $A_B(x_B - y_B) = 0$;
- ▶ this means that $x_B = y_B$ since A_B has linearly independent columns
- ▶ we get $y = x$
- ▶ hence, x is a vertex of P

Theorem 23

Let $P = \{x \mid Ax = b, x \geq 0\}$. For $x \in P$, define $B = \{j \mid x_j > 0\}$. If A_B has linearly independent columns then x is a vertex of P .

- ▶ define $c_j = \begin{cases} 0 & j \in B \\ -1 & j \notin B \end{cases}$
- ▶ then $c^T x = 0$ and $c^T y \leq 0$ for $y \in P$
 - ▶ assume $c^T y = 0$; then $y_j = 0$ for all $j \notin B$
 - ▶ $b = Ay = A_B y_B = Ax = A_B x_B$ gives that $A_B(x_B - y_B) = 0$;
 - ▶ this means that $x_B = y_B$ since A_B has linearly independent columns
 - ▶ we get $y = x$
 - ▶ hence, x is a vertex of P

Theorem 23

Let $P = \{x \mid Ax = b, x \geq 0\}$. For $x \in P$, define $B = \{j \mid x_j > 0\}$. If A_B has linearly independent columns then x is a vertex of P .

- ▶ define $c_j = \begin{cases} 0 & j \in B \\ -1 & j \notin B \end{cases}$
- ▶ then $c^T x = 0$ and $c^T y \leq 0$ for $y \in P$
- ▶ assume $c^T y = 0$; then $y_j = 0$ for all $j \notin B$
- ▶ $b = Ay = A_B y_B = Ax = A_B x_B$ gives that $A_B(x_B - y_B) = 0$;
- ▶ this means that $x_B = y_B$ since A_B has linearly independent columns
- ▶ we get $y = x$
- ▶ hence, x is a vertex of P

Theorem 23

Let $P = \{x \mid Ax = b, x \geq 0\}$. For $x \in P$, define $B = \{j \mid x_j > 0\}$. If A_B has linearly independent columns then x is a vertex of P .

- ▶ define $c_j = \begin{cases} 0 & j \in B \\ -1 & j \notin B \end{cases}$
- ▶ then $c^T x = 0$ and $c^T y \leq 0$ for $y \in P$
- ▶ assume $c^T y = 0$; then $y_j = 0$ for all $j \notin B$
- ▶ $b = Ay = A_B y_B = Ax = A_B x_B$ gives that $A_B(x_B - y_B) = 0$;
 - ▶ this means that $x_B = y_B$ since A_B has linearly independent columns
 - ▶ we get $y = x$
 - ▶ hence, x is a vertex of P

Theorem 23

Let $P = \{x \mid Ax = b, x \geq 0\}$. For $x \in P$, define $B = \{j \mid x_j > 0\}$. If A_B has linearly independent columns then x is a vertex of P .

- ▶ define $c_j = \begin{cases} 0 & j \in B \\ -1 & j \notin B \end{cases}$
- ▶ then $c^T x = 0$ and $c^T y \leq 0$ for $y \in P$
- ▶ assume $c^T y = 0$; then $y_j = 0$ for all $j \notin B$
- ▶ $b = Ay = A_B y_B = Ax = A_B x_B$ gives that $A_B(x_B - y_B) = 0$;
- ▶ this means that $x_B = y_B$ since A_B has linearly independent columns
- ▶ we get $y = x$
- ▶ hence, x is a vertex of P

Theorem 23

Let $P = \{x \mid Ax = b, x \geq 0\}$. For $x \in P$, define $B = \{j \mid x_j > 0\}$. If A_B has linearly independent columns then x is a vertex of P .

- ▶ define $c_j = \begin{cases} 0 & j \in B \\ -1 & j \notin B \end{cases}$
- ▶ then $c^T x = 0$ and $c^T y \leq 0$ for $y \in P$
- ▶ assume $c^T y = 0$; then $y_j = 0$ for all $j \notin B$
- ▶ $b = Ay = A_B y_B = Ax = A_B x_B$ gives that $A_B(x_B - y_B) = 0$;
- ▶ this means that $x_B = y_B$ since A_B has linearly independent columns
- ▶ we get $y = x$
- ▶ hence, x is a vertex of P

Theorem 23

Let $P = \{x \mid Ax = b, x \geq 0\}$. For $x \in P$, define $B = \{j \mid x_j > 0\}$. If A_B has linearly independent columns then x is a vertex of P .

- ▶ define $c_j = \begin{cases} 0 & j \in B \\ -1 & j \notin B \end{cases}$
- ▶ then $c^T x = 0$ and $c^T y \leq 0$ for $y \in P$
- ▶ assume $c^T y = 0$; then $y_j = 0$ for all $j \notin B$
- ▶ $b = Ay = A_B y_B = Ax = A_B x_B$ gives that $A_B(x_B - y_B) = 0$;
- ▶ this means that $x_B = y_B$ since A_B has linearly independent columns
- ▶ we get $y = x$
- ▶ hence, x is a vertex of P

Observation

For an LP we can assume wlog. that the matrix A has full row-rank. This means $\text{rank}(A) = m$.

Observation

For an LP we can assume wlog. that the matrix A has full row-rank. This means $\text{rank}(A) = m$.

- ▶ assume that $\text{rank}(A) < m$
- ▶ assume wlog. that the first row A_1 lies in the span of the other rows $A_2, \dots, A_m$; this means

C1 if now $b_1 = \sum_{i=2}^m \lambda_i \cdot b_i$ then for all x with $A_2 x \leq b_2, \dots, A_m x \leq b_m$ we also have $A_1 x \leq b_1$, hence the first constraint is superfluous

C2 if $b_1 \neq \sum_{i=2}^m \lambda_i \cdot b_i$ then the LP is infeasible, since for all x that fulfill constraints $A_2, \dots, A_m$ we have

Observation

For an LP we can assume wlog. that the matrix A has full row-rank. This means $\text{rank}(A) = m$.

- ▶ assume that $\text{rank}(A) < m$
- ▶ assume wlog. that the first row A_1 lies in the span of the other rows $A_2, \dots, A_m$; this means

$$A_1 = \sum_{i=2}^m \lambda_i \cdot A_i, \text{ for suitable } \lambda_i$$

C1 if now $b_1 = \sum_{i=2}^m \lambda_i \cdot b_i$ then for all x with $A_2 x \leq b_2, \dots, A_m x \leq b_m$ we also have $A_1 x \leq b_1$, hence the first constraint is superfluous

C2 if $b_1 \neq \sum_{i=2}^m \lambda_i \cdot b_i$ then the LP is infeasible, since for all x that fulfill constraints $A_2, \dots, A_m$ we have

Observation

For an LP we can assume wlog. that the matrix A has full row-rank. This means $\text{rank}(A) = m$.

- ▶ assume that $\text{rank}(A) < m$
- ▶ assume wlog. that the first row A_1 lies in the span of the other rows $A_2, \dots, A_m$; this means

$$A_1 = \sum_{i=2}^m \lambda_i \cdot A_i, \text{ for suitable } \lambda_i$$

C1 if now $b_1 = \sum_{i=2}^m \lambda_i \cdot b_i$ then for all x with $A_2 x \leq b_2, \dots, A_m x \leq b_m$ we also have $A_1 x \leq b_1$, hence the first constraint is superfluous

C2 if $b_1 \neq \sum_{i=2}^m \lambda_i \cdot b_i$ then the LP is infeasible, since for all x that fulfill constraints $A_2, \dots, A_m$ we have

Observation

For an LP we can assume wlog. that the matrix A has full row-rank. This means $\text{rank}(A) = m$.

- ▶ assume that $\text{rank}(A) < m$
- ▶ assume wlog. that the first row A_1 lies in the span of the other rows $A_2, \dots, A_m$; this means

$$A_1 = \sum_{i=2}^m \lambda_i \cdot A_i, \text{ for suitable } \lambda_i$$

C1 if now $b_1 = \sum_{i=2}^m \lambda_i \cdot b_i$ then for all x with $A_i x = b_i$ we also have $A_1 x = b_1$; hence the first constraint is superfluous

C2 if $b_1 \neq \sum_{i=2}^m \lambda_i \cdot b_i$ then the LP is infeasible, since for all x that fulfill constraints $A_2, \dots, A_m$ we have

Observation

For an LP we can assume wlog. that the matrix A has full row-rank. This means $\text{rank}(A) = m$.

- ▶ assume that $\text{rank}(A) < m$
- ▶ assume wlog. that the first row A_1 lies in the span of the other rows $A_2, \dots, A_m$; this means

$$A_1 = \sum_{i=2}^m \lambda_i \cdot A_i, \text{ for suitable } \lambda_i$$

C1 if now $b_1 = \sum_{i=2}^m \lambda_i \cdot b_i$ then for all x with $A_i x = b_i$ we also have $A_1 x = b_1$; hence the first constraint is superfluous

C2 if $b_1 \neq \sum_{i=2}^m \lambda_i \cdot b_i$ then the LP is infeasible, since for all x that fulfill constraints $A_2, \dots, A_m$ we have

Observation

For an LP we can assume wlog. that the matrix A has full row-rank. This means $\text{rank}(A) = m$.

- ▶ assume that $\text{rank}(A) < m$
- ▶ assume wlog. that the first row A_1 lies in the span of the other rows $A_2, \dots, A_m$; this means

$$A_1 = \sum_{i=2}^m \lambda_i \cdot A_i, \text{ for suitable } \lambda_i$$

- C1** if now $b_1 = \sum_{i=2}^m \lambda_i \cdot b_i$ then for all x with $A_i x = b_i$ we also have $A_1 x = b_1$; hence the first constraint is superfluous
- C2** if $b_1 \neq \sum_{i=2}^m \lambda_i \cdot b_i$ then the LP is infeasible, since for all x that fulfill constraints $A_2, \dots, A_m$ we have

$$A_1 x = \sum_{i=2}^m \lambda_i \cdot A_i x = \sum_{i=2}^m \lambda_i \cdot b_i \neq b_1$$

Observation

For an LP we can assume wlog. that the matrix A has full row-rank. This means $\text{rank}(A) = m$.

- ▶ assume that $\text{rank}(A) < m$
- ▶ assume wlog. that the first row A_1 lies in the span of the other rows $A_2, \dots, A_m$; this means

$$A_1 = \sum_{i=2}^m \lambda_i \cdot A_i, \text{ for suitable } \lambda_i$$

- C1** if now $b_1 = \sum_{i=2}^m \lambda_i \cdot b_i$ then for all x with $A_i x = b_i$ we also have $A_1 x = b_1$; hence the first constraint is superfluous
- C2** if $b_1 \neq \sum_{i=2}^m \lambda_i \cdot b_i$ then the LP is infeasible, since for all x that fulfill constraints $A_2, \dots, A_m$ we have

$$A_1 x = \sum_{i=2}^m \lambda_i \cdot A_i x = \sum_{i=2}^m \lambda_i \cdot b_i \neq b_1$$

Observation

For an LP we can assume wlog. that the matrix A has full row-rank. This means $\text{rank}(A) = m$.

- ▶ assume that $\text{rank}(A) < m$
- ▶ assume wlog. that the first row A_1 lies in the span of the other rows $A_2, \dots, A_m$; this means

$$A_1 = \sum_{i=2}^m \lambda_i \cdot A_i, \text{ for suitable } \lambda_i$$

- C1** if now $b_1 = \sum_{i=2}^m \lambda_i \cdot b_i$ then for all x with $A_i x = b_i$ we also have $A_1 x = b_1$; hence the first constraint is superfluous
- C2** if $b_1 \neq \sum_{i=2}^m \lambda_i \cdot b_i$ then the LP is infeasible, since for all x that fulfill constraints $A_2, \dots, A_m$ we have

$$A_1 x = \sum_{i=2}^m \lambda_i \cdot A_i x = \sum_{i=2}^m \lambda_i \cdot b_i \neq b_1$$

Observation

For an LP we can assume wlog. that the matrix A has full row-rank. This means $\text{rank}(A) = m$.

- ▶ assume that $\text{rank}(A) < m$
- ▶ assume wlog. that the first row A_1 lies in the span of the other rows $A_2, \dots, A_m$; this means

$$A_1 = \sum_{i=2}^m \lambda_i \cdot A_i, \text{ for suitable } \lambda_i$$

- C1** if now $b_1 = \sum_{i=2}^m \lambda_i \cdot b_i$ then for all x with $A_i x = b_i$ we also have $A_1 x = b_1$; hence the first constraint is superfluous
- C2** if $b_1 \neq \sum_{i=2}^m \lambda_i \cdot b_i$ then the LP is infeasible, since for all x that fulfill constraints $A_2, \dots, A_m$ we have

$$A_1 x = \sum_{i=2}^m \lambda_i \cdot A_i x = \sum_{i=2}^m \lambda_i \cdot b_i \neq b_1$$

Observation

For an LP we can assume wlog. that the matrix A has full row-rank. This means $\text{rank}(A) = m$.

- ▶ assume that $\text{rank}(A) < m$
- ▶ assume wlog. that the first row A_1 lies in the span of the other rows $A_2, \dots, A_m$; this means

$$A_1 = \sum_{i=2}^m \lambda_i \cdot A_i, \text{ for suitable } \lambda_i$$

- C1** if now $b_1 = \sum_{i=2}^m \lambda_i \cdot b_i$ then for all x with $A_i x = b_i$ we also have $A_1 x = b_1$; hence the first constraint is superfluous
- C2** if $b_1 \neq \sum_{i=2}^m \lambda_i \cdot b_i$ then the LP is infeasible, since for all x that fulfill constraints $A_2, \dots, A_m$ we have

$$A_1 x = \sum_{i=2}^m \lambda_i \cdot A_i x = \sum_{i=2}^m \lambda_i \cdot b_i \neq b_1$$

From now on we will always assume that the constraint matrix of a standard form LP has full row rank.

Theorem 24

Given $P = \{x \mid Ax = b, x \geq 0\}$. x is extreme point iff there exists $B \subseteq \{1, \dots, n\}$ with $|B| = m$ and

- ▶ A_B is non-singular
- ▶ $x_B = A_B^{-1}b \geq 0$
- ▶ $x_N = 0$

where $N = \{1, \dots, n\} \setminus B$.

Proof

Take $B = \{j \mid x_j > 0\}$ and augment with linearly independent columns until $|B| = m$; always possible since $\text{rank}(A) = m$.

Theorem 24

Given $P = \{x \mid Ax = b, x \geq 0\}$. x is extreme point iff there exists $B \subseteq \{1, \dots, n\}$ with $|B| = m$ and

- ▶ A_B is non-singular
- ▶ $x_B = A_B^{-1}b \geq 0$
- ▶ $x_N = 0$

where $N = \{1, \dots, n\} \setminus B$.

Proof

Take $B = \{j \mid x_j > 0\}$ and augment with linearly independent columns until $|B| = m$; always possible since $\text{rank}(A) = m$.

Basic Feasible Solutions

$x \in \mathbb{R}^n$ is called **basic solution** (Basislösung) if $Ax = b$ and $\text{rank}(A_J) = |J|$ where $J = \{j \mid x_j \neq 0\}$;

x is a **basic feasible solution** (gültige Basislösung) if in addition $x \geq 0$.

A **basis** (Basis) is an index set $B \subseteq \{1, \dots, n\}$ with $\text{rank}(A_B) = m$ and $|B| = m$.

$x \in \mathbb{R}^n$ with $A_B x_B = b$ and $x_j = 0$ for all $j \notin B$ is **the basic solution associated to basis B** (die zu B assoziierte Basislösung)

Basic Feasible Solutions

$x \in \mathbb{R}^n$ is called **basic solution** (**Basislösung**) if $Ax = b$ and $\text{rank}(A_J) = |J|$ where $J = \{j \mid x_j \neq 0\}$;

x is a **basic feasible solution** (gültige Basislösung) if in addition $x \geq 0$.

A **basis** (Basis) is an index set $B \subseteq \{1, \dots, n\}$ with $\text{rank}(A_B) = m$ and $|B| = m$.

$x \in \mathbb{R}^n$ with $A_B x_B = b$ and $x_j = 0$ for all $j \notin B$ is the **basic solution associated to basis B** (die zu B assoziierte Basislösung)

Basic Feasible Solutions

$x \in \mathbb{R}^n$ is called **basic solution** (**Basislösung**) if $Ax = b$ and $\text{rank}(A_J) = |J|$ where $J = \{j \mid x_j \neq 0\}$;

x is a **basic feasible solution** (**gültige Basislösung**) if in addition $x \geq 0$.

A **basis** (**Basis**) is an index set $B \subseteq \{1, \dots, n\}$ with $\text{rank}(A_B) = m$ and $|B| = m$.

$x \in \mathbb{R}^n$ with $A_B x_B = b$ and $x_j = 0$ for all $j \notin B$ is the **basic solution associated to basis B** (die zu B assoziierte Basislösung)

Basic Feasible Solutions

$x \in \mathbb{R}^n$ is called **basic solution** (**Basislösung**) if $Ax = b$ and $\text{rank}(A_J) = |J|$ where $J = \{j \mid x_j \neq 0\}$;

x is a **basic feasible solution** (**gültige Basislösung**) if in addition $x \geq 0$.

A **basis** (**Basis**) is an index set $B \subseteq \{1, \dots, n\}$ with $\text{rank}(A_B) = m$ and $|B| = m$.

$x \in \mathbb{R}^n$ with $A_B x_B = b$ and $x_j = 0$ for all $j \notin B$ is the **basic solution associated to basis B** (die zu B assoziierte Basislösung)

Basic Feasible Solutions

$x \in \mathbb{R}^n$ is called **basic solution** (**Basislösung**) if $Ax = b$ and $\text{rank}(A_J) = |J|$ where $J = \{j \mid x_j \neq 0\}$;

x is a **basic feasible solution** (**gültige Basislösung**) if in addition $x \geq 0$.

A **basis** (**Basis**) is an index set $B \subseteq \{1, \dots, n\}$ with $\text{rank}(A_B) = m$ and $|B| = m$.

$x \in \mathbb{R}^n$ with $A_B x_B = b$ and $x_j = 0$ for all $j \notin B$ is the **basic solution associated to basis B** (**die zu B assoziierte Basislösung**)

Basic Feasible Solutions

A BFS fulfills the m equality constraints.

In addition, at least $n - m$ of the x_i 's are zero. The corresponding non-negativity constraint is fulfilled with equality.

Fact:

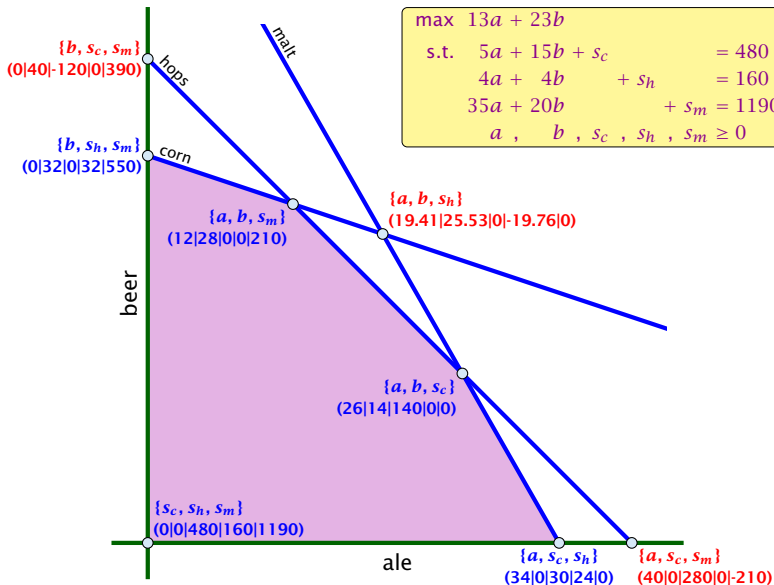
In a BFS at least n constraints are fulfilled with equality.

Basic Feasible Solutions

Definition 25

For a general LP ($\max\{c^T x \mid Ax \leq b\}$) with n variables a point x is a **basic feasible solution** if x is feasible and there exist n (linearly independent) constraints that are tight.

Algebraic View



Fundamental Questions

Linear Programming Problem (LP)

Let $A \in \mathbb{Q}^{m \times n}$, $b \in \mathbb{Q}^m$, $c \in \mathbb{Q}^n$, $\alpha \in \mathbb{Q}$. Does there exist $x \in \mathbb{Q}^n$
s.t. $Ax = b$, $x \geq 0$, $c^T x \geq \alpha$?

Questions:

- ▶ Is LP in NP? yes!
- ▶ Is LP in co-NP?
- ▶ Is LP in P?

Proof:

- ▶ Given a basis B we can compute the associated basis solution by calculating $A_B^{-1}b$ in polynomial time; then we can also compute the profit.

Fundamental Questions

Linear Programming Problem (LP)

Let $A \in \mathbb{Q}^{m \times n}$, $b \in \mathbb{Q}^m$, $c \in \mathbb{Q}^n$, $\alpha \in \mathbb{Q}$. Does there exist $x \in \mathbb{Q}^n$ s.t. $Ax = b$, $x \geq 0$, $c^T x \geq \alpha$?

Questions:

- ▶ Is LP in NP? yes!
- ▶ Is LP in co-NP?
- ▶ Is LP in P?

Proof:

- ▶ Given a basis B we can compute the associated basis solution by calculating $A_B^{-1}b$ in polynomial time; then we can also compute the profit.

Observation

We can compute an optimal solution to a linear program in time $\mathcal{O}\left(\binom{n}{m} \cdot \text{poly}(n, m)\right)$.

- ▶ there are only $\binom{n}{m}$ different bases.
- ▶ compute the profit of each of them and take the maximum

What happens if LP is unbounded?

4 Simplex Algorithm

Enumerating all basic feasible solutions (BFS), in order to find the optimum is slow.

Simplex Algorithm [George Dantzig 1947]

Move from BFS to adjacent BFS, without decreasing objective function.

Two BFSs are called adjacent if the bases just differ in one variable.

4 Simplex Algorithm

Enumerating all basic feasible solutions (BFS), in order to find the optimum is slow.

Simplex Algorithm [George Dantzig 1947]

Move from BFS to **adjacent** BFS, without decreasing objective function.

Two BFSs are called **adjacent** if the bases just differ in one variable.

4 Simplex Algorithm

$$\max 13a + 23b$$

$$\text{s.t. } 5a + 15b + s_c = 480$$

$$4a + 4b + s_h = 160$$

$$35a + 20b + s_m = 1190$$

$$a, b, s_c, s_h, s_m \geq 0$$

$$\max Z$$

$$13a + 23b - Z = 0$$

$$5a + 15b + s_c = 480$$

$$4a + 4b + s_h = 160$$

$$35a + 20b + s_m = 1190$$

$$a, b, s_c, s_h, s_m \geq 0$$

$$\text{basis} = \{s_c, s_h, s_m\}$$

$$a = b = 0$$

$$Z = 0$$

$$s_c = 480$$

$$s_h = 160$$

$$s_m = 1190$$

4 Simplex Algorithm

$$\begin{array}{llll} \max & 13a + 23b & & \\ \text{s.t.} & 5a + 15b + s_c & = & 480 \\ & 4a + 4b + s_h & = & 160 \\ & 35a + 20b + s_m & = & 1190 \\ & a, b, s_c, s_h, s_m & \geq & 0 \end{array}$$

$$\begin{array}{llll} \max & Z & & \\ & 13a + 23b & - Z & = 0 \\ & 5a + 15b + s_c & & = 480 \\ & 4a + 4b + s_h & & = 160 \\ & 35a + 20b + s_m & & = 1190 \\ & a, b, s_c, s_h, s_m & \geq & 0 \end{array}$$

$$\text{basis} = \{s_c, s_h, s_m\}$$

$$a = b = 0$$

$$Z = 0$$

$$s_c = 480$$

$$s_h = 160$$

$$s_m = 1190$$

Pivoting Step

max Z

$$13a + 23b \quad \quad \quad - Z = 0$$

$$5a + 15b + s_c \quad \quad \quad = 480$$

$$4a + 4b \quad \quad + s_h \quad \quad \quad = 160$$

$$35a + 20b \quad \quad \quad + s_m \quad \quad \quad = 1190$$

$$a, \quad b, \quad s_c, \quad s_h, \quad s_m \quad \geq 0$$

basis = $\{s_c, s_h, s_m\}$

$$a = b = 0$$

$$Z = 0$$

$$s_c = 480$$

$$s_h = 160$$

$$s_m = 1190$$

Pivoting Step

max Z

$$13a + 23b - Z = 0$$

$$5a + 15b + s_c = 480$$

$$4a + 4b + s_h = 160$$

$$35a + 20b + s_m = 1190$$

$$a, b, s_c, s_h, s_m \geq 0$$

basis = $\{s_c, s_h, s_m\}$

$$a = b = 0$$

$$Z = 0$$

$$s_c = 480$$

$$s_h = 160$$

$$s_m = 1190$$

- ▶ choose variable to bring into the basis
- ▶ chosen variable should have positive coefficient in objective function
- ▶ apply min-ratio test to find out by how much the variable can be increased
- ▶ pivot on row found by min-ratio test
- ▶ the existing basis variable in this row leaves the basis

Pivoting Step

max Z

$$13a + 23b - Z = 0$$

$$5a + 15b + s_c = 480$$

$$4a + 4b + s_h = 160$$

$$35a + 20b + s_m = 1190$$

$$a, b, s_c, s_h, s_m \geq 0$$

basis = $\{s_c, s_h, s_m\}$

$a = b = 0$

$Z = 0$

$s_c = 480$

$s_h = 160$

$s_m = 1190$

- ▶ choose variable to bring into the basis
- ▶ chosen variable should have positive coefficient in objective function
- ▶ apply min-ratio test to find out by how much the variable can be increased
- ▶ pivot on row found by min-ratio test
- ▶ the existing basis variable in this row leaves the basis

Pivoting Step

max Z

$$13a + 23b - Z = 0$$

$$5a + 15b + s_c = 480$$

$$4a + 4b + s_h = 160$$

$$35a + 20b + s_m = 1190$$

$$a, b, s_c, s_h, s_m \geq 0$$

basis = $\{s_c, s_h, s_m\}$

$a = b = 0$

$Z = 0$

$s_c = 480$

$s_h = 160$

$s_m = 1190$

- ▶ choose variable to bring into the basis
- ▶ chosen variable should have positive coefficient in objective function
- ▶ apply **min-ratio** test to find out by how much the variable can be increased
- ▶ pivot on row found by min-ratio test
- ▶ the existing basis variable in this row leaves the basis

Pivoting Step

max Z

$$13a + 23b - Z = 0$$

$$5a + 15b + s_c = 480$$

$$4a + 4b + s_h = 160$$

$$35a + 20b + s_m = 1190$$

$$a, b, s_c, s_h, s_m \geq 0$$

basis = $\{s_c, s_h, s_m\}$

$a = b = 0$

$Z = 0$

$s_c = 480$

$s_h = 160$

$s_m = 1190$

- ▶ choose variable to bring into the basis
- ▶ chosen variable should have positive coefficient in objective function
- ▶ apply **min-ratio** test to find out by how much the variable can be increased
- ▶ pivot on row found by min-ratio test
- ▶ the existing basis variable in this row leaves the basis

Pivoting Step

max Z

$$13a + 23b - Z = 0$$

$$5a + 15b + s_c = 480$$

$$4a + 4b + s_h = 160$$

$$35a + 20b + s_m = 1190$$

$$a, b, s_c, s_h, s_m \geq 0$$

basis = $\{s_c, s_h, s_m\}$

$$a = b = 0$$

$$Z = 0$$

$$s_c = 480$$

$$s_h = 160$$

$$s_m = 1190$$

- ▶ choose variable to bring into the basis
- ▶ chosen variable should have positive coefficient in objective function
- ▶ apply **min-ratio** test to find out by how much the variable can be increased
- ▶ pivot on row found by min-ratio test
- ▶ the existing basis variable in this row leaves the basis

max Z

$$13a + 23b \quad \quad \quad - Z = 0$$

$$5a + 15b + s_c \quad \quad \quad = 480$$

$$4a + 4b \quad \quad + s_h \quad \quad \quad = 160$$

$$35a + 20b \quad \quad \quad + s_m \quad \quad \quad = 1190$$

$$a, \quad b, \quad s_c, \quad s_h, \quad s_m \quad \quad \geq 0$$

basis = $\{s_c, s_h, s_m\}$

$a = b = 0$

$Z = 0$

$s_c = 480$

$s_h = 160$

$s_m = 1190$

max Z

$$13a + 23b - Z = 0$$

$$5a + 15b + s_c = 480$$

$$4a + 4b + s_h = 160$$

$$35a + 20b + s_m = 1190$$

$$a, b, s_c, s_h, s_m \geq 0$$

$$\text{basis} = \{s_c, s_h, s_m\}$$

$$a = b = 0$$

$$Z = 0$$

$$s_c = 480$$

$$s_h = 160$$

$$s_m = 1190$$

- Choose variable with coefficient > 0 as entering variable.

max Z

$$13a + 23b - Z = 0$$

$$5a + 15b + s_c = 480$$

$$4a + 4b + s_h = 160$$

$$35a + 20b + s_m = 1190$$

$$a, b, s_c, s_h, s_m \geq 0$$

basis = $\{s_c, s_h, s_m\}$

$$a = b = 0$$

$$Z = 0$$

$$s_c = 480$$

$$s_h = 160$$

$$s_m = 1190$$

- ▶ Choose variable with coefficient > 0 as entering variable.
- ▶ If we keep $a = 0$ and increase b from 0 to $\theta > 0$ s.t. all constraints ($Ax = b, x \geq 0$) are still fulfilled the objective value Z will strictly increase.

max Z

$$13a + 23b - Z = 0$$

$$5a + 15b + s_c = 480$$

$$4a + 4b + s_h = 160$$

$$35a + 20b + s_m = 1190$$

$$a, b, s_c, s_h, s_m \geq 0$$

basis = $\{s_c, s_h, s_m\}$

$a = b = 0$

$Z = 0$

$s_c = 480$

$s_h = 160$

$s_m = 1190$

- ▶ Choose variable with coefficient > 0 as entering variable.
- ▶ If we keep $a = 0$ and increase b from 0 to $\theta > 0$ s.t. all constraints ($Ax = b, x \geq 0$) are still fulfilled the objective value Z will strictly increase.
- ▶ For maintaining $Ax = b$ we need e.g. to set $s_c = 480 - 15\theta$.

max Z

$$13a + 23b - Z = 0$$

$$5a + 15b + s_c = 480$$

$$4a + 4b + s_h = 160$$

$$35a + 20b + s_m = 1190$$

$$a, b, s_c, s_h, s_m \geq 0$$

basis = $\{s_c, s_h, s_m\}$

$a = b = 0$

$Z = 0$

$s_c = 480$

$s_h = 160$

$s_m = 1190$

- ▶ Choose variable with coefficient > 0 as entering variable.
- ▶ If we keep $a = 0$ and increase b from 0 to $\theta > 0$ s.t. all constraints ($Ax = b, x \geq 0$) are still fulfilled the objective value Z will strictly increase.
- ▶ For maintaining $Ax = b$ we need e.g. to set $s_c = 480 - 15\theta$.
- ▶ Choosing $\theta = \min\{480/15, 160/4, 1190/20\}$ ensures that in the new solution one current basic variable becomes 0, and no variable goes negative.

max Z

$$13a + 23b - Z = 0$$

$$5a + 15b + s_c = 480$$

$$4a + 4b + s_h = 160$$

$$35a + 20b + s_m = 1190$$

$$a, b, s_c, s_h, s_m \geq 0$$

basis = $\{s_c, s_h, s_m\}$

$a = b = 0$

$Z = 0$

$s_c = 480$

$s_h = 160$

$s_m = 1190$

- ▶ Choose variable with coefficient > 0 as **entering variable**.
- ▶ If we keep $a = 0$ and increase b from 0 to $\theta > 0$ s.t. all constraints ($Ax = b, x \geq 0$) are still fulfilled the objective value Z will strictly increase.
- ▶ For maintaining $Ax = b$ we need e.g. to set $s_c = 480 - 15\theta$.
- ▶ Choosing $\theta = \min\{480/15, 160/4, 1190/20\}$ ensures that in the new solution one current basic variable becomes 0, and no variable goes negative.
- ▶ The basic variable in the row that gives $\min\{480/15, 160/4, 1190/20\}$ becomes the **leaving variable**.

max Z

$$13a + 23b - Z = 0$$

$$5a + 15b + s_c = 480$$

$$4a + 4b + s_h = 160$$

$$35a + 20b + s_m = 1190$$

$$a, b, s_c, s_h, s_m \geq 0$$

basis = $\{s_c, s_h, s_m\}$

$$a = b = 0$$

$$Z = 0$$

$$s_c = 480$$

$$s_h = 160$$

$$s_m = 1190$$

max Z

$$13a + 23b - Z = 0$$

$$5a + 15b + s_c = 480$$

$$4a + 4b + s_h = 160$$

$$35a + 20b + s_m = 1190$$

$$a, b, s_c, s_h, s_m \geq 0$$

basis = $\{s_c, s_h, s_m\}$

$$a = b = 0$$

$$Z = 0$$

$$s_c = 480$$

$$s_h = 160$$

$$s_m = 1190$$

Substitute $b = \frac{1}{15}(480 - 5a - s_c)$.

max Z

$$13a + 23b - Z = 0$$

$$5a + 15b + s_c = 480$$

$$4a + 4b + s_h = 160$$

$$35a + 20b + s_m = 1190$$

$$a, b, s_c, s_h, s_m \geq 0$$

basis = $\{s_c, s_h, s_m\}$

$$a = b = 0$$

$$Z = 0$$

$$s_c = 480$$

$$s_h = 160$$

$$s_m = 1190$$

Substitute $b = \frac{1}{15}(480 - 5a - s_c)$.

max Z

$$\frac{16}{3}a - \frac{23}{15}s_c - Z = -736$$

$$\frac{1}{3}a + b + \frac{1}{15}s_c = 32$$

$$\frac{8}{3}a - \frac{4}{15}s_c + s_h = 32$$

$$\frac{85}{3}a - \frac{4}{3}s_c + s_m = 550$$

$$a, b, s_c, s_h, s_m \geq 0$$

basis = $\{b, s_h, s_m\}$

$$a = s_c = 0$$

$$Z = 736$$

$$b = 32$$

$$s_h = 32$$

$$s_m = 550$$

max Z

$$\frac{16}{3}a - \frac{23}{15}s_c - Z = -736$$

$$\frac{1}{3}a + b + \frac{1}{15}s_c = 32$$

$$\frac{8}{3}a - \frac{4}{15}s_c + s_h = 32$$

$$\frac{85}{3}a - \frac{4}{3}s_c + s_m = 550$$

$$a, b, s_c, s_h, s_m \geq 0$$

basis = $\{b, s_h, s_m\}$

$$a = s_c = 0$$

$$Z = 736$$

$$b = 32$$

$$s_h = 32$$

$$s_m = 550$$

max Z

$$\frac{16}{3}a - \frac{23}{15}s_c - Z = -736$$

$$\frac{1}{3}a + b + \frac{1}{15}s_c = 32$$

$$\frac{8}{3}a - \frac{4}{15}s_c + s_h = 32$$

$$\frac{85}{3}a - \frac{4}{3}s_c + s_m = 550$$

$$a, b, s_c, s_h, s_m \geq 0$$

basis = $\{b, s_h, s_m\}$

$$a = s_c = 0$$

$$Z = 736$$

$$b = 32$$

$$s_h = 32$$

$$s_m = 550$$

Choose variable a to bring into basis.

max Z

$$\frac{16}{3}a - \frac{23}{15}s_c - Z = -736$$

$$\frac{1}{3}a + b + \frac{1}{15}s_c = 32$$

$$\frac{8}{3}a - \frac{4}{15}s_c + s_h = 32$$

$$\frac{85}{3}a - \frac{4}{3}s_c + s_m = 550$$

$$a, b, s_c, s_h, s_m \geq 0$$

basis = $\{b, s_h, s_m\}$

$a = s_c = 0$

$Z = 736$

$b = 32$

$s_h = 32$

$s_m = 550$

Choose variable a to bring into basis.

Computing $\min\{3 \cdot 32, 3 \cdot 32/8, 3 \cdot 550/85\}$ means pivot on line 2.

max Z

$$\frac{16}{3}a - \frac{23}{15}s_c - Z = -736$$

$$\frac{1}{3}a + b + \frac{1}{15}s_c = 32$$

$$\frac{8}{3}a - \frac{4}{15}s_c + s_h = 32$$

$$\frac{85}{3}a - \frac{4}{3}s_c + s_m = 550$$

$$a, b, s_c, s_h, s_m \geq 0$$

basis = $\{b, s_h, s_m\}$

$$a = s_c = 0$$

$$Z = 736$$

$$b = 32$$

$$s_h = 32$$

$$s_m = 550$$

Choose variable a to bring into basis.

Computing $\min\{3 \cdot 32, 3 \cdot 32/8, 3 \cdot 550/85\}$ means pivot on line 2.

Substitute $a = \frac{3}{8}(32 + \frac{4}{15}s_c - s_h)$.

max Z

$$\frac{16}{3}a - \frac{23}{15}s_c - Z = -736$$

$$\frac{1}{3}a + b + \frac{1}{15}s_c = 32$$

$$\frac{8}{3}a - \frac{4}{15}s_c + s_h = 32$$

$$\frac{85}{3}a - \frac{4}{3}s_c + s_m = 550$$

$$a, b, s_c, s_h, s_m \geq 0$$

basis = $\{b, s_h, s_m\}$

$$a = s_c = 0$$

$$Z = 736$$

$$b = 32$$

$$s_h = 32$$

$$s_m = 550$$

Choose variable a to bring into basis.

Computing $\min\{3 \cdot 32, 3 \cdot 32/8, 3 \cdot 550/85\}$ means pivot on line 2.

Substitute $a = \frac{3}{8}(32 + \frac{4}{15}s_c - s_h)$.

max Z

$$-s_c - 2s_h - Z = -800$$

$$b + \frac{1}{10}s_c - \frac{1}{8}s_h = 28$$

$$a - \frac{1}{10}s_c + \frac{3}{8}s_h = 12$$

$$\frac{3}{2}s_c - \frac{85}{8}s_h + s_m = 210$$

$$a, b, s_c, s_h, s_m \geq 0$$

basis = $\{a, b, s_m\}$

$$s_c = s_h = 0$$

$$Z = 800$$

$$b = 28$$

$$a = 12$$

$$s_m = 210$$

4 Simplex Algorithm

Pivoting stops when all coefficients in the objective function are non-positive.

Solution is optimal:

- If any feasible solution satisfies all constraints in the problem, then any feasible solution satisfies all constraints in the dual problem.
- If any feasible solution satisfies all constraints in the dual problem, then any feasible solution satisfies all constraints in the primal problem.
- If a primal solution value $\leq$ at most every dual solution value, then the current solution has value $\geq$ at least every dual solution value.

4 Simplex Algorithm

Pivoting stops when all coefficients in the objective function are non-positive.

Solution is optimal:

4 Simplex Algorithm

Pivoting stops when all coefficients in the objective function are non-positive.

Solution is optimal:

- ▶ any feasible solution satisfies all equations in the tableaux
- ▶ in particular: $Z = 800 - s_c - 2s_h$, $s_c \geq 0, s_h \geq 0$
- ▶ hence optimum solution value is at most 800
- ▶ the current solution has value 800

4 Simplex Algorithm

Pivoting stops when all coefficients in the objective function are non-positive.

Solution is optimal:

- ▶ any feasible solution satisfies all equations in the tableaux
- ▶ in particular: $Z = 800 - s_c - 2s_h$, $s_c \geq 0, s_h \geq 0$
 - ▶ hence optimum solution value is at most 800
 - ▶ the current solution has value 800

4 Simplex Algorithm

Pivoting stops when all coefficients in the objective function are non-positive.

Solution is optimal:

- ▶ any feasible solution satisfies all equations in the tableaux
- ▶ in particular: $Z = 800 - s_c - 2s_h$, $s_c \geq 0, s_h \geq 0$
- ▶ hence optimum solution value is at most 800
- ▶ the current solution has value 800

4 Simplex Algorithm

Pivoting stops when all coefficients in the objective function are non-positive.

Solution is optimal:

- ▶ any feasible solution satisfies all equations in the tableaux
- ▶ in particular: $Z = 800 - s_c - 2s_h$, $s_c \geq 0, s_h \geq 0$
- ▶ hence optimum solution value is at most 800
- ▶ the current solution has value 800

Matrix View

Let our linear program be

$$\begin{aligned}c_B^T x_B + c_N^T x_N &= Z \\ A_B x_B + A_N x_N &= b \\ x_B, x_N &\geq 0\end{aligned}$$

The simplex tableaux for basis B is

$$\begin{aligned}I x_B + (c_N^T - c_B^T A_B^{-1} A_N) x_N &= Z - c_B^T A_B^{-1} b \\ A_B^{-1} A_N x_N &= A_B^{-1} b \\ x_B, x_N &\geq 0\end{aligned}$$

The BFS is given by $x_N = 0, x_B = A_B^{-1} b$.

If $(c_N^T - c_B^T A_B^{-1} A_N) \leq 0$ we know that we have an optimum solution.

Matrix View

Let our linear program be

$$\begin{aligned}c_B^T x_B + c_N^T x_N &= Z \\ A_B x_B + A_N x_N &= b \\ x_B, x_N &\geq 0\end{aligned}$$

The simplex tableaux for basis B is

$$\begin{aligned}(c_N^T - c_B^T A_B^{-1} A_N) x_N &= Z - c_B^T A_B^{-1} b \\ I x_B + A_B^{-1} A_N x_N &= A_B^{-1} b \\ x_B, x_N &\geq 0\end{aligned}$$

The BFS is given by $x_N = 0, x_B = A_B^{-1} b$.

If $(c_N^T - c_B^T A_B^{-1} A_N) \leq 0$ we know that we have an optimum solution.

Matrix View

Let our linear program be

$$\begin{aligned}c_B^T x_B + c_N^T x_N &= Z \\ A_B x_B + A_N x_N &= b \\ x_B, x_N &\geq 0\end{aligned}$$

The simplex tableaux for basis B is

$$\begin{aligned}(c_N^T - c_B^T A_B^{-1} A_N) x_N &= Z - c_B^T A_B^{-1} b \\ I x_B + A_B^{-1} A_N x_N &= A_B^{-1} b \\ x_B, x_N &\geq 0\end{aligned}$$

The BFS is given by $x_N = 0, x_B = A_B^{-1} b$.

If $(c_N^T - c_B^T A_B^{-1} A_N) \leq 0$ we know that we have an optimum solution.

Matrix View

Let our linear program be

$$\begin{aligned}c_B^T x_B + c_N^T x_N &= Z \\ A_B x_B + A_N x_N &= b \\ x_B, x_N &\geq 0\end{aligned}$$

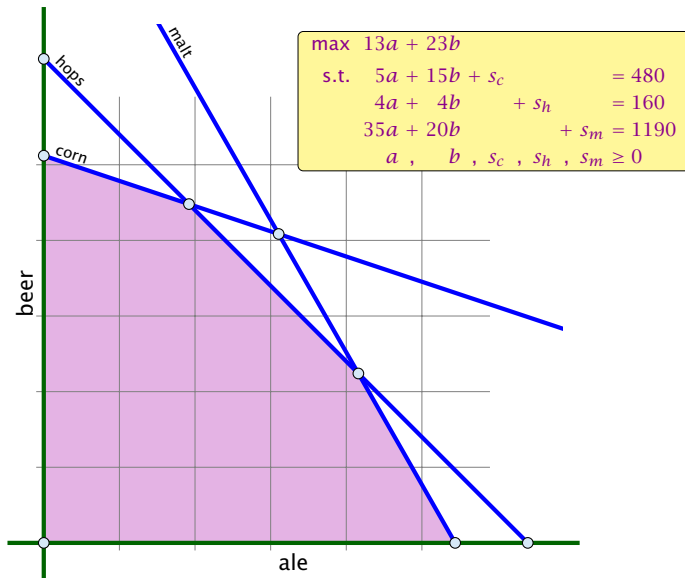
The simplex tableaux for basis B is

$$\begin{aligned}(c_N^T - c_B^T A_B^{-1} A_N) x_N &= Z - c_B^T A_B^{-1} b \\ I x_B + A_B^{-1} A_N x_N &= A_B^{-1} b \\ x_B, x_N &\geq 0\end{aligned}$$

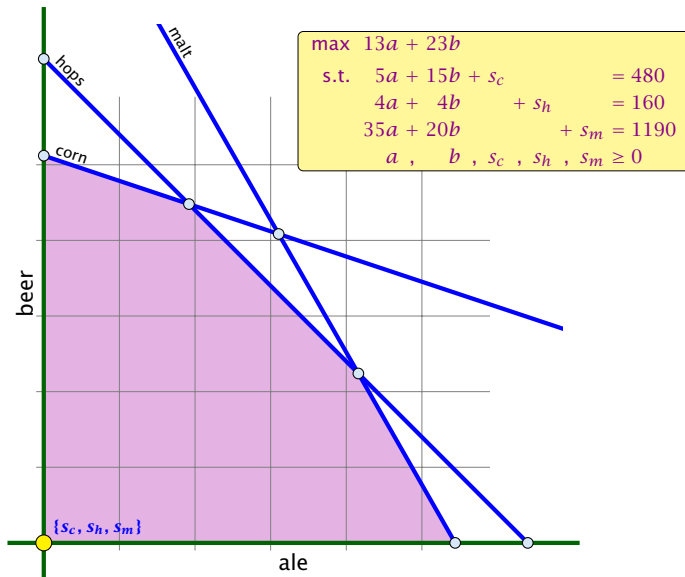
The BFS is given by $x_N = 0, x_B = A_B^{-1} b$.

If $(c_N^T - c_B^T A_B^{-1} A_N) \leq 0$ we know that we have an optimum solution.

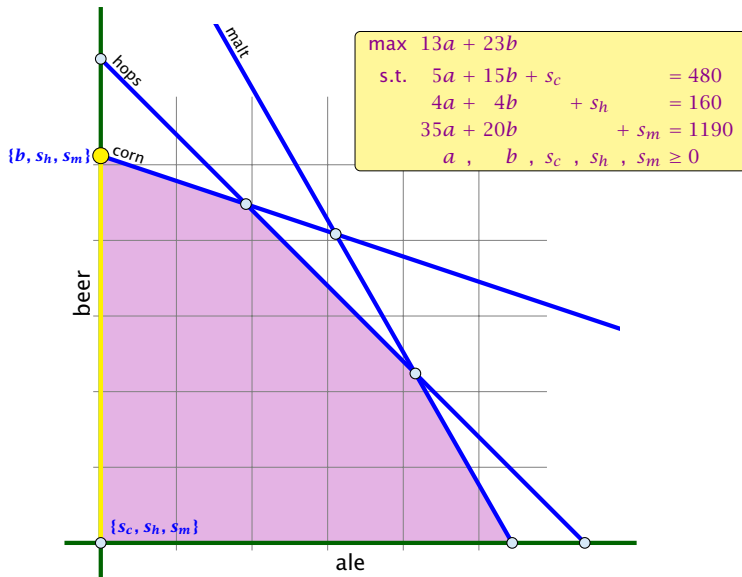
Geometric View of Pivoting



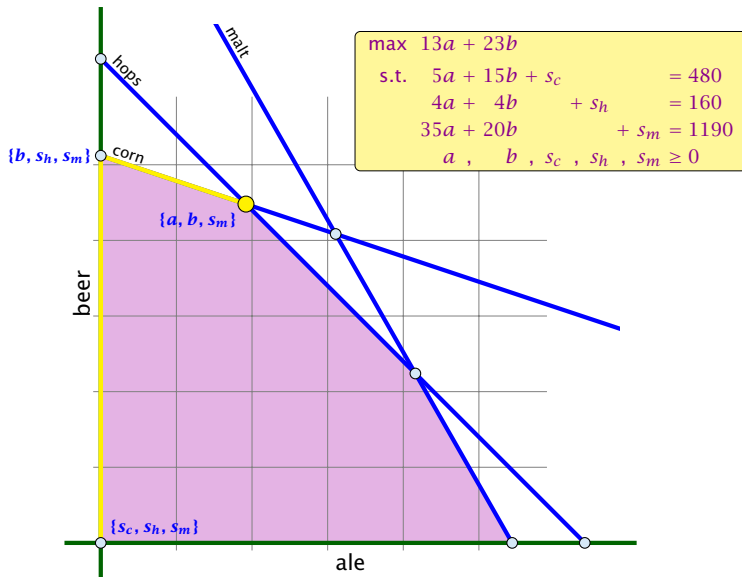
Geometric View of Pivoting



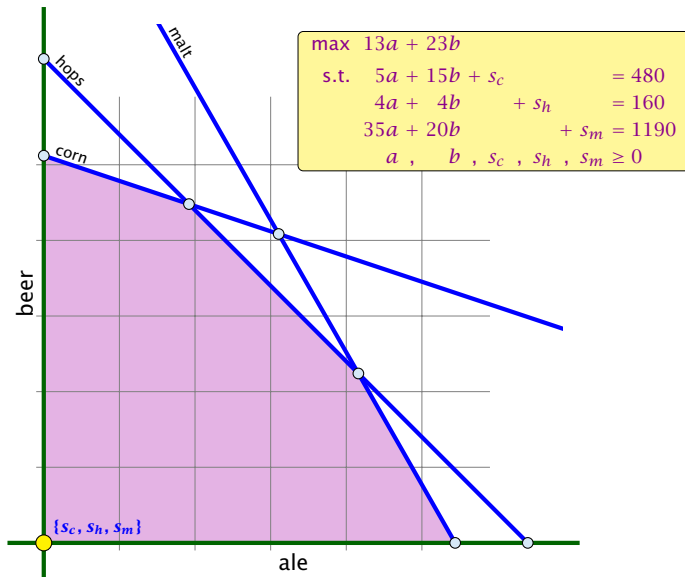
Geometric View of Pivoting



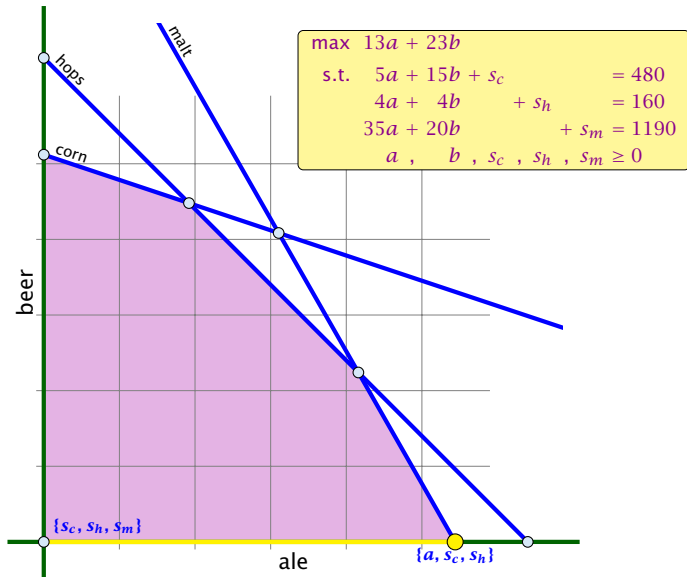
Geometric View of Pivoting



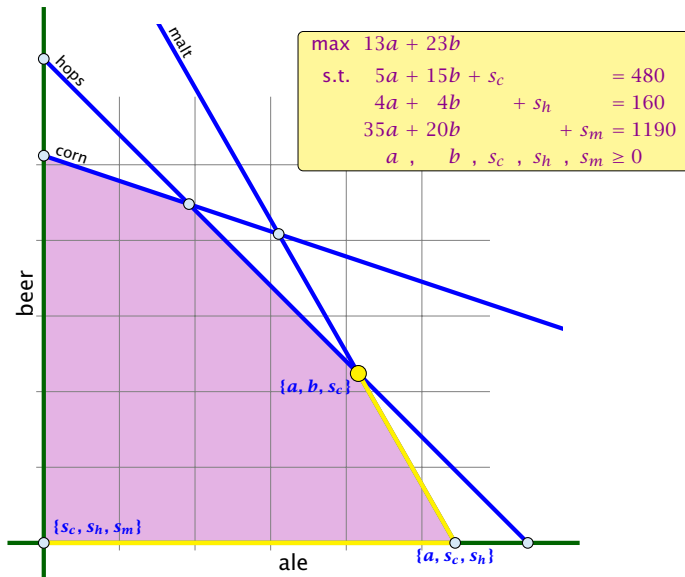
Geometric View of Pivoting



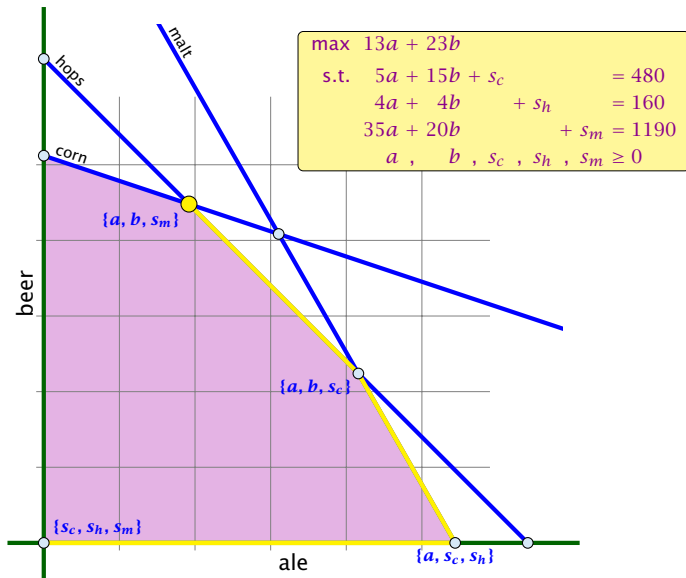
Geometric View of Pivoting



Geometric View of Pivoting



Geometric View of Pivoting



Algebraic Definition of Pivoting

- ▶ Given basis B with BFS x^* .
- ▶ Choose index $j \notin B$ in order to increase x_j^* from 0 to $\theta > 0$.
 - Other non-basis variables should stay at 0.
 - Basis variables change to maintain feasibility.
- ▶ Go from x^* to $x^* + \theta \cdot d$.

Requirements for d :

1. $d_j = 1$ (normalization)

2. $d_B = -A_B^{-1} A_j$ (feasibility, d is a direction)

3. d must hold, hence $d_B = -A_B^{-1} A_j$ (feasibility, d is a direction)

4. d must hold, hence $d_B = -A_B^{-1} A_j$ (feasibility, d is a direction)

5. d must hold, hence $d_B = -A_B^{-1} A_j$ (feasibility, d is a direction)

Algebraic Definition of Pivoting

- ▶ Given basis B with BFS x^* .
- ▶ Choose index $j \notin B$ in order to increase x_j^* from 0 to $\theta > 0$.
 - ▶ Other non-basis variables should stay at 0.
 - ▶ Basis variables change to maintain feasibility.
- ▶ Go from x^* to $x^* + \theta \cdot d$.

Requirements for d :

1. $d_j = 1$ (normalization)

2. $d_B = -A_B^{-1} A_j$ (feasibility, d_B is the pivot row)

3. $d_{\text{non-basis}} = 0$ must hold, hence d is the j -th column of $A_B^{-1} A$

4. d is the j -th column of $A_B^{-1} A$ (which gives $d_j = 1$)

Algebraic Definition of Pivoting

- ▶ Given basis B with BFS x^* .
- ▶ Choose index $j \notin B$ in order to increase x_j^* from 0 to $\theta > 0$.
 - ▶ Other non-basis variables should stay at 0.
 - ▶ Basis variables change to maintain feasibility.
- ▶ Go from x^* to $x^* + \theta \cdot d$.

Requirements for d :

- ▶ $d_j = 1$ (normalization)
- ▶ $d_i = 0$ for all $i \notin B$
- ▶ $d_i \leq 0$ for all $i \in B$ (feasibility must hold, hence $\theta \geq 0$)
- ▶ d is a direction

Algebraic Definition of Pivoting

- ▶ Given basis B with BFS x^* .
- ▶ Choose index $j \notin B$ in order to increase x_j^* from 0 to $\theta > 0$.
 - ▶ Other non-basis variables should stay at 0.
 - ▶ Basis variables change to maintain feasibility.
- ▶ Go from x^* to $x^* + \theta \cdot d$.

Requirements for d :

1. $d_j = 1$ (normalization)

2. $d_B = -A_B^{-1} A_j$

3. d must be feasible, hence $d \geq 0$

4. d must be a direction, hence $d \neq 0$

Algebraic Definition of Pivoting

- ▶ Given basis B with BFS x^* .
- ▶ Choose index $j \notin B$ in order to increase x_j^* from 0 to $\theta > 0$.
 - ▶ Other non-basis variables should stay at 0.
 - ▶ Basis variables change to maintain feasibility.
- ▶ Go from x^* to $x^* + \theta \cdot d$.

Requirements for d :

• $d_j = 1$ (normalization)

• $d_i = 0$ for all $i \notin B$

• $d_B = -a_B^{-1} a_j$ (maintain feasibility)

• $d_B \geq 0$ (feasibility of $x^* + \theta \cdot d$ for all $\theta \geq 0$)

Algebraic Definition of Pivoting

- ▶ Given basis B with BFS x^* .
- ▶ Choose index $j \notin B$ in order to increase x_j^* from 0 to $\theta > 0$.
 - ▶ Other non-basis variables should stay at 0.
 - ▶ Basis variables change to maintain feasibility.
- ▶ Go from x^* to $x^* + \theta \cdot d$.

Requirements for d :

- ▶ $d_j = 1$ (normalization)
- ▶ $d_\ell = 0, \ell \notin B, \ell \neq j$
- ▶ $A(x^* + \theta d) = b$ must hold. Hence $Ad = 0$.
- ▶ Altogether: $A_B d_B + A_{*j} = Ad = 0$, which gives $d_B = -A_B^{-1} A_{*j}$.

Algebraic Definition of Pivoting

- ▶ Given basis B with BFS x^* .
- ▶ Choose index $j \notin B$ in order to increase x_j^* from 0 to $\theta > 0$.
 - ▶ Other non-basis variables should stay at 0.
 - ▶ Basis variables change to maintain feasibility.
- ▶ Go from x^* to $x^* + \theta \cdot d$.

Requirements for d :

- ▶ $d_j = 1$ (normalization)
- ▶ $d_\ell = 0, \ell \notin B, \ell \neq j$
- ▶ $A(x^* + \theta d) = b$ must hold. Hence $Ad = 0$.
- ▶ Altogether: $A_B d_B + A_{*j} = Ad = 0$, which gives $d_B = -A_B^{-1} A_{*j}$.

Algebraic Definition of Pivoting

- ▶ Given basis B with BFS x^* .
- ▶ Choose index $j \notin B$ in order to increase x_j^* from 0 to $\theta > 0$.
 - ▶ Other non-basis variables should stay at 0.
 - ▶ Basis variables change to maintain feasibility.
- ▶ Go from x^* to $x^* + \theta \cdot d$.

Requirements for d :

- ▶ $d_j = 1$ (normalization)
- ▶ $d_\ell = 0, \ell \notin B, \ell \neq j$
- ▶ $A(x^* + \theta d) = b$ must hold. Hence $Ad = 0$.
- ▶ Altogether: $A_B d_B + A_{*j} = Ad = 0$, which gives
 $d_B = -A_B^{-1} A_{*j}$.

Algebraic Definition of Pivoting

- ▶ Given basis B with BFS x^* .
- ▶ Choose index $j \notin B$ in order to increase x_j^* from 0 to $\theta > 0$.
 - ▶ Other non-basis variables should stay at 0.
 - ▶ Basis variables change to maintain feasibility.
- ▶ Go from x^* to $x^* + \theta \cdot d$.

Requirements for d :

- ▶ $d_j = 1$ (normalization)
- ▶ $d_\ell = 0, \ell \notin B, \ell \neq j$
- ▶ $A(x^* + \theta d) = b$ must hold. Hence $Ad = 0$.
- ▶ Altogether: $A_B d_B + A_{*j} = Ad = 0$, which gives $d_B = -A_B^{-1} A_{*j}$.

Algebraic Definition of Pivoting

Definition 26 (j -th basis direction)

Let B be a basis, and let $j \notin B$. The vector d with $d_j = 1$ and $d_\ell = 0, \ell \notin B, \ell \neq j$ and $d_B = -A_B^{-1}A_{*j}$ is called the j -th basis direction for B .

Going from x^* to $x^* + \theta \cdot d$ the objective function changes by

$$\theta \cdot c^T d = \theta(c_j - c_B^T A_B^{-1} A_{*j})$$

Algebraic Definition of Pivoting

Definition 26 (j -th basis direction)

Let B be a basis, and let $j \notin B$. The vector d with $d_j = 1$ and $d_\ell = 0, \ell \notin B, \ell \neq j$ and $d_B = -A_B^{-1}A_{*j}$ is called the j -th basis direction for B .

Going from x^* to $x^* + \theta \cdot d$ the objective function changes by

$$\theta \cdot c^T d = \theta(c_j - c_B^T A_B^{-1} A_{*j})$$

Algebraic Definition of Pivoting

Definition 27 (Reduced Cost)

For a basis B the value

$$\tilde{c}_j = c_j - c_B^T A_B^{-1} A_{*j}$$

is called the **reduced cost** for variable x_j .

Note that this is defined for every j . If $j \in B$ then the above term is 0.

Algebraic Definition of Pivoting

Let our linear program be

$$\begin{aligned}c_B^T x_B + c_N^T x_N &= Z \\ A_B x_B + A_N x_N &= b \\ x_B, x_N &\geq 0\end{aligned}$$

The simplex tableaux for basis B is

$$\begin{aligned}(c_N^T - c_B^T A_B^{-1} A_N) x_N &= Z - c_B^T A_B^{-1} b \\ I x_B + A_B^{-1} A_N x_N &= A_B^{-1} b \\ x_B, x_N &\geq 0\end{aligned}$$

The BFS is given by $x_N = 0, x_B = A_B^{-1} b$.

If $(c_N^T - c_B^T A_B^{-1} A_N) \leq 0$ we know that we have an optimum solution.

Algebraic Definition of Pivoting

Let our linear program be

$$\begin{aligned}c_B^T x_B + c_N^T x_N &= Z \\ A_B x_B + A_N x_N &= b \\ x_B, x_N &\geq 0\end{aligned}$$

The simplex tableaux for basis B is

$$\begin{aligned}(c_N^T - c_B^T A_B^{-1} A_N) x_N &= Z - c_B^T A_B^{-1} b \\ I x_B + A_B^{-1} A_N x_N &= A_B^{-1} b \\ x_B, x_N &\geq 0\end{aligned}$$

The BFS is given by $x_N = 0, x_B = A_B^{-1} b$.

If $(c_N^T - c_B^T A_B^{-1} A_N) \leq 0$ we know that we have an optimum solution.

Algebraic Definition of Pivoting

Let our linear program be

$$\begin{aligned}c_B^T x_B + c_N^T x_N &= Z \\ A_B x_B + A_N x_N &= b \\ x_B, x_N &\geq 0\end{aligned}$$

The simplex tableaux for basis B is

$$\begin{aligned}(c_N^T - c_B^T A_B^{-1} A_N) x_N &= Z - c_B^T A_B^{-1} b \\ I x_B + A_B^{-1} A_N x_N &= A_B^{-1} b \\ x_B, x_N &\geq 0\end{aligned}$$

The BFS is given by $x_N = 0, x_B = A_B^{-1} b$.

If $(c_N^T - c_B^T A_B^{-1} A_N) \leq 0$ we know that we have an optimum solution.

Algebraic Definition of Pivoting

Let our linear program be

$$\begin{aligned}c_B^T x_B + c_N^T x_N &= Z \\ A_B x_B + A_N x_N &= b \\ x_B, x_N &\geq 0\end{aligned}$$

The simplex tableaux for basis B is

$$\begin{aligned}(c_N^T - c_B^T A_B^{-1} A_N) x_N &= Z - c_B^T A_B^{-1} b \\ I x_B + A_B^{-1} A_N x_N &= A_B^{-1} b \\ x_B, x_N &\geq 0\end{aligned}$$

The BFS is given by $x_N = 0, x_B = A_B^{-1} b$.

If $(c_N^T - c_B^T A_B^{-1} A_N) \leq 0$ we know that we have an optimum solution.

4 Simplex Algorithm

Questions:

What happens if the min ratio test fails to give us a value by which we can safely increase the entering variable?

How do we find the initial basic feasible solution?

Is there always a basis? Why/How?

What happens if the min ratio test fails to give us a value by which we can safely increase the entering variable?

Then we can terminate, because we know that the solution is unbounded.

What happens if the min ratio test fails to give us a value by which we can safely increase the entering variable?

What happens if the min ratio test fails to give us a value by which we can safely increase the entering variable?

4 Simplex Algorithm

Questions:

- ▶ What happens if the min ratio test fails to give us a value θ by which we can safely increase the entering variable?
- ▶ How do we find the initial basic feasible solution?
- ▶ Is there always a basis B such that

$$(c_N^T - c_B^T A_B^{-1} A_N) \leq 0 \quad ?$$

Then we can terminate because we know that the solution is optimal.

- ▶ If yes how do we make sure that we reach such a basis?

4 Simplex Algorithm

Questions:

- ▶ What happens if the min ratio test fails to give us a value θ by which we can safely increase the entering variable?
- ▶ How do we find the initial basic feasible solution?
- ▶ Is there always a basis B such that

$$(c_N^T - c_B^T A_B^{-1} A_N) \leq 0 \quad ?$$

Then we can terminate because we know that the solution is optimal.

- ▶ If yes how do we make sure that we reach such a basis?

4 Simplex Algorithm

Questions:

- ▶ What happens if the min ratio test fails to give us a value θ by which we can safely increase the entering variable?
- ▶ How do we find the initial basic feasible solution?
- ▶ Is there always a basis B such that

$$(c_N^T - c_B^T A_B^{-1} A_N) \leq 0 \quad ?$$

Then we can terminate because we know that the solution is optimal.

- ▶ If yes how do we make sure that we reach such a basis?

4 Simplex Algorithm

Questions:

- ▶ What happens if the min ratio test fails to give us a value θ by which we can safely increase the entering variable?
- ▶ How do we find the initial basic feasible solution?
- ▶ Is there always a basis B such that

$$(c_N^T - c_B^T A_B^{-1} A_N) \leq 0 \quad ?$$

Then we can terminate because we know that the solution is optimal.

- ▶ If yes how do we make sure that we reach such a basis?

Min Ratio Test

The min ratio test computes a value $\theta \geq 0$ such that after setting the entering variable to θ the leaving variable becomes 0 and all other variables stay non-negative.

For this, one computes b_i/A_{ie} for all constraints i and calculates the minimum positive value.

What does it mean that the ratio b_i/A_{ie} (and hence A_{ie}) is negative for a constraint?

This means that the corresponding basic variable will increase if we increase b . Hence, there is no danger of this basic variable becoming negative

What happens if all b_i/A_{ie} are negative? Then we do not have a leaving variable. Then the LP is unbounded!

Min Ratio Test

The min ratio test computes a value $\theta \geq 0$ such that after setting the entering variable to θ the leaving variable becomes 0 and all other variables stay non-negative.

For this, one computes b_i/A_{ie} for all constraints i and calculates the minimum positive value.

What does it mean that the ratio b_i/A_{ie} (and hence A_{ie}) is negative for a constraint?

This means that the corresponding basic variable will increase if we increase b . Hence, there is no danger of this basic variable becoming negative

What happens if all b_i/A_{ie} are negative? Then we do not have a leaving variable. Then the LP is unbounded!

Min Ratio Test

The min ratio test computes a value $\theta \geq 0$ such that after setting the entering variable to θ the leaving variable becomes 0 and all other variables stay non-negative.

For this, one computes b_i/A_{ie} for all constraints i and calculates the minimum positive value.

What does it mean that the ratio b_i/A_{ie} (and hence A_{ie}) is negative for a constraint?

This means that the corresponding basic variable will increase if we increase b . Hence, there is no danger of this basic variable becoming negative

What happens if all b_i/A_{ie} are negative? Then we do not have a leaving variable. Then the LP is unbounded!

Min Ratio Test

The min ratio test computes a value $\theta \geq 0$ such that after setting the entering variable to θ the leaving variable becomes 0 and all other variables stay non-negative.

For this, one computes b_i/A_{ie} for all constraints i and calculates the minimum positive value.

What does it mean that the ratio b_i/A_{ie} (and hence A_{ie}) is negative for a constraint?

This means that the corresponding basic variable will increase if we increase b . Hence, there is no danger of this basic variable becoming negative

What happens if all b_i/A_{ie} are negative? Then we do not have a leaving variable. Then the LP is unbounded!

Min Ratio Test

The min ratio test computes a value $\theta \geq 0$ such that after setting the entering variable to θ the leaving variable becomes 0 and all other variables stay non-negative.

For this, one computes b_i/A_{ie} for all constraints i and calculates the minimum positive value.

What does it mean that the ratio b_i/A_{ie} (and hence A_{ie}) is negative for a constraint?

This means that the corresponding basic variable will increase if we increase b . Hence, there is no danger of this basic variable becoming negative

What happens if **all** b_i/A_{ie} are negative? Then we do not have a leaving variable. Then the LP is unbounded!

Min Ratio Test

The min ratio test computes a value $\theta \geq 0$ such that after setting the entering variable to θ the leaving variable becomes 0 and all other variables stay non-negative.

For this, one computes b_i/A_{ie} for all constraints i and calculates the minimum positive value.

What does it mean that the ratio b_i/A_{ie} (and hence A_{ie}) is negative for a constraint?

This means that the corresponding basic variable will increase if we increase b . Hence, there is no danger of this basic variable becoming negative

What happens if **all** b_i/A_{ie} are negative? Then we do not have a leaving variable. **Then the LP is unbounded!**

Termination

The objective function does not decrease during one iteration of the simplex-algorithm.

Does it always increase?

Termination

The objective function does not decrease during one iteration of the simplex-algorithm.

Does it always increase?

Termination

The objective function does not decrease during one iteration of the simplex-algorithm.

Does it always increase?

Termination

The objective function may not increase!

Because a variable x_ℓ with $\ell \in B$ is already 0.

The set of inequalities is **degenerate** (also the basis is degenerate).

Definition 28 (Degeneracy)

A BFS x^* is called **degenerate** if the set $J = \{j \mid x_j^* > 0\}$ fulfills $|J| < m$.

It is possible that the algorithm **cycles**, i.e., it cycles through a sequence of different bases without ever terminating. Happens, very rarely in practise.

Termination

The objective function may not increase!

Because a variable x_ℓ with $\ell \in B$ is already 0.

The set of inequalities is **degenerate** (also the basis is degenerate).

Definition 28 (Degeneracy)

A BFS x^* is called **degenerate** if the set $J = \{j \mid x_j^* > 0\}$ fulfills $|J| < m$.

It is possible that the algorithm **cycles**, i.e., it cycles through a sequence of different bases without ever terminating. Happens, very rarely in practise.

Termination

The objective function may not increase!

Because a variable x_ℓ with $\ell \in B$ is already 0.

The set of inequalities is **degenerate** (also the basis is degenerate).

Definition 28 (Degeneracy)

A BFS x^* is called **degenerate** if the set $J = \{j \mid x_j^* > 0\}$ fulfills $|J| < m$.

It is possible that the algorithm **cycles**, i.e., it cycles through a sequence of different bases without ever terminating. Happens, very rarely in practise.

Termination

The objective function may not increase!

Because a variable x_ℓ with $\ell \in B$ is already 0.

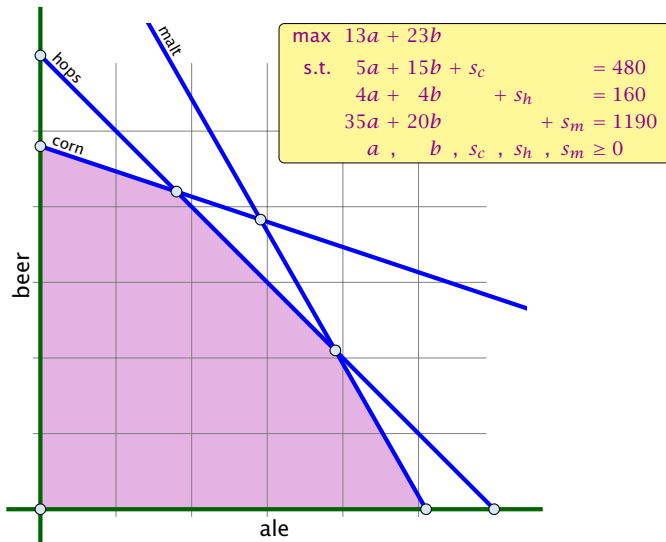
The set of inequalities is **degenerate** (also the basis is degenerate).

Definition 28 (Degeneracy)

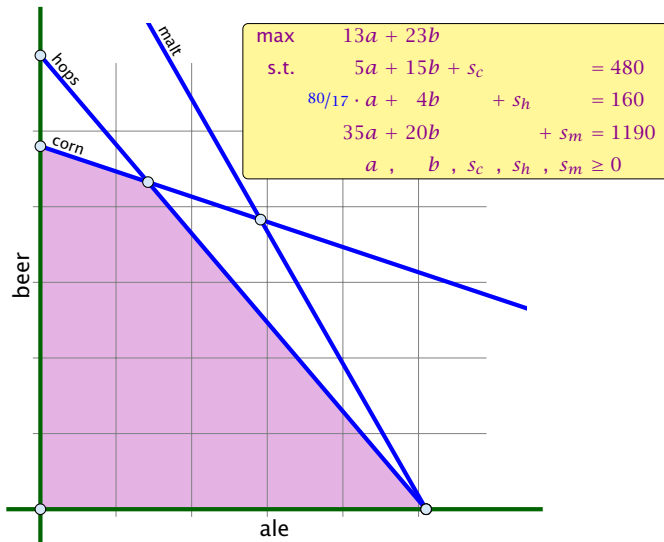
A BFS x^* is called **degenerate** if the set $J = \{j \mid x_j^* > 0\}$ fulfills $|J| < m$.

It is possible that the algorithm **cycles**, i.e., it cycles through a sequence of different bases without ever terminating. Happens, very rarely in practise.

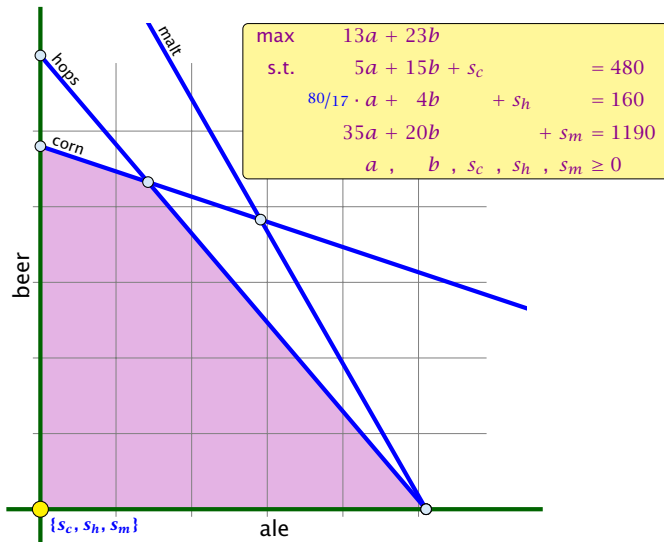
Non Degenerate Example



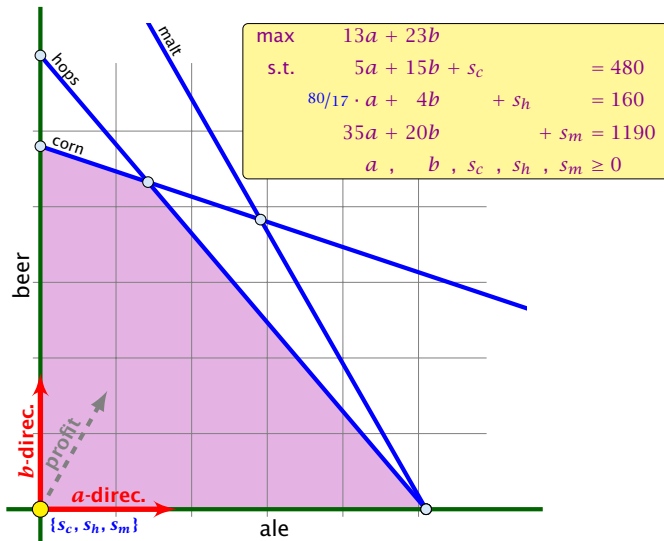
Degenerate Example



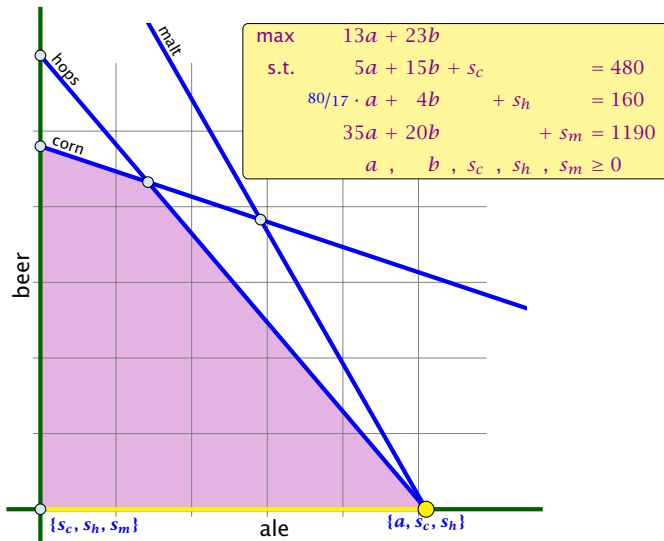
Degenerate Example



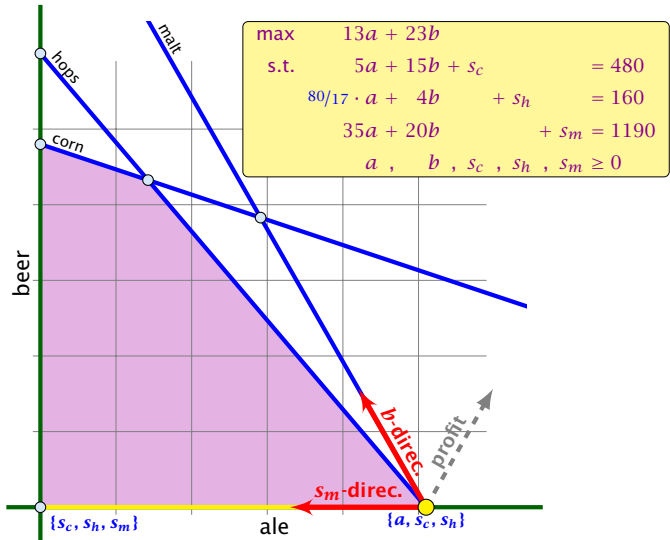
Degenerate Example



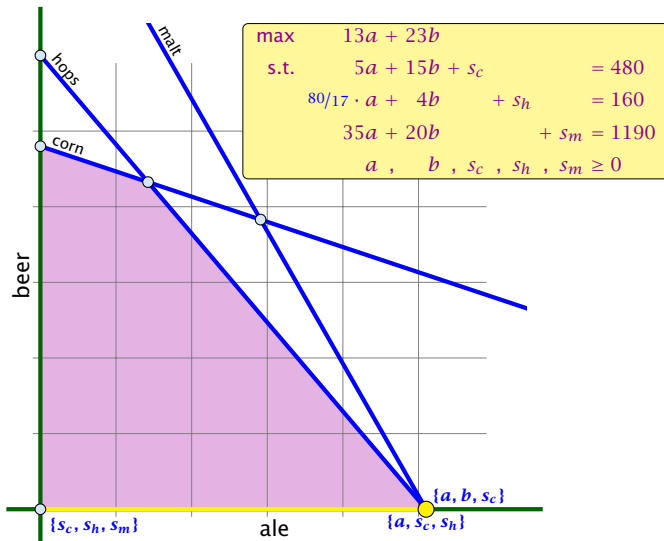
Degenerate Example



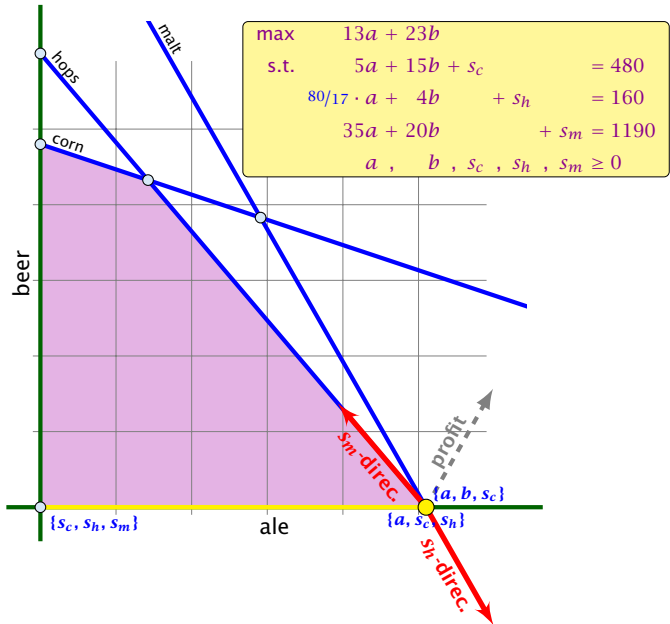
Degenerate Example



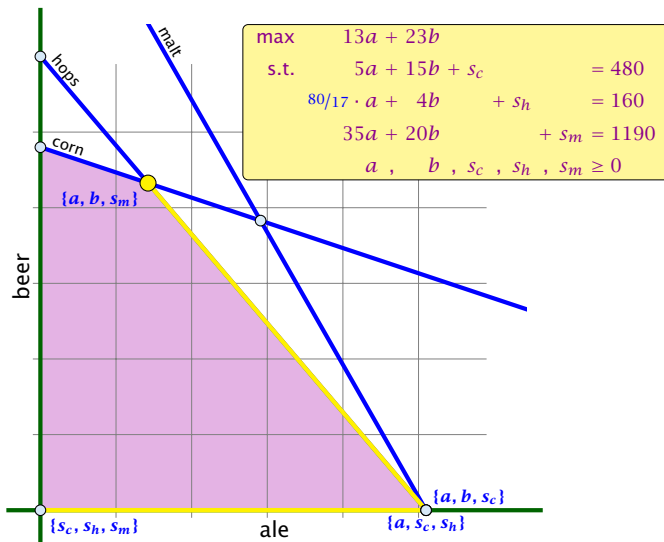
Degenerate Example



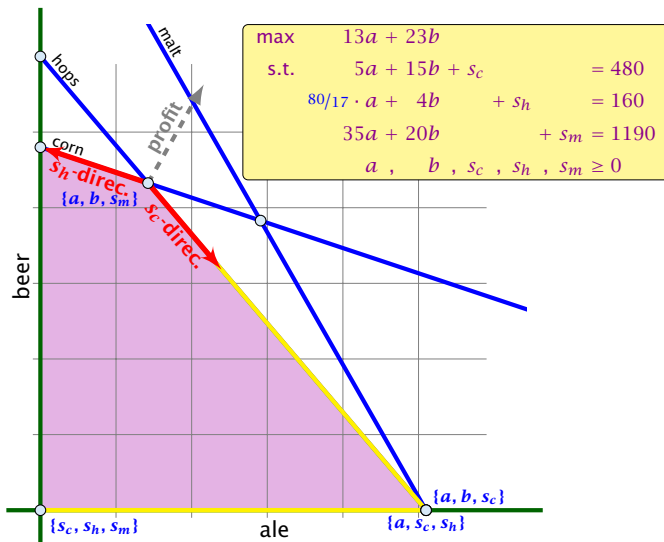
Degenerate Example



Degenerate Example



Degenerate Example



Summary: How to choose pivot-elements

- ▶ We can choose a column e as an entering variable if $\tilde{c}_e > 0$ ($\tilde{c}_e$ is reduced cost for x_e).
- ▶ The standard choice is the column that maximizes $\tilde{c}_e$.
- ▶ If $A_{ie} \leq 0$ for all $i \in \{1, \dots, m\}$ then the maximum is not bounded.
- ▶ Otw. choose a leaving variable ℓ such that $b_\ell / A_{\ell e}$ is minimal among all variables i with $A_{ie} > 0$.
- ▶ If several variables have minimum $b_\ell / A_{\ell e}$ you reach a **degenerate** basis.
- ▶ Depending on the choice of ℓ it may happen that the algorithm runs into a cycle where it does not escape from a degenerate vertex.

Summary: How to choose pivot-elements

- ▶ We can choose a column e as an entering variable if $\tilde{c}_e > 0$ ($\tilde{c}_e$ is reduced cost for x_e).
- ▶ The standard choice is the column that maximizes $\tilde{c}_e$.
- ▶ If $A_{ie} \leq 0$ for all $i \in \{1, \dots, m\}$ then the maximum is not bounded.
- ▶ Otw. choose a leaving variable ℓ such that $b_\ell / A_{\ell e}$ is minimal among all variables i with $A_{ie} > 0$.
- ▶ If several variables have minimum $b_\ell / A_{\ell e}$ you reach a **degenerate** basis.
- ▶ Depending on the choice of ℓ it may happen that the algorithm runs into a cycle where it does not escape from a degenerate vertex.

Summary: How to choose pivot-elements

- ▶ We can choose a column e as an entering variable if $\tilde{c}_e > 0$ ($\tilde{c}_e$ is reduced cost for x_e).
- ▶ The standard choice is the column that maximizes $\tilde{c}_e$.
- ▶ If $A_{ie} \leq 0$ for all $i \in \{1, \dots, m\}$ then the maximum is not bounded.
- ▶ Otw. choose a leaving variable ℓ such that $b_\ell / A_{\ell e}$ is minimal among all variables i with $A_{ie} > 0$.
- ▶ If several variables have minimum $b_\ell / A_{\ell e}$ you reach a **degenerate** basis.
- ▶ Depending on the choice of ℓ it may happen that the algorithm runs into a cycle where it does not escape from a degenerate vertex.

Summary: How to choose pivot-elements

- ▶ We can choose a column e as an entering variable if $\tilde{c}_e > 0$ ($\tilde{c}_e$ is reduced cost for x_e).
- ▶ The standard choice is the column that maximizes $\tilde{c}_e$.
- ▶ If $A_{ie} \leq 0$ for all $i \in \{1, \dots, m\}$ then the maximum is not bounded.
- ▶ Otw. choose a leaving variable ℓ such that $b_\ell / A_{\ell e}$ is minimal among all variables i with $A_{ie} > 0$.
- ▶ If several variables have minimum $b_\ell / A_{\ell e}$ you reach a **degenerate** basis.
- ▶ Depending on the choice of ℓ it may happen that the algorithm runs into a cycle where it does not escape from a degenerate vertex.

Summary: How to choose pivot-elements

- ▶ We can choose a column e as an entering variable if $\tilde{c}_e > 0$ ($\tilde{c}_e$ is reduced cost for x_e).
- ▶ The standard choice is the column that maximizes $\tilde{c}_e$.
- ▶ If $A_{ie} \leq 0$ for all $i \in \{1, \dots, m\}$ then the maximum is not bounded.
- ▶ Otw. choose a leaving variable ℓ such that $b_\ell / A_{\ell e}$ is minimal among all variables i with $A_{ie} > 0$.
- ▶ If several variables have minimum $b_\ell / A_{\ell e}$ you reach a **degenerate** basis.
- ▶ Depending on the choice of ℓ it may happen that the algorithm runs into a cycle where it does not escape from a degenerate vertex.

Summary: How to choose pivot-elements

- ▶ We can choose a column e as an entering variable if $\tilde{c}_e > 0$ ($\tilde{c}_e$ is reduced cost for x_e).
- ▶ The standard choice is the column that maximizes $\tilde{c}_e$.
- ▶ If $A_{ie} \leq 0$ for all $i \in \{1, \dots, m\}$ then the maximum is not bounded.
- ▶ Otw. choose a leaving variable ℓ such that $b_\ell / A_{\ell e}$ is minimal among all variables i with $A_{ie} > 0$.
- ▶ If several variables have minimum $b_\ell / A_{\ell e}$ you reach a **degenerate** basis.
- ▶ Depending on the choice of ℓ it may happen that the algorithm runs into a cycle where it does not escape from a degenerate vertex.

Termination

What do we have so far?

Suppose we are given an initial feasible solution to an LP. If the LP is non-degenerate then Simplex will terminate.

Note that we either terminate because the min-ratio test fails and we can conclude that the LP is **unbounded**, or we terminate because the vector of reduced cost is non-positive. In the latter case we have an **optimum solution**.

How do we come up with an initial solution?

- ▶ $Ax \leq b, x \geq 0$, and $b \geq 0$.
- ▶ The standard slack form for this problem is $Ax + Is = b, x \geq 0, s \geq 0$, where s denotes the vector of slack variables.
- ▶ Then $s = b, x = 0$ is a basic feasible solution (how?).
- ▶ We directly can start the simplex algorithm.

How do we find an initial basic feasible solution for an arbitrary problem?

How do we come up with an initial solution?

- ▶ $Ax \leq b, x \geq 0$, and $b \geq 0$.
- ▶ The standard slack form for this problem is $Ax + Is = b, x \geq 0, s \geq 0$, where s denotes the vector of slack variables.
- ▶ Then $s = b, x = 0$ is a basic feasible solution (how?).
- ▶ We directly can start the simplex algorithm.

How do we find an initial basic feasible solution for an arbitrary problem?

How do we come up with an initial solution?

- ▶ $Ax \leq b, x \geq 0$, and $b \geq 0$.
- ▶ The standard slack form for this problem is $Ax + Is = b, x \geq 0, s \geq 0$, where s denotes the vector of slack variables.
- ▶ Then $s = b, x = 0$ is a basic feasible solution (how?).
- ▶ We directly can start the simplex algorithm.

How do we find an initial basic feasible solution for an arbitrary problem?

How do we come up with an initial solution?

- ▶ $Ax \leq b, x \geq 0$, and $b \geq 0$.
- ▶ The standard slack form for this problem is $Ax + Is = b, x \geq 0, s \geq 0$, where s denotes the vector of slack variables.
- ▶ Then $s = b, x = 0$ is a basic feasible solution (how?).
- ▶ We directly can start the simplex algorithm.

How do we find an initial basic feasible solution for an arbitrary problem?

How do we come up with an initial solution?

- ▶ $Ax \leq b, x \geq 0$, and $b \geq 0$.
- ▶ The standard slack form for this problem is $Ax + Is = b, x \geq 0, s \geq 0$, where s denotes the vector of slack variables.
- ▶ Then $s = b, x = 0$ is a basic feasible solution (how?).
- ▶ We directly can start the simplex algorithm.

How do we find an initial basic feasible solution for an arbitrary problem?

Two phase algorithm

Suppose we want to maximize $c^T x$ s.t. $Ax = b, x \geq 0$.

Multiply all rows with $b_i < 0$ by -1 .

maximize $-c^T x$ s.t. $Ax = b, x \geq 0$, $b \geq 0$ using
Simplex. If $x_0, x_0 \geq 0$ is initial feasible.

If $x_0 = 0$ then the original problem is infeasible.

Otherwise you have $x_0 \geq 0$ with $Ax_0 = b$.

From this you can get basic feasible solution.

Now you can start the Simplex for the original problem.

Two phase algorithm

Suppose we want to maximize $c^T x$ s.t. $Ax = b, x \geq 0$.

1. Multiply all rows with $b_i < 0$ by -1 .
2. maximize $-\sum_i v_i$ s.t. $Ax + Iv = b, x \geq 0, v \geq 0$ using Simplex. $x = 0, v = b$ is initial feasible.
3. If $\sum_i v_i > 0$ then the original problem is **infeasible**.
4. Otw. you have $x \geq 0$ with $Ax = b$.
5. From this you can get basic feasible solution.
6. Now you can start the Simplex for the original problem.

Two phase algorithm

Suppose we want to maximize $c^T x$ s.t. $Ax = b, x \geq 0$.

1. Multiply all rows with $b_i < 0$ by -1 .
2. maximize $-\sum_i v_i$ s.t. $Ax + Iv = b, x \geq 0, v \geq 0$ using Simplex. $x = 0, v = b$ is initial feasible.
3. If $\sum_i v_i > 0$ then the original problem is infeasible.
4. Otw. you have $x \geq 0$ with $Ax = b$.
5. From this you can get basic feasible solution.
6. Now you can start the Simplex for the original problem.

Two phase algorithm

Suppose we want to maximize $c^T x$ s.t. $Ax = b, x \geq 0$.

1. Multiply all rows with $b_i < 0$ by -1 .
2. maximize $-\sum_i v_i$ s.t. $Ax + Iv = b, x \geq 0, v \geq 0$ using Simplex. $x = 0, v = b$ is initial feasible.
3. If $\sum_i v_i > 0$ then the original problem is **infeasible**.
4. Otw. you have $x \geq 0$ with $Ax = b$.
5. From this you can get basic feasible solution.
6. Now you can start the Simplex for the original problem.

Two phase algorithm

Suppose we want to maximize $c^T x$ s.t. $Ax = b, x \geq 0$.

1. Multiply all rows with $b_i < 0$ by -1 .
2. maximize $-\sum_i v_i$ s.t. $Ax + Iv = b, x \geq 0, v \geq 0$ using Simplex. $x = 0, v = b$ is initial feasible.
3. If $\sum_i v_i > 0$ then the original problem is **infeasible**.
4. Otw. you have $x \geq 0$ with $Ax = b$.
5. From this you can get basic feasible solution.
6. Now you can start the Simplex for the original problem.

Two phase algorithm

Suppose we want to maximize $c^T x$ s.t. $Ax = b, x \geq 0$.

1. Multiply all rows with $b_i < 0$ by -1 .
2. maximize $-\sum_i v_i$ s.t. $Ax + Iv = b, x \geq 0, v \geq 0$ using Simplex. $x = 0, v = b$ is initial feasible.
3. If $\sum_i v_i > 0$ then the original problem is **infeasible**.
4. Otw. you have $x \geq 0$ with $Ax = b$.
5. From this you can get basic feasible solution.
6. Now you can start the Simplex for the original problem.

Two phase algorithm

Suppose we want to maximize $c^T x$ s.t. $Ax = b, x \geq 0$.

1. Multiply all rows with $b_i < 0$ by -1 .
2. maximize $-\sum_i v_i$ s.t. $Ax + Iv = b, x \geq 0, v \geq 0$ using Simplex. $x = 0, v = b$ is initial feasible.
3. If $\sum_i v_i > 0$ then the original problem is **infeasible**.
4. Otw. you have $x \geq 0$ with $Ax = b$.
5. From this you can get basic feasible solution.
6. Now you can start the Simplex for the original problem.

Lemma 29

Let B be a basis and x^ a BFS corresponding to basis B . $\tilde{c} \leq 0$ implies that x^* is an optimum solution to the LP.*

Duality

How do we get an upper bound to a maximization LP?

$$\begin{array}{llll} \max & 13a & + & 23b \\ \text{s.t.} & 5a & + & 15b \leq 480 \\ & 4a & + & 4b \leq 160 \\ & 35a & + & 20b \leq 1190 \\ & & & a, b \geq 0 \end{array}$$

Note that a lower bound is easy to derive. Every choice of $a, b \geq 0$ gives us a lower bound (e.g. $a = 12, b = 28$ gives us a lower bound of 800).

If you take a conic combination of the rows (multiply the i -th row with $y_i \geq 0$) such that $\sum_i y_i a_{ij} \geq c_j$ then $\sum_i y_i b_i$ will be an upper bound.

Duality

How do we get an upper bound to a maximization LP?

$$\begin{array}{ll} \max & 13a + 23b \\ \text{s.t.} & 5a + 15b \leq 480 \\ & 4a + 4b \leq 160 \\ & 35a + 20b \leq 1190 \\ & a, b \geq 0 \end{array}$$

Note that a lower bound is easy to derive. Every choice of $a, b \geq 0$ gives us a lower bound (e.g. $a = 12, b = 28$ gives us a lower bound of 800).

If you take a conic combination of the rows (multiply the i -th row with $y_i \geq 0$) such that $\sum_i y_i a_{ij} \geq c_j$ then $\sum_i y_i b_i$ will be an upper bound.

Duality

How do we get an upper bound to a maximization LP?

$$\begin{array}{llll} \max & 13a & + & 23b \\ \text{s.t.} & 5a & + & 15b \leq 480 \\ & 4a & + & 4b \leq 160 \\ & 35a & + & 20b \leq 1190 \\ & & & a, b \geq 0 \end{array}$$

Note that a lower bound is easy to derive. Every choice of $a, b \geq 0$ gives us a lower bound (e.g. $a = 12, b = 28$ gives us a lower bound of 800).

If you take a conic combination of the rows (multiply the i -th row with $y_i \geq 0$) such that $\sum_i y_i a_{ij} \geq c_j$ then $\sum_i y_i b_i$ will be an upper bound.

Definition 30

Let $z = \max\{c^T x \mid Ax \leq b, x \geq 0\}$ be a linear program P (called the primal linear program).

The linear program D defined by

$$w = \min\{b^T y \mid A^T y \geq c, y \geq 0\}$$

is called the **dual problem**.

Lemma 31

The dual of the dual problem is the primal problem.

Proof:

The dual problem is

Lemma 31

The dual of the dual problem is the primal problem.

Proof:

$$\blacktriangleright w = \min\{b^T y \mid A^T y \geq c, y \geq 0\}$$

$$\blacktriangleright w = -\max\{-b^T y \mid -A^T y \leq -c, y \geq 0\}$$

The dual problem is

Lemma 31

The dual of the dual problem is the primal problem.

Proof:

- ▶ $w = \min\{b^T y \mid A^T y \geq c, y \geq 0\}$
- ▶ $w = -\max\{-b^T y \mid -A^T y \leq -c, y \geq 0\}$

The dual problem is

Lemma 31

The dual of the dual problem is the primal problem.

Proof:

- ▶ $w = \min\{b^T y \mid A^T y \geq c, y \geq 0\}$
- ▶ $w = -\max\{-b^T y \mid -A^T y \leq -c, y \geq 0\}$

The dual problem is

- ▶ $z = -\min\{-c^T x \mid -Ax \geq -b, x \geq 0\}$
- ▶ $z = \max\{c^T x \mid Ax \leq b, x \geq 0\}$

Lemma 31

The dual of the dual problem is the primal problem.

Proof:

- ▶ $w = \min\{b^T y \mid A^T y \geq c, y \geq 0\}$
- ▶ $w = -\max\{-b^T y \mid -A^T y \leq -c, y \geq 0\}$

The dual problem is

- ▶ $z = -\min\{-c^T x \mid -Ax \geq -b, x \geq 0\}$
- ▶ $z = \max\{c^T x \mid Ax \leq b, x \geq 0\}$

Weak Duality

Let $z = \max\{c^T x \mid Ax \leq b, x \geq 0\}$ and
 $w = \min\{b^T y \mid A^T y \geq c, y \geq 0\}$ be a primal dual pair.

x is primal feasible iff $x \in \{x \mid Ax \leq b, x \geq 0\}$

y is dual feasible, iff $y \in \{y \mid A^T y \geq c, y \geq 0\}$.

Theorem 32 (Weak Duality)

Let $\hat{x}$ be primal feasible and let $\hat{y}$ be dual feasible. Then

$$c^T \hat{x} \leq z \leq w \leq b^T \hat{y} .$$

Weak Duality

Let $z = \max\{c^T x \mid Ax \leq b, x \geq 0\}$ and
 $w = \min\{b^T y \mid A^T y \geq c, y \geq 0\}$ be a primal dual pair.

x is primal feasible iff $x \in \{x \mid Ax \leq b, x \geq 0\}$

y is dual feasible, iff $y \in \{y \mid A^T y \geq c, y \geq 0\}$.

Theorem 32 (Weak Duality)

Let $\hat{x}$ be primal feasible and let $\hat{y}$ be dual feasible. Then

$$c^T \hat{x} \leq z \leq w \leq b^T \hat{y} .$$

Weak Duality

$$A^T \hat{y} \geq c \Rightarrow \hat{x}^T A^T \hat{y} \geq \hat{x}^T c \quad (\hat{x} \geq 0)$$

$$A \hat{x} \leq b \Rightarrow y^T A \hat{x} \leq y^T b \quad (y \geq 0)$$

This gives

$$c^T \hat{x} \leq \hat{y}^T A \hat{x} \leq b^T \hat{y} .$$

Since, there exists primal feasible $\hat{x}$ with $c^T \hat{x} = z$, and dual feasible $\hat{y}$ with $b^T \hat{y} = w$ we get $z \leq w$.

If P is unbounded then D is infeasible.

Weak Duality

$$A^T \hat{y} \geq c \Rightarrow \hat{x}^T A^T \hat{y} \geq \hat{x}^T c \quad (\hat{x} \geq 0)$$

$$A\hat{x} \leq b \Rightarrow y^T A\hat{x} \leq y^T b \quad (y \geq 0)$$

This gives

$$c^T \hat{x} \leq y^T A\hat{x} \leq b^T y .$$

Since, there exists primal feasible $\hat{x}$ with $c^T \hat{x} = z$, and dual feasible $\hat{y}$ with $b^T \hat{y} = w$ we get $z \leq w$.

If P is unbounded then D is infeasible.

Weak Duality

$$A^T \hat{y} \geq c \Rightarrow \hat{x}^T A^T \hat{y} \geq \hat{x}^T c \quad (\hat{x} \geq 0)$$

$$A\hat{x} \leq b \Rightarrow y^T A\hat{x} \leq y^T b \quad (y \geq 0)$$

This gives

$$c^T \hat{x} \leq y^T A\hat{x} \leq b^T y .$$

Since, there exists primal feasible $\hat{x}$ with $c^T \hat{x} = z$, and dual feasible $\hat{y}$ with $b^T \hat{y} = w$ we get $z \leq w$.

If P is unbounded then D is infeasible.

Weak Duality

$$A^T \hat{y} \geq c \Rightarrow \hat{x}^T A^T \hat{y} \geq \hat{x}^T c \quad (\hat{x} \geq 0)$$

$$A \hat{x} \leq b \Rightarrow y^T A \hat{x} \leq y^T b \quad (y \geq 0)$$

This gives

$$c^T \hat{x} \leq y^T A \hat{x} \leq b^T y .$$

Since, there exists primal feasible $\hat{x}$ with $c^T \hat{x} = z$, and dual feasible $\hat{y}$ with $b^T \hat{y} = w$ we get $z \leq w$.

If P is unbounded then D is infeasible.

Weak Duality

$$A^T \hat{y} \geq c \Rightarrow \hat{x}^T A^T \hat{y} \geq \hat{x}^T c \quad (\hat{x} \geq 0)$$

$$A\hat{x} \leq b \Rightarrow y^T A\hat{x} \leq y^T b \quad (y \geq 0)$$

This gives

$$c^T \hat{x} \leq \hat{y}^T A\hat{x} \leq b^T \hat{y} .$$

Since, there exists primal feasible $\hat{x}$ with $c^T \hat{x} = z$, and dual feasible $\hat{y}$ with $b^T \hat{y} = w$ we get $z \leq w$.

If P is unbounded then D is infeasible.

Weak Duality

$$A^T \hat{y} \geq c \Rightarrow \hat{x}^T A^T \hat{y} \geq \hat{x}^T c \quad (\hat{x} \geq 0)$$

$$A\hat{x} \leq b \Rightarrow y^T A\hat{x} \leq y^T b \quad (y \geq 0)$$

This gives

$$c^T \hat{x} \leq \hat{y}^T A\hat{x} \leq b^T \hat{y} .$$

Since, there exists primal feasible $\hat{x}$ with $c^T \hat{x} = z$, and dual feasible $\hat{y}$ with $b^T \hat{y} = w$ we get $z \leq w$.

If P is unbounded then D is infeasible.

Weak Duality

$$A^T \hat{y} \geq c \Rightarrow \hat{x}^T A^T \hat{y} \geq \hat{x}^T c \quad (\hat{x} \geq 0)$$

$$A\hat{x} \leq b \Rightarrow y^T A\hat{x} \leq y^T b \quad (y \geq 0)$$

This gives

$$c^T \hat{x} \leq \hat{y}^T A\hat{x} \leq b^T \hat{y} .$$

Since, there exists primal feasible $\hat{x}$ with $c^T \hat{x} = z$, and dual feasible $\hat{y}$ with $b^T \hat{y} = w$ we get $z \leq w$.

If P is unbounded then D is infeasible.

Weak Duality

$$A^T \hat{y} \geq c \Rightarrow \hat{x}^T A^T \hat{y} \geq \hat{x}^T c \quad (\hat{x} \geq 0)$$

$$A\hat{x} \leq b \Rightarrow y^T A\hat{x} \leq \hat{y}^T b \quad (\hat{y} \geq 0)$$

This gives

$$c^T \hat{x} \leq \hat{y}^T A\hat{x} \leq b^T \hat{y} .$$

Since, there exists primal feasible $\hat{x}$ with $c^T \hat{x} = z$, and dual feasible $\hat{y}$ with $b^T \hat{y} = w$ we get $z \leq w$.

If P is unbounded then D is infeasible.

Weak Duality

$$A^T \hat{y} \geq c \Rightarrow \hat{x}^T A^T \hat{y} \geq \hat{x}^T c \quad (\hat{x} \geq 0)$$

$$A\hat{x} \leq b \Rightarrow y^T A\hat{x} \leq \hat{y}^T b \quad (\hat{y} \geq 0)$$

This gives

$$c^T \hat{x} \leq \hat{y}^T A\hat{x} \leq b^T \hat{y} .$$

Since, there exists primal feasible $\hat{x}$ with $c^T \hat{x} = z$, and dual feasible $\hat{y}$ with $b^T \hat{y} = w$ we get $z \leq w$.

If P is unbounded then D is infeasible.

5.2 Simplex and Duality

The following linear programs form a primal dual pair:

$$z = \max\{c^T x \mid Ax = b, x \geq 0\}$$
$$w = \min\{b^T y \mid A^T y \geq c\}$$

This means for computing the dual of a standard form LP, we do not have non-negativity constraints for the dual variables.

Proof

Primal:

$$\max\{c^T x \mid Ax = b, x \geq 0\}$$

Proof

Primal:

$$\begin{aligned}\max\{c^T x \mid Ax = b, x \geq 0\} \\ = \max\{c^T x \mid Ax \leq b, -Ax \leq -b, x \geq 0\}\end{aligned}$$

Proof

Primal:

$$\begin{aligned} & \max\{c^T x \mid Ax = b, x \geq 0\} \\ &= \max\{c^T x \mid Ax \leq b, -Ax \leq -b, x \geq 0\} \\ &= \max\{c^T x \mid \begin{bmatrix} A \\ -A \end{bmatrix} x \leq \begin{bmatrix} b \\ -b \end{bmatrix}, x \geq 0\} \end{aligned}$$

Proof

Primal:

$$\begin{aligned}\max\{c^T x \mid Ax = b, x \geq 0\} \\&= \max\{c^T x \mid Ax \leq b, -Ax \leq -b, x \geq 0\} \\&= \max\{c^T x \mid \begin{bmatrix} A \\ -A \end{bmatrix} x \leq \begin{bmatrix} b \\ -b \end{bmatrix}, x \geq 0\}\end{aligned}$$

Dual:

$$\min\{[b^T \ -b^T]y \mid [A^T \ -A^T]y \geq c, y \geq 0\}$$

Proof

Primal:

$$\begin{aligned} & \max\{c^T x \mid Ax = b, x \geq 0\} \\ &= \max\{c^T x \mid Ax \leq b, -Ax \leq -b, x \geq 0\} \\ &= \max\{c^T x \mid \begin{bmatrix} A \\ -A \end{bmatrix} x \leq \begin{bmatrix} b \\ -b \end{bmatrix}, x \geq 0\} \end{aligned}$$

Dual:

$$\begin{aligned} & \min\{[b^T \ -b^T]y \mid [A^T \ -A^T]y \geq c, y \geq 0\} \\ &= \min\left\{[b^T \ -b^T] \cdot \begin{bmatrix} y^+ \\ y^- \end{bmatrix} \mid [A^T \ -A^T] \cdot \begin{bmatrix} y^+ \\ y^- \end{bmatrix} \geq c, y^- \geq 0, y^+ \geq 0\right\} \end{aligned}$$

Proof

Primal:

$$\begin{aligned} \max\{c^T x \mid Ax = b, x \geq 0\} \\ = \max\{c^T x \mid Ax \leq b, -Ax \leq -b, x \geq 0\} \\ = \max\{c^T x \mid \begin{bmatrix} A \\ -A \end{bmatrix} x \leq \begin{bmatrix} b \\ -b \end{bmatrix}, x \geq 0\} \end{aligned}$$

Dual:

$$\begin{aligned} \min\{[b^T \ -b^T]y \mid [A^T \ -A^T]y \geq c, y \geq 0\} \\ = \min\left\{[b^T \ -b^T] \cdot \begin{bmatrix} y^+ \\ y^- \end{bmatrix} \mid [A^T \ -A^T] \cdot \begin{bmatrix} y^+ \\ y^- \end{bmatrix} \geq c, y^- \geq 0, y^+ \geq 0\right\} \\ = \min\{b^T \cdot (y^+ - y^-) \mid A^T \cdot (y^+ - y^-) \geq c, y^- \geq 0, y^+ \geq 0\} \end{aligned}$$

Proof

Primal:

$$\begin{aligned} & \max\{c^T x \mid Ax = b, x \geq 0\} \\ &= \max\{c^T x \mid Ax \leq b, -Ax \leq -b, x \geq 0\} \\ &= \max\{c^T x \mid \begin{bmatrix} A \\ -A \end{bmatrix} x \leq \begin{bmatrix} b \\ -b \end{bmatrix}, x \geq 0\} \end{aligned}$$

Dual:

$$\begin{aligned} & \min\{[b^T \ -b^T]y \mid [A^T \ -A^T]y \geq c, y \geq 0\} \\ &= \min\left\{[b^T \ -b^T] \cdot \begin{bmatrix} y^+ \\ y^- \end{bmatrix} \mid [A^T \ -A^T] \cdot \begin{bmatrix} y^+ \\ y^- \end{bmatrix} \geq c, y^- \geq 0, y^+ \geq 0\right\} \\ &= \min\{b^T \cdot (y^+ - y^-) \mid A^T \cdot (y^+ - y^-) \geq c, y^- \geq 0, y^+ \geq 0\} \\ &= \min\{b^T y' \mid A^T y' \geq c\} \end{aligned}$$

Proof of Optimality Criterion for Simplex

Suppose that we have a basic feasible solution with **reduced cost**

$$\tilde{c} = c^T - c_B^T A_B^{-1} A \leq 0$$

This is equivalent to $A^T (A_B^{-1})^T c_B \geq c$

$y^* = (A_B^{-1})^T c_B$ is solution to the **dual** $\min\{b^T y \mid A^T y \geq c\}$.

Since y^* is feasible and $y^* b = c^T x^*$, x^* is optimal.

Hence, the solution is optimal.

Proof of Optimality Criterion for Simplex

Suppose that we have a basic feasible solution with **reduced cost**

$$\tilde{c} = c^T - c_B^T A_B^{-1} A \leq 0$$

This is equivalent to $A^T (A_B^{-1})^T c_B \geq c$

$y^* = (A_B^{-1})^T c_B$ is solution to the **dual** $\min\{b^T y \mid A^T y \geq c\}$.

Hence, the solution is optimal.

Proof of Optimality Criterion for Simplex

Suppose that we have a basic feasible solution with **reduced cost**

$$\tilde{c} = c^T - c_B^T A_B^{-1} A \leq 0$$

This is equivalent to $A^T (A_B^{-1})^T c_B \geq c$

$y^* = (A_B^{-1})^T c_B$ is solution to the **dual** $\min\{b^T y \mid A^T y \geq c\}$.

$$\begin{aligned} b^T y^* &= (Ax^*)^T y^* = (A_B x_B^*)^T y^* \\ &= (A_B x_B^*)^T (A_B^{-1})^T c_B = (x_B^*)^T A_B^T (A_B^{-1})^T c_B \\ &= c^T x^* \end{aligned}$$

Hence, the solution is optimal.

Proof of Optimality Criterion for Simplex

Suppose that we have a basic feasible solution with **reduced cost**

$$\tilde{c} = c^T - c_B^T A_B^{-1} A \leq 0$$

This is equivalent to $A^T (A_B^{-1})^T c_B \geq c$

$y^* = (A_B^{-1})^T c_B$ is solution to the **dual** $\min\{b^T y \mid A^T y \geq c\}$.

$$\begin{aligned} b^T y^* &= (Ax^*)^T y^* = (A_B x_B^*)^T y^* \\ &= (A_B x_B^*)^T (A_B^{-1})^T c_B = (x_B^*)^T A_B^T (A_B^{-1})^T c_B \\ &= c^T x^* \end{aligned}$$

Hence, the solution is optimal.

Proof of Optimality Criterion for Simplex

Suppose that we have a basic feasible solution with **reduced cost**

$$\tilde{c} = c^T - c_B^T A_B^{-1} A \leq 0$$

This is equivalent to $A^T (A_B^{-1})^T c_B \geq c$

$y^* = (A_B^{-1})^T c_B$ is solution to the **dual** $\min\{b^T y \mid A^T y \geq c\}$.

$$\begin{aligned} b^T y^* &= (Ax^*)^T y^* = (A_B x_B^*)^T y^* \\ &= (A_B x_B^*)^T (A_B^{-1})^T c_B = (x_B^*)^T A_B^T (A_B^{-1})^T c_B \\ &= c^T x^* \end{aligned}$$

Hence, the solution is optimal.

Proof of Optimality Criterion for Simplex

Suppose that we have a basic feasible solution with **reduced cost**

$$\tilde{c} = c^T - c_B^T A_B^{-1} A \leq 0$$

This is equivalent to $A^T (A_B^{-1})^T c_B \geq c$

$y^* = (A_B^{-1})^T c_B$ is solution to the **dual** $\min\{b^T y \mid A^T y \geq c\}$.

$$\begin{aligned} b^T y^* &= (Ax^*)^T y^* = (A_B x_B^*)^T y^* \\ &= (A_B x_B^*)^T (A_B^{-1})^T c_B = (x_B^*)^T A_B^T (A_B^{-1})^T c_B \\ &= c^T x^* \end{aligned}$$

Hence, the solution is optimal.

Proof of Optimality Criterion for Simplex

Suppose that we have a basic feasible solution with **reduced cost**

$$\tilde{c} = c^T - c_B^T A_B^{-1} A \leq 0$$

This is equivalent to $A^T (A_B^{-1})^T c_B \geq c$

$y^* = (A_B^{-1})^T c_B$ is solution to the **dual** $\min\{b^T y \mid A^T y \geq c\}$.

$$\begin{aligned} b^T y^* &= (Ax^*)^T y^* = (A_B x_B^*)^T y^* \\ &= (A_B x_B^*)^T (A_B^{-1})^T c_B = (x_B^*)^T A_B^T (A_B^{-1})^T c_B \\ &= c^T x^* \end{aligned}$$

Hence, the solution is optimal.

Proof of Optimality Criterion for Simplex

Suppose that we have a basic feasible solution with **reduced cost**

$$\tilde{c} = c^T - c_B^T A_B^{-1} A \leq 0$$

This is equivalent to $A^T (A_B^{-1})^T c_B \geq c$

$y^* = (A_B^{-1})^T c_B$ is solution to the **dual** $\min\{b^T y \mid A^T y \geq c\}$.

$$\begin{aligned} b^T y^* &= (Ax^*)^T y^* = (A_B x_B^*)^T y^* \\ &= (A_B x_B^*)^T (A_B^{-1})^T c_B = (x_B^*)^T A_B^T (A_B^{-1})^T c_B \\ &= c^T x^* \end{aligned}$$

Hence, the solution is optimal.

Proof of Optimality Criterion for Simplex

Suppose that we have a basic feasible solution with **reduced cost**

$$\tilde{c} = c^T - c_B^T A_B^{-1} A \leq 0$$

This is equivalent to $A^T (A_B^{-1})^T c_B \geq c$

$y^* = (A_B^{-1})^T c_B$ is solution to the **dual** $\min\{b^T y \mid A^T y \geq c\}$.

$$\begin{aligned} b^T y^* &= (Ax^*)^T y^* = (A_B x_B^*)^T y^* \\ &= (A_B x_B^*)^T (A_B^{-1})^T c_B = (x_B^*)^T A_B^T (A_B^{-1})^T c_B \\ &= c^T x^* \end{aligned}$$

Hence, the solution is optimal.

5.3 Strong Duality

$$P = \max\{c^T x \mid Ax \leq b, x \geq 0\}$$

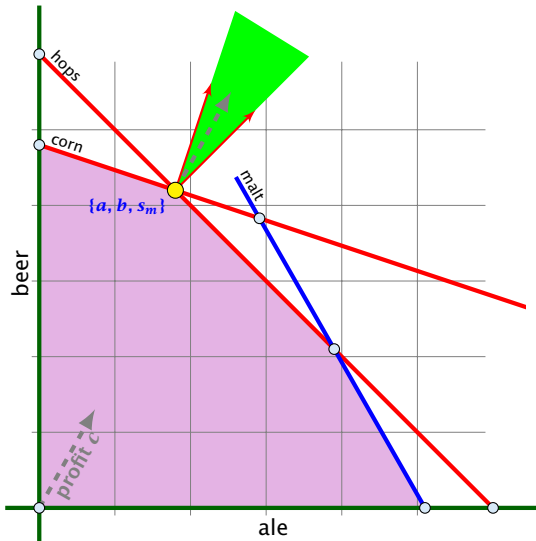
n_A : number of variables, m_A : number of constraints

We can put the non-negativity constraints into A (which gives us unrestricted variables): $\bar{P} = \max\{c^T x \mid \bar{A}x \leq \bar{b}\}$

$$n_{\bar{A}} = n_A, m_{\bar{A}} = m_A + n_A$$

$$\text{Dual } D = \min\{\bar{b}^T y \mid \bar{A}^T y = c, y \geq 0\}.$$

5.3 Strong Duality



The profit vector c lies in the cone generated by the normals for the hops and the corn constraint (the tight constraints).

Strong Duality

Theorem 33 (Strong Duality)

Let P and D be a primal dual pair of linear programs, and let z^* and w^* denote the optimal solution to P and D , respectively. Then

$$z^* = w^*$$

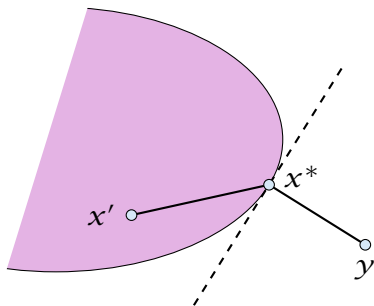
Lemma 34 (Weierstrass)

Let X be a compact set and let $f(x)$ be a continuous function on X . Then $\min\{f(x) : x \in X\}$ exists.

(without proof)

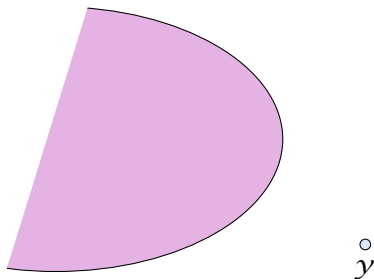
Lemma 35 (Projection Lemma)

Let $X \subseteq \mathbb{R}^m$ be a non-empty convex set, and let $y \notin X$. Then there exist $x^* \in X$ with minimum distance from y . Moreover for all $x \in X$ we have $(y - x^*)^T(x - x^*) \leq 0$.



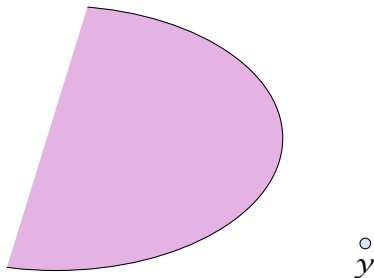
Proof of the Projection Lemma

- ▶ Define $f(x) = \|y - x\|$.
- ▶ We want to apply Weierstrass but X may not be bounded.
- ▶ $X \neq \emptyset$. Hence, there exists $x' \in X$.
- ▶ Define $X' = \{x \in X \mid \|y - x\| \leq \|y - x'\|\}$. This set is closed and bounded.
- ▶ Applying Weierstrass gives the existence.



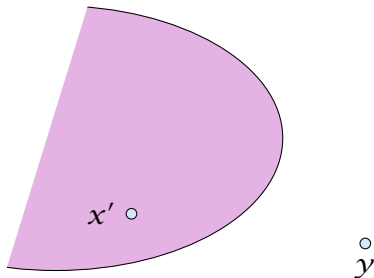
Proof of the Projection Lemma

- ▶ Define $f(x) = \|y - x\|$.
- ▶ We want to apply Weierstrass but X may not be bounded.
- ▶ $X \neq \emptyset$. Hence, there exists $x' \in X$.
- ▶ Define $X' = \{x \in X \mid \|y - x\| \leq \|y - x'\|\}$. This set is closed and bounded.
- ▶ Applying Weierstrass gives the existence.



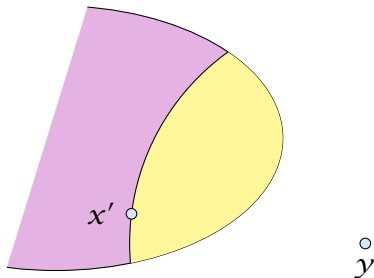
Proof of the Projection Lemma

- ▶ Define $f(x) = \|y - x\|$.
- ▶ We want to apply Weierstrass but X may not be bounded.
- ▶ $X \neq \emptyset$. Hence, there exists $x' \in X$.
- ▶ Define $X' = \{x \in X \mid \|y - x\| \leq \|y - x'\|\}$. This set is closed and bounded.
- ▶ Applying Weierstrass gives the existence.



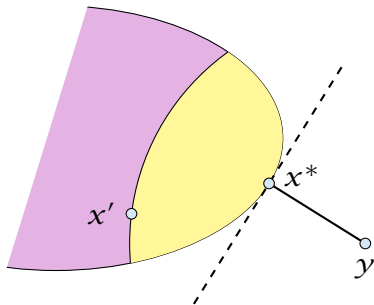
Proof of the Projection Lemma

- ▶ Define $f(x) = \|y - x\|$.
- ▶ We want to apply Weierstrass but X may not be bounded.
- ▶ $X \neq \emptyset$. Hence, there exists $x' \in X$.
- ▶ Define $X' = \{x \in X \mid \|y - x\| \leq \|y - x'\|\}$. This set is closed and bounded.
- ▶ Applying Weierstrass gives the existence.



Proof of the Projection Lemma

- ▶ Define $f(x) = \|y - x\|$.
- ▶ We want to apply Weierstrass but X may not be bounded.
- ▶ $X \neq \emptyset$. Hence, there exists $x' \in X$.
- ▶ Define $X' = \{x \in X \mid \|y - x\| \leq \|y - x'\|\}$. This set is closed and bounded.
- ▶ Applying Weierstrass gives the existence.



Proof of the Projection Lemma (continued)

Proof of the Projection Lemma (continued)

x^* is minimum. Hence $\|y - x^*\|^2 \leq \|y - x\|^2$ for all $x \in X$.

Proof of the Projection Lemma (continued)

x^* is minimum. Hence $\|y - x^*\|^2 \leq \|y - x\|^2$ for all $x \in X$.

By **convexity**: $x \in X$ then $x^* + \epsilon(x - x^*) \in X$ for all $0 \leq \epsilon \leq 1$.

Proof of the Projection Lemma (continued)

x^* is minimum. Hence $\|y - x^*\|^2 \leq \|y - x\|^2$ for all $x \in X$.

By **convexity**: $x \in X$ then $x^* + \epsilon(x - x^*) \in X$ for all $0 \leq \epsilon \leq 1$.

$$\|y - x^*\|^2$$

Proof of the Projection Lemma (continued)

x^* is minimum. Hence $\|y - x^*\|^2 \leq \|y - x\|^2$ for all $x \in X$.

By **convexity**: $x \in X$ then $x^* + \epsilon(x - x^*) \in X$ for all $0 \leq \epsilon \leq 1$.

$$\|y - x^*\|^2 \leq \|y - x^* - \epsilon(x - x^*)\|^2$$

Proof of the Projection Lemma (continued)

x^* is minimum. Hence $\|y - x^*\|^2 \leq \|y - x\|^2$ for all $x \in X$.

By **convexity**: $x \in X$ then $x^* + \epsilon(x - x^*) \in X$ for all $0 \leq \epsilon \leq 1$.

$$\begin{aligned}\|y - x^*\|^2 &\leq \|y - x^* - \epsilon(x - x^*)\|^2 \\ &= \|y - x^*\|^2 + \epsilon^2 \|x - x^*\|^2 - 2\epsilon(y - x^*)^T(x - x^*)\end{aligned}$$

Proof of the Projection Lemma (continued)

x^* is minimum. Hence $\|y - x^*\|^2 \leq \|y - x\|^2$ for all $x \in X$.

By **convexity**: $x \in X$ then $x^* + \epsilon(x - x^*) \in X$ for all $0 \leq \epsilon \leq 1$.

$$\begin{aligned}\|y - x^*\|^2 &\leq \|y - x^* - \epsilon(x - x^*)\|^2 \\ &= \|y - x^*\|^2 + \epsilon^2 \|x - x^*\|^2 - 2\epsilon(y - x^*)^T(x - x^*)\end{aligned}$$

Hence, $(y - x^*)^T(x - x^*) \leq \frac{1}{2}\epsilon\|x - x^*\|^2$.

Proof of the Projection Lemma (continued)

x^* is minimum. Hence $\|y - x^*\|^2 \leq \|y - x\|^2$ for all $x \in X$.

By **convexity**: $x \in X$ then $x^* + \epsilon(x - x^*) \in X$ for all $0 \leq \epsilon \leq 1$.

$$\begin{aligned}\|y - x^*\|^2 &\leq \|y - x^* - \epsilon(x - x^*)\|^2 \\ &= \|y - x^*\|^2 + \epsilon^2 \|x - x^*\|^2 - 2\epsilon(y - x^*)^T(x - x^*)\end{aligned}$$

Hence, $(y - x^*)^T(x - x^*) \leq \frac{1}{2}\epsilon\|x - x^*\|^2$.

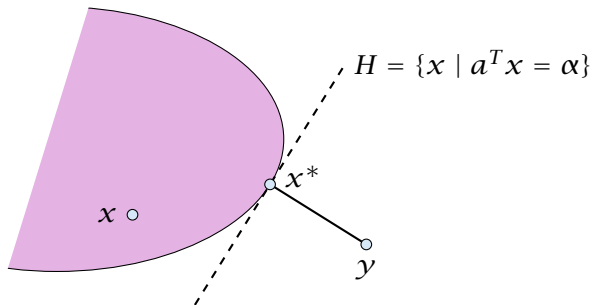
Letting $\epsilon \rightarrow 0$ gives the result.

Theorem 36 (Separating Hyperplane)

Let $X \subseteq \mathbb{R}^m$ be a non-empty closed convex set, and let $y \notin X$. Then there exists a separating hyperplane $\{x \in \mathbb{R} : a^T x = \alpha\}$ where $a \in \mathbb{R}^m$, $\alpha \in \mathbb{R}$ that separates y from X . ($a^T y < \alpha$; $a^T x \geq \alpha$ for all $x \in X$)

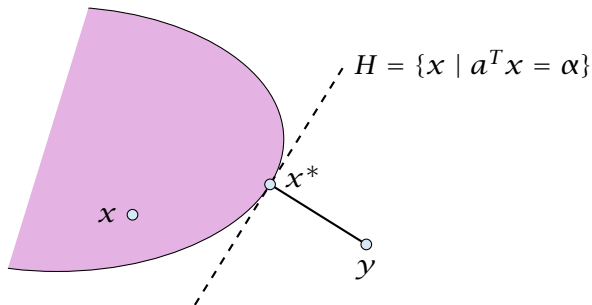
Proof of the Hyperplane Lemma

- ▶ Let $x^* \in X$ be closest point to y in X .
- ▶ By previous lemma $(y - x^*)^T(x - x^*) \leq 0$ for all $x \in X$.
- ▶ Choose $a = (x^* - y)$ and $\alpha = a^T x^*$.
- ▶ For $x \in X$: $a^T(x - x^*) \geq 0$, and, hence, $a^T x \geq \alpha$.
- ▶ Also, $a^T y = a^T(x^* - a) = \alpha - \|a\|^2 < \alpha$



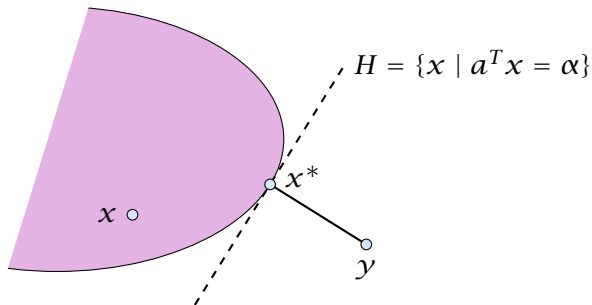
Proof of the Hyperplane Lemma

- ▶ Let $x^* \in X$ be closest point to y in X .
- ▶ By previous lemma $(y - x^*)^T(x - x^*) \leq 0$ for all $x \in X$.
- ▶ Choose $a = (x^* - y)$ and $\alpha = a^T x^*$.
- ▶ For $x \in X$: $a^T(x - x^*) \geq 0$, and, hence, $a^T x \geq \alpha$.
- ▶ Also, $a^T y = a^T(x^* - a) = \alpha - \|a\|^2 < \alpha$



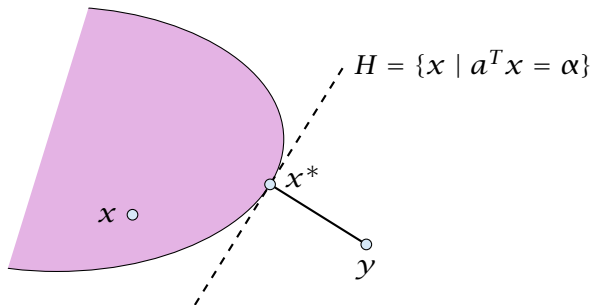
Proof of the Hyperplane Lemma

- ▶ Let $x^* \in X$ be closest point to y in X .
- ▶ By previous lemma $(y - x^*)^T(x - x^*) \leq 0$ for all $x \in X$.
- ▶ Choose $a = (x^* - y)$ and $\alpha = a^T x^*$.
- ▶ For $x \in X$: $a^T(x - x^*) \geq 0$, and, hence, $a^T x \geq \alpha$.
- ▶ Also, $a^T y = a^T(x^* - a) = \alpha - \|a\|^2 < \alpha$



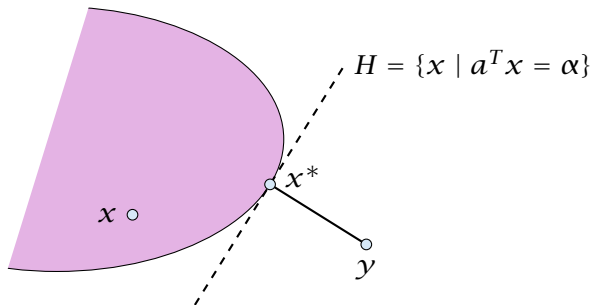
Proof of the Hyperplane Lemma

- ▶ Let $x^* \in X$ be closest point to y in X .
- ▶ By previous lemma $(y - x^*)^T(x - x^*) \leq 0$ for all $x \in X$.
- ▶ Choose $a = (x^* - y)$ and $\alpha = a^T x^*$.
- ▶ For $x \in X : a^T(x - x^*) \geq 0$, and, hence, $a^T x \geq \alpha$.
- ▶ Also, $a^T y = a^T(x^* - a) = \alpha - \|a\|^2 < \alpha$



Proof of the Hyperplane Lemma

- ▶ Let $x^* \in X$ be closest point to y in X .
- ▶ By previous lemma $(y - x^*)^T(x - x^*) \leq 0$ for all $x \in X$.
- ▶ Choose $a = (x^* - y)$ and $\alpha = a^T x^*$.
- ▶ For $x \in X : a^T(x - x^*) \geq 0$, and, hence, $a^T x \geq \alpha$.
- ▶ Also, $a^T y = a^T(x^* - a) = \alpha - \|a\|^2 < \alpha$



Lemma 37 (Farkas Lemma)

Let A be an $m \times n$ matrix, $b \in \mathbb{R}^m$. Then *exactly one* of the following statements holds.

1. $\exists x \in \mathbb{R}^n$ with $Ax = b, x \geq 0$
2. $\exists y \in \mathbb{R}^m$ with $A^T y \geq 0, b^T y < 0$

Assume $\hat{x}$ satisfies 1. and $\hat{y}$ satisfies 2. Then

$$0 > \hat{y}^T b = \hat{y}^T A \hat{x} \geq 0$$

Hence, at most one of the statements can hold.

Lemma 37 (Farkas Lemma)

Let A be an $m \times n$ matrix, $b \in \mathbb{R}^m$. Then *exactly one* of the following statements holds.

1. $\exists x \in \mathbb{R}^n$ with $Ax = b, x \geq 0$
2. $\exists y \in \mathbb{R}^m$ with $A^T y \geq 0, b^T y < 0$

Assume $\hat{x}$ satisfies 1. and $\hat{y}$ satisfies 2. Then

$$0 > \hat{y}^T b = \hat{y}^T A \hat{x} \geq 0$$

Hence, at most one of the statements can hold.

Lemma 37 (Farkas Lemma)

Let A be an $m \times n$ matrix, $b \in \mathbb{R}^m$. Then *exactly one* of the following statements holds.

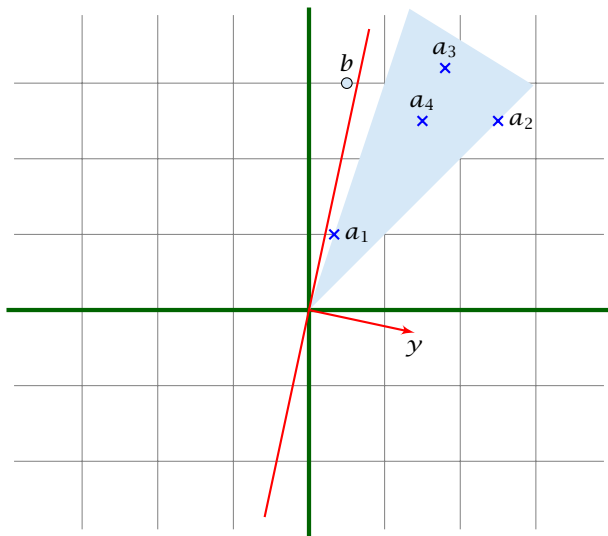
1. $\exists x \in \mathbb{R}^n$ with $Ax = b, x \geq 0$
2. $\exists y \in \mathbb{R}^m$ with $A^T y \geq 0, b^T y < 0$

Assume $\hat{x}$ satisfies 1. and $\hat{y}$ satisfies 2. Then

$$0 > \hat{y}^T b = \hat{y}^T A \hat{x} \geq 0$$

Hence, at most one of the statements can hold.

Farkas Lemma



If b is not in the cone generated by the columns of A , there exists a hyperplane y that separates b from the cone.

Proof of Farkas Lemma

Now, assume that 1. does not hold.

Consider $S = \{Ax : x \geq 0\}$ so that S closed, convex, $b \notin S$.

We want to show that there is y with $A^T y \geq 0$, $b^T y < 0$.

Let y be a hyperplane that separates b from S . Hence, $y^T b < \alpha$ and $y^T s \geq \alpha$ for all $s \in S$.

$$0 \in S \Rightarrow \alpha \leq 0 \Rightarrow y^T b < 0$$

$y^T Ax \geq \alpha$ for all $x \geq 0$. Hence, $y^T A \geq 0$ as we can choose x arbitrarily large.

Proof of Farkas Lemma

Now, assume that 1. does not hold.

Consider $S = \{Ax : x \geq 0\}$ so that S closed, convex, $b \notin S$.

We want to show that there is y with $A^T y \geq 0$, $b^T y < 0$.

Let y be a hyperplane that separates b from S . Hence, $y^T b < \alpha$ and $y^T s \geq \alpha$ for all $s \in S$.

$$0 \in S \Rightarrow \alpha \leq 0 \Rightarrow y^T b < 0$$

$y^T Ax \geq \alpha$ for all $x \geq 0$. Hence, $y^T A \geq 0$ as we can choose x arbitrarily large.

Proof of Farkas Lemma

Now, assume that 1. does not hold.

Consider $S = \{Ax : x \geq 0\}$ so that S closed, convex, $b \notin S$.

We want to show that there is y with $A^T y \geq 0$, $b^T y < 0$.

Let y be a hyperplane that separates b from S . Hence, $y^T b < \alpha$ and $y^T s \geq \alpha$ for all $s \in S$.

$$0 \in S \Rightarrow \alpha \leq 0 \Rightarrow y^T b < 0$$

$y^T Ax \geq \alpha$ for all $x \geq 0$. Hence, $y^T A \geq 0$ as we can choose x arbitrarily large.

Proof of Farkas Lemma

Now, assume that 1. does not hold.

Consider $S = \{Ax : x \geq 0\}$ so that S closed, convex, $b \notin S$.

We want to show that there is y with $A^T y \geq 0$, $b^T y < 0$.

Let y be a hyperplane that separates b from S . Hence, $y^T b < \alpha$ and $y^T s \geq \alpha$ for all $s \in S$.

$$0 \in S \Rightarrow \alpha \leq 0 \Rightarrow y^T b < 0$$

$y^T Ax \geq \alpha$ for all $x \geq 0$. Hence, $y^T A \geq 0$ as we can choose x arbitrarily large.

Proof of Farkas Lemma

Now, assume that 1. does not hold.

Consider $S = \{Ax : x \geq 0\}$ so that S closed, convex, $b \notin S$.

We want to show that there is y with $A^T y \geq 0$, $b^T y < 0$.

Let y be a hyperplane that separates b from S . Hence, $y^T b < \alpha$ and $y^T s \geq \alpha$ for all $s \in S$.

$$0 \in S \Rightarrow \alpha \leq 0 \Rightarrow y^T b < 0$$

$y^T Ax \geq \alpha$ for all $x \geq 0$. Hence, $y^T A \geq 0$ as we can choose x arbitrarily large.

Proof of Farkas Lemma

Now, assume that 1. does not hold.

Consider $S = \{Ax : x \geq 0\}$ so that S closed, convex, $b \notin S$.

We want to show that there is y with $A^T y \geq 0$, $b^T y < 0$.

Let y be a hyperplane that separates b from S . Hence, $y^T b < \alpha$ and $y^T s \geq \alpha$ for all $s \in S$.

$$0 \in S \Rightarrow \alpha \leq 0 \Rightarrow y^T b < 0$$

$y^T Ax \geq \alpha$ for all $x \geq 0$. Hence, $y^T A \geq 0$ as we can choose x arbitrarily large.

Proof of Farkas Lemma

Now, assume that 1. does not hold.

Consider $S = \{Ax : x \geq 0\}$ so that S closed, convex, $b \notin S$.

We want to show that there is y with $A^T y \geq 0$, $b^T y < 0$.

Let y be a hyperplane that separates b from S . Hence, $y^T b < \alpha$ and $y^T s \geq \alpha$ for all $s \in S$.

$$0 \in S \Rightarrow \alpha \leq 0 \Rightarrow y^T b < 0$$

$y^T Ax \geq \alpha$ for all $x \geq 0$. Hence, $y^T A \geq 0$ as we can choose x arbitrarily large.

Proof of Farkas Lemma

Now, assume that 1. does not hold.

Consider $S = \{Ax : x \geq 0\}$ so that S closed, convex, $b \notin S$.

We want to show that there is y with $A^T y \geq 0$, $b^T y < 0$.

Let y be a hyperplane that separates b from S . Hence, $y^T b < \alpha$ and $y^T s \geq \alpha$ for all $s \in S$.

$$0 \in S \Rightarrow \alpha \leq 0 \Rightarrow y^T b < 0$$

$y^T Ax \geq \alpha$ for all $x \geq 0$. Hence, $y^T A \geq 0$ as we can choose x arbitrarily large.

Lemma 38 (Farkas Lemma; different version)

Let A be an $m \times n$ matrix, $b \in \mathbb{R}^m$. Then exactly one of the following statements holds.

1. $\exists x \in \mathbb{R}^n$ with $Ax \leq b, x \geq 0$
2. $\exists y \in \mathbb{R}^m$ with $A^T y \geq 0, b^T y < 0, y \geq 0$

Rewrite the conditions:

1. $\exists x \in \mathbb{R}^n$ with $\begin{bmatrix} A & I \end{bmatrix} \cdot \begin{bmatrix} x \\ s \end{bmatrix} = b, x \geq 0, s \geq 0$
2. $\exists y \in \mathbb{R}^m$ with $\begin{bmatrix} A^T \\ I \end{bmatrix} y \geq 0, b^T y < 0$

Lemma 38 (Farkas Lemma; different version)

Let A be an $m \times n$ matrix, $b \in \mathbb{R}^m$. Then exactly one of the following statements holds.

1. $\exists x \in \mathbb{R}^n$ with $Ax \leq b, x \geq 0$
2. $\exists y \in \mathbb{R}^m$ with $A^T y \geq 0, b^T y < 0, y \geq 0$

Rewrite the conditions:

1. $\exists x \in \mathbb{R}^n$ with $\begin{bmatrix} A & I \end{bmatrix} \cdot \begin{bmatrix} x \\ s \end{bmatrix} = b, x \geq 0, s \geq 0$
2. $\exists y \in \mathbb{R}^m$ with $\begin{bmatrix} A^T \\ I \end{bmatrix} y \geq 0, b^T y < 0$

Proof of Strong Duality

$$P: z = \max\{c^T x \mid Ax \leq b, x \geq 0\}$$

$$D: w = \min\{b^T y \mid A^T y \geq c, y \geq 0\}$$

Theorem 39 (Strong Duality)

Let P and D be a primal dual pair of linear programs, and let z and w denote the optimal solution to P and D , respectively (i.e., P and D are non-empty). Then

$$z = w \text{ .}$$

Proof of Strong Duality

Proof of Strong Duality

$z \leq w$: follows from weak duality

Proof of Strong Duality

$z \leq w$: follows from weak duality

$z \geq w$:

Proof of Strong Duality

$z \leq w$: follows from weak duality

$z \geq w$:

We show $z < \alpha$ implies $w < \alpha$.

Proof of Strong Duality

$z \leq w$: follows from weak duality

$z \geq w$:

We show $z < \alpha$ implies $w < \alpha$.

$$\begin{array}{lll} \exists x \in \mathbb{R}^n & & \\ \text{s.t.} & Ax & \leq b \\ & -c^T x & \leq -\alpha \\ & x & \geq 0 \end{array}$$

Proof of Strong Duality

$z \leq w$: follows from weak duality

$z \geq w$:

We show $z < \alpha$ implies $w < \alpha$.

$$\exists x \in \mathbb{R}^n$$

$$\begin{array}{lll} \text{s.t.} & Ax & \leq b \\ & -c^T x & \leq -\alpha \\ & x & \geq 0 \end{array}$$

$$\exists y \in \mathbb{R}^m; v \in \mathbb{R}$$

$$\begin{array}{lll} \text{s.t.} & A^T y - cv & \geq 0 \\ & b^T y - \alpha v & < 0 \\ & y, v & \geq 0 \end{array}$$

Proof of Strong Duality

$z \leq w$: follows from weak duality

$z \geq w$:

We show $z < \alpha$ implies $w < \alpha$.

$$\exists x \in \mathbb{R}^n$$

$$\begin{array}{lll} \text{s.t.} & Ax & \leq b \\ & -c^T x & \leq -\alpha \\ & x & \geq 0 \end{array}$$

$$\exists y \in \mathbb{R}^m; v \in \mathbb{R}$$

$$\begin{array}{lll} \text{s.t.} & A^T y - cv & \geq 0 \\ & b^T y - \alpha v & < 0 \\ & y, v & \geq 0 \end{array}$$

From the definition of α we know that the first system is infeasible; hence the second must be feasible.

Proof of Strong Duality

$$\begin{array}{ll}\exists \boldsymbol{y} \in \mathbb{R}^m; v \in \mathbb{R} & \\ \text{s.t.} & A^T \boldsymbol{y} - c v \geq 0 \\ & b^T \boldsymbol{y} - \alpha v < 0 \\ & \boldsymbol{y}, v \geq 0\end{array}$$

Proof of Strong Duality

$$\begin{array}{ll}\exists \mathbf{y} \in \mathbb{R}^m; v \in \mathbb{R} & \\ \text{s.t.} & A^T \mathbf{y} - c v \geq 0 \\ & b^T \mathbf{y} - \alpha v < 0 \\ & \mathbf{y}, v \geq 0\end{array}$$

If the solution $\mathbf{y}, v$ has $v = 0$ we have that

$$\begin{array}{ll}\exists \mathbf{y} \in \mathbb{R}^m & \\ \text{s.t.} & A^T \mathbf{y} \geq 0 \\ & b^T \mathbf{y} < 0 \\ & \mathbf{y} \geq 0\end{array}$$

is feasible.

Proof of Strong Duality

$$\begin{array}{ll}\exists \mathbf{y} \in \mathbb{R}^m; v \in \mathbb{R} & \\ \text{s.t.} & A^T \mathbf{y} - c v \geq 0 \\ & b^T \mathbf{y} - \alpha v < 0 \\ & \mathbf{y}, v \geq 0\end{array}$$

If the solution $\mathbf{y}, v$ has $v = 0$ we have that

$$\begin{array}{ll}\exists \mathbf{y} \in \mathbb{R}^m & \\ \text{s.t.} & A^T \mathbf{y} \geq 0 \\ & b^T \mathbf{y} < 0 \\ & \mathbf{y} \geq 0\end{array}$$

is feasible. By Farkas lemma this gives that LP P is infeasible.
Contradiction to the assumption of the lemma.

Proof of Strong Duality

Hence, there exists a solution y, v with $v > 0$.

We can rescale this solution (scaling both y and v) s.t. $v = 1$.

Then y is feasible for the dual but $b^T y < \alpha$. This means that $w < \alpha$.

Proof of Strong Duality

Hence, there exists a solution y, v with $v > 0$.

We can rescale this solution (scaling both y and v) s.t. $v = 1$.

Then y is feasible for the dual but $b^T y < \alpha$. This means that $w < \alpha$.

Proof of Strong Duality

Hence, there exists a solution y, v with $v > 0$.

We can rescale this solution (scaling both y and v) s.t. $v = 1$.

Then y is feasible for the dual but $b^T y < \alpha$. This means that $w < \alpha$.

Proof of Strong Duality

Hence, there exists a solution y, v with $v > 0$.

We can rescale this solution (scaling both y and v) s.t. $v = 1$.

Then y is feasible for the dual but $b^T y < \alpha$. This means that $w < \alpha$.

Fundamental Questions

Definition 40 (Linear Programming Problem (LP))

Let $A \in \mathbb{Q}^{m \times n}$, $b \in \mathbb{Q}^m$, $c \in \mathbb{Q}^n$, $\alpha \in \mathbb{Q}$. Does there exist $x \in \mathbb{Q}^n$ s.t. $Ax = b$, $x \geq 0$, $c^T x \geq \alpha$?

Questions:

- ▶ Is LP in NP?
- ▶ Is LP in co-NP? yes!
- ▶ Is LP in P?

Proof:

Fundamental Questions

Definition 40 (Linear Programming Problem (LP))

Let $A \in \mathbb{Q}^{m \times n}$, $b \in \mathbb{Q}^m$, $c \in \mathbb{Q}^n$, $\alpha \in \mathbb{Q}$. Does there exist $x \in \mathbb{Q}^n$ s.t. $Ax = b$, $x \geq 0$, $c^T x \geq \alpha$?

Questions:

- ▶ Is LP in NP?
- ▶ Is LP in co-NP? yes!
- ▶ Is LP in P?

Proof:

- ▶ Given a primal maximization problem P and a parameter α . Suppose that $\alpha > \text{opt}(P)$.
 - ▶ We can prove this by providing an optimal basis for the dual.
 - ▶ A verifier can check that the associated dual solution fulfills all dual constraints and that it has dual cost $< \alpha$.

Fundamental Questions

Definition 40 (Linear Programming Problem (LP))

Let $A \in \mathbb{Q}^{m \times n}$, $b \in \mathbb{Q}^m$, $c \in \mathbb{Q}^n$, $\alpha \in \mathbb{Q}$. Does there exist $x \in \mathbb{Q}^n$ s.t. $Ax = b$, $x \geq 0$, $c^T x \geq \alpha$?

Questions:

- ▶ Is LP in NP?
- ▶ Is LP in co-NP? yes!
- ▶ Is LP in P?

Proof:

- ▶ Given a primal maximization problem P and a parameter α . Suppose that $\alpha > \text{opt}(P)$.
- ▶ We can prove this by providing an optimal basis for the dual.
- ▶ A verifier can check that the associated dual solution fulfills all dual constraints and that it has dual cost $< \alpha$.

Fundamental Questions

Definition 40 (Linear Programming Problem (LP))

Let $A \in \mathbb{Q}^{m \times n}$, $b \in \mathbb{Q}^m$, $c \in \mathbb{Q}^n$, $\alpha \in \mathbb{Q}$. Does there exist $x \in \mathbb{Q}^n$ s.t. $Ax = b$, $x \geq 0$, $c^T x \geq \alpha$?

Questions:

- ▶ Is LP in NP?
- ▶ Is LP in co-NP? yes!
- ▶ Is LP in P?

Proof:

- ▶ Given a primal maximization problem P and a parameter α . Suppose that $\alpha > \text{opt}(P)$.
- ▶ We can prove this by providing an optimal basis for the dual.
- ▶ A verifier can check that the associated dual solution fulfills all dual constraints and that it has dual cost $< \alpha$.

Complementary Slackness

Lemma 41

Assume a linear program $P = \max\{c^T x \mid Ax \leq b; x \geq 0\}$ has solution x^* and its dual $D = \min\{b^T y \mid A^T y \geq c; y \geq 0\}$ has solution y^* .

1. If $x_j^* > 0$ then the j -th constraint in D is tight.
2. If the j -th constraint in D is not tight then $x_j^* = 0$.
3. If $y_i^* > 0$ then the i -th constraint in P is tight.
4. If the i -th constraint in P is not tight then $y_i^* = 0$.

Complementary Slackness

Lemma 41

Assume a linear program $P = \max\{c^T x \mid Ax \leq b; x \geq 0\}$ has solution x^* and its dual $D = \min\{b^T y \mid A^T y \geq c; y \geq 0\}$ has solution y^* .

1. If $x_j^* > 0$ then the j -th constraint in D is tight.
2. If the j -th constraint in D is not tight then $x_j^* = 0$.
3. If $y_i^* > 0$ then the i -th constraint in P is tight.
4. If the i -th constraint in P is not tight then $y_i^* = 0$.

If we say that a variable x_j^* (y_i^*) has slack if $x_j^* > 0$ ($y_i^* > 0$), (i.e., the corresponding variable restriction is not tight) and a constraint has slack if it is not tight, then the above says that for a primal-dual solution pair it is not possible that a constraint **and** its corresponding (dual) variable has slack.

Proof: Complementary Slackness

Analogous to the proof of weak duality we obtain

$$c^T x^* \leq y^{*T} A x^* \leq b^T y^*$$

Proof: Complementary Slackness

Analogous to the proof of weak duality we obtain

$$c^T x^* \leq y^{*T} A x^* \leq b^T y^*$$

Because of strong duality we then get

$$c^T x^* = y^{*T} A x^* = b^T y^*$$

This gives e.g.

$$\sum_j (y^T A - c^T)_j x_j^* = 0$$

Proof: Complementary Slackness

Analogous to the proof of weak duality we obtain

$$c^T x^* \leq y^{*T} A x^* \leq b^T y^*$$

Because of strong duality we then get

$$c^T x^* = y^{*T} A x^* = b^T y^*$$

This gives e.g.

$$\sum_j (y^T A - c^T)_j x_j^* = 0$$

From the constraint of the dual it follows that $y^T A \geq c^T$. Hence the left hand side is a sum over the product of non-negative numbers. Hence, if e.g. $(y^T A - c^T)_j > 0$ (the j -th constraint in the dual is not tight) then $x_j^* = 0$ (2.). The result for (1./3./4.) follows similarly.

Interpretation of Dual Variables

- Brewer: find mix of ale and beer that maximizes profits

$$\begin{array}{lll} \max & 13a & + \quad 23b \\ \text{s.t.} & 5a & + \quad 15b \leq 480 \\ & 4a & + \quad 4b \leq 160 \\ & 35a & + \quad 20b \leq 1190 \\ & & a, b \geq 0 \end{array}$$

- Entrepreneur: buy resources from brewer at minimum cost
 C, H, M : unit price for corn, hops and malt.

$$\begin{array}{lll} \min & 480C & + \quad 160H \quad + \quad 1190M \\ \text{s.t.} & 5C & + \quad 4H \quad + \quad 35M \geq 13 \\ & 15C & + \quad 4H \quad + \quad 20M \geq 23 \\ & & C, H, M \geq 0 \end{array}$$

Note that brewer won't sell (at least not all) if e.g.

$5C + 4H + 35M < 13$ as then brewing ale would be advantageous.

Interpretation of Dual Variables

- Brewer: find mix of ale and beer that maximizes profits

$$\begin{array}{llll} \max & 13a & + & 23b \\ \text{s.t.} & 5a & + & 15b \leq 480 \\ & 4a & + & 4b \leq 160 \\ & 35a & + & 20b \leq 1190 \\ & & & a, b \geq 0 \end{array}$$

- Entrepreneur: buy resources from brewer at minimum cost
 C, H, M : unit price for corn, hops and malt.

$$\begin{array}{llllll} \min & 480C & + & 160H & + & 1190M \\ \text{s.t.} & 5C & + & 4H & + & 35M \geq 13 \\ & 15C & + & 4H & + & 20M \geq 23 \\ & & & & & C, H, M \geq 0 \end{array}$$

Note that brewer won't sell (at least not all) if e.g.
 $5C + 4H + 35M < 13$ as then brewing ale would be advantageous.

Interpretation of Dual Variables

- Brewer: find mix of ale and beer that maximizes profits

$$\begin{array}{llll} \max & 13a & + & 23b \\ \text{s.t.} & 5a & + & 15b \leq 480 \\ & 4a & + & 4b \leq 160 \\ & 35a & + & 20b \leq 1190 \\ & & & a, b \geq 0 \end{array}$$

- Entrepreneur: buy resources from brewer at minimum cost
 C, H, M : unit price for corn, hops and malt.

$$\begin{array}{llllll} \min & 480C & + & 160H & + & 1190M \\ \text{s.t.} & 5C & + & 4H & + & 35M \geq 13 \\ & 15C & + & 4H & + & 20M \geq 23 \\ & & & & & C, H, M \geq 0 \end{array}$$

Note that brewer won't sell (at least not all) if e.g.

$5C + 4H + 35M < 13$ as then brewing ale would be advantageous.

Interpretation of Dual Variables

Marginal Price:

- ▶ How much money is the brewer willing to pay for additional amount of Corn, Hops, or Malt?
- ▶ We are interested in the marginal price, i.e., what happens if we increase the amount of Corn, Hops, and Malt by ϵ_C , ϵ_H , and ϵ_M , respectively.

The profit increases to $\max\{c^T x \mid Ax \leq b + \epsilon; x \geq 0\}$. Because of strong duality this is equal to

$$\begin{array}{ll} \min & (b^T + \epsilon^T)y \\ \text{s.t.} & A^T y \geq c \\ & y \geq 0 \end{array}$$

Interpretation of Dual Variables

Marginal Price:

- ▶ How much money is the brewer willing to pay for additional amount of Corn, Hops, or Malt?
- ▶ We are interested in the marginal price, i.e., what happens if we increase the amount of Corn, Hops, and Malt by ϵ_C , ϵ_H , and ϵ_M , respectively.

The profit increases to $\max\{c^T x \mid Ax \leq b + \epsilon; x \geq 0\}$. Because of strong duality this is equal to

$$\begin{array}{ll} \min & (b^T + \epsilon^T)y \\ \text{s.t.} & A^T y \geq c \\ & y \geq 0 \end{array}$$

Interpretation of Dual Variables

Marginal Price:

- ▶ How much money is the brewer willing to pay for additional amount of Corn, Hops, or Malt?
- ▶ We are interested in the marginal price, i.e., what happens if we increase the amount of Corn, Hops, and Malt by ϵ_C , ϵ_H , and ϵ_M , respectively.

The profit increases to $\max\{c^T x \mid Ax \leq b + \epsilon; x \geq 0\}$. Because of strong duality this is equal to

$$\begin{array}{ll} \min & (b^T + \epsilon^T)y \\ \text{s.t.} & A^T y \geq c \\ & y \geq 0 \end{array}$$

Interpretation of Dual Variables

Marginal Price:

- ▶ How much money is the brewer willing to pay for additional amount of Corn, Hops, or Malt?
- ▶ We are interested in the marginal price, i.e., what happens if we increase the amount of Corn, Hops, and Malt by ϵ_C , ϵ_H , and ϵ_M , respectively.

The profit increases to $\max\{c^T x \mid Ax \leq b + \epsilon; x \geq 0\}$. Because of strong duality this is equal to

$$\begin{array}{ll} \min & (b^T + \epsilon^T)y \\ \text{s.t.} & A^T y \geq c \\ & y \geq 0 \end{array}$$

Interpretation of Dual Variables

If ϵ is “small” enough then the optimum dual solution y^* might not change. Therefore the profit increases by $\sum_i \epsilon_i y_i^*$.

Therefore we can interpret the dual variables as **marginal prices**.

Note that with this interpretation, complementary slackness becomes obvious.

• If the buyer has slack of some resource (e.g. food) then he is not willing to pay anything for it (corresponding dual variable is zero).

• If the dual variable for some resource is non-zero, then an increase of this resource increases the profit of the buyer. Hence, it makes no sense to have left-overs of this resource. Therefore no slack must be zero.

Interpretation of Dual Variables

If ϵ is “small” enough then the optimum dual solution y^* might not change. Therefore the profit increases by $\sum_i \epsilon_i y_i^*$.

Therefore we can interpret the dual variables as **marginal prices**.

Note that with this interpretation, complementary slackness becomes obvious.

• If the buyer has slack of some resource (e.g. food) then he is not willing to pay anything for it (corresponding dual variable is zero).

• If the dual variable for some resource is non-zero, then an increase of this resource increases the profit of the buyer. Hence, it makes no sense to have an excess of this resource. Therefore slack must be zero.

Interpretation of Dual Variables

If ϵ is “small” enough then the optimum dual solution y^* might not change. Therefore the profit increases by $\sum_i \epsilon_i y_i^*$.

Therefore we can interpret the dual variables as **marginal prices**.

Note that with this interpretation, complementary slackness becomes obvious.

If the primal has slack on some resource (e.g. capacity) then the primal is not willing to pay anything for it (corresponding dual variable is zero). If the primal is not willing to pay for a resource, then the primal must have slack on this resource.

If the primal is willing to pay for a resource, then the primal must not have slack on this resource. If the primal has no slack on a resource, then the primal must be willing to pay for it.

Complementary slackness states that if the primal has slack on a resource, then the dual variable is zero. If the primal is not willing to pay for a resource, then the primal must have slack on this resource.

Interpretation of Dual Variables

If ϵ is “small” enough then the optimum dual solution y^* might not change. Therefore the profit increases by $\sum_i \epsilon_i y_i^*$.

Therefore we can interpret the dual variables as **marginal prices**.

Note that with this interpretation, complementary slackness becomes obvious.

- ▶ If the brewer has slack of some resource (e.g. corn) then he is not willing to pay anything for it (corresponding dual variable is zero).
- ▶ If the dual variable for some resource is non-zero, then an increase of this resource increases the profit of the brewer. Hence, it makes no sense to have left-overs of this resource. Therefore its slack must be zero.

Interpretation of Dual Variables

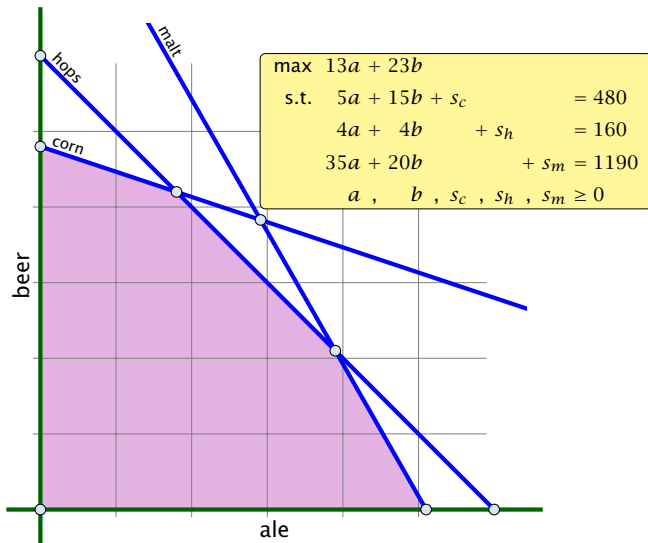
If ϵ is “small” enough then the optimum dual solution y^* might not change. Therefore the profit increases by $\sum_i \epsilon_i y_i^*$.

Therefore we can interpret the dual variables as **marginal prices**.

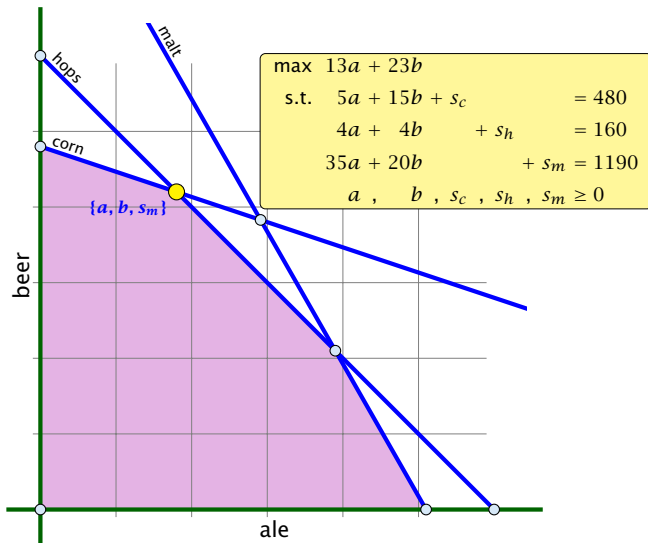
Note that with this interpretation, complementary slackness becomes obvious.

- ▶ If the brewer has slack of some resource (e.g. corn) then he is not willing to pay anything for it (corresponding dual variable is zero).
- ▶ If the dual variable for some resource is non-zero, then an increase of this resource increases the profit of the brewer. Hence, it makes no sense to have left-overs of this resource. Therefore its slack must be zero.

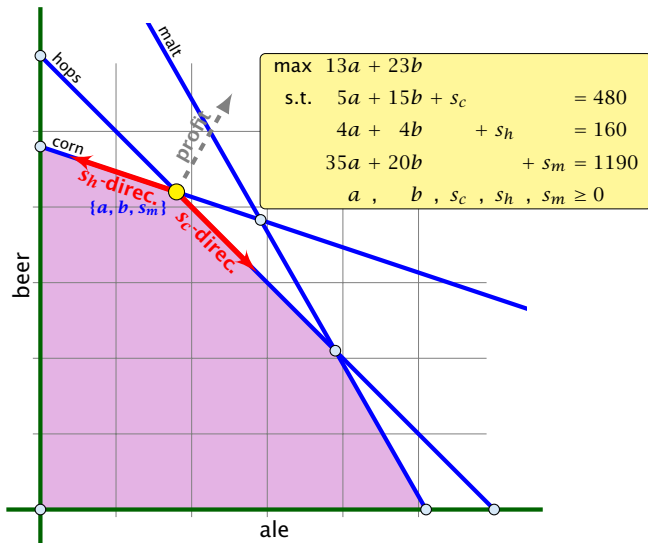
Example



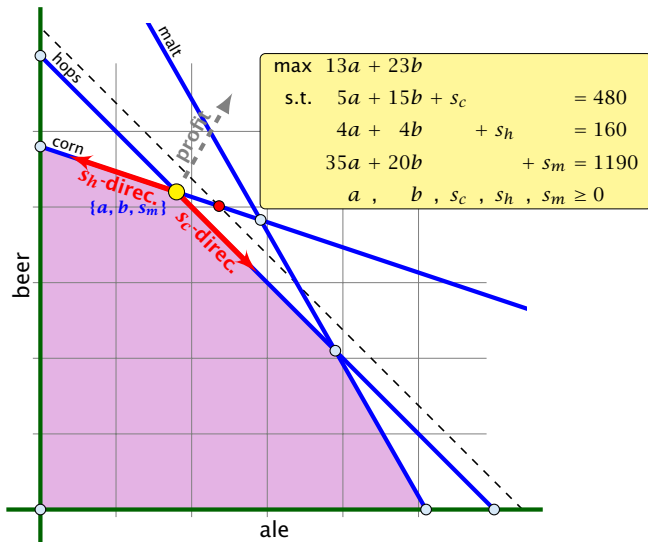
Example



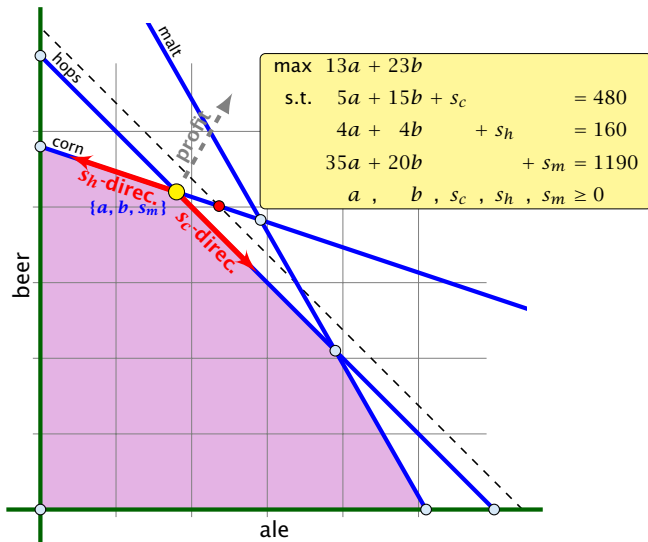
Example



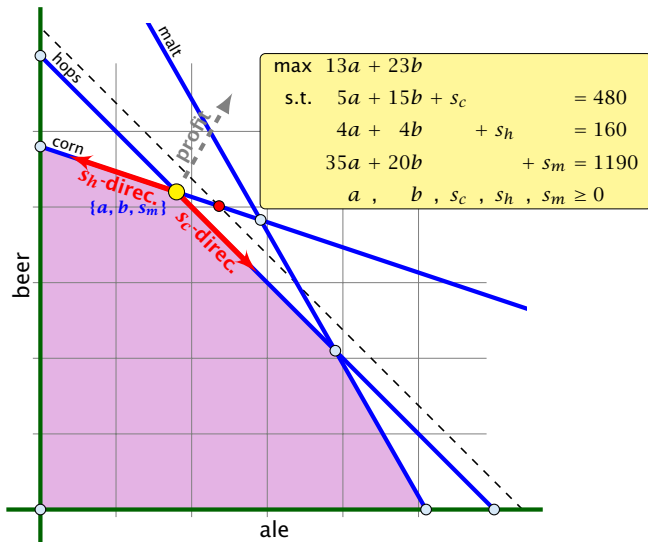
Example



Example



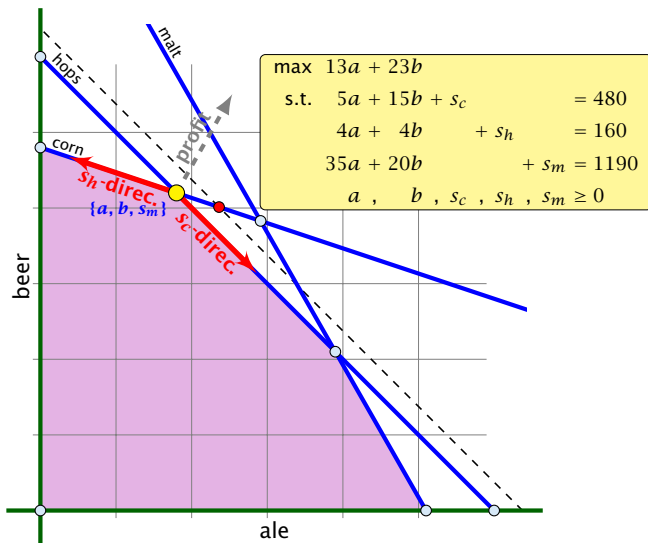
Example



The change in profit when increasing hops by one unit is

$$= c_B^T A_B^{-1} e_h.$$

Example



The change in profit when increasing hops by one unit is

$$= \underbrace{c_B^T A_B^{-1}}_{y^*} e_h.$$

Of course, the previous argument about the increase in the primal objective only holds for the non-degenerate case.

If the optimum basis is degenerate then increasing the supply of one resource may not allow the objective value to increase.

Definition 42

An (s, t) -flow in a (complete) directed graph $G = (V, V \times V, c)$ is a function $f : V \times V \mapsto \mathbb{R}_0^+$ that satisfies

1. For each edge (x, y)

$$0 \leq f_{xy} \leq c_{xy} .$$

(capacity constraints)

2. For each $v \in V \setminus \{s, t\}$

$$\sum_x f_{vx} = \sum_x f_{xv} .$$

(flow conservation constraints)

Definition 42

An (s, t) -flow in a (complete) directed graph $G = (V, V \times V, c)$ is a function $f : V \times V \mapsto \mathbb{R}_0^+$ that satisfies

1. For each edge (x, y)

$$0 \leq f_{xy} \leq c_{xy} .$$

(capacity constraints)

2. For each $v \in V \setminus \{s, t\}$

$$\sum_x f_{vx} = \sum_x f_{xv} .$$

(flow conservation constraints)

Definition 43

The **value of an (s, t) -flow f** is defined as

$$\text{val}(f) = \sum_x f_{sx} - \sum_x f_{xs} .$$

Maximum Flow Problem:

Find an (s, t) -flow with maximum value.

Definition 43

The **value of an (s, t) -flow f** is defined as

$$\text{val}(f) = \sum_x f_{sx} - \sum_x f_{xs} .$$

Maximum Flow Problem:

Find an (s, t) -flow with maximum value.

LP-Formulation of Maxflow

$$\begin{array}{ll}\max & \sum_z f_{sz} - \sum_z f_{zs} \\ \text{s.t.} & \forall (z, w) \in V \times V \quad f_{zw} \leq c_{zw} \quad \ell_{zw} \\ & \forall w \neq s, t \quad \sum_z f_{zw} - \sum_z f_{wz} = 0 \quad p_w \\ & f_{zw} \geq 0\end{array}$$

LP-Formulation of Maxflow

$$\begin{array}{ll}
 \max & \sum_z f_{sz} - \sum_z f_{zs} \\
 \text{s.t.} & \forall (z, w) \in V \times V \quad f_{zw} \leq c_{zw} \quad \ell_{zw} \\
 & \forall w \neq s, t \quad \sum_z f_{zw} - \sum_z f_{wz} = 0 \quad p_w \\
 & f_{zw} \geq 0
 \end{array}$$

$$\begin{array}{ll}
 \min & \sum_{(xy)} c_{xy} \ell_{xy} \\
 \text{s.t.} & f_{xy} \ (x, y \neq s, t) : \quad 1\ell_{xy} - 1p_x + 1p_y \geq 0 \\
 & f_{sy} \ (y \neq s, t) : \quad 1\ell_{sy} \quad \quad + 1p_y \geq 1 \\
 & f_{xs} \ (x \neq s, t) : \quad 1\ell_{xs} - 1p_x \quad \quad \geq -1 \\
 & f_{ty} \ (y \neq s, t) : \quad 1\ell_{ty} \quad \quad + 1p_y \geq 0 \\
 & f_{xt} \ (x \neq s, t) : \quad 1\ell_{xt} - 1p_x \quad \quad \geq 0 \\
 & f_{st} : \quad \quad \quad 1\ell_{st} \quad \quad \geq 1 \\
 & f_{ts} : \quad \quad \quad 1\ell_{ts} \quad \quad \geq -1 \\
 & \quad \quad \quad \ell_{xy} \quad \quad \geq 0
 \end{array}$$

LP-Formulation of Maxflow

$$\begin{array}{ll} \min & \sum_{(x,y)} c_{xy} \ell_{xy} \\ \text{s.t.} & f_{xy} \ (x, y \neq s, t) : \quad 1\ell_{xy} - 1p_x + 1p_y \geq 0 \\ & f_{sy} \ (y \neq s, t) : \quad 1\ell_{sy} - \quad 1 + 1p_y \geq 0 \\ & f_{xs} \ (x \neq s, t) : \quad 1\ell_{xs} - 1p_x + \quad 1 \geq 0 \\ & f_{ty} \ (y \neq s, t) : \quad 1\ell_{ty} - \quad 0 + 1p_y \geq 0 \\ & f_{xt} \ (x \neq s, t) : \quad 1\ell_{xt} - 1p_x + \quad 0 \geq 0 \\ & f_{st} : \quad 1\ell_{st} - \quad 1 + \quad 0 \geq 0 \\ & f_{ts} : \quad 1\ell_{ts} - \quad 0 + \quad 1 \geq 0 \\ & \ell_{xy} \geq 0 \end{array}$$

LP-Formulation of Maxflow

$$\begin{array}{ll} \min & \sum_{(x,y)} c_{xy} \ell_{xy} \\ \text{s.t.} & f_{xy} \ (x, y \neq s, t) : \quad 1\ell_{xy} - 1p_x + 1p_y \geq 0 \\ & f_{sy} \ (y \neq s, t) : \quad 1\ell_{sy} - p_s + 1p_y \geq 0 \\ & f_{xs} \ (x \neq s, t) : \quad 1\ell_{xs} - 1p_x + p_s \geq 0 \\ & f_{ty} \ (y \neq s, t) : \quad 1\ell_{ty} - p_t + 1p_y \geq 0 \\ & f_{xt} \ (x \neq s, t) : \quad 1\ell_{xt} - 1p_x + p_t \geq 0 \\ & f_{st} : \quad 1\ell_{st} - p_s + p_t \geq 0 \\ & f_{ts} : \quad 1\ell_{ts} - p_t + p_s \geq 0 \\ & \ell_{xy} \geq 0 \end{array}$$

with $p_t = 0$ and $p_s = 1$.

LP-Formulation of Maxflow

$$\begin{array}{ll}\min & \sum_{(xy)} c_{xy} \ell_{xy} \\ \text{s.t.} & f_{xy} : \quad 1\ell_{xy} - 1p_x + 1p_y \geq 0 \\ & \ell_{xy} \geq 0 \\ & p_s = 1 \\ & p_t = 0\end{array}$$

We can interpret the ℓ_{xy} value as assigning a length to every edge.

The value p_x for a variable, then can be seen as the distance of x to t (where the distance from s to t is required to be 1 since $p_s = 1$).

The constraint $p_x \leq \ell_{xy} + p_y$ then simply follows from triangle inequality ($d(x, t) \leq d(x, y) + d(y, t) \Rightarrow d(x, t) \leq \ell_{xy} + d(y, t)$).

LP-Formulation of Maxflow

$$\begin{array}{ll}\min & \sum_{(x,y)} c_{xy} \ell_{xy} \\ \text{s.t. } f_{xy} : & 1\ell_{xy} - 1p_x + 1p_y \geq 0 \\ & \ell_{xy} \geq 0 \\ & p_s = 1 \\ & p_t = 0\end{array}$$

We can interpret the ℓ_{xy} value as assigning a length to every edge.

The value p_x for a variable, then can be seen as the distance of x to t (where the distance from s to t is required to be 1 since $p_s = 1$).

The constraint $p_x \leq \ell_{xy} + p_y$ then simply follows from triangle inequality ($d(x,t) \leq d(x,y) + d(y,t) \Rightarrow d(x,t) \leq \ell_{xy} + d(y,t)$).

LP-Formulation of Maxflow

$$\begin{array}{ll}\min & \sum_{(xy)} c_{xy} \ell_{xy} \\ \text{s.t.} & f_{xy} : \quad 1\ell_{xy} - 1p_x + 1p_y \geq 0 \\ & \ell_{xy} \geq 0 \\ & p_s = 1 \\ & p_t = 0\end{array}$$

We can interpret the ℓ_{xy} value as assigning a length to every edge.

The value p_x for a variable, then can be seen as the distance of x to t (where the distance from s to t is required to be 1 since $p_s = 1$).

The constraint $p_x \leq \ell_{xy} + p_y$ then simply follows from triangle inequality ($d(x, t) \leq d(x, y) + d(y, t) \Rightarrow d(x, t) \leq \ell_{xy} + d(y, t)$).

LP-Formulation of Maxflow

$$\begin{array}{ll}\min & \sum_{(xy)} c_{xy} \ell_{xy} \\ \text{s.t.} & f_{xy} : \quad 1\ell_{xy} - 1p_x + 1p_y \geq 0 \\ & \ell_{xy} \geq 0 \\ & p_s = 1 \\ & p_t = 0\end{array}$$

We can interpret the ℓ_{xy} value as assigning a length to every edge.

The value p_x for a variable, then can be seen as the distance of x to t (where the distance from s to t is required to be 1 since $p_s = 1$).

The constraint $p_x \leq \ell_{xy} + p_y$ then simply follows from triangle inequality ($d(x, t) \leq d(x, y) + d(y, t) \Rightarrow d(x, t) \leq \ell_{xy} + d(y, t)$).

One can show that there is an optimum LP-solution for the dual problem that gives an integral assignment of variables.

This means $p_x = 1$ or $p_x = 0$ for our case. This gives rise to a cut in the graph with vertices having value 1 on one side and the other vertices on the other side. The objective function then evaluates the capacity of this cut.

This shows that the Maxflow/Mincut theorem follows from linear programming duality.

One can show that there is an optimum LP-solution for the dual problem that gives an integral assignment of variables.

This means $p_x = 1$ or $p_x = 0$ for our case. This gives rise to a cut in the graph with vertices having value 1 on one side and the other vertices on the other side. The objective function then evaluates the capacity of this cut.

This shows that the Maxflow/Mincut theorem follows from linear programming duality.

One can show that there is an optimum LP-solution for the dual problem that gives an integral assignment of variables.

This means $p_x = 1$ or $p_x = 0$ for our case. This gives rise to a cut in the graph with vertices having value 1 on one side and the other vertices on the other side. The objective function then evaluates the capacity of this cut.

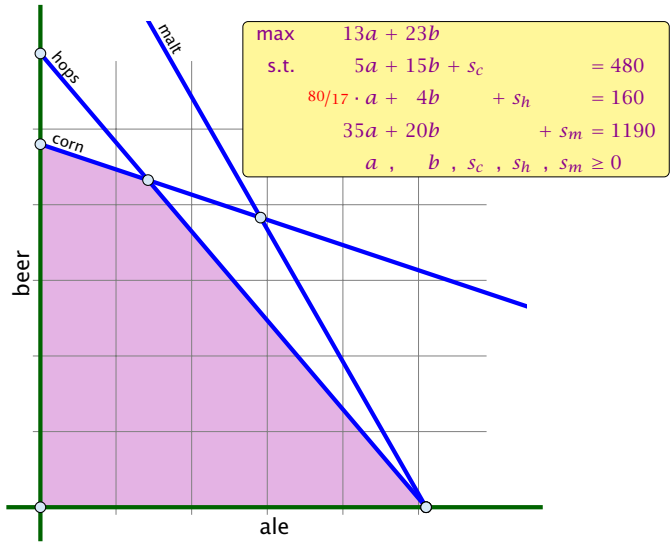
This shows that the Maxflow/Mincut theorem follows from linear programming duality.

Degeneracy Revisited

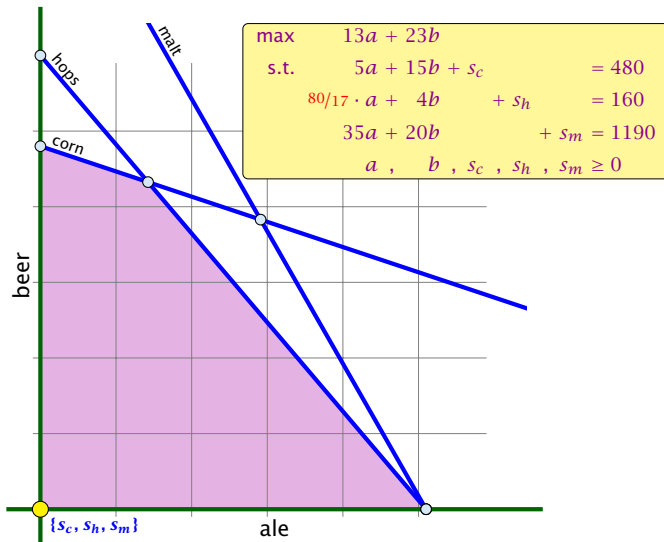
Degeneracy Revisited

If a basis variable is 0 in the basic feasible solution then we may not make progress during an iteration of simplex.

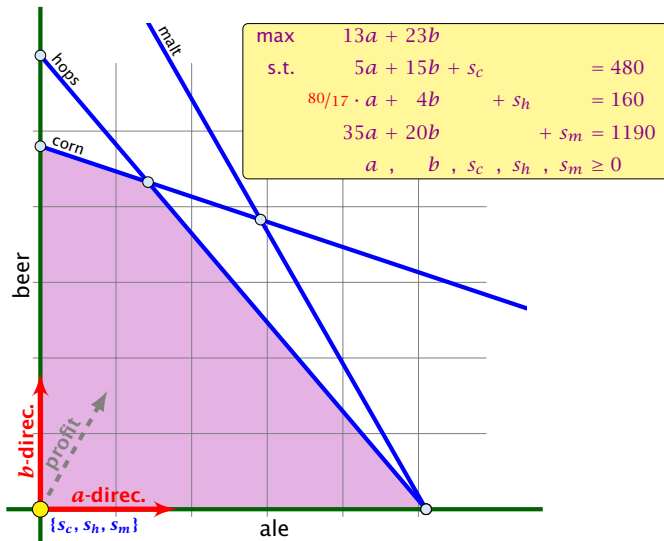
Degenerate Example



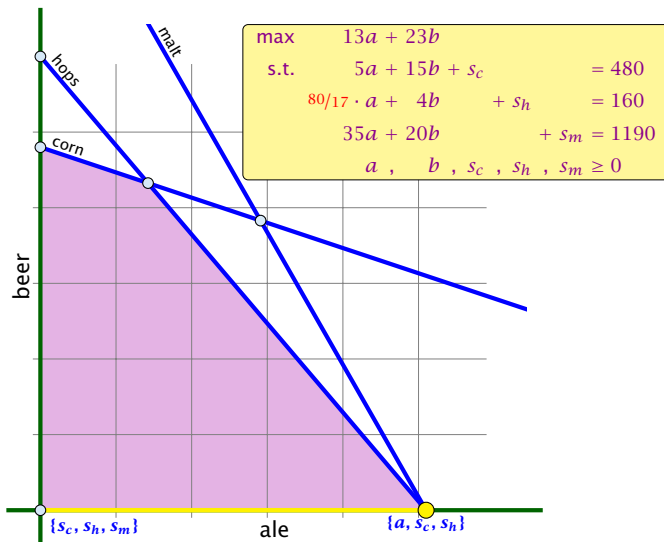
Degenerate Example



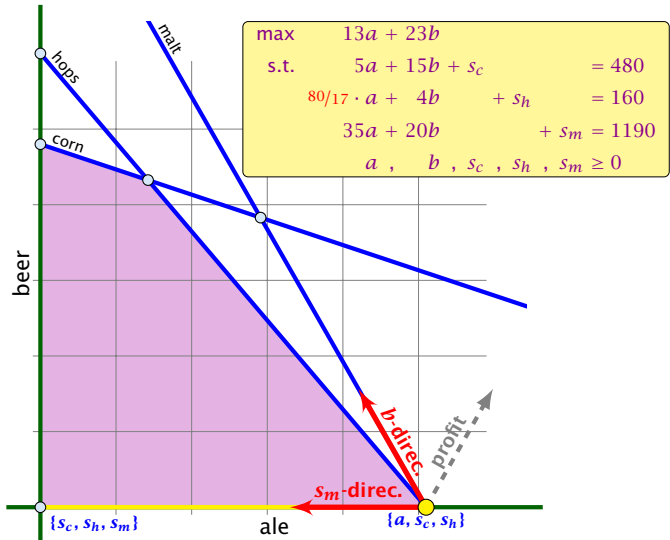
Degenerate Example



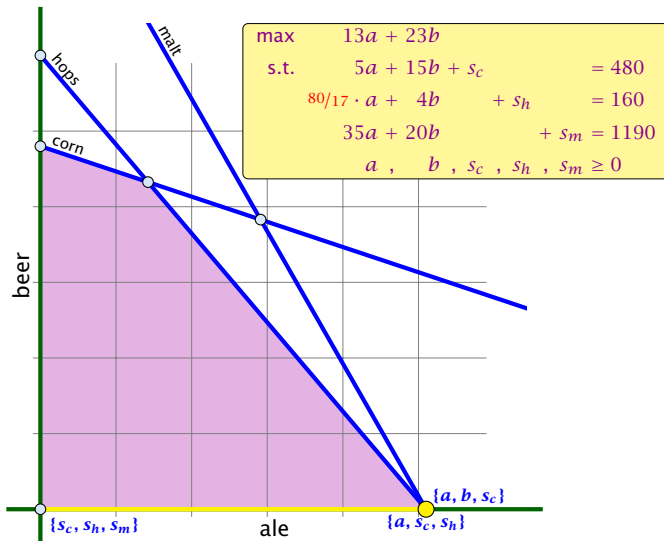
Degenerate Example



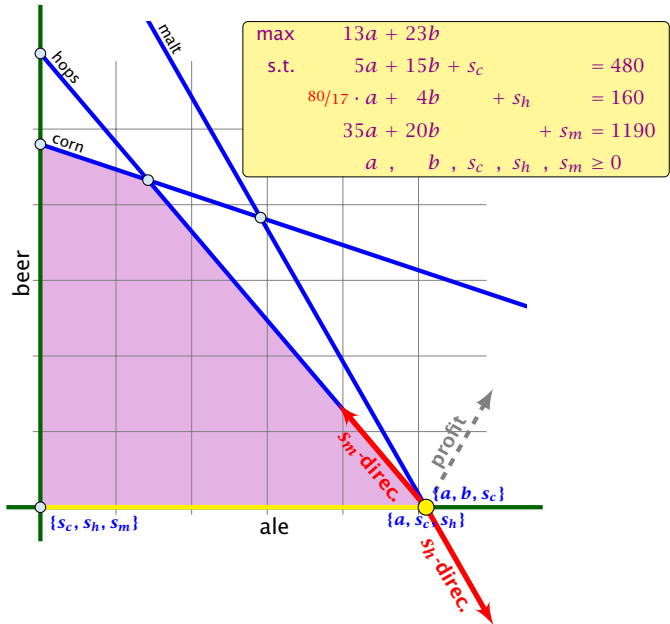
Degenerate Example



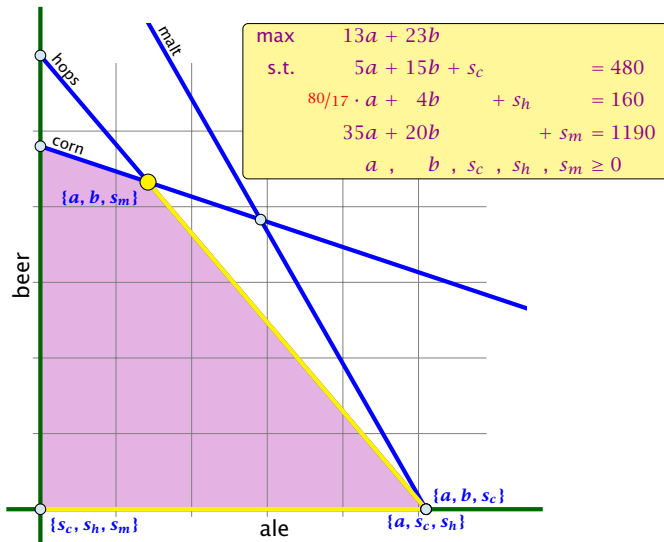
Degenerate Example



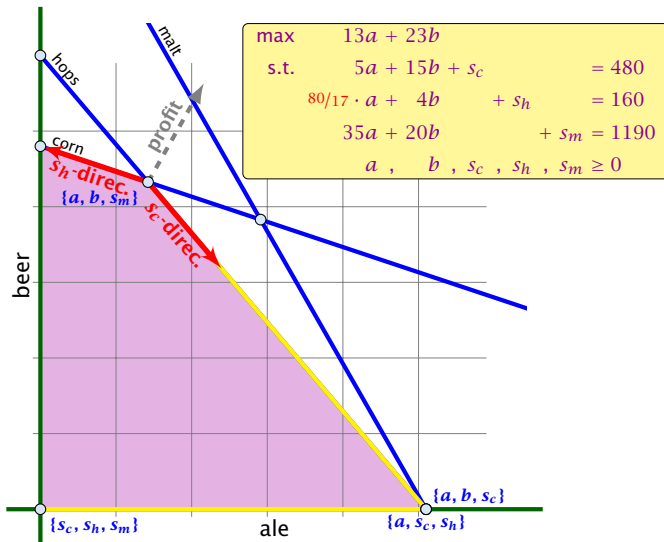
Degenerate Example



Degenerate Example



Degenerate Example



Degeneracy Revisited

If a basis variable is 0 in the basic feasible solution then we may not make progress during an iteration of simplex.

Idea:

Given feasible $LP := \max\{c^T x, Ax = b; x \geq 0\}$. Change it into $LP' := \max\{c^T x, Ax = b', x \geq 0\}$ such that

1. LP' is feasible

2. If a set of basis variables corresponds to an optimal basis of LP' (e.g., B') then it corresponds to an infeasible basis of LP .

3. If a set of columns in A are linearly independent, then LP' has no degenerate basic solution.

Degeneracy Revisited

If a basis variable is 0 in the basic feasible solution then we may not make progress during an iteration of simplex.

Idea:

Given feasible $LP := \max\{c^T x, Ax = b; x \geq 0\}$. Change it into $LP' := \max\{c^T x, Ax = b', x \geq 0\}$ such that

Degeneracy Revisited

If a basis variable is 0 in the basic feasible solution then we may not make progress during an iteration of simplex.

Idea:

Given feasible $LP := \max\{c^T x, Ax = b; x \geq 0\}$. Change it into $LP' := \max\{c^T x, Ax = b', x \geq 0\}$ such that

- I. LP' is feasible
- II. If a set B of basis variables corresponds to an infeasible basis (i.e. $A_B^{-1}b \not\geq 0$) then B corresponds to an infeasible basis in LP' (note that columns in A_B are linearly independent).
- III. LP' has no degenerate basic solutions

Degeneracy Revisited

If a basis variable is **0** in the basic feasible solution then we may not make progress during an iteration of simplex.

Idea:

Given feasible $LP := \max\{c^T x, Ax = b; x \geq 0\}$. Change it into $LP' := \max\{c^T x, Ax = b', x \geq 0\}$ such that

- I. LP' is feasible
- II. If a set B of basis variables corresponds to an **infeasible** basis (i.e. $A_B^{-1}b \not\geq 0$) then B corresponds to an infeasible basis in LP' (note that columns in A_B are linearly independent).
- III. LP' has no degenerate basic solutions

Degeneracy Revisited

If a basis variable is 0 in the basic feasible solution then we may not make progress during an iteration of simplex.

Idea:

Given feasible $LP := \max\{c^T x, Ax = b; x \geq 0\}$. Change it into $LP' := \max\{c^T x, Ax = b', x \geq 0\}$ such that

- I. LP' is feasible
- II. If a set B of basis variables corresponds to an infeasible basis (i.e. $A_B^{-1}b \not\geq 0$) then B corresponds to an infeasible basis in LP' (note that columns in A_B are linearly independent).
- III. LP' has no degenerate basic solutions

Perturbation

Let B be index set of **some** basis with basic solution

$$x_B^* = A_B^{-1}b \geq 0, x_N^* = 0 \quad (\text{i.e. } B \text{ is feasible})$$

Fix

$$b' := b + A_B \begin{pmatrix} \varepsilon \\ \vdots \\ \varepsilon^m \end{pmatrix} \quad \text{for } \varepsilon > 0 .$$

This is the perturbation that we are using.

Perturbation

Let B be index set of **some** basis with basic solution

$$x_B^* = A_B^{-1}b \geq 0, x_N^* = 0 \quad (\text{i.e. } B \text{ is feasible})$$

Fix

$$b' := b + A_B \begin{pmatrix} \varepsilon \\ \vdots \\ \varepsilon^m \end{pmatrix} \quad \text{for } \varepsilon > 0 .$$

This is the perturbation that we are using.

Property I

The new LP is feasible because the set B of basis variables provides a feasible basis:

$$A_B^{-1} \left(b + A_B \begin{pmatrix} \varepsilon \\ \vdots \\ \varepsilon^m \end{pmatrix} \right) = x_B^* + \begin{pmatrix} \varepsilon \\ \vdots \\ \varepsilon^m \end{pmatrix} \geq 0 \ .$$

Property I

The new LP is feasible because the set B of basis variables provides a feasible basis:

$$A_B^{-1} \left(b + A_B \begin{pmatrix} \varepsilon \\ \vdots \\ \varepsilon^m \end{pmatrix} \right) = x_B^* + \begin{pmatrix} \varepsilon \\ \vdots \\ \varepsilon^m \end{pmatrix} \geq 0 .$$

Property II

Let $\tilde{B}$ be a non-feasible basis. This means $(A_{\tilde{B}}^{-1}b)_i < 0$ for some row i .

Property II

Let $\tilde{B}$ be a non-feasible basis. This means $(A_{\tilde{B}}^{-1}b)_i < 0$ for some row i .

Then for small enough $\epsilon > 0$

$$\left(A_{\tilde{B}}^{-1} \left(b + A_B \begin{pmatrix} \epsilon \\ \vdots \\ \epsilon^m \end{pmatrix} \right) \right)_i$$

Property II

Let $\tilde{B}$ be a non-feasible basis. This means $(A_{\tilde{B}}^{-1}b)_i < 0$ for some row i .

Then for small enough $\epsilon > 0$

$$\left(A_{\tilde{B}}^{-1} \left(b + A_B \begin{pmatrix} \epsilon \\ \vdots \\ \epsilon^m \end{pmatrix} \right) \right)_i = (A_{\tilde{B}}^{-1}b)_i + \left(A_{\tilde{B}}^{-1}A_B \begin{pmatrix} \epsilon \\ \vdots \\ \epsilon^m \end{pmatrix} \right)_i < 0$$

Property II

Let $\tilde{B}$ be a non-feasible basis. This means $(A_{\tilde{B}}^{-1}b)_i < 0$ for some row i .

Then for small enough $\epsilon > 0$

$$\left(A_{\tilde{B}}^{-1} \left(b + A_B \begin{pmatrix} \epsilon \\ \vdots \\ \epsilon^m \end{pmatrix} \right) \right)_i = (A_{\tilde{B}}^{-1}b)_i + \left(A_{\tilde{B}}^{-1}A_B \begin{pmatrix} \epsilon \\ \vdots \\ \epsilon^m \end{pmatrix} \right)_i < 0$$

Hence, $\tilde{B}$ is not feasible.

Property III

Let $\tilde{B}$ be a basis. It has an associated solution

$$x_{\tilde{B}}^* = A_{\tilde{B}}^{-1}b + A_{\tilde{B}}^{-1}A_B \begin{pmatrix} \varepsilon \\ \vdots \\ \varepsilon^m \end{pmatrix}$$

in the perturbed instance.

We can view each component of the vector as a polynomial with variable ε of degree at most m .

$A_{\tilde{B}}^{-1}A_B$ has rank m . Therefore no polynomial is 0.

A polynomial of degree at most m has at most m roots (Nullstellen).

Hence, $\varepsilon > 0$ small enough gives that no component of the above vector is 0. Hence, no degeneracies.

Property III

Let $\tilde{B}$ be a basis. It has an associated solution

$$x_{\tilde{B}}^* = A_{\tilde{B}}^{-1}b + A_{\tilde{B}}^{-1}A_B \begin{pmatrix} \varepsilon \\ \vdots \\ \varepsilon^m \end{pmatrix}$$

in the perturbed instance.

We can view each component of the vector as a polynomial with variable ε of degree at most m .

$A_{\tilde{B}}^{-1}A_B$ has rank m . Therefore no polynomial is 0.

A polynomial of degree at most m has at most m roots (Nullstellen).

Hence, $\varepsilon > 0$ small enough gives that no component of the above vector is 0. Hence, no degeneracies.

Property III

Let $\tilde{B}$ be a basis. It has an associated solution

$$x_{\tilde{B}}^* = A_{\tilde{B}}^{-1}b + A_{\tilde{B}}^{-1}A_B \begin{pmatrix} \varepsilon \\ \vdots \\ \varepsilon^m \end{pmatrix}$$

in the perturbed instance.

We can view each component of the vector as a polynomial with variable ε of degree at most m .

$A_{\tilde{B}}^{-1}A_B$ has rank m . Therefore no polynomial is 0.

A polynomial of degree at most m has at most m roots (Nullstellen).

Hence, $\varepsilon > 0$ small enough gives that no component of the above vector is 0. Hence, no degeneracies.

Property III

Let $\tilde{B}$ be a basis. It has an associated solution

$$x_{\tilde{B}}^* = A_{\tilde{B}}^{-1}b + A_{\tilde{B}}^{-1}A_B \begin{pmatrix} \varepsilon \\ \vdots \\ \varepsilon^m \end{pmatrix}$$

in the perturbed instance.

We can view each component of the vector as a polynomial with variable ε of degree at most m .

$A_{\tilde{B}}^{-1}A_B$ has rank m . Therefore no polynomial is 0.

A polynomial of degree at most m has at most m roots (Nullstellen).

Hence, $\varepsilon > 0$ small enough gives that no component of the above vector is 0. Hence, no degeneracies.

Property III

Let $\tilde{B}$ be a basis. It has an associated solution

$$x_{\tilde{B}}^* = A_{\tilde{B}}^{-1}b + A_{\tilde{B}}^{-1}A_B \begin{pmatrix} \varepsilon \\ \vdots \\ \varepsilon^m \end{pmatrix}$$

in the perturbed instance.

We can view each component of the vector as a polynomial with variable ε of degree at most m .

$A_{\tilde{B}}^{-1}A_B$ has rank m . Therefore no polynomial is 0.

A polynomial of degree at most m has at most m roots (Nullstellen).

Hence, $\varepsilon > 0$ small enough gives that no component of the above vector is 0. Hence, no degeneracies.

Property III

Let $\tilde{B}$ be a basis. It has an associated solution

$$x_{\tilde{B}}^* = A_{\tilde{B}}^{-1}b + A_{\tilde{B}}^{-1}A_B \begin{pmatrix} \varepsilon \\ \vdots \\ \varepsilon^m \end{pmatrix}$$

in the perturbed instance.

We can view each component of the vector as a polynomial with variable ε of degree at most m .

$A_{\tilde{B}}^{-1}A_B$ has rank m . Therefore no polynomial is 0.

A polynomial of degree at most m has at most m roots (Nullstellen).

Hence, $\varepsilon > 0$ small enough gives that no component of the above vector is 0. Hence, no degeneracies.

Since, there are no degeneracies Simplex will terminate when run on LP' .

Since, there are no degeneracies Simplex will terminate when run on LP' .

- ▶ If it terminates because the reduced cost vector fulfills

$$\tilde{c} = (c^T - c_B^T A_B^{-1} A) \leq 0$$

then we have found an optimal basis.

Since, there are no degeneracies Simplex will terminate when run on LP' .

- ▶ If it terminates because the reduced cost vector fulfills

$$\tilde{c} = (c^T - c_B^T A_B^{-1} A) \leq 0$$

then we have found an optimal basis. **Note that this basis is also optimal for LP , as the above constraint does not depend on b .**

Since, there are no degeneracies Simplex will terminate when run on LP' .

- ▶ If it terminates because the reduced cost vector fulfills

$$\tilde{c} = (c^T - c_B^T A_B^{-1} A) \leq 0$$

then we have found an optimal basis. **Note that this basis is also optimal for LP , as the above constraint does not depend on b .**

- ▶ If it terminates because it finds a variable x_j with $\tilde{c}_j > 0$ for which the j -th basis direction d , fulfills $d \geq 0$ we know that LP' is unbounded. The basis direction **does not depend on b** . Hence, we also know that LP is unbounded.

Lexicographic Pivoting

Doing calculations with perturbed instances may be costly. Also the right choice of ε is difficult.

Idea:

Simulate behaviour of LP' without explicitly doing a perturbation.

Lexicographic Pivoting

Doing calculations with perturbed instances may be costly. Also the right choice of ε is difficult.

Idea:

Simulate behaviour of LP' without explicitly doing a perturbation.

Lexicographic Pivoting

Doing calculations with perturbed instances may be costly. Also the right choice of ε is difficult.

Idea:

Simulate behaviour of LP' without explicitly doing a perturbation.

Lexicographic Pivoting

We choose the entering variable arbitrarily as before ($\tilde{c}_e > 0$, of course).

If we do not have a choice for the leaving variable then LP' and LP do the same (i.e., choose the same variable).

Otherwise we have to be careful.

Lexicographic Pivoting

We choose the entering variable arbitrarily as before ($\tilde{c}_e > 0$, of course).

If we do not have a choice for the leaving variable then LP' and LP do the same (i.e., choose the same variable).

Otherwise we have to be careful.

Lexicographic Pivoting

We choose the entering variable arbitrarily as before ($\tilde{c}_e > 0$, of course).

If we do not have a choice for the leaving variable then LP' and LP do the same (i.e., choose the same variable).

Otherwise we have to be careful.

Lexicographic Pivoting

We choose the entering variable arbitrarily as before ($\tilde{c}_e > 0$, of course).

If we do not have a choice for the leaving variable then LP' and LP do the same (i.e., choose the same variable).

Otherwise we have to be careful.

Lexicographic Pivoting

In the following we assume that $b \geq 0$. This can be obtained by replacing the initial system $(A \mid b)$ by $(A_B^{-1}A \mid A_B^{-1}b)$ where B is the index set of a feasible basis (found e.g. by the first phase of the Two-phase algorithm).

Then the perturbed instance is

$$b' = b + \begin{pmatrix} \varepsilon \\ \vdots \\ \varepsilon^m \end{pmatrix}$$

Lexicographic Pivoting

In the following we assume that $b \geq 0$. This can be obtained by replacing the initial system $(A \mid b)$ by $(A_B^{-1}A \mid A_B^{-1}b)$ where B is the index set of a feasible basis (found e.g. by the first phase of the Two-phase algorithm).

Then the perturbed instance is

$$b' = b + \begin{pmatrix} \varepsilon \\ \vdots \\ \varepsilon^m \end{pmatrix}$$

Matrix View

Let our linear program be

$$\begin{aligned}c_B^T x_B + c_N^T x_N &= Z \\ A_B x_B + A_N x_N &= b \\ x_B, x_N &\geq 0\end{aligned}$$

The simplex tableaux for basis B is

$$\begin{aligned}(c_N^T - c_B^T A_B^{-1} A_N) x_N &= Z - c_B^T A_B^{-1} b \\ I x_B + A_B^{-1} A_N x_N &= A_B^{-1} b \\ x_B, x_N &\geq 0\end{aligned}$$

The BFS is given by $x_N = 0, x_B = A_B^{-1} b$.

If $(c_N^T - c_B^T A_B^{-1} A_N) \leq 0$ we know that we have an optimum solution.

Lexicographic Pivoting

LP chooses an arbitrary leaving variable that has $\hat{A}_{\ell e} > 0$ and minimizes

$$\theta_{\ell} = \frac{\hat{b}_{\ell}}{\hat{A}_{\ell e}} = \frac{(A_B^{-1}b)_{\ell}}{(A_B^{-1}A_{*e})_{\ell}} \ .$$

ℓ is the index of a leaving variable within B . This means if e.g. $B = \{1, 3, 7, 14\}$ and leaving variable is 3 then $\ell = 2$.

Lexicographic Pivoting

LP chooses an arbitrary leaving variable that has $\hat{A}_{\ell e} > 0$ and minimizes

$$\theta_{\ell} = \frac{\hat{b}_{\ell}}{\hat{A}_{\ell e}} = \frac{(A_B^{-1}b)_{\ell}}{(A_B^{-1}A_{*e})_{\ell}} .$$

ℓ is the index of a leaving variable within B . This means if e.g. $B = \{1, 3, 7, 14\}$ and leaving variable is 3 then $\ell = 2$.

Lexicographic Pivoting

LP chooses an arbitrary leaving variable that has $\hat{A}_{\ell e} > 0$ and minimizes

$$\theta_{\ell} = \frac{\hat{b}_{\ell}}{\hat{A}_{\ell e}} = \frac{(A_B^{-1}b)_{\ell}}{(A_B^{-1}A_{*e})_{\ell}} \ .$$

ℓ is the index of a leaving variable within B . This means if e.g. $B = \{1, 3, 7, 14\}$ and leaving variable is 3 then $\ell = 2$.

Lexicographic Pivoting

LP chooses an arbitrary leaving variable that has $\hat{A}_{\ell e} > 0$ and minimizes

$$\theta_{\ell} = \frac{\hat{b}_{\ell}}{\hat{A}_{\ell e}} = \frac{(A_B^{-1}b)_{\ell}}{(A_B^{-1}A_{*e})_{\ell}} .$$

ℓ is the index of a leaving variable within B . This means if e.g. $B = \{1, 3, 7, 14\}$ and leaving variable is 3 then $\ell = 2$.

Lexicographic Pivoting

Definition 44

$u \leq_{\text{lex}} v$ if and only if the first component in which u and v differ fulfills $u_i \leq v_i$.

Lexicographic Pivoting

LP' chooses an index that minimizes

$$\theta_\ell$$

Lexicographic Pivoting

LP' chooses an index that minimizes

$$\theta_\ell = \frac{\left(A_B^{-1} \left(b + \begin{pmatrix} \varepsilon \\ \vdots \\ \varepsilon^m \end{pmatrix} \right) \right)_\ell}{(A_B^{-1} A_{*e})_\ell}$$

Lexicographic Pivoting

LP' chooses an index that minimizes

$$\theta_\ell = \frac{\left(A_B^{-1} \left(b + \begin{pmatrix} \varepsilon \\ \vdots \\ \varepsilon^m \end{pmatrix} \right) \right)_\ell}{(A_B^{-1} A_{*e})_\ell} = \frac{\left(A_B^{-1} (b \mid I) \begin{pmatrix} 1 \\ \varepsilon \\ \vdots \\ \varepsilon^m \end{pmatrix} \right)_\ell}{(A_B^{-1} A_{*e})_\ell}$$

Lexicographic Pivoting

LP' chooses an index that minimizes

$$\begin{aligned}\theta_\ell &= \frac{\left(A_B^{-1} \left(b + \begin{pmatrix} \varepsilon \\ \vdots \\ \varepsilon^m \end{pmatrix} \right) \right)_\ell}{(A_B^{-1} A_{*e})_\ell} = \frac{\left(A_B^{-1} (b \mid I) \begin{pmatrix} 1 \\ \varepsilon \\ \vdots \\ \varepsilon^m \end{pmatrix} \right)_\ell}{(A_B^{-1} A_{*e})_\ell} \\ &= \frac{\ell\text{-th row of } A_B^{-1} (b \mid I)}{(A_B^{-1} A_{*e})_\ell} \begin{pmatrix} 1 \\ \varepsilon \\ \vdots \\ \varepsilon^m \end{pmatrix}\end{aligned}$$

Lexicographic Pivoting

This means you can choose the variable/row ℓ for which the vector

$$\frac{\ell\text{-th row of } A_B^{-1}(b \mid I)}{(A_B^{-1}A_{*e})_\ell}$$

is lexicographically minimal.

Of course only including rows with $(A_B^{-1}A_{*e})_\ell > 0$.

This technique guarantees that your pivoting is the same as in the perturbed case. This guarantees that cycling does not occur.

Lexicographic Pivoting

This means you can choose the variable/row ℓ for which the vector

$$\frac{\ell\text{-th row of } A_B^{-1}(b \mid I)}{(A_B^{-1}A_{*e})_\ell}$$

is lexicographically minimal.

Of course only including rows with $(A_B^{-1}A_{*e})_\ell > 0$.

This technique guarantees that your pivoting is the same as in the perturbed case. This guarantees that cycling does not occur.

Lexicographic Pivoting

This means you can choose the variable/row ℓ for which the vector

$$\frac{\ell\text{-th row of } A_B^{-1}(b \mid I)}{(A_B^{-1}A_{*e})_\ell}$$

is lexicographically minimal.

Of course only including rows with $(A_B^{-1}A_{*e})_\ell > 0$.

This technique guarantees that your pivoting is the same as in the perturbed case. This guarantees that cycling does not occur.

Number of Simplex Iterations

Number of Simplex Iterations

Each iteration of Simplex can be implemented in polynomial time.

Number of Simplex Iterations

Each iteration of Simplex can be implemented in polynomial time.

If we use lexicographic pivoting we know that Simplex requires at most $\binom{n}{m}$ iterations, because it will not visit a basis twice.

Number of Simplex Iterations

Each iteration of Simplex can be implemented in polynomial time.

If we use lexicographic pivoting we know that Simplex requires at most $\binom{n}{m}$ iterations, because it will not visit a basis twice.

The input size is $L \cdot n \cdot m$, where n is the number of variables, m is the number of constraints, and L is the length of the binary representation of the largest coefficient in the matrix A .

Number of Simplex Iterations

Each iteration of Simplex can be implemented in polynomial time.

If we use lexicographic pivoting we know that Simplex requires at most $\binom{n}{m}$ iterations, because it will not visit a basis twice.

The input size is $L \cdot n \cdot m$, where n is the number of variables, m is the number of constraints, and L is the length of the binary representation of the largest coefficient in the matrix A .

If we really require $\binom{n}{m}$ iterations then Simplex is not a polynomial time algorithm.

Number of Simplex Iterations

Each iteration of Simplex can be implemented in polynomial time.

If we use lexicographic pivoting we know that Simplex requires at most $\binom{n}{m}$ iterations, because it will not visit a basis twice.

The input size is $L \cdot n \cdot m$, where n is the number of variables, m is the number of constraints, and L is the length of the binary representation of the largest coefficient in the matrix A .

If we really require $\binom{n}{m}$ iterations then Simplex is not a polynomial time algorithm.

Can we obtain a better analysis?

Number of Simplex Iterations

Observation

Simplex visits every **feasible** basis at most once.

Number of Simplex Iterations

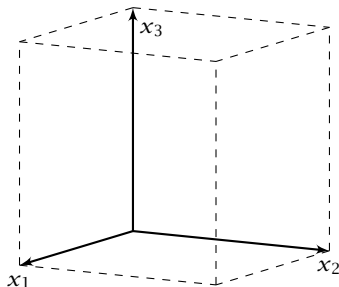
Observation

Simplex visits every **feasible** basis at most once.

However, also the number of feasible bases can be very large.

Example

$$\begin{array}{ll}\max & c^T x \\ \text{s.t.} & 0 \leq x_1 \leq 1 \\ & 0 \leq x_2 \leq 1 \\ & \vdots \\ & 0 \leq x_n \leq 1\end{array}$$

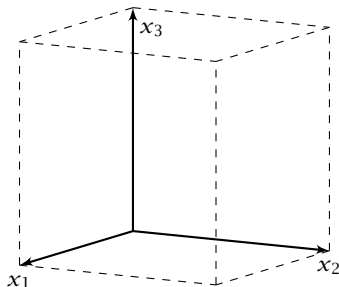


$2n$ constraint on n variables define an n -dimensional hypercube as feasible region.

The feasible region has 2^n vertices.

Example

$$\begin{array}{ll}\max & c^T x \\ \text{s.t.} & 0 \leq x_1 \leq 1 \\ & 0 \leq x_2 \leq 1 \\ & \vdots \\ & 0 \leq x_n \leq 1\end{array}$$



However, Simplex may still run quickly as it usually does not visit all feasible bases.

In the following we give an example of a feasible region for which there is a bad **Pivoting Rule**.

Pivoting Rule

A Pivoting Rule defines how to choose the entering and leaving variable for an iteration of Simplex.

In the non-degenerate case after choosing the entering variable the leaving variable is unique.

Klee Minty Cube

$$\max x_n$$

$$\text{s.t.} \quad 0 \leq x_1 \leq 1$$

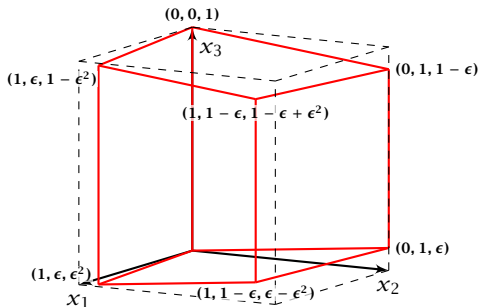
$$\epsilon x_1 \leq x_2 \leq 1 - \epsilon x_1$$

$$\epsilon x_2 \leq x_3 \leq 1 - \epsilon x_2$$

$$\vdots$$

$$\epsilon x_{n-1} \leq x_n \leq 1 - \epsilon x_{n-1}$$

$$x_i \geq 0$$



Observations

- ▶ We have $2n$ constraints, and $3n$ variables (after adding slack variables to every constraint).
- ▶ Every basis is defined by $2n$ variables, and n non-basic variables.
- ▶ There exist degenerate vertices.
- ▶ The degeneracies come from the non-negativity constraints, which are superfluous.
- ▶ In the following all variables x_i stay in the basis at all times.
- ▶ Then, we can uniquely specify a basis by choosing for each variable whether it should be equal to its lower bound, or equal to its upper bound (the slack variable corresponding to the non-tight constraint is part of the basis).
- ▶ We can also simply identify each basis/vertex with the corresponding hypercube vertex obtained by letting $\epsilon \rightarrow 0$.

Observations

- ▶ We have $2n$ constraints, and $3n$ variables (after adding slack variables to every constraint).
- ▶ Every basis is defined by $2n$ variables, and n non-basic variables.
- ▶ There exist degenerate vertices.
- ▶ The degeneracies come from the non-negativity constraints, which are superfluous.
- ▶ In the following all variables x_i stay in the basis at all times.
- ▶ Then, we can uniquely specify a basis by choosing for each variable whether it should be equal to its lower bound, or equal to its upper bound (the slack variable corresponding to the non-tight constraint is part of the basis).
- ▶ We can also simply identify each basis/vertex with the corresponding hypercube vertex obtained by letting $\epsilon \rightarrow 0$.

Observations

- ▶ We have $2n$ constraints, and $3n$ variables (after adding slack variables to every constraint).
- ▶ Every basis is defined by $2n$ variables, and n non-basic variables.
- ▶ There exist degenerate vertices.
- ▶ The degeneracies come from the non-negativity constraints, which are superfluous.
- ▶ In the following all variables x_i stay in the basis at all times.
- ▶ Then, we can uniquely specify a basis by choosing for each variable whether it should be equal to its lower bound, or equal to its upper bound (the slack variable corresponding to the non-tight constraint is part of the basis).
- ▶ We can also simply identify each basis/vertex with the corresponding hypercube vertex obtained by letting $\epsilon \rightarrow 0$.

Observations

- ▶ We have $2n$ constraints, and $3n$ variables (after adding slack variables to every constraint).
- ▶ Every basis is defined by $2n$ variables, and n non-basic variables.
- ▶ There exist degenerate vertices.
- ▶ The degeneracies come from the non-negativity constraints, which are superfluous.
- ▶ In the following all variables x_i stay in the basis at all times.
- ▶ Then, we can uniquely specify a basis by choosing for each variable whether it should be equal to its lower bound, or equal to its upper bound (the slack variable corresponding to the non-tight constraint is part of the basis).
- ▶ We can also simply identify each basis/vertex with the corresponding hypercube vertex obtained by letting $\epsilon \rightarrow 0$.

Observations

- ▶ We have $2n$ constraints, and $3n$ variables (after adding slack variables to every constraint).
- ▶ Every basis is defined by $2n$ variables, and n non-basic variables.
- ▶ There exist degenerate vertices.
- ▶ The degeneracies come from the non-negativity constraints, which are superfluous.
- ▶ In the following all variables x_i stay in the basis at all times.
- ▶ Then, we can uniquely specify a basis by choosing for each variable whether it should be equal to its lower bound, or equal to its upper bound (the slack variable corresponding to the non-tight constraint is part of the basis).
- ▶ We can also simply identify each basis/vertex with the corresponding hypercube vertex obtained by letting $\epsilon \rightarrow 0$.

Observations

- ▶ We have $2n$ constraints, and $3n$ variables (after adding slack variables to every constraint).
- ▶ Every basis is defined by $2n$ variables, and n non-basic variables.
- ▶ There exist degenerate vertices.
- ▶ The degeneracies come from the non-negativity constraints, which are superfluous.
- ▶ In the following all variables x_i stay in the basis at all times.
- ▶ Then, we can uniquely specify a basis by choosing for each variable whether it should be equal to its lower bound, or equal to its upper bound (the slack variable corresponding to the non-tight constraint is part of the basis).
- ▶ We can also simply identify each basis/vertex with the corresponding hypercube vertex obtained by letting $\epsilon \rightarrow 0$.

Observations

- ▶ We have $2n$ constraints, and $3n$ variables (after adding slack variables to every constraint).
- ▶ Every basis is defined by $2n$ variables, and n non-basic variables.
- ▶ There exist degenerate vertices.
- ▶ The degeneracies come from the non-negativity constraints, which are superfluous.
- ▶ In the following all variables x_i stay in the basis at all times.
- ▶ Then, we can uniquely specify a basis by choosing for each variable whether it should be equal to its lower bound, or equal to its upper bound (the slack variable corresponding to the non-tight constraint is part of the basis).
- ▶ We can also simply identify each basis/vertex with the corresponding hypercube vertex obtained by letting $\epsilon \rightarrow 0$.

- ▶ In the following we specify a sequence of bases (identified by the corresponding hypercube node) along which the objective function strictly increases.
- ▶ The basis $(0, \dots, 0, 1)$ is the unique optimal basis.
- ▶ Our sequence S_n starts at $(0, \dots, 0)$ ends with $(0, \dots, 0, 1)$ and visits every node of the hypercube.
- ▶ An unfortunate Pivoting Rule may choose this sequence, and, hence, require an exponential number of iterations.

Analysis

- ▶ In the following we specify a sequence of bases (identified by the corresponding hypercube node) along which the objective function strictly increases.
- ▶ The basis $(0, \dots, 0, 1)$ is the unique optimal basis.
- ▶ Our sequence S_n starts at $(0, \dots, 0)$ ends with $(0, \dots, 0, 1)$ and visits every node of the hypercube.
- ▶ An unfortunate Pivoting Rule may choose this sequence, and, hence, require an exponential number of iterations.

Analysis

- ▶ In the following we specify a sequence of bases (identified by the corresponding hypercube node) along which the objective function strictly increases.
- ▶ The basis $(0, \dots, 0, 1)$ is the unique optimal basis.
- ▶ Our sequence S_n starts at $(0, \dots, 0)$ ends with $(0, \dots, 0, 1)$ and visits every node of the hypercube.
- ▶ An unfortunate Pivoting Rule may choose this sequence, and, hence, require an exponential number of iterations.

Analysis

- ▶ In the following we specify a sequence of bases (identified by the corresponding hypercube node) along which the objective function strictly increases.
- ▶ The basis $(0, \dots, 0, 1)$ is the unique optimal basis.
- ▶ Our sequence S_n starts at $(0, \dots, 0)$ ends with $(0, \dots, 0, 1)$ and visits every node of the hypercube.
- ▶ An unfortunate Pivoting Rule may choose this sequence, and, hence, require an exponential number of iterations.

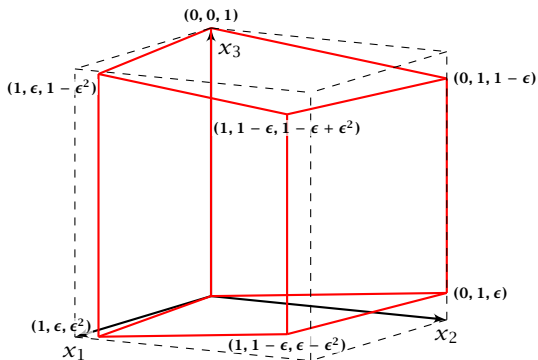
Klee Minty Cube

$$\max x_n$$

$$\text{s.t.} \quad 0 \leq x_1 \leq 1$$

$$\epsilon x_1 \leq x_2 \leq 1 - \epsilon x_1$$

$$\epsilon x_2 \leq x_3 \leq 1 - \epsilon x_2$$



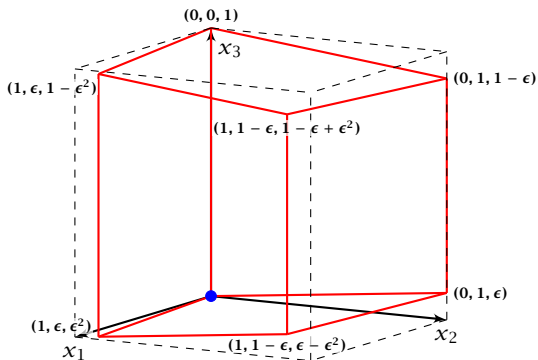
Klee Minty Cube

$$\max x_n$$

$$\text{s.t.} \quad 0 \leq x_1 \leq 1$$

$$\epsilon x_1 \leq x_2 \leq 1 - \epsilon x_1$$

$$\epsilon x_2 \leq x_3 \leq 1 - \epsilon x_2$$



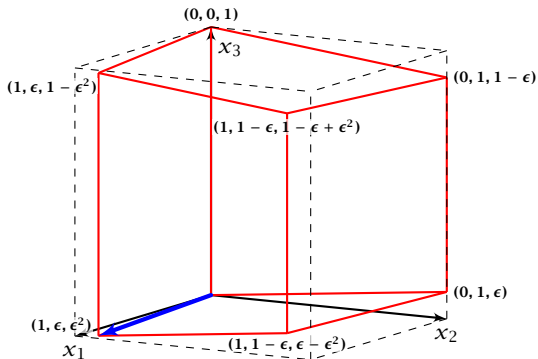
Klee Minty Cube

$$\max x_n$$

$$\text{s.t.} \quad 0 \leq x_1 \leq 1$$

$$\epsilon x_1 \leq x_2 \leq 1 - \epsilon x_1$$

$$\epsilon x_2 \leq x_3 \leq 1 - \epsilon x_2$$



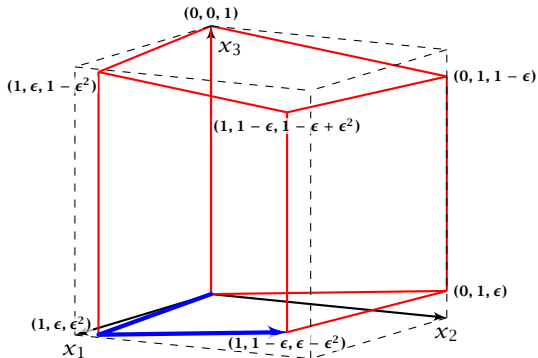
Klee Minty Cube

$$\max x_n$$

$$\text{s.t.} \quad 0 \leq x_1 \leq 1$$

$$\epsilon x_1 \leq x_2 \leq 1 - \epsilon x_1$$

$$\epsilon x_2 \leq x_3 \leq 1 - \epsilon x_2$$



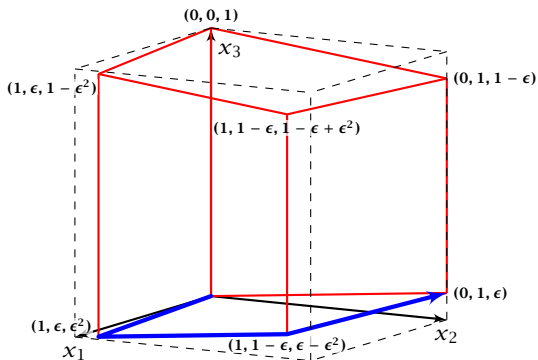
Klee Minty Cube

$$\max x_n$$

$$\text{s.t.} \quad 0 \leq x_1 \leq 1$$

$$\epsilon x_1 \leq x_2 \leq 1 - \epsilon x_1$$

$$\epsilon x_2 \leq x_3 \leq 1 - \epsilon x_2$$



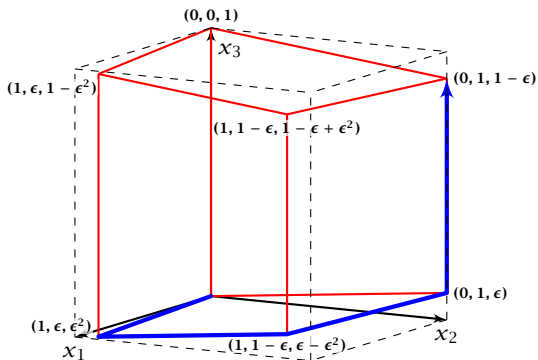
Klee Minty Cube

$$\max x_n$$

$$\text{s.t.} \quad 0 \leq x_1 \leq 1$$

$$\epsilon x_1 \leq x_2 \leq 1 - \epsilon x_1$$

$$\epsilon x_2 \leq x_3 \leq 1 - \epsilon x_2$$



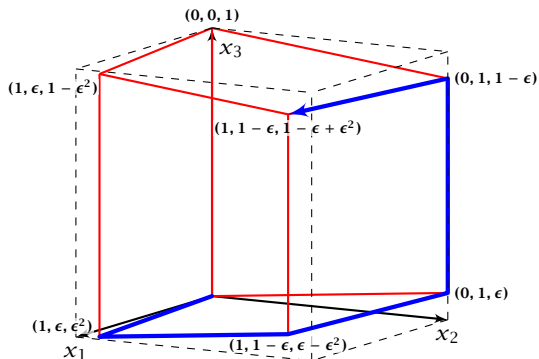
Klee Minty Cube

$$\max x_n$$

$$\text{s.t.} \quad 0 \leq x_1 \leq 1$$

$$\epsilon x_1 \leq x_2 \leq 1 - \epsilon x_1$$

$$\epsilon x_2 \leq x_3 \leq 1 - \epsilon x_2$$



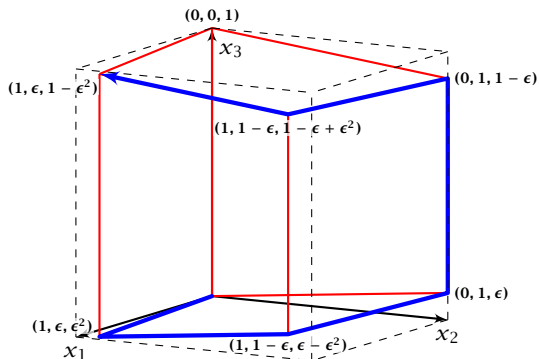
Klee Minty Cube

$$\max x_n$$

$$\text{s.t.} \quad 0 \leq x_1 \leq 1$$

$$\epsilon x_1 \leq x_2 \leq 1 - \epsilon x_1$$

$$\epsilon x_2 \leq x_3 \leq 1 - \epsilon x_2$$



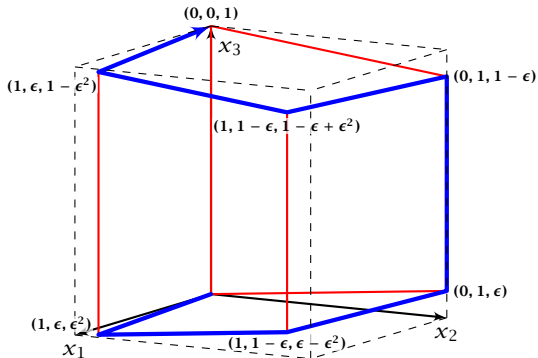
Klee Minty Cube

$$\max x_n$$

$$\text{s.t.} \quad 0 \leq x_1 \leq 1$$

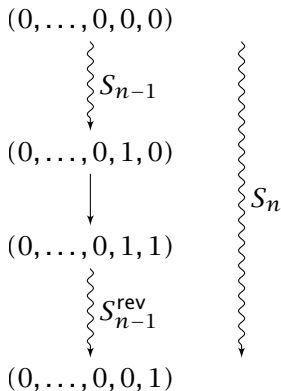
$$\epsilon x_1 \leq x_2 \leq 1 - \epsilon x_1$$

$$\epsilon x_2 \leq x_3 \leq 1 - \epsilon x_2$$



Analysis

The sequence S_n that visits every node of the hypercube is defined recursively



The non-recursive case is $S_1 = 0 \rightarrow 1$

Analysis

Lemma 45

The objective value x_n is increasing along path S_n .

Proof by induction:

$n = 1$: obvious, since $S_1 = 0 \rightarrow 1$, and $1 > 0$.

$n - 1 \rightarrow n$

For the first part the value of x_n is increasing along S_n because

by induction hypothesis x_{n-1} is increasing along S_{n-1} .
Hence, x_{n-1} is increasing twice, also

Going from S_{n-1} to S_n , $0 \rightarrow 1$ increases x_n for
values small enough (i.e. x_{n-1} is small enough).

For the remaining path S_n we have x_{n-1} is increasing

by induction hypothesis x_{n-1} is increasing along S_{n-1} , hence
it is increasing along S_n .

Analysis

Lemma 45

The objective value x_n is increasing along path S_n .

Proof by induction:

$n = 1$: obvious, since $S_1 = 0 \rightarrow 1$, and $1 > 0$.

$n - 1 \rightarrow n$

For the first part the value of x_n is increasing along S_n because of the induction hypothesis. x_{n-1} is increasing along S_{n-1} twice, also. Going from S_{n-1} to S_n , x_{n-1} decreases a bit, but for ϵ small enough x_n is still increasing along S_n . For the remaining path S_n we have x_{n-1} increasing again. By induction hypothesis x_n is increasing along S_n , hence x_n is increasing along S_n .

Analysis

Lemma 45

The objective value x_n is increasing along path S_n .

Proof by induction:

$n = 1$: obvious, since $S_1 = 0 \rightarrow 1$, and $1 > 0$.

$n - 1 \rightarrow n$

At the first part the value of x_n is increasing along S_n by induction hypothesis. It is increasing along S_{n-1} as well, also. Along S_{n-1} the value of x_{n-1} is increasing. For n small enough, x_{n-1} is small enough, and x_n is large enough. So, along the remaining part of S_n we have $x_n > x_{n-1}$. The induction hypothesis is x_{n-1} is increasing along S_{n-1} . It is increasing along S_n .

Analysis

Lemma 45

The objective value x_n is increasing along path S_n .

Proof by induction:

$n = 1$: obvious, since $S_1 = 0 \rightarrow 1$, and $1 > 0$.

$n - 1 \rightarrow n$

- ▶ For the first part the value of $x_n = \epsilon x_{n-1}$.
- ▶ By induction hypothesis x_{n-1} is increasing along S_{n-1} , hence, also x_n .
- ▶ Going from $(0, \dots, 0, 1, 0)$ to $(0, \dots, 0, 1, 1)$ increases x_n for small enough ϵ .
- ▶ For the remaining path S_{n-1}^{rev} we have $x_n = 1 - \epsilon x_{n-1}$.
- ▶ By induction hypothesis x_{n-1} is increasing along S_{n-1} , hence $-\epsilon x_{n-1}$ is increasing along S_{n-1}^{rev} .

Analysis

Lemma 45

The objective value x_n is increasing along path S_n .

Proof by induction:

$n = 1$: obvious, since $S_1 = 0 \rightarrow 1$, and $1 > 0$.

$n - 1 \rightarrow n$

- ▶ For the first part the value of $x_n = \epsilon x_{n-1}$.
- ▶ By induction hypothesis x_{n-1} is increasing along S_{n-1} , hence, also x_n .
- ▶ Going from $(0, \dots, 0, 1, 0)$ to $(0, \dots, 0, 1, 1)$ increases x_n for small enough ϵ .
- ▶ For the remaining path S_{n-1}^{rev} we have $x_n = 1 - \epsilon x_{n-1}$.
- ▶ By induction hypothesis x_{n-1} is increasing along S_{n-1} , hence $-\epsilon x_{n-1}$ is increasing along S_{n-1}^{rev} .

Analysis

Lemma 45

The objective value x_n is increasing along path S_n .

Proof by induction:

$n = 1$: obvious, since $S_1 = 0 \rightarrow 1$, and $1 > 0$.

$n - 1 \rightarrow n$

- ▶ For the first part the value of $x_n = \epsilon x_{n-1}$.
- ▶ By induction hypothesis x_{n-1} is increasing along S_{n-1} , hence, also x_n .
- ▶ Going from $(0, \dots, 0, 1, 0)$ to $(0, \dots, 0, 1, 1)$ increases x_n for small enough ϵ .
- ▶ For the remaining path S_{n-1}^{rev} we have $x_n = 1 - \epsilon x_{n-1}$.
- ▶ By induction hypothesis x_{n-1} is increasing along S_{n-1} , hence $-\epsilon x_{n-1}$ is increasing along S_{n-1}^{rev} .

Analysis

Lemma 45

The objective value x_n is increasing along path S_n .

Proof by induction:

$n = 1$: obvious, since $S_1 = 0 \rightarrow 1$, and $1 > 0$.

$n - 1 \rightarrow n$

- ▶ For the first part the value of $x_n = \epsilon x_{n-1}$.
- ▶ By induction hypothesis x_{n-1} is increasing along S_{n-1} , hence, also x_n .
- ▶ Going from $(0, \dots, 0, 1, 0)$ to $(0, \dots, 0, 1, 1)$ increases x_n for small enough ϵ .
- ▶ For the remaining path S_{n-1}^{rev} we have $x_n = 1 - \epsilon x_{n-1}$.
 - ▶ By induction hypothesis x_{n-1} is increasing along S_{n-1} , hence $-\epsilon x_{n-1}$ is increasing along S_{n-1}^{rev} .

Analysis

Lemma 45

The objective value x_n is increasing along path S_n .

Proof by induction:

$n = 1$: obvious, since $S_1 = 0 \rightarrow 1$, and $1 > 0$.

$n - 1 \rightarrow n$

- ▶ For the first part the value of $x_n = \epsilon x_{n-1}$.
- ▶ By induction hypothesis x_{n-1} is increasing along S_{n-1} , hence, also x_n .
- ▶ Going from $(0, \dots, 0, 1, 0)$ to $(0, \dots, 0, 1, 1)$ increases x_n for small enough ϵ .
- ▶ For the remaining path S_{n-1}^{rev} we have $x_n = 1 - \epsilon x_{n-1}$.
- ▶ By induction hypothesis x_{n-1} is increasing along S_{n-1} , hence $-\epsilon x_{n-1}$ is increasing along S_{n-1}^{rev} .

Remarks about Simplex

Observation

The simplex algorithm takes at most $\binom{n}{m}$ iterations. Each iteration can be implemented in time $\mathcal{O}(mn)$.

In practise it usually takes a linear number of iterations.

Remarks about Simplex

Theorem

For almost all known **deterministic** pivoting rules (rules for choosing entering and leaving variables) there exist lower bounds that require the algorithm to have exponential running time ($\Omega(2^{\Omega(n)})$) (e.g. Klee Minty 1972).

Remarks about Simplex

Theorem

For some standard **randomized** pivoting rules there exist subexponential lower bounds ($\Omega(2^{\Omega(n^\alpha)})$ for $\alpha > 0$) (Friedmann, Hansen, Zwick 2011).

Remarks about Simplex

Conjecture (Hirsch 1957)

The edge-vertex graph of an m -facet polytope in d -dimensional Euclidean space has diameter no more than $m - d$.

The conjecture has been proven wrong in 2010.

But the question whether the diameter is perhaps of the form $\mathcal{O}(\text{poly}(m, d))$ is open.

8 Seidels LP-algorithm

- ▶ Suppose we want to solve $\min\{c^T x \mid Ax \geq b; x \geq 0\}$, where $x \in \mathbb{R}^d$ and we have m constraints.
- ▶ In the worst-case Simplex runs in time roughly $\mathcal{O}(m(m+d)\binom{m+d}{m}) \approx (m+d)^m$. (slightly better bounds on the running time exist, but will not be discussed here).
- ▶ If d is much smaller than m one can do a lot better.
- ▶ In the following we develop an algorithm with running time $\mathcal{O}(d! \cdot m)$, i.e., linear in m .

8 Seidels LP-algorithm

- ▶ Suppose we want to solve $\min\{c^T x \mid Ax \geq b; x \geq 0\}$, where $x \in \mathbb{R}^d$ and we have m constraints.
- ▶ In the worst-case Simplex runs in time roughly $\mathcal{O}(m(m+d)\binom{m+d}{m}) \approx (m+d)^m$. (slightly better bounds on the running time exist, but will not be discussed here).
- ▶ If d is much smaller than m one can do a lot better.
- ▶ In the following we develop an algorithm with running time $\mathcal{O}(d! \cdot m)$, i.e., linear in m .

8 Seidels LP-algorithm

- ▶ Suppose we want to solve $\min\{c^T x \mid Ax \geq b; x \geq 0\}$, where $x \in \mathbb{R}^d$ and we have m constraints.
- ▶ In the worst-case Simplex runs in time roughly $\mathcal{O}(m(m+d)\binom{m+d}{m}) \approx (m+d)^m$. (slightly better bounds on the running time exist, but will not be discussed here).
- ▶ If d is much smaller than m one can do a lot better.
- ▶ In the following we develop an algorithm with running time $\mathcal{O}(d! \cdot m)$, i.e., linear in m .

8 Seidels LP-algorithm

- ▶ Suppose we want to solve $\min\{c^T x \mid Ax \geq b; x \geq 0\}$, where $x \in \mathbb{R}^d$ and we have m constraints.
- ▶ In the worst-case Simplex runs in time roughly $\mathcal{O}(m(m+d)\binom{m+d}{m}) \approx (m+d)^m$. (slightly better bounds on the running time exist, but will not be discussed here).
- ▶ If d is much smaller than m one can do a lot better.
- ▶ In the following we develop an algorithm with running time $\mathcal{O}(d! \cdot m)$, i.e., **linear in m** .

8 Seidels LP-algorithm

Setting:

- ▶ We assume an LP of the form

$$\begin{array}{ll} \min & c^T x \\ \text{s.t.} & Ax \geq b \\ & x \geq 0 \end{array}$$

- ▶ We assume that the LP is **bounded**.

Ensuring Conditions

Given a **standard minimization LP**

$$\begin{array}{ll}\min & c^T x \\ \text{s.t.} & Ax \geq b \\ & x \geq 0\end{array}$$

how can we obtain an LP of the required form?

- ▶ **Compute a lower bound on $c^T x$ for any basic feasible solution.**

Computing a Lower Bound

Let s denote the smallest common multiple of all denominators of entries in A, b .

Multiply entries in A, b by s to obtain integral entries. This does not change the feasible region.

Add slack variables to A ; denote the resulting matrix with $\tilde{A}$.

If B is an optimal basis then x_B with $\tilde{A}_B x_B = \tilde{b}$, gives an optimal assignment to the basis variables (non-basic variables are 0).

Computing a Lower Bound

Let s denote the smallest common multiple of all denominators of entries in A, b .

Multiply entries in A, b by s to obtain integral entries. **This does not change the feasible region.**

Add slack variables to A ; denote the resulting matrix with $\tilde{A}$.

If B is an optimal basis then x_B with $\tilde{A}_B x_B = \tilde{b}$, gives an optimal assignment to the basis variables (non-basic variables are 0).

Computing a Lower Bound

Let s denote the smallest common multiple of all denominators of entries in A, b .

Multiply entries in A, b by s to obtain integral entries. **This does not change the feasible region.**

Add slack variables to A ; denote the resulting matrix with $\bar{A}$.

If B is an optimal basis then x_B with $\bar{A}_B x_B = \bar{b}$, gives an optimal assignment to the basis variables (non-basic variables are 0).

Computing a Lower Bound

Let s denote the smallest common multiple of all denominators of entries in A, b .

Multiply entries in A, b by s to obtain integral entries. **This does not change the feasible region.**

Add slack variables to A ; denote the resulting matrix with $\bar{A}$.

If B is an optimal basis then x_B with $\bar{A}_B x_B = \bar{b}$, gives an optimal assignment to the basis variables (non-basic variables are 0).

Theorem 46 (Cramers Rule)

Let M be a matrix with $\det(M) \neq 0$. Then the solution to the system $Mx = b$ is given by

$$x_i = \frac{\det(M_i)}{\det(M)} ,$$

where M_i is the matrix obtained from M by replacing the i -th column by the vector b .

Proof:

Proof:

- ▶ Define

$$X_i = \begin{pmatrix} | & & | & | & | & & | \\ e_1 & \cdots & e_{i-1} & x & e_{i+1} & \cdots & e_n \\ | & & | & | & | & & | \end{pmatrix}$$

Note that expanding along the i -th column gives that $\det(X_i) = x_i$.

- ▶ Further, we have

$$MX_i = \begin{pmatrix} | & & | & | & | & & | \\ Me_1 & \cdots & Me_{i-1} & Mx & Me_{i+1} & \cdots & Me_n \\ | & & | & | & | & & | \end{pmatrix} = M_i$$

- ▶ Hence,

$$x_i = \det(X_i) = \frac{\det(M_i)}{\det(M)}$$

Proof:

- ▶ Define

$$X_i = \begin{pmatrix} | & & | & | & | & & | \\ e_1 & \cdots & e_{i-1} & x & e_{i+1} & \cdots & e_n \\ | & & | & | & | & & | \end{pmatrix}$$

Note that expanding along the i -th column gives that $\det(X_i) = x_i$.

- ▶ Further, we have

$$MX_i = \begin{pmatrix} | & & | & | & | & & | \\ Me_1 & \cdots & Me_{i-1} & Mx & Me_{i+1} & \cdots & Me_n \\ | & & | & | & | & & | \end{pmatrix} = M_i$$

- ▶ Hence,

$$x_i = \det(X_i) = \frac{\det(M_i)}{\det(M)}$$

Proof:

- Define

$$X_i = \begin{pmatrix} | & & | & | & | & \\ e_1 & \cdots & e_{i-1} & x & e_{i+1} & \cdots & e_n \\ | & & | & | & | & \end{pmatrix}$$

Note that expanding along the i -th column gives that $\det(X_i) = x_i$.

- Further, we have

$$MX_i = \begin{pmatrix} | & & | & | & | & \\ Me_1 & \cdots & Me_{i-1} & Mx & Me_{i+1} & \cdots & Me_n \\ | & & | & | & | & \end{pmatrix} = M_i$$

- Hence,

$$x_i = \det(X_i) = \frac{\det(M_i)}{\det(M)}$$

Proof:

- Define

$$X_i = \begin{pmatrix} | & & | & | & | & \\ e_1 & \cdots & e_{i-1} & x & e_{i+1} & \cdots & e_n \\ | & & | & | & | & \end{pmatrix}$$

Note that expanding along the i -th column gives that $\det(X_i) = x_i$.

- Further, we have

$$MX_i = \begin{pmatrix} | & & | & | & | & \\ Me_1 & \cdots & Me_{i-1} & Mx & Me_{i+1} & \cdots & Me_n \\ | & & | & | & | & \end{pmatrix} = M_i$$

- Hence,

$$x_i = \det(X_i) = \frac{\det(M_i)}{\det(M)}$$

Bounding the Determinant

Let Z be the maximum absolute entry occurring in $\bar{A}$, $\bar{b}$ or c . Let C denote the matrix obtained from $\bar{A}_B$ by replacing the j -th column with vector $\bar{b}$ (for some j).

Observe that

$$|\det(C)|$$

Bounding the Determinant

Let Z be the maximum absolute entry occurring in $\bar{A}$, $\bar{b}$ or c . Let C denote the matrix obtained from $\bar{A}_B$ by replacing the j -th column with vector $\bar{b}$ (for some j).

Observe that

$$|\det(C)| = \left| \sum_{\pi \in S_m} \operatorname{sgn}(\pi) \prod_{1 \leq i \leq m} C_{i\pi(i)} \right|$$

Bounding the Determinant

Let Z be the maximum absolute entry occurring in $\bar{A}$, $\bar{b}$ or c . Let C denote the matrix obtained from $\bar{A}_B$ by replacing the j -th column with vector $\bar{b}$ (for some j).

Observe that

$$\begin{aligned} |\det(C)| &= \left| \sum_{\pi \in S_m} \operatorname{sgn}(\pi) \prod_{1 \leq i \leq m} C_{i\pi(i)} \right| \\ &\leq \sum_{\pi \in S_m} \prod_{1 \leq i \leq m} |C_{i\pi(i)}| \end{aligned}$$

Bounding the Determinant

Let Z be the maximum absolute entry occurring in $\bar{A}, \bar{b}$ or c . Let C denote the matrix obtained from $\bar{A}_B$ by replacing the j -th column with vector $\bar{b}$ (for some j).

Observe that

$$\begin{aligned} |\det(C)| &= \left| \sum_{\pi \in S_m} \operatorname{sgn}(\pi) \prod_{1 \leq i \leq m} C_{i\pi(i)} \right| \\ &\leq \sum_{\pi \in S_m} \prod_{1 \leq i \leq m} |C_{i\pi(i)}| \\ &\leq m! \cdot Z^m . \end{aligned}$$

Bounding the Determinant

Alternatively, Hadamards inequality gives

$$|\det(C)|$$

Bounding the Determinant

Alternatively, Hadamard's inequality gives

$$|\det(C)| \leq \prod_{i=1}^m \|C_{*i}\|$$

Bounding the Determinant

Alternatively, Hadamard's inequality gives

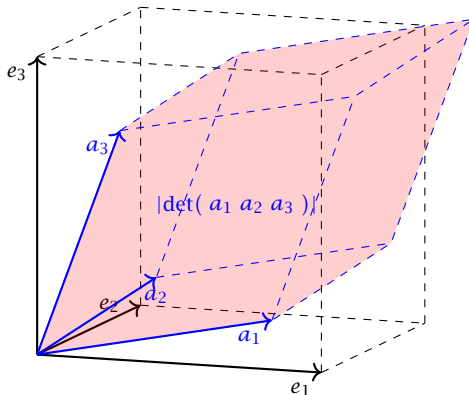
$$|\det(C)| \leq \prod_{i=1}^m \|C_{*i}\| \leq \prod_{i=1}^m (\sqrt{m}Z)$$

Bounding the Determinant

Alternatively, Hadamard's inequality gives

$$\begin{aligned} |\det(C)| &\leq \prod_{i=1}^m \|C_{*i}\| \leq \prod_{i=1}^m (\sqrt{m}Z) \\ &\leq m^{m/2} Z^m . \end{aligned}$$

Hadamards Inequality



Hadamards inequality says that the volume of the red parallelepiped (**Spat**) is smaller than the volume in the black cube (if $\|e_1\| = \|a_1\|$, $\|e_2\| = \|a_2\|$, $\|e_3\| = \|a_3\|$).

Ensuring Conditions

Given a **standard minimization LP**

$$\begin{array}{ll} \min & c^T x \\ \text{s.t.} & Ax \geq b \\ & x \geq 0 \end{array}$$

how can we obtain an LP of the required form?

- ▶ **Compute a lower bound on $c^T x$ for any basic feasible solution.** Add the constraint $c^T x \geq -dZ(m! \cdot Z^m) - 1$. **Note that this constraint is superfluous unless the LP is unbounded.**

Ensuring Conditions

Compute an optimum basis for the new LP.

- ▶ If the cost is $c^T x = -(dZ)(m! \cdot Z^m) - 1$ we know that the original LP is unbounded.
- ▶ Otw. we have an optimum basis.

In the following we use $\mathcal{H}$ to denote the set of all constraints apart from the constraint $c^T x \geq -dZ(m! \cdot Z^m) - 1$.

We give a routine $\text{SeidelLP}(\mathcal{H}, d)$ that is given a set $\mathcal{H}$ of **explicit, non-degenerate** constraints over d variables, and minimizes $c^T x$ over all feasible points.

In addition it obeys the implicit constraint $c^T x \geq -(dZ)(m! \cdot Z^m) - 1$.

In the following we use $\mathcal{H}$ to denote the set of all constraints apart from the constraint $c^T x \geq -dZ(m! \cdot Z^m) - 1$.

We give a routine $\text{SeidelLP}(\mathcal{H}, d)$ that is given a set $\mathcal{H}$ of **explicit, non-degenerate** constraints over d variables, and minimizes $c^T x$ over all feasible points.

In addition it obeys the implicit constraint $c^T x \geq -(dZ)(m! \cdot Z^m) - 1$.

In the following we use $\mathcal{H}$ to denote the set of all constraints apart from the constraint $c^T x \geq -dZ(m! \cdot Z^m) - 1$.

We give a routine $\text{SeidelLP}(\mathcal{H}, d)$ that is given a set $\mathcal{H}$ of **explicit, non-degenerate** constraints over d variables, and minimizes $c^T x$ over all feasible points.

In addition it obeys the implicit constraint $c^T x \geq -(dZ)(m! \cdot Z^m) - 1$.

In the following we use $\mathcal{H}$ to denote the set of all constraints apart from the constraint $c^T x \geq -dZ(m! \cdot Z^m) - 1$.

We give a routine $\text{SeidelLP}(\mathcal{H}, d)$ that is given a set $\mathcal{H}$ of **explicit, non-degenerate** constraints over d variables, and minimizes $c^T x$ over all feasible points.

In addition it obeys the implicit constraint $c^T x \geq -(dZ)(m! \cdot Z^m) - 1$.

Algorithm 1 SeidelLP($\mathcal{H}, d$)

1: **if** $d = 1$ **then** solve 1-dimensional problem and return;

Algorithm 1 SeidelLP($\mathcal{H}, d$)

- 1: **if** $d = 1$ **then** solve 1-dimensional problem and return;
- 2: **if** $\mathcal{H} = \emptyset$ **then** return x on implicit constraint hyperplane

Algorithm 1 SeidelLP($\mathcal{H}, d$)

- 1: **if** $d = 1$ **then** solve 1-dimensional problem and return;
- 2: **if** $\mathcal{H} = \emptyset$ **then** return x on implicit constraint hyperplane
- 3: choose **random** constraint $h \in \mathcal{H}$

Algorithm 1 SeidelLP($\mathcal{H}, d$)

- 1: **if** $d = 1$ **then** solve 1-dimensional problem and return;
- 2: **if** $\mathcal{H} = \emptyset$ **then** return x on implicit constraint hyperplane
- 3: choose **random** constraint $h \in \mathcal{H}$
- 4: $\hat{\mathcal{H}} \leftarrow \mathcal{H} \setminus \{h\}$

Algorithm 1 SeidelLP($\mathcal{H}, d$)

- 1: **if** $d = 1$ **then** solve 1-dimensional problem and return;
- 2: **if** $\mathcal{H} = \emptyset$ **then** return x on implicit constraint hyperplane
- 3: choose **random** constraint $h \in \mathcal{H}$
- 4: $\hat{\mathcal{H}} \leftarrow \mathcal{H} \setminus \{h\}$
- 5: $\hat{x}^* \leftarrow \text{SeidelLP}(\hat{\mathcal{H}}, d)$

Algorithm 1 SeidelLP($\mathcal{H}, d$)

- 1: **if** $d = 1$ **then** solve 1-dimensional problem and return;
- 2: **if** $\mathcal{H} = \emptyset$ **then** return x on implicit constraint hyperplane
- 3: choose **random** constraint $h \in \mathcal{H}$
- 4: $\hat{\mathcal{H}} \leftarrow \mathcal{H} \setminus \{h\}$
- 5: $\hat{x}^* \leftarrow \text{SeidelLP}(\hat{\mathcal{H}}, d)$
- 6: **if** $\hat{x}^* = \text{infeasible}$ **then return** infeasible

Algorithm 1 SeidelLP($\mathcal{H}, d$)

- 1: **if** $d = 1$ **then** solve 1-dimensional problem and return;
- 2: **if** $\mathcal{H} = \emptyset$ **then** return x on implicit constraint hyperplane
- 3: choose **random** constraint $h \in \mathcal{H}$
- 4: $\hat{\mathcal{H}} \leftarrow \mathcal{H} \setminus \{h\}$
- 5: $\hat{x}^* \leftarrow \text{SeidelLP}(\hat{\mathcal{H}}, d)$
- 6: **if** $\hat{x}^* = \text{infeasible}$ **then return** infeasible
- 7: **if** $\hat{x}^*$ fulfills h **then return** $\hat{x}^*$

Algorithm 1 SeidelLP($\mathcal{H}, d$)

- 1: **if** $d = 1$ **then** solve 1-dimensional problem and return;
- 2: **if** $\mathcal{H} = \emptyset$ **then** return x on implicit constraint hyperplane
- 3: choose **random** constraint $h \in \mathcal{H}$
- 4: $\hat{\mathcal{H}} \leftarrow \mathcal{H} \setminus \{h\}$
- 5: $\hat{x}^* \leftarrow \text{SeidelLP}(\hat{\mathcal{H}}, d)$
- 6: **if** $\hat{x}^* = \text{infeasible}$ **then return** infeasible
- 7: **if** $\hat{x}^*$ fulfills h **then return** $\hat{x}^*$
- 8: // **optimal solution fulfills h with equality, i.e., $a_h^T x = b_h$**

Algorithm 1 SeidelLP($\mathcal{H}, d$)

- 1: **if** $d = 1$ **then** solve 1-dimensional problem and return;
- 2: **if** $\mathcal{H} = \emptyset$ **then** return x on implicit constraint hyperplane
- 3: choose **random** constraint $h \in \mathcal{H}$
- 4: $\hat{\mathcal{H}} \leftarrow \mathcal{H} \setminus \{h\}$
- 5: $\hat{x}^* \leftarrow \text{SeidelLP}(\hat{\mathcal{H}}, d)$
- 6: **if** $\hat{x}^* = \text{infeasible}$ **then return** infeasible
- 7: **if** $\hat{x}^*$ fulfills h **then return** $\hat{x}^*$
- 8: // **optimal solution fulfills h with equality, i.e., $a_h^T x = b_h$**
- 9: solve $a_h^T x = b_h$ for some variable x_ℓ ;
- 10: eliminate x_ℓ in constraints from $\hat{\mathcal{H}}$ and in implicit constr.;

Algorithm 1 SeidelLP($\mathcal{H}, d$)

- 1: **if** $d = 1$ **then** solve 1-dimensional problem and return;
- 2: **if** $\mathcal{H} = \emptyset$ **then** return x on implicit constraint hyperplane
- 3: choose **random** constraint $h \in \mathcal{H}$
- 4: $\hat{\mathcal{H}} \leftarrow \mathcal{H} \setminus \{h\}$
- 5: $\hat{x}^* \leftarrow \text{SeidelLP}(\hat{\mathcal{H}}, d)$
- 6: **if** $\hat{x}^* = \text{infeasible}$ **then return** infeasible
- 7: **if** $\hat{x}^*$ fulfills h **then return** $\hat{x}^*$
- 8: // **optimal solution fulfills h with equality, i.e., $a_h^T x = b_h$**
- 9: solve $a_h^T x = b_h$ for some variable x_ℓ ;
- 10: eliminate x_ℓ in constraints from $\hat{\mathcal{H}}$ and in implicit constr.;
- 11: $\hat{x}^* \leftarrow \text{SeidelLP}(\hat{\mathcal{H}}, d - 1)$

Algorithm 1 SeidelLP($\mathcal{H}, d$)

- 1: **if** $d = 1$ **then** solve 1-dimensional problem and return;
- 2: **if** $\mathcal{H} = \emptyset$ **then** return x on implicit constraint hyperplane
- 3: choose **random** constraint $h \in \mathcal{H}$
- 4: $\hat{\mathcal{H}} \leftarrow \mathcal{H} \setminus \{h\}$
- 5: $\hat{x}^* \leftarrow \text{SeidelLP}(\hat{\mathcal{H}}, d)$
- 6: **if** $\hat{x}^* = \text{infeasible}$ **then return** infeasible
- 7: **if** $\hat{x}^*$ fulfills h **then return** $\hat{x}^*$
- 8: // **optimal solution fulfills h with equality, i.e., $a_h^T x = b_h$**
- 9: solve $a_h^T x = b_h$ for some variable x_ℓ ;
- 10: eliminate x_ℓ in constraints from $\hat{\mathcal{H}}$ and in implicit constr.;
- 11: $\hat{x}^* \leftarrow \text{SeidelLP}(\hat{\mathcal{H}}, d - 1)$
- 12: **if** $\hat{x}^* = \text{infeasible}$ **then**
- 13: **return** infeasible
- 14: **else**
- 15: add the value of x_ℓ to $\hat{x}^*$ and return the solution

8 Seidels LP-algorithm

- ▶ If $d = 1$ we can solve the 1-dimensional problem in time $\mathcal{O}(\max\{m, 1\})$.
- ▶ If $d > 1$ and $m = 0$ we take time $\mathcal{O}(d)$ to return d -dimensional vector x .
- ▶ The first recursive call takes time $T(m-1, d)$ for the call plus $\mathcal{O}(d)$ for checking whether the solution fulfills h .
- ▶ If we are unlucky and $\hat{x}^*$ does not fulfill h we need time $\mathcal{O}(d(m+1)) = \mathcal{O}(dm)$ to eliminate x_ℓ . Then we make a recursive call that takes time $T(m-1, d-1)$.
- ▶ The probability of being unlucky is at most d/m as there are at most d constraints whose removal will decrease the objective function

8 Seidels LP-algorithm

- ▶ If $d = 1$ we can solve the 1-dimensional problem in time $\mathcal{O}(\max\{m, 1\})$.
- ▶ If $d > 1$ and $m = 0$ we take time $\mathcal{O}(d)$ to return d -dimensional vector x .
- ▶ The first recursive call takes time $T(m-1, d)$ for the call plus $\mathcal{O}(d)$ for checking whether the solution fulfills h .
- ▶ If we are unlucky and $\hat{x}^*$ does not fulfill h we need time $\mathcal{O}(d(m+1)) = \mathcal{O}(dm)$ to eliminate x_ℓ . Then we make a recursive call that takes time $T(m-1, d-1)$.
- ▶ The probability of being unlucky is at most d/m as there are at most d constraints whose removal will decrease the objective function

8 Seidels LP-algorithm

- ▶ If $d = 1$ we can solve the 1-dimensional problem in time $\mathcal{O}(\max\{m, 1\})$.
- ▶ If $d > 1$ and $m = 0$ we take time $\mathcal{O}(d)$ to return d -dimensional vector x .
- ▶ The first recursive call takes time $T(m - 1, d)$ for the call plus $\mathcal{O}(d)$ for checking whether the solution fulfills h .
- ▶ If we are unlucky and $\hat{x}^*$ does not fulfill h we need time $\mathcal{O}(d(m + 1)) = \mathcal{O}(dm)$ to eliminate x_ℓ . Then we make a recursive call that takes time $T(m - 1, d - 1)$.
- ▶ The probability of being unlucky is at most d/m as there are at most d constraints whose removal will decrease the objective function

8 Seidels LP-algorithm

- ▶ If $d = 1$ we can solve the 1-dimensional problem in time $\mathcal{O}(\max\{m, 1\})$.
- ▶ If $d > 1$ and $m = 0$ we take time $\mathcal{O}(d)$ to return d -dimensional vector x .
- ▶ The first recursive call takes time $T(m - 1, d)$ for the call plus $\mathcal{O}(d)$ for checking whether the solution fulfills h .
- ▶ If we are unlucky and $\hat{x}^*$ does not fulfill h we need time $\mathcal{O}(d(m + 1)) = \mathcal{O}(dm)$ to eliminate x_ℓ . Then we make a recursive call that takes time $T(m - 1, d - 1)$.
- ▶ The probability of being unlucky is at most d/m as there are at most d constraints whose removal will decrease the objective function

8 Seidels LP-algorithm

- ▶ If $d = 1$ we can solve the 1-dimensional problem in time $\mathcal{O}(\max\{m, 1\})$.
- ▶ If $d > 1$ and $m = 0$ we take time $\mathcal{O}(d)$ to return d -dimensional vector x .
- ▶ The first recursive call takes time $T(m - 1, d)$ for the call plus $\mathcal{O}(d)$ for checking whether the solution fulfills h .
- ▶ If we are unlucky and $\hat{x}^*$ does not fulfill h we need time $\mathcal{O}(d(m + 1)) = \mathcal{O}(dm)$ to eliminate x_ℓ . Then we make a recursive call that takes time $T(m - 1, d - 1)$.
- ▶ The probability of being unlucky is at most d/m as there are at most d constraints whose removal will decrease the objective function

8 Seidels LP-algorithm

This gives the recurrence

$$T(m, d) = \begin{cases} \mathcal{O}(\max\{1, m\}) & \text{if } d = 1 \\ \mathcal{O}(d) & \text{if } d > 1 \text{ and } m = 0 \\ \mathcal{O}(d) + T(m-1, d) + \frac{d}{m}(\mathcal{O}(dm) + T(m-1, d-1)) & \text{otw.} \end{cases}$$

Note that $T(m, d)$ denotes the **expected running time**.

8 Seidels LP-algorithm

Let C be the largest constant in the $\mathcal{O}$ -notations.

$$T(m, d) = \begin{cases} C \max\{1, m\} & \text{if } d = 1 \\ Cd & \text{if } d > 1 \text{ and } m = 0 \\ Cd + T(m - 1, d) + \frac{d}{m}(Cdm + T(m - 1, d - 1)) & \text{otw.} \end{cases}$$

Note that $T(m, d)$ denotes the **expected running time**.

8 Seidels LP-algorithm

8 Seidels LP-algorithm

Let C be the largest constant in the $\mathcal{O}$ -notations.

8 Seidels LP-algorithm

Let C be the largest constant in the $\mathcal{O}$ -notations.

We show $T(m, d) \leq C f(d) \max\{1, m\}$.

8 Seidels LP-algorithm

Let C be the largest constant in the $\mathcal{O}$ -notations.

We show $T(m, d) \leq C f(d) \max\{1, m\}$.

$d = 1$:

8 Seidels LP-algorithm

Let C be the largest constant in the $\mathcal{O}$ -notations.

We show $T(m, d) \leq C f(d) \max\{1, m\}$.

$d = 1$:

$$T(m, 1)$$

8 Seidels LP-algorithm

Let C be the largest constant in the $\mathcal{O}$ -notations.

We show $T(m, d) \leq C f(d) \max\{1, m\}$.

$d = 1$:

$$T(m, 1) \leq C \max\{1, m\}$$

8 Seidels LP-algorithm

Let C be the largest constant in the $\mathcal{O}$ -notations.

We show $T(m, d) \leq C f(d) \max\{1, m\}$.

$d = 1$:

$$T(m, 1) \leq C \max\{1, m\} \leq C f(1) \max\{1, m\}$$

8 Seidels LP-algorithm

Let C be the largest constant in the $\mathcal{O}$ -notations.

We show $T(m, d) \leq C f(d) \max\{1, m\}$.

$d = 1$:

$$T(m, 1) \leq C \max\{1, m\} \leq C f(1) \max\{1, m\} \text{ for } f(1) \geq 1$$

8 Seidels LP-algorithm

Let C be the largest constant in the $\mathcal{O}$ -notations.

We show $T(m, d) \leq C f(d) \max\{1, m\}$.

$d = 1$:

$$T(m, 1) \leq C \max\{1, m\} \leq C f(1) \max\{1, m\} \text{ for } f(1) \geq 1$$

$d > 1; m = 0$:

$$T(0, d) \leq \mathcal{O}(d)$$

8 Seidels LP-algorithm

Let C be the largest constant in the $\mathcal{O}$ -notations.

We show $T(m, d) \leq C f(d) \max\{1, m\}$.

$d = 1$:

$$T(m, 1) \leq C \max\{1, m\} \leq C f(1) \max\{1, m\} \text{ for } f(1) \geq 1$$

$d > 1; m = 0$:

$$T(0, d) \leq \mathcal{O}(d) \leq Cd$$

8 Seidels LP-algorithm

Let C be the largest constant in the $\mathcal{O}$ -notations.

We show $T(m, d) \leq Cf(d) \max\{1, m\}$.

$d = 1$:

$$T(m, 1) \leq C \max\{1, m\} \leq Cf(1) \max\{1, m\} \text{ for } f(1) \geq 1$$

$d > 1; m = 0$:

$$T(0, d) \leq \mathcal{O}(d) \leq Cd \leq Cf(d) \max\{1, m\}$$

8 Seidels LP-algorithm

Let C be the largest constant in the $\mathcal{O}$ -notations.

We show $T(m, d) \leq Cf(d) \max\{1, m\}$.

$d = 1$:

$$T(m, 1) \leq C \max\{1, m\} \leq Cf(1) \max\{1, m\} \text{ for } f(1) \geq 1$$

$d > 1; m = 0$:

$$T(0, d) \leq \mathcal{O}(d) \leq Cd \leq Cf(d) \max\{1, m\} \text{ for } f(d) \geq d$$

8 Seidels LP-algorithm

Let C be the largest constant in the $\mathcal{O}$ -notations.

We show $T(m, d) \leq C f(d) \max\{1, m\}$.

$d = 1$:

$$T(m, 1) \leq C \max\{1, m\} \leq C f(1) \max\{1, m\} \text{ for } f(1) \geq 1$$

$d > 1; m = 0$:

$$T(0, d) \leq \mathcal{O}(d) \leq C d \leq C f(d) \max\{1, m\} \text{ for } f(d) \geq d$$

$d > 1; m = 1$:

$$T(1, d) = \mathcal{O}(d) + T(0, d) + d(\mathcal{O}(d) + T(0, d - 1))$$

8 Seidels LP-algorithm

Let C be the largest constant in the $\mathcal{O}$ -notations.

We show $T(m, d) \leq Cf(d) \max\{1, m\}$.

$d = 1$:

$$T(m, 1) \leq C \max\{1, m\} \leq Cf(1) \max\{1, m\} \text{ for } f(1) \geq 1$$

$d > 1; m = 0$:

$$T(0, d) \leq \mathcal{O}(d) \leq Cd \leq Cf(d) \max\{1, m\} \text{ for } f(d) \geq d$$

$d > 1; m = 1$:

$$\begin{aligned} T(1, d) &= \mathcal{O}(d) + T(0, d) + d(\mathcal{O}(d) + T(0, d-1)) \\ &\leq Cd + Cd + Cd^2 + dCf(d-1) \end{aligned}$$

8 Seidels LP-algorithm

Let C be the largest constant in the $\mathcal{O}$ -notations.

We show $T(m, d) \leq Cf(d) \max\{1, m\}$.

$d = 1$:

$$T(m, 1) \leq C \max\{1, m\} \leq Cf(1) \max\{1, m\} \text{ for } f(1) \geq 1$$

$d > 1; m = 0$:

$$T(0, d) \leq \mathcal{O}(d) \leq Cd \leq Cf(d) \max\{1, m\} \text{ for } f(d) \geq d$$

$d > 1; m = 1$:

$$\begin{aligned} T(1, d) &= \mathcal{O}(d) + T(0, d) + d(\mathcal{O}(d) + T(0, d-1)) \\ &\leq Cd + Cd + Cd^2 + dCf(d-1) \\ &\leq Cf(d) \max\{1, m\} \end{aligned}$$

8 Seidels LP-algorithm

Let C be the largest constant in the $\mathcal{O}$ -notations.

We show $T(m, d) \leq Cf(d) \max\{1, m\}$.

$d = 1$:

$$T(m, 1) \leq C \max\{1, m\} \leq Cf(1) \max\{1, m\} \text{ for } f(1) \geq 1$$

$d > 1; m = 0$:

$$T(0, d) \leq \mathcal{O}(d) \leq Cd \leq Cf(d) \max\{1, m\} \text{ for } f(d) \geq d$$

$d > 1; m = 1$:

$$\begin{aligned} T(1, d) &= \mathcal{O}(d) + T(0, d) + d(\mathcal{O}(d) + T(0, d-1)) \\ &\leq Cd + Cd + Cd^2 + dCf(d-1) \\ &\leq Cf(d) \max\{1, m\} \text{ for } f(d) \geq 3d^2 + df(d-1) \end{aligned}$$

8 Seidels LP-algorithm

$d > 1; m > 1 :$

(by induction hypothesis statm. true for $d' < d, m' \geq 0$;

and for $d' = d, m' < m$)

8 Seidels LP-algorithm

$d > 1; m > 1 :$

(by induction hypothesis statm. true for $d' < d, m' \geq 0$;

and for $d' = d, m' < m$)

$$T(m, d) = \mathcal{O}(d) + T(m - 1, d) + \frac{d}{m} \left(\mathcal{O}(dm) + T(m - 1, d - 1) \right)$$

8 Seidels LP-algorithm

$d > 1; m > 1 :$

(by induction hypothesis statm. true for $d' < d, m' \geq 0$;

and for $d' = d, m' < m$)

$$\begin{aligned} T(m, d) &= \mathcal{O}(d) + T(m-1, d) + \frac{d}{m} \left(\mathcal{O}(dm) + T(m-1, d-1) \right) \\ &\leq Cd + Cf(d)(m-1) + Cd^2 + \frac{d}{m} Cf(d-1)(m-1) \end{aligned}$$

8 Seidels LP-algorithm

$d > 1; m > 1 :$

(by induction hypothesis statm. true for $d' < d, m' \geq 0$;

and for $d' = d, m' < m$)

$$\begin{aligned} T(m, d) &= \mathcal{O}(d) + T(m-1, d) + \frac{d}{m} \left(\mathcal{O}(dm) + T(m-1, d-1) \right) \\ &\leq Cd + Cf(d)(m-1) + Cd^2 + \frac{d}{m} Cf(d-1)(m-1) \\ &\leq 2Cd^2 + Cf(d)(m-1) + dCf(d-1) \end{aligned}$$

8 Seidels LP-algorithm

$d > 1; m > 1 :$

(by induction hypothesis statm. true for $d' < d, m' \geq 0$;

and for $d' = d, m' < m$)

$$\begin{aligned}T(m, d) &= \mathcal{O}(d) + T(m-1, d) + \frac{d}{m} \left(\mathcal{O}(dm) + T(m-1, d-1) \right) \\&\leq Cd + Cf(d)(m-1) + Cd^2 + \frac{d}{m} Cf(d-1)(m-1) \\&\leq 2Cd^2 + Cf(d)(m-1) + dCf(d-1) \\&\leq Cf(d)m\end{aligned}$$

8 Seidels LP-algorithm

$d > 1; m > 1 :$

(by induction hypothesis statm. true for $d' < d, m' \geq 0$;

and for $d' = d, m' < m$)

$$\begin{aligned}T(m, d) &= \mathcal{O}(d) + T(m-1, d) + \frac{d}{m} \left(\mathcal{O}(dm) + T(m-1, d-1) \right) \\&\leq Cd + Cf(d)(m-1) + Cd^2 + \frac{d}{m} Cf(d-1)(m-1) \\&\leq 2Cd^2 + Cf(d)(m-1) + dCf(d-1) \\&\leq Cf(d)m\end{aligned}$$

if $f(d) \geq df(d-1) + 2d^2$.

8 Seidels LP-algorithm

- ▶ Define $f(1) = 3 \cdot 1^2$ and $f(d) = df(d-1) + 3d^2$ for $d > 1$.

8 Seidels LP-algorithm

- ▶ Define $f(1) = 3 \cdot 1^2$ and $f(d) = df(d-1) + 3d^2$ for $d > 1$.

Then

$$f(d)$$

8 Seidels LP-algorithm

- ▶ Define $f(1) = 3 \cdot 1^2$ and $f(d) = df(d-1) + 3d^2$ for $d > 1$.

Then

$$f(d) = 3d^2 + df(d-1)$$

8 Seidels LP-algorithm

- Define $f(1) = 3 \cdot 1^2$ and $f(d) = df(d-1) + 3d^2$ for $d > 1$.

Then

$$\begin{aligned} f(d) &= 3d^2 + df(d-1) \\ &= 3d^2 + d \left[3(d-1)^2 + (d-1)f(d-2) \right] \end{aligned}$$

8 Seidels LP-algorithm

- Define $f(1) = 3 \cdot 1^2$ and $f(d) = df(d-1) + 3d^2$ for $d > 1$.

Then

$$\begin{aligned}f(d) &= 3d^2 + df(d-1) \\&= 3d^2 + d \left[3(d-1)^2 + (d-1)f(d-2) \right] \\&= 3d^2 + d \left[3(d-1)^2 + (d-1) \left[3(d-2)^2 + (d-2)f(d-3) \right] \right]\end{aligned}$$

8 Seidels LP-algorithm

- Define $f(1) = 3 \cdot 1^2$ and $f(d) = df(d-1) + 3d^2$ for $d > 1$.

Then

$$\begin{aligned}f(d) &= 3d^2 + df(d-1) \\&= 3d^2 + d \left[3(d-1)^2 + (d-1)f(d-2) \right] \\&= 3d^2 + d \left[3(d-1)^2 + (d-1) \left[3(d-2)^2 + (d-2)f(d-3) \right] \right] \\&= 3d^2 + 3d(d-1)^2 + 3d(d-1)(d-2)^2 + \dots \\&\quad + 3d(d-1)(d-2) \cdot \dots \cdot 4 \cdot 3 \cdot 2 \cdot 1^2\end{aligned}$$

8 Seidels LP-algorithm

- Define $f(1) = 3 \cdot 1^2$ and $f(d) = df(d-1) + 3d^2$ for $d > 1$.

Then

$$\begin{aligned} f(d) &= 3d^2 + df(d-1) \\ &= 3d^2 + d \left[3(d-1)^2 + (d-1)f(d-2) \right] \\ &= 3d^2 + d \left[3(d-1)^2 + (d-1) \left[3(d-2)^2 + (d-2)f(d-3) \right] \right] \\ &= 3d^2 + 3d(d-1)^2 + 3d(d-1)(d-2)^2 + \dots \\ &\quad + 3d(d-1)(d-2) \cdot \dots \cdot 4 \cdot 3 \cdot 2 \cdot 1^2 \\ &= 3d! \left(\frac{d^2}{d!} + \frac{(d-1)^2}{(d-1)!} + \frac{(d-2)^2}{(d-2)!} + \dots \right) \end{aligned}$$

8 Seidels LP-algorithm

- Define $f(1) = 3 \cdot 1^2$ and $f(d) = df(d-1) + 3d^2$ for $d > 1$.

Then

$$\begin{aligned} f(d) &= 3d^2 + df(d-1) \\ &= 3d^2 + d \left[3(d-1)^2 + (d-1)f(d-2) \right] \\ &= 3d^2 + d \left[3(d-1)^2 + (d-1) \left[3(d-2)^2 + (d-2)f(d-3) \right] \right] \\ &= 3d^2 + 3d(d-1)^2 + 3d(d-1)(d-2)^2 + \dots \\ &\quad + 3d(d-1)(d-2) \cdot \dots \cdot 4 \cdot 3 \cdot 2 \cdot 1^2 \\ &= 3d! \left(\frac{d^2}{d!} + \frac{(d-1)^2}{(d-1)!} + \frac{(d-2)^2}{(d-2)!} + \dots \right) \\ &= \mathcal{O}(d!) \end{aligned}$$

8 Seidels LP-algorithm

- Define $f(1) = 3 \cdot 1^2$ and $f(d) = df(d-1) + 3d^2$ for $d > 1$.

Then

$$\begin{aligned} f(d) &= 3d^2 + df(d-1) \\ &= 3d^2 + d \left[3(d-1)^2 + (d-1)f(d-2) \right] \\ &= 3d^2 + d \left[3(d-1)^2 + (d-1) \left[3(d-2)^2 + (d-2)f(d-3) \right] \right] \\ &= 3d^2 + 3d(d-1)^2 + 3d(d-1)(d-2)^2 + \dots \\ &\quad + 3d(d-1)(d-2) \cdot \dots \cdot 4 \cdot 3 \cdot 2 \cdot 1^2 \\ &= 3d! \left(\frac{d^2}{d!} + \frac{(d-1)^2}{(d-1)!} + \frac{(d-2)^2}{(d-2)!} + \dots \right) \\ &= \mathcal{O}(d!) \end{aligned}$$

since $\sum_{i \geq 1} \frac{i^2}{i!}$ is a constant.

Complexity

LP Feasibility Problem (LP feasibility A)

Given $A \in \mathbb{Z}^{m \times n}$, $b \in \mathbb{Z}^m$. Does there exist $x \in \mathbb{R}^n$ with $Ax \leq b$, $x \geq 0$?

LP Feasibility Problem (LP feasibility B)

Given $A \in \mathbb{Z}^{m \times n}$, $b \in \mathbb{Z}^m$. Find $x \in \mathbb{R}^n$ with $Ax \leq b$, $x \geq 0$!

LP Optimization A

Given $A \in \mathbb{Z}^{m \times n}$, $b \in \mathbb{Z}^m$, $c \in \mathbb{Z}^n$. What is the maximum value of $c^T x$ for a feasible point $x \in \mathbb{R}^n$?

LP Optimization B

Given $A \in \mathbb{Z}^{m \times n}$, $b \in \mathbb{Z}^m$, $c \in \mathbb{Z}^n$. Return feasible point $x \in \mathbb{R}^n$ with maximum value of $c^T x$?

The Bit Model

Input size

- ▶ The number of bits to represent a number $a \in \mathbb{Z}$ is

$$\lceil \log_2(|a|) \rceil + 1$$

- ▶ Let for an $m \times n$ matrix M , $L(M)$ denote the number of bits required to encode all the numbers in M .

$$\langle M \rangle := \sum_{i,j} \lceil \log_2(|m_{ij}|) \rceil + 1$$

- ▶ In the following we assume that input matrices are encoded in a standard way, where each number is encoded in binary and then suitable separators are added in order to separate distinct number from each other.
- ▶ Then the input length is $L = \Theta(\langle A \rangle + \langle b \rangle)$.

The Bit Model

Input size

- ▶ The number of bits to represent a number $a \in \mathbb{Z}$ is

$$\lceil \log_2(|a|) \rceil + 1$$

- ▶ Let for an $m \times n$ matrix M , $L(M)$ denote the number of bits required to encode all the numbers in M .

$$\langle M \rangle := \sum_{i,j} \lceil \log_2(|m_{ij}|) \rceil + 1$$

- ▶ In the following we assume that input matrices are encoded in a standard way, where each number is encoded in binary and then suitable separators are added in order to separate distinct number from each other.
- ▶ Then the input length is $L = \Theta(\langle A \rangle + \langle b \rangle)$.

The Bit Model

Input size

- ▶ The number of bits to represent a number $a \in \mathbb{Z}$ is

$$\lceil \log_2(|a|) \rceil + 1$$

- ▶ Let for an $m \times n$ matrix M , $L(M)$ denote the number of bits required to encode all the numbers in M .

$$\langle M \rangle := \sum_{i,j} \lceil \log_2(|m_{ij}|) \rceil + 1$$

- ▶ In the following we assume that input matrices are encoded in a standard way, where each number is encoded in binary and then suitable separators are added in order to separate distinct number from each other.
- ▶ Then the input length is $L = \Theta(\langle A \rangle + \langle b \rangle)$.

The Bit Model

Input size

- ▶ The number of bits to represent a number $a \in \mathbb{Z}$ is

$$\lceil \log_2(|a|) \rceil + 1$$

- ▶ Let for an $m \times n$ matrix M , $L(M)$ denote the number of bits required to encode all the numbers in M .

$$\langle M \rangle := \sum_{i,j} \lceil \log_2(|m_{ij}|) \rceil + 1$$

- ▶ In the following we assume that input matrices are encoded in a standard way, where each number is encoded in binary and then suitable separators are added in order to separate distinct number from each other.
- ▶ Then the input length is $L = \Theta(\langle A \rangle + \langle b \rangle)$.

- ▶ In the following we sometimes refer to $L := \langle A \rangle + \langle b \rangle$ as the input size (even though the real input size is something in $\Theta(\langle A \rangle + \langle b \rangle)$).
- ▶ Sometimes we may also refer to $L := \langle A \rangle + \langle b \rangle + n \log_2 n$ as the input size. Note that $n \log_2 n = \Theta(\langle A \rangle + \langle b \rangle)$.
- ▶ In order to show that LP-decision is in NP we show that if there is a solution x then there exists a small solution for which feasibility can be verified in polynomial time (polynomial in L).

Suppose that $\bar{A}x = b; x \geq 0$ is feasible.

Then there exists a basic feasible solution. This means a set B of basic variables such that

$$x_B = \bar{A}_B^{-1}b$$

and all other entries in x are 0.

Suppose that $\bar{A}x = b$; $x \geq 0$ is feasible.

Then there exists a basic feasible solution. This means a set B of basic variables such that

$$x_B = \bar{A}_B^{-1}b$$

and all other entries in x are 0.

Size of a Basic Feasible Solution

- ▶ A : original input matrix
- ▶ $\bar{A}$: transformation of A into standard form
- ▶ $\bar{A}_B$: submatrix of $\bar{A}$ corresponding to basis B

Lemma 47

Let $\bar{A}_B \in \mathbb{Z}^{m \times m}$ and $b \in \mathbb{Z}^m$. Define $L = \langle A \rangle + \langle b \rangle + n \log_2 n$. Then a solution to $\bar{A}_B x_B = b$ has rational components x_j of the form $\frac{D_j}{D}$, where $|D_j| \leq 2^L$ and $|D| \leq 2^L$.

Proof:

Cramers rules says that we can compute x_j as

$$x_j = \frac{\det(\bar{A}_B^j)}{\det(\bar{A}_B)}$$

where $\bar{A}_B^j$ is the matrix obtained from $\bar{A}_B$ by replacing the j -th column by the vector b .

Size of a Basic Feasible Solution

- ▶ A : original input matrix
- ▶ $\bar{A}$: transformation of A into standard form
- ▶ $\bar{A}_B$: submatrix of $\bar{A}$ corresponding to basis B

Lemma 47

Let $\bar{A}_B \in \mathbb{Z}^{m \times m}$ and $b \in \mathbb{Z}^m$. Define $L = \langle A \rangle + \langle b \rangle + n \log_2 n$. Then a solution to $\bar{A}_B x_B = b$ has rational components x_j of the form $\frac{D_j}{D}$, where $|D_j| \leq 2^L$ and $|D| \leq 2^L$.

Proof:

Cramers rules says that we can compute x_j as

$$x_j = \frac{\det(\bar{A}_B^j)}{\det(\bar{A}_B)}$$

where $\bar{A}_B^j$ is the matrix obtained from $\bar{A}_B$ by replacing the j -th column by the vector b .

Bounding the Determinant

Let $X = \bar{A}_B$. Then

$$|\det(X)|$$

Bounding the Determinant

Let $X = \tilde{A}_B$. Then

$$|\det(X)| = |\det(\tilde{X})|$$

Bounding the Determinant

Let $X = \bar{A}_B$. Then

$$\begin{aligned} |\det(X)| &= |\det(\tilde{X})| \\ &= \left| \sum_{\pi \in S_{\tilde{n}}} \text{sgn}(\pi) \prod_{1 \leq i \leq \tilde{n}} \tilde{X}_{i\pi(i)} \right| \end{aligned}$$

Bounding the Determinant

Let $X = \bar{A}_B$. Then

$$\begin{aligned} |\det(X)| &= |\det(\bar{X})| \\ &= \left| \sum_{\pi \in S_{\tilde{n}}} \operatorname{sgn}(\pi) \prod_{1 \leq i \leq \tilde{n}} \bar{X}_{i\pi(i)} \right| \\ &\leq \sum_{\pi \in S_{\tilde{n}}} \prod_{1 \leq i \leq \tilde{n}} |\bar{X}_{i\pi(i)}| \end{aligned}$$

Bounding the Determinant

Let $X = \bar{A}_B$. Then

$$\begin{aligned} |\det(X)| &= |\det(\tilde{X})| \\ &= \left| \sum_{\pi \in S_{\tilde{n}}} \text{sgn}(\pi) \prod_{1 \leq i \leq \tilde{n}} \tilde{X}_{i\pi(i)} \right| \\ &\leq \sum_{\pi \in S_{\tilde{n}}} \prod_{1 \leq i \leq \tilde{n}} |\tilde{X}_{i\pi(i)}| \\ &\leq n! \cdot 2^{\langle A \rangle + \langle b \rangle} \end{aligned}$$

Bounding the Determinant

Let $X = \bar{A}_B$. Then

$$\begin{aligned} |\det(X)| &= |\det(\tilde{X})| \\ &= \left| \sum_{\pi \in S_{\tilde{n}}} \text{sgn}(\pi) \prod_{1 \leq i \leq \tilde{n}} \tilde{X}_{i\pi(i)} \right| \\ &\leq \sum_{\pi \in S_{\tilde{n}}} \prod_{1 \leq i \leq \tilde{n}} |\tilde{X}_{i\pi(i)}| \\ &\leq n! \cdot 2^{\langle A \rangle + \langle b \rangle} \leq 2^L . \end{aligned}$$

Bounding the Determinant

Let $X = \bar{A}_B$. Then

$$\begin{aligned} |\det(X)| &= |\det(\tilde{X})| \\ &= \left| \sum_{\pi \in S_{\tilde{n}}} \text{sgn}(\pi) \prod_{1 \leq i \leq \tilde{n}} \tilde{X}_{i\pi(i)} \right| \\ &\leq \sum_{\pi \in S_{\tilde{n}}} \prod_{1 \leq i \leq \tilde{n}} |\tilde{X}_{i\pi(i)}| \\ &\leq n! \cdot 2^{\langle A \rangle + \langle b \rangle} \leq 2^L. \end{aligned}$$

Here $\tilde{X}$ is an $\tilde{n} \times \tilde{n}$ submatrix of A with $\tilde{n} \leq n$.

Bounding the Determinant

Let $X = \bar{A}_B$. Then

$$\begin{aligned} |\det(X)| &= |\det(\bar{X})| \\ &= \left| \sum_{\pi \in S_{\tilde{n}}} \text{sgn}(\pi) \prod_{1 \leq i \leq \tilde{n}} \bar{X}_{i\pi(i)} \right| \\ &\leq \sum_{\pi \in S_{\tilde{n}}} \prod_{1 \leq i \leq \tilde{n}} |\bar{X}_{i\pi(i)}| \\ &\leq n! \cdot 2^{\langle A \rangle + \langle b \rangle} \leq 2^L. \end{aligned}$$

Here $\bar{X}$ is an $\tilde{n} \times \tilde{n}$ submatrix of A with $\tilde{n} \leq n$.

Analogously for $\det(A_B^j)$.

Reducing LP-solving to LP decision.

Given an LP $\max\{c^T x \mid Ax \leq b; x \geq 0\}$ do a **binary search** for the optimum solution

(Add constraint $c^T x \geq M$). Then checking for feasibility shows whether optimum solution is larger or smaller than M).

If the LP is feasible then the binary search finishes in at most

$$\log_2 \left(\frac{2n2^{2L'}}{1/2^{L'}} \right) = \mathcal{O}(L') ,$$

as the range of the search is at most $-n2^{2L'}, \dots, n2^{2L'}$ and the distance between two adjacent values is at least $\frac{1}{\det(A)} \geq \frac{1}{2^{L'}}$.

Here we use $L' = \langle A \rangle + \langle b \rangle + \langle c \rangle + n \log_2 n$ (it also includes the encoding size of c).

Reducing LP-solving to LP decision.

Given an LP $\max\{c^T x \mid Ax \leq b; x \geq 0\}$ do a **binary search** for the optimum solution

(Add constraint $c^T x \geq M$). Then checking for feasibility shows whether optimum solution is larger or smaller than M).

If the LP is feasible then the binary search finishes in at most

$$\log_2 \left(\frac{2n2^{2L'}}{1/2^{L'}} \right) = \mathcal{O}(L') ,$$

as the range of the search is at most $-n2^{2L'}, \dots, n2^{2L'}$ and the distance between two adjacent values is at least $\frac{1}{\det(A)} \geq \frac{1}{2^{L'}}$.

Here we use $L' = \langle A \rangle + \langle b \rangle + \langle c \rangle + n \log_2 n$ (it also includes the encoding size of c).

Reducing LP-solving to LP decision.

Given an LP $\max\{c^T x \mid Ax \leq b; x \geq 0\}$ do a **binary search** for the optimum solution

(Add constraint $c^T x \geq M$). Then checking for feasibility shows whether optimum solution is larger or smaller than M).

If the LP is feasible then the binary search finishes in at most

$$\log_2 \left(\frac{2n2^{2L'}}{1/2^{L'}} \right) = \mathcal{O}(L') ,$$

as the range of the search is at most $-n2^{2L'}, \dots, n2^{2L'}$ and the distance between two adjacent values is at least $\frac{1}{\det(A)} \geq \frac{1}{2^{L'}}$.

Here we use $L' = \langle A \rangle + \langle b \rangle + \langle c \rangle + n \log_2 n$ (it also includes the encoding size of c).

Reducing LP-solving to LP decision.

Given an LP $\max\{c^T x \mid Ax \leq b; x \geq 0\}$ do a **binary search** for the optimum solution

(Add constraint $c^T x \geq M$). Then checking for feasibility shows whether optimum solution is larger or smaller than M).

If the LP is feasible then the binary search finishes in at most

$$\log_2 \left(\frac{2n2^{2L'}}{1/2^{L'}} \right) = \mathcal{O}(L') ,$$

as the range of the search is at most $-n2^{2L'}, \dots, n2^{2L'}$ and the distance between two adjacent values is at least $\frac{1}{\det(A)} \geq \frac{1}{2^{L'}}$.

Here we use $L' = \langle A \rangle + \langle b \rangle + \langle c \rangle + n \log_2 n$ (it also includes the encoding size of c).

Reducing LP-solving to LP decision.

Given an LP $\max\{c^T x \mid Ax \leq b; x \geq 0\}$ do a **binary search** for the optimum solution

(Add constraint $c^T x \geq M$). Then checking for feasibility shows whether optimum solution is larger or smaller than M).

If the LP is feasible then the binary search finishes in at most

$$\log_2 \left(\frac{2n2^{2L'}}{1/2^{L'}} \right) = \mathcal{O}(L') ,$$

as the range of the search is at most $-n2^{2L'}, \dots, n2^{2L'}$ and the distance between two adjacent values is at least $\frac{1}{\det(A)} \geq \frac{1}{2^{L'}}$.

Here we use $L' = \langle A \rangle + \langle b \rangle + \langle c \rangle + n \log_2 n$ (it also includes the encoding size of c).

How do we detect whether the LP is unbounded?

Let $M_{\max} = n2^{2L'}$ be an upper bound on the objective value of a basic feasible solution.

We can add a constraint $c^T x \geq M_{\max} + 1$ and check for feasibility.

How do we detect whether the LP is unbounded?

Let $M_{\max} = n2^{2L'}$ be an upper bound on the objective value of a **basic feasible solution**.

We can add a constraint $c^T x \geq M_{\max} + 1$ and check for feasibility.

How do we detect whether the LP is unbounded?

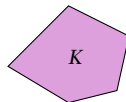
Let $M_{\max} = n2^{2L'}$ be an upper bound on the objective value of a **basic feasible solution**.

We can add a constraint $c^T x \geq M_{\max} + 1$ and check for feasibility.

Ellipsoid Method

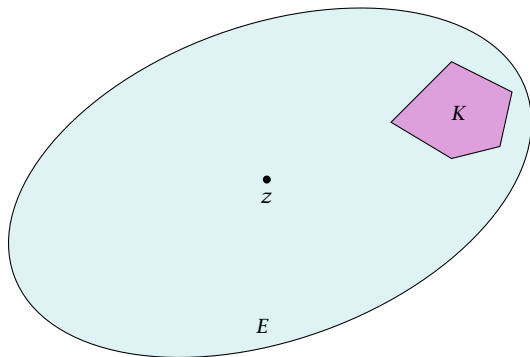
Ellipsoid Method

- ▶ Let K be a convex set.



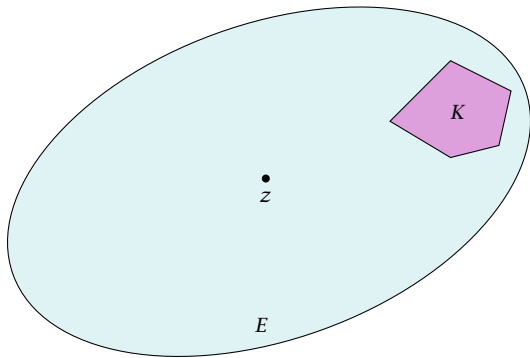
Ellipsoid Method

- ▶ Let K be a convex set.
- ▶ Maintain ellipsoid E that is guaranteed to contain K provided that K is non-empty.



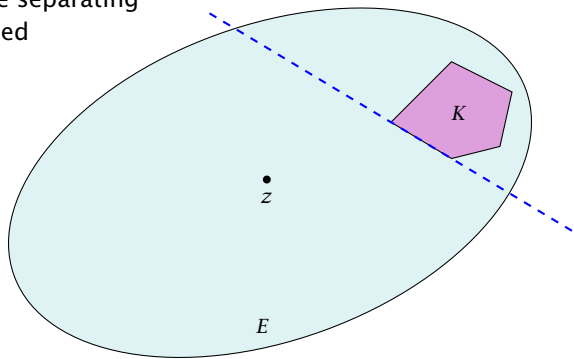
Ellipsoid Method

- ▶ Let K be a convex set.
- ▶ Maintain ellipsoid E that is guaranteed to contain K provided that K is non-empty.
- ▶ If center $z \in K$ STOP.



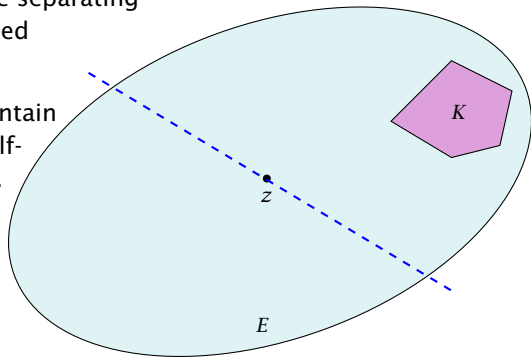
Ellipsoid Method

- ▶ Let K be a convex set.
- ▶ Maintain ellipsoid E that is guaranteed to contain K provided that K is non-empty.
- ▶ If center $z \in K$ STOP.
- ▶ Otw. find a hyperplane separating K from z (e.g. a violated constraint in the LP).



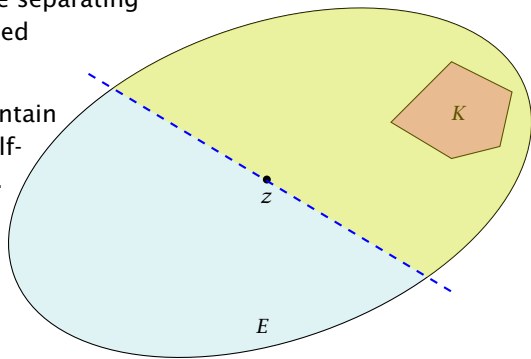
Ellipsoid Method

- ▶ Let K be a convex set.
- ▶ Maintain ellipsoid E that is guaranteed to contain K provided that K is non-empty.
- ▶ If center $z \in K$ STOP.
- ▶ Otw. find a hyperplane separating K from z (e.g. a violated constraint in the LP).
- ▶ Shift hyperplane to contain node z . H denotes half-space that contains K .



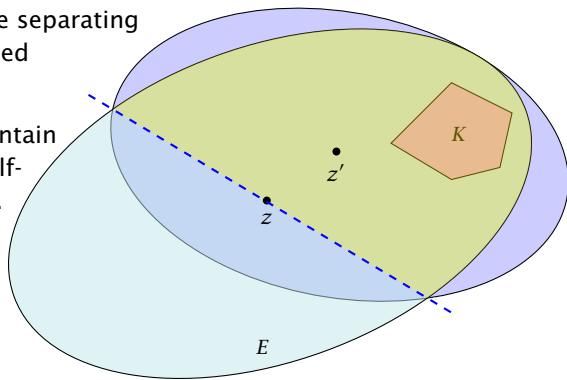
Ellipsoid Method

- ▶ Let K be a convex set.
- ▶ Maintain ellipsoid E that is guaranteed to contain K provided that K is non-empty.
- ▶ If center $z \in K$ STOP.
- ▶ Otw. find a hyperplane separating K from z (e.g. a violated constraint in the LP).
- ▶ Shift hyperplane to contain node z . H denotes half-space that contains K .



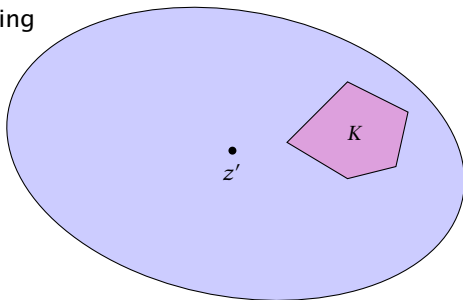
Ellipsoid Method

- ▶ Let K be a convex set.
- ▶ Maintain ellipsoid E that is guaranteed to contain K provided that K is non-empty.
- ▶ If center $z \in K$ STOP.
- ▶ Otw. find a hyperplane separating K from z (e.g. a violated constraint in the LP).
- ▶ Shift hyperplane to contain node z . H denotes half-space that contains K .
- ▶ Compute (smallest) ellipsoid E' that contains $E \cap H$.



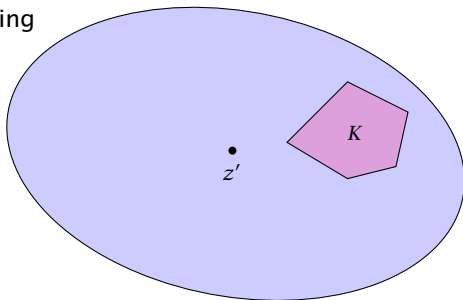
Ellipsoid Method

- ▶ Let K be a convex set.
- ▶ Maintain ellipsoid E that is guaranteed to contain K provided that K is non-empty.
- ▶ If center $z \in K$ STOP.
- ▶ Otw. find a hyperplane separating K from z (e.g. a violated constraint in the LP).
- ▶ Shift hyperplane to contain node z . H denotes half-space that contains K .
- ▶ Compute (smallest) ellipsoid E' that contains $E \cap H$.



Ellipsoid Method

- ▶ Let K be a convex set.
- ▶ Maintain ellipsoid E that is guaranteed to contain K provided that K is non-empty.
- ▶ If center $z \in K$ STOP.
- ▶ Otw. find a hyperplane separating K from z (e.g. a violated constraint in the LP).
- ▶ Shift hyperplane to contain node z . H denotes half-space that contains K .
- ▶ Compute (smallest) ellipsoid E' that contains $E \cap H$.
- ▶ REPEAT



Issues/Questions:

- ▶ How do you choose the first Ellipsoid? What is its volume?
- ▶ How do you measure progress? By how much does the volume decrease in each iteration?
- ▶ When can you stop? What is the minimum volume of a non-empty polytop?

Definition 48

A mapping $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$ with $f(x) = Lx + t$, where L is an invertible matrix is called an **affine transformation**.

Definition 49

A ball in $\mathbb{R}^n$ with center c and radius r is given by

$$\begin{aligned} B(c, r) &= \{x \mid (x - c)^T (x - c) \leq r^2\} \\ &= \{x \mid \sum_i (x - c)_i^2 / r^2 \leq 1\} \end{aligned}$$

$B(0, 1)$ is called the **unit ball**.

Definition 50

An affine transformation of the unit ball is called an **ellipsoid**.

Definition 50

An affine transformation of the unit ball is called an **ellipsoid**.

From $f(x) = Lx + t$ follows $x = L^{-1}(f(x) - t)$.

Definition 50

An affine transformation of the unit ball is called an **ellipsoid**.

From $f(x) = Lx + t$ follows $x = L^{-1}(f(x) - t)$.

$$f(B(0, 1))$$

Definition 50

An affine transformation of the unit ball is called an **ellipsoid**.

From $f(x) = Lx + t$ follows $x = L^{-1}(f(x) - t)$.

$$f(B(0, 1)) = \{f(x) \mid x \in B(0, 1)\}$$

Definition 50

An affine transformation of the unit ball is called an **ellipsoid**.

From $f(x) = Lx + t$ follows $x = L^{-1}(f(x) - t)$.

$$\begin{aligned} f(B(0, 1)) &= \{f(x) \mid x \in B(0, 1)\} \\ &= \{y \in \mathbb{R}^n \mid L^{-1}(y - t) \in B(0, 1)\} \end{aligned}$$

Definition 50

An affine transformation of the unit ball is called an **ellipsoid**.

From $f(x) = Lx + t$ follows $x = L^{-1}(f(x) - t)$.

$$\begin{aligned} f(B(0,1)) &= \{f(x) \mid x \in B(0,1)\} \\ &= \{y \in \mathbb{R}^n \mid L^{-1}(y - t) \in B(0,1)\} \\ &= \{y \in \mathbb{R}^n \mid (y - t)^T L^{-1^T} L^{-1} (y - t) \leq 1\} \end{aligned}$$

Definition 50

An affine transformation of the unit ball is called an **ellipsoid**.

From $f(x) = Lx + t$ follows $x = L^{-1}(f(x) - t)$.

$$\begin{aligned} f(B(0,1)) &= \{f(x) \mid x \in B(0,1)\} \\ &= \{y \in \mathbb{R}^n \mid L^{-1}(y - t) \in B(0,1)\} \\ &= \{y \in \mathbb{R}^n \mid (y - t)^T L^{-1^T} L^{-1} (y - t) \leq 1\} \\ &= \{y \in \mathbb{R}^n \mid (y - t)^T Q^{-1} (y - t) \leq 1\} \end{aligned}$$

Definition 50

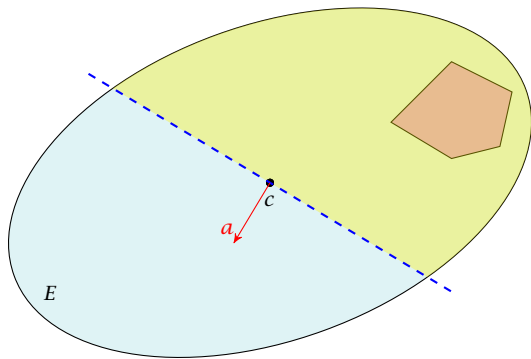
An affine transformation of the unit ball is called an **ellipsoid**.

From $f(x) = Lx + t$ follows $x = L^{-1}(f(x) - t)$.

$$\begin{aligned} f(B(0, 1)) &= \{f(x) \mid x \in B(0, 1)\} \\ &= \{y \in \mathbb{R}^n \mid L^{-1}(y - t) \in B(0, 1)\} \\ &= \{y \in \mathbb{R}^n \mid (y - t)^T L^{-1^T} L^{-1} (y - t) \leq 1\} \\ &= \{y \in \mathbb{R}^n \mid (y - t)^T Q^{-1} (y - t) \leq 1\} \end{aligned}$$

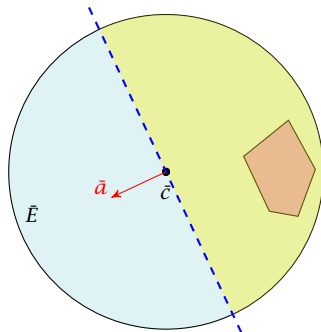
where $Q = LL^T$ is an invertible matrix.

How to Compute the New Ellipsoid



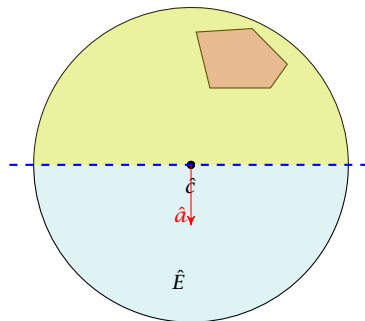
How to Compute the New Ellipsoid

- Use f^{-1} (recall that $f = Lx + t$ is the affine transformation of the unit ball) to rotate/distort the ellipsoid (back) into the unit ball.



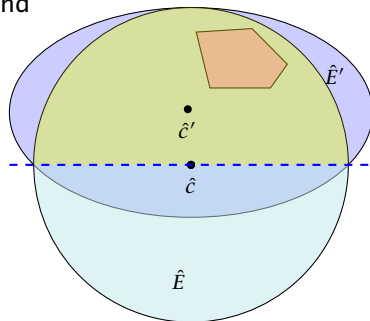
How to Compute the New Ellipsoid

- ▶ Use f^{-1} (recall that $f = Lx + t$ is the affine transformation of the unit ball) to rotate/distort the ellipsoid (back) into the unit ball.
- ▶ Use a rotation R^{-1} to rotate the unit ball such that the normal vector of the halfspace is parallel to e_1 .



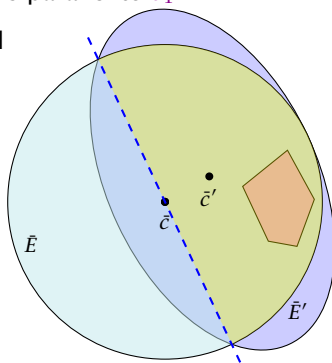
How to Compute the New Ellipsoid

- ▶ Use f^{-1} (recall that $f = Lx + t$ is the affine transformation of the unit ball) to rotate/distort the ellipsoid (back) into the unit ball.
- ▶ Use a rotation R^{-1} to rotate the unit ball such that the normal vector of the halfspace is parallel to e_1 .
- ▶ Compute the new center $\hat{c}'$ and the new matrix $\hat{Q}'$ for this simplified setting.



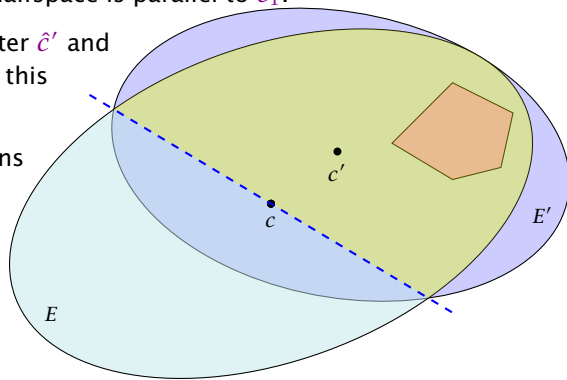
How to Compute the New Ellipsoid

- ▶ Use f^{-1} (recall that $f = Lx + t$ is the affine transformation of the unit ball) to rotate/distort the ellipsoid (back) into the unit ball.
- ▶ Use a rotation R^{-1} to rotate the unit ball such that the normal vector of the halfspace is parallel to e_1 .
- ▶ Compute the new center $\tilde{c}'$ and the new matrix $\tilde{Q}'$ for this simplified setting.
- ▶ Use the transformations R and f to get the new center c' and the new matrix Q' for the original ellipsoid E .

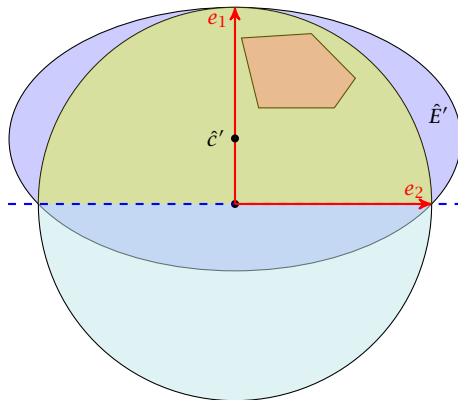


How to Compute the New Ellipsoid

- ▶ Use f^{-1} (recall that $f = Lx + t$ is the affine transformation of the unit ball) to rotate/distort the ellipsoid (back) into the unit ball.
- ▶ Use a rotation R^{-1} to rotate the unit ball such that the normal vector of the halfspace is parallel to e_1 .
- ▶ Compute the new center $\hat{c}'$ and the new matrix $\hat{Q}'$ for this simplified setting.
- ▶ Use the transformations R and f to get the new center c' and the new matrix Q' for the original ellipsoid E .

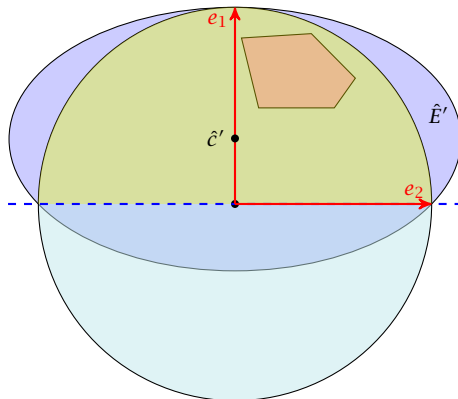


The Easy Case



- ▶ The new center lies on axis x_1 . Hence, $\hat{c}' = te_1$ for $t > 0$.
- ▶ The vectors $e_1, e_2, \dots$ have to fulfill the ellipsoid constraint with equality. Hence $(e_i - \hat{c}')^T \hat{Q}'^{-1} (e_i - \hat{c}') = 1$.

The Easy Case



- ▶ The new center lies on axis x_1 . Hence, $\hat{c}' = te_1$ for $t > 0$.
- ▶ The vectors $e_1, e_2, \dots$ have to fulfill the ellipsoid constraint with equality. Hence $(e_i - \hat{c}')^T \hat{Q}'^{-1} (e_i - \hat{c}') = 1$.

The Easy Case

- ▶ To obtain the matrix $\hat{Q}'^{-1}$ for our ellipsoid $\hat{E}'$ note that $\hat{E}'$ is **axis-parallel**.
- ▶ Let a denote the radius along the x_1 -axis and let b denote the (common) radius for the other axes.
- ▶ The matrix

$$\hat{L}' = \begin{pmatrix} a & 0 & \dots & 0 \\ 0 & b & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & b \end{pmatrix}$$

maps the unit ball (via function $\hat{f}'(x) = \hat{L}'x$) to an axis-parallel ellipsoid with radius a in direction x_1 and b in all other directions.

The Easy Case

- ▶ To obtain the matrix $\hat{Q}'^{-1}$ for our ellipsoid $\hat{E}'$ note that $\hat{E}'$ is **axis-parallel**.
- ▶ Let a denote the radius along the x_1 -axis and let b denote the (common) radius for the other axes.
- ▶ The matrix

$$\hat{L}' = \begin{pmatrix} a & 0 & \dots & 0 \\ 0 & b & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & b \end{pmatrix}$$

maps the unit ball (via function $\hat{f}'(x) = \hat{L}'x$) to an axis-parallel ellipsoid with radius a in direction x_1 and b in all other directions.

The Easy Case

- ▶ To obtain the matrix $\hat{Q}'^{-1}$ for our ellipsoid $\hat{E}'$ note that $\hat{E}'$ is **axis-parallel**.
- ▶ Let a denote the radius along the x_1 -axis and let b denote the (common) radius for the other axes.
- ▶ The matrix

$$\hat{L}' = \begin{pmatrix} a & 0 & \dots & 0 \\ 0 & b & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & b \end{pmatrix}$$

maps the unit ball (via function $\hat{f}'(x) = \hat{L}'x$) to an axis-parallel ellipsoid with radius a in direction x_1 and b in all other directions.

The Easy Case

- As $\hat{Q}' = \hat{L}'\hat{L}'^t$ the matrix $\hat{Q}'^{-1}$ is of the form

$$\hat{Q}'^{-1} = \begin{pmatrix} \frac{1}{a^2} & 0 & \dots & 0 \\ 0 & \frac{1}{b^2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \frac{1}{b^2} \end{pmatrix}$$

The Easy Case

- ▶ $(e_1 - \hat{c}')^T \hat{Q}'^{-1} (e_1 - \hat{c}') = 1$ gives

$$\begin{pmatrix} 1-t \\ 0 \\ \vdots \\ 0 \end{pmatrix}^T \cdot \begin{pmatrix} \frac{1}{a^2} & 0 & \dots & 0 \\ 0 & \frac{1}{b^2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \frac{1}{b^2} \end{pmatrix} \cdot \begin{pmatrix} 1-t \\ 0 \\ \vdots \\ 0 \end{pmatrix} = 1$$

- ▶ This gives $(1-t)^2 = a^2$.

The Easy Case

- ▶ For $i \neq 1$ the equation $(e_i - \hat{c}')^T \hat{Q}'^{-1} (e_i - \hat{c}') = 1$ looks like (here $i = 2$)

$$\begin{pmatrix} -t \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}^T \cdot \begin{pmatrix} \frac{1}{a^2} & 0 & \dots & 0 \\ 0 & \frac{1}{b^2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \frac{1}{b^2} \end{pmatrix} \cdot \begin{pmatrix} -t \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = 1$$

- ▶ This gives $\frac{t^2}{a^2} + \frac{1}{b^2} = 1$, and hence

$$\frac{1}{b^2} = 1 - \frac{t^2}{a^2}$$

The Easy Case

- ▶ For $i \neq 1$ the equation $(e_i - \hat{c}')^T \hat{Q}'^{-1} (e_i - \hat{c}') = 1$ looks like (here $i = 2$)

$$\begin{pmatrix} -t \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}^T \cdot \begin{pmatrix} \frac{1}{a^2} & 0 & \dots & 0 \\ 0 & \frac{1}{b^2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \frac{1}{b^2} \end{pmatrix} \cdot \begin{pmatrix} -t \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = 1$$

- ▶ This gives $\frac{t^2}{a^2} + \frac{1}{b^2} = 1$, and hence

$$\frac{1}{b^2} = 1 - \frac{t^2}{a^2} = 1 - \frac{t^2}{(1-t)^2}$$

The Easy Case

- ▶ For $i \neq 1$ the equation $(e_i - \hat{c}')^T \hat{Q}'^{-1} (e_i - \hat{c}') = 1$ looks like (here $i = 2$)

$$\begin{pmatrix} -t \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}^T \cdot \begin{pmatrix} \frac{1}{a^2} & 0 & \dots & 0 \\ 0 & \frac{1}{b^2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & \frac{1}{b^2} \end{pmatrix} \cdot \begin{pmatrix} -t \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = 1$$

- ▶ This gives $\frac{t^2}{a^2} + \frac{1}{b^2} = 1$, and hence

$$\frac{1}{b^2} = 1 - \frac{t^2}{a^2} = 1 - \frac{t^2}{(1-t)^2} = \frac{1-2t}{(1-t)^2}$$

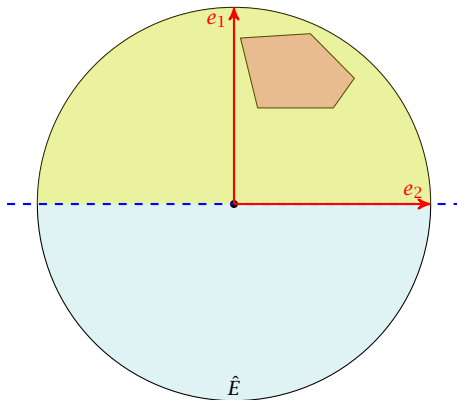
Summary

So far we have

$$a = 1 - t \quad \text{and} \quad b = \frac{1 - t}{\sqrt{1 - 2t}}$$

The Easy Case

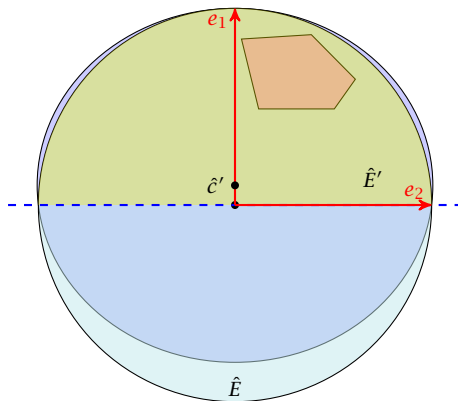
We still have many choices for t :



Choose t such that the volume of $\hat{E}'$ is minimal!!!

The Easy Case

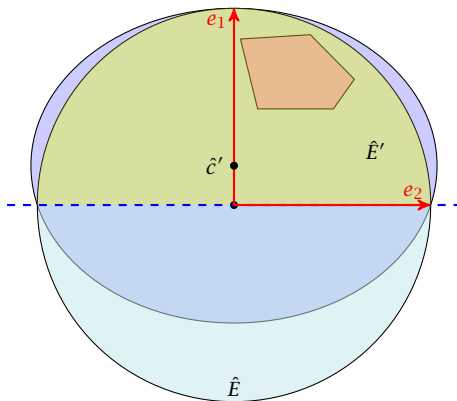
We still have many choices for t :



Choose t such that the volume of $\hat{E}'$ is minimal!!!

The Easy Case

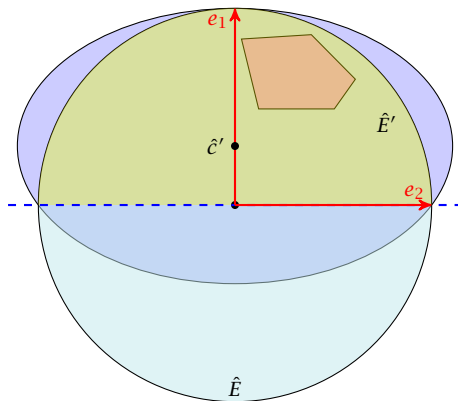
We still have many choices for t :



Choose t such that the volume of $\hat{E}'$ is minimal!!!

The Easy Case

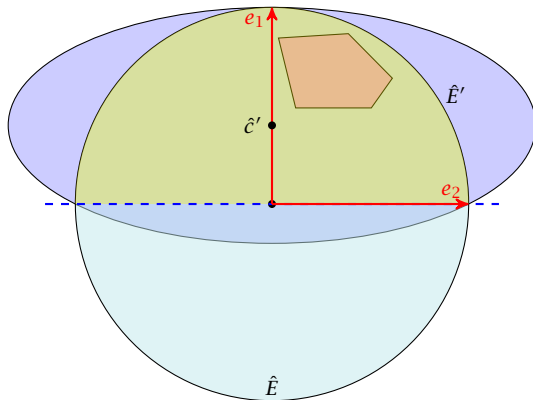
We still have many choices for t :



Choose t such that the volume of $\hat{E}'$ is minimal!!!

The Easy Case

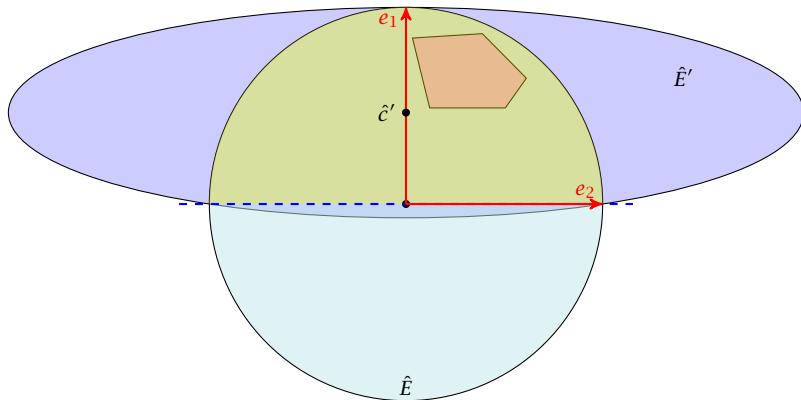
We still have many choices for t :



Choose t such that the volume of $\hat{E}'$ is minimal!!!

The Easy Case

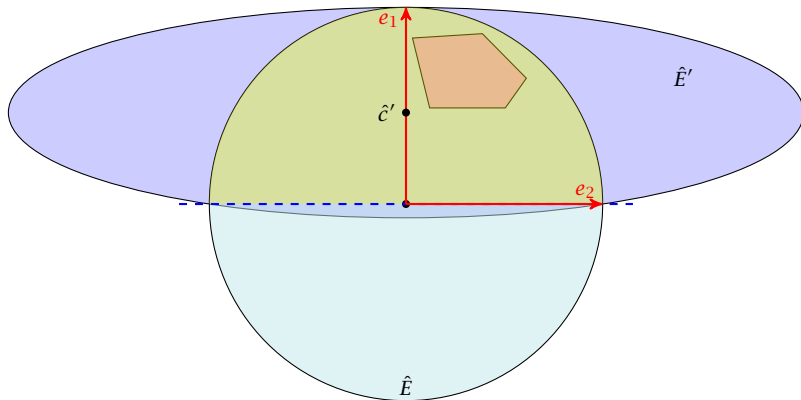
We still have many choices for t :



Choose t such that the volume of $\hat{E}'$ is minimal!!!

The Easy Case

We still have many choices for t :



Choose t such that the volume of $\hat{E}'$ is minimal!!!

The Easy Case

We want to choose t such that the volume of $\hat{E}'$ is minimal.

Lemma 51

Let L be an affine transformation and $K \subseteq \mathbb{R}^n$. Then

$$\text{vol}(L(K)) = |\det(L)| \cdot \text{vol}(K) .$$

The Easy Case

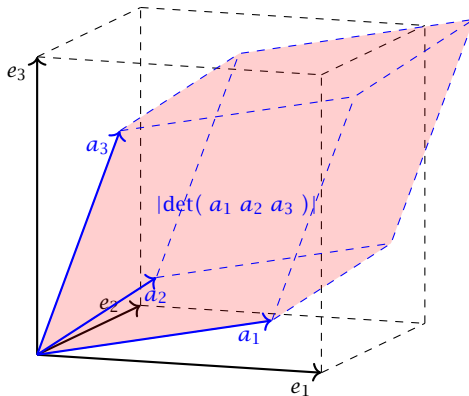
We want to choose t such that the volume of $\hat{E}'$ is minimal.

Lemma 51

Let L be an affine transformation and $K \subseteq \mathbb{R}^n$. Then

$$\text{vol}(L(K)) = |\det(L)| \cdot \text{vol}(K) .$$

n-dimensional volume



The Easy Case

- ▶ We want to choose t such that the volume of $\hat{E}'$ is minimal.

$$\text{vol}(\hat{E}') = \text{vol}(B(0,1)) \cdot |\det(\hat{L}')| ,$$

- ▶ Recall that

$$\hat{L}' = \begin{pmatrix} a & 0 & \dots & 0 \\ 0 & b & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & b \end{pmatrix}$$

- ▶ Note that a and b in the above equations depend on t , by the previous equations.

The Easy Case

- ▶ We want to choose t such that the volume of $\hat{E}'$ is minimal.

$$\text{vol}(\hat{E}') = \text{vol}(B(0, 1)) \cdot |\det(\hat{L}')| ,$$

- ▶ Recall that

$$\hat{L}' = \begin{pmatrix} a & 0 & \dots & 0 \\ 0 & b & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & b \end{pmatrix}$$

- ▶ Note that a and b in the above equations depend on t , by the previous equations.

The Easy Case

- ▶ We want to choose t such that the volume of $\hat{E}'$ is minimal.

$$\text{vol}(\hat{E}') = \text{vol}(B(0,1)) \cdot |\det(\hat{L}')| ,$$

- ▶ Recall that

$$\hat{L}' = \begin{pmatrix} a & 0 & \dots & 0 \\ 0 & b & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & b \end{pmatrix}$$

- ▶ Note that a and b in the above equations depend on t , by the previous equations.

The Easy Case

$$\text{vol}(\hat{E}')$$

The Easy Case

$$\text{vol}(\hat{E}') = \text{vol}(B(0, 1)) \cdot |\det(\hat{L}')|$$

The Easy Case

$$\begin{aligned}\text{vol}(\hat{E}') &= \text{vol}(B(0, 1)) \cdot |\det(\hat{L}')| \\ &= \text{vol}(B(0, 1)) \cdot ab^{n-1}\end{aligned}$$

The Easy Case

$$\begin{aligned}\text{vol}(\hat{E}') &= \text{vol}(B(0, 1)) \cdot |\det(\hat{L}')| \\ &= \text{vol}(B(0, 1)) \cdot ab^{n-1} \\ &= \text{vol}(B(0, 1)) \cdot (1 - t) \cdot \left(\frac{1 - t}{\sqrt{1 - 2t}} \right)^{n-1}\end{aligned}$$

The Easy Case

$$\begin{aligned}\text{vol}(\hat{E}') &= \text{vol}(B(0, 1)) \cdot |\det(\hat{L}')| \\ &= \text{vol}(B(0, 1)) \cdot ab^{n-1} \\ &= \text{vol}(B(0, 1)) \cdot (1-t) \cdot \left(\frac{1-t}{\sqrt{1-2t}} \right)^{n-1} \\ &= \text{vol}(B(0, 1)) \cdot \frac{(1-t)^n}{(\sqrt{1-2t})^{n-1}}\end{aligned}$$

The Easy Case

$$\begin{aligned}\text{vol}(\hat{E}') &= \text{vol}(B(0, 1)) \cdot |\det(\hat{L}')| \\ &= \text{vol}(B(0, 1)) \cdot ab^{n-1} \\ &= \text{vol}(B(0, 1)) \cdot (1-t) \cdot \left(\frac{1-t}{\sqrt{1-2t}} \right)^{n-1} \\ &= \text{vol}(B(0, 1)) \cdot \frac{(1-t)^n}{(\sqrt{1-2t})^{n-1}}\end{aligned}$$

We use the shortcut $\Phi := \text{vol}(B(0, 1))$.

The Easy Case

$$\frac{d \operatorname{vol}(\hat{E}')}{d t}$$

The Easy Case

$$\frac{d \operatorname{vol}(\hat{E}')}{d t} = \frac{d}{d t} \left(\Phi \frac{(1-t)^n}{(\sqrt{1-2t})^{n-1}} \right)$$

The Easy Case

$$\begin{aligned}\frac{d \operatorname{vol}(\hat{E}')}{d t} &= \frac{d}{d t} \left(\Phi \frac{(1-t)^n}{(\sqrt{1-2t})^{n-1}} \right) \\ &= \frac{\Phi}{N^2}\end{aligned}$$

$N = \text{denominator}$

The Easy Case

$$\begin{aligned}\frac{d \operatorname{vol}(\hat{E}')}{d t} &= \frac{d}{d t} \left(\Phi \frac{(1-t)^n}{(\sqrt{1-2t})^{n-1}} \right) \\ &= \frac{\Phi}{N^2} \cdot \left(\begin{array}{l} (-1) \cdot n(1-t)^{n-1} \\ \text{derivative of numerator} \end{array} \right)\end{aligned}$$

The Easy Case

$$\begin{aligned}\frac{d \operatorname{vol}(\hat{E}')}{d t} &= \frac{d}{d t} \left(\Phi \frac{(1-t)^n}{(\sqrt{1-2t})^{n-1}} \right) \\ &= \frac{\Phi}{N^2} \cdot \left((-1) \cdot n(1-t)^{n-1} \cdot (\sqrt{1-2t})^{n-1} \right.\end{aligned}$$

denominator

The Easy Case

$$\begin{aligned}\frac{d \operatorname{vol}(\hat{E}')}{d t} &= \frac{d}{d t} \left(\Phi \frac{(1-t)^n}{(\sqrt{1-2t})^{n-1}} \right) \\ &= \frac{\Phi}{N^2} \cdot \left((-1) \cdot n(1-t)^{n-1} \cdot (\sqrt{1-2t})^{n-1} \right. \\ &\quad \left. - (n-1)(\sqrt{1-2t})^{n-2} \right) \\ &\quad \text{outer derivative}\end{aligned}$$

The Easy Case

$$\begin{aligned}\frac{d \operatorname{vol}(\hat{E}')}{d t} &= \frac{d}{d t} \left(\Phi \frac{(1-t)^n}{(\sqrt{1-2t})^{n-1}} \right) \\ &= \frac{\Phi}{N^2} \cdot \left((-1) \cdot n(1-t)^{n-1} \cdot (\sqrt{1-2t})^{n-1} \right. \\ &\quad \left. - (n-1)(\sqrt{1-2t})^{n-2} \cdot \frac{1}{2\sqrt{1-2t}} \cdot (-2) \right)\end{aligned}$$

inner derivative

The Easy Case

$$\begin{aligned}\frac{d \operatorname{vol}(\hat{E}')}{d t} &= \frac{d}{d t} \left(\Phi \frac{(1-t)^n}{(\sqrt{1-2t})^{n-1}} \right) \\ &= \frac{\Phi}{N^2} \cdot \left((-1) \cdot n(1-t)^{n-1} \cdot (\sqrt{1-2t})^{n-1} \right. \\ &\quad \left. - (n-1)(\sqrt{1-2t})^{n-2} \cdot \frac{1}{2\sqrt{1-2t}} \cdot (-2) \cdot (1-t)^n \right) \\ &\quad \text{numerator}\end{aligned}$$

The Easy Case

$$\begin{aligned}\frac{d \operatorname{vol}(\hat{E}')}{d t} &= \frac{d}{d t} \left(\Phi \frac{(1-t)^n}{(\sqrt{1-2t})^{n-1}} \right) \\&= \frac{\Phi}{N^2} \cdot \left((-1) \cdot n(1-t)^{n-1} \cdot (\sqrt{1-2t})^{n-1} \right. \\&\quad \left. - (n-1)(\sqrt{1-2t})^{n-2} \cdot \frac{1}{2\sqrt{1-2t}} \cdot (-2) \cdot (1-t)^n \right) \\&= \frac{\Phi}{N^2} \cdot (\sqrt{1-2t})^{n-3} \cdot (1-t)^{n-1}\end{aligned}$$

The Easy Case

$$\begin{aligned}\frac{d \operatorname{vol}(\hat{E}')}{d t} &= \frac{d}{d t} \left(\Phi \frac{(1-t)^n}{(\sqrt{1-2t})^{n-1}} \right) \\&= \frac{\Phi}{N^2} \cdot \left((-1) \cdot n(1-t)^{n-1} \cdot \frac{1-2t}{\sqrt{1-2t}} \cdot (\sqrt{1-2t})^{n-1} \right. \\&\quad \left. - (n-1)(\sqrt{1-2t})^{n-2} \cdot \frac{1}{2\sqrt{1-2t}} \cdot (-2) \cdot (1-t)^n \right) \\&= \frac{\Phi}{N^2} \cdot (\sqrt{1-2t})^{n-3} \cdot (1-t)^{n-1}\end{aligned}$$

The Easy Case

$$\begin{aligned}\frac{d \operatorname{vol}(\hat{E}')}{d t} &= \frac{d}{d t} \left(\Phi \frac{(1-t)^n}{(\sqrt{1-2t})^{n-1}} \right) \\&= \frac{\Phi}{N^2} \cdot \left((-1) \cdot n(1-t)^{n-1} \cdot \frac{1-2t}{(\sqrt{1-2t})^{n-1}} \right. \\&\quad \left. - (n-1) \frac{1}{2\sqrt{1-2t}} \cdot (-2) \cdot (1-t)^n \right) \\&= \frac{\Phi}{N^2} \cdot (\sqrt{1-2t})^{n-3} \cdot (1-t)^{n-1}\end{aligned}$$

The Easy Case

$$\begin{aligned}\frac{d \operatorname{vol}(\hat{E}')}{d t} &= \frac{d}{d t} \left(\Phi \frac{(1-t)^n}{(\sqrt{1-2t})^{n-1}} \right) \\&= \frac{\Phi}{N^2} \cdot \left((-1) \cdot n \cancel{(1-t)^{n-1}} \cdot \cancel{(\sqrt{1-2t})^{n-1}}^{1-2t} \right. \\&\quad \left. - (n-1) \cancel{(\sqrt{1-2t})^{n-2}} \cdot \frac{1}{2\sqrt{1-2t}} \cdot (-2) \cdot (1-t)^n \right) \\&= \frac{\Phi}{N^2} \cdot (\sqrt{1-2t})^{n-3} \cdot (1-t)^{n-1}\end{aligned}$$

The Easy Case

$$\begin{aligned}
 \frac{d \operatorname{vol}(\hat{E}')}{d t} &= \frac{d}{d t} \left(\Phi \frac{(1-t)^n}{(\sqrt{1-2t})^{n-1}} \right) \\
 &= \frac{\Phi}{N^2} \cdot \left((-1) \cdot n \cancel{(1-t)^{n-1}} \cdot \cancel{(\sqrt{1-2t})^{n-1}}^{1-2t} \right. \\
 &\quad \left. - (n-1) \cancel{(\sqrt{1-2t})^{n-2}} \cdot \frac{1}{2\sqrt{1-2t}} \cdot (-2) \cdot \cancel{(1-t)^n}^{1-t} \right) \\
 &= \frac{\Phi}{N^2} \cdot (\sqrt{1-2t})^{n-3} \cdot (1-t)^{n-1}
 \end{aligned}$$

The Easy Case

$$\begin{aligned}
 \frac{d \operatorname{vol}(\hat{E}')}{d t} &= \frac{d}{d t} \left(\Phi \frac{(1-t)^n}{(\sqrt{1-2t})^{n-1}} \right) \\
 &= \frac{\Phi}{N^2} \cdot \left((-1) \cdot n \cancel{(1-t)^{n-1}} \cdot \cancel{(\sqrt{1-2t})^{n-1}}^{1-2t} \right. \\
 &\quad \left. \cancel{(n-1)(\sqrt{1-2t})^{n-2}} \cdot \frac{1}{2\sqrt{1-2t}} \cdot \cancel{(-2)} \cdot \cancel{(1-t)^n}^{1-t} \right) \\
 &= \frac{\Phi}{N^2} \cdot (\sqrt{1-2t})^{n-3} \cdot (1-t)^{n-1}
 \end{aligned}$$

The Easy Case

$$\begin{aligned}
 \frac{d \operatorname{vol}(\hat{E}')}{d t} &= \frac{d}{d t} \left(\Phi \frac{(1-t)^n}{(\sqrt{1-2t})^{n-1}} \right) \\
 &= \frac{\Phi}{N^2} \cdot \left((-1) \cdot n \cancel{(1-t)^{n-1}} \cdot \cancel{(\sqrt{1-2t})^{n-1}}^{1-2t} \right. \\
 &\quad \left. \cancel{(n-1)(\sqrt{1-2t})^{n-2}} \cdot \frac{1}{2\sqrt{1-2t}} \cdot \cancel{(-2)} \cdot \cancel{(1-t)^n}^{1-t} \right) \\
 &= \frac{\Phi}{N^2} \cdot (\sqrt{1-2t})^{n-3} \cdot (1-t)^{n-1} \\
 &\quad \cdot \left((n-1)(1-t) - n(1-2t) \right)
 \end{aligned}$$

The Easy Case

$$\begin{aligned}
 \frac{d \operatorname{vol}(\hat{E}')}{d t} &= \frac{d}{d t} \left(\Phi \frac{(1-t)^n}{(\sqrt{1-2t})^{n-1}} \right) \\
 &= \frac{\Phi}{N^2} \cdot \left((-1) \cdot n \cancel{(1-t)^{n-1}} \cdot \cancel{(\sqrt{1-2t})^{n-1}}^{1-2t} \right. \\
 &\quad \left. \cancel{(n-1)(\sqrt{1-2t})^{n-2}} \cdot \frac{1}{2\sqrt{1-2t}} \cdot \cancel{(-2)} \cdot \cancel{(1-t)^n}^{1-t} \right) \\
 &= \frac{\Phi}{N^2} \cdot (\sqrt{1-2t})^{n-3} \cdot (1-t)^{n-1} \\
 &\quad \cdot \left((n-1)(1-t) - n(1-2t) \right) \\
 &= \frac{\Phi}{N^2} \cdot (\sqrt{1-2t})^{n-3} \cdot (1-t)^{n-1} \cdot \left((n+1)t - 1 \right)
 \end{aligned}$$

The Easy Case

- ▶ We obtain the minimum for $t = \frac{1}{n+1}$.
- ▶ For this value we obtain

a

The Easy Case

- ▶ We obtain the minimum for $t = \frac{1}{n+1}$.
- ▶ For this value we obtain

$$a = 1 - t$$

The Easy Case

- ▶ We obtain the minimum for $t = \frac{1}{n+1}$.
- ▶ For this value we obtain

$$a = 1 - t = \frac{n}{n+1}$$

The Easy Case

- ▶ We obtain the minimum for $t = \frac{1}{n+1}$.
- ▶ For this value we obtain

$$a = 1 - t = \frac{n}{n+1} \text{ and } b =$$

The Easy Case

- ▶ We obtain the minimum for $t = \frac{1}{n+1}$.
- ▶ For this value we obtain

$$a = 1 - t = \frac{n}{n+1} \text{ and } b = \frac{1-t}{\sqrt{1-2t}}$$

The Easy Case

- ▶ We obtain the minimum for $t = \frac{1}{n+1}$.
- ▶ For this value we obtain

$$a = 1 - t = \frac{n}{n+1} \text{ and } b = \frac{1-t}{\sqrt{1-2t}} = \frac{n}{\sqrt{n^2-1}}$$

The Easy Case

- ▶ We obtain the minimum for $t = \frac{1}{n+1}$.
- ▶ For this value we obtain

$$a = 1 - t = \frac{n}{n+1} \text{ and } b = \frac{1-t}{\sqrt{1-2t}} = \frac{n}{\sqrt{n^2-1}}$$

To see the equation for b , observe that

$$b^2$$

The Easy Case

- ▶ We obtain the minimum for $t = \frac{1}{n+1}$.
- ▶ For this value we obtain

$$a = 1 - t = \frac{n}{n+1} \text{ and } b = \frac{1-t}{\sqrt{1-2t}} = \frac{n}{\sqrt{n^2-1}}$$

To see the equation for b , observe that

$$b^2 = \frac{(1-t)^2}{1-2t}$$

The Easy Case

- ▶ We obtain the minimum for $t = \frac{1}{n+1}$.
- ▶ For this value we obtain

$$a = 1 - t = \frac{n}{n+1} \text{ and } b = \frac{1-t}{\sqrt{1-2t}} = \frac{n}{\sqrt{n^2-1}}$$

To see the equation for b , observe that

$$b^2 = \frac{(1-t)^2}{1-2t} = \frac{(1-\frac{1}{n+1})^2}{1-\frac{2}{n+1}}$$

The Easy Case

- ▶ We obtain the minimum for $t = \frac{1}{n+1}$.
- ▶ For this value we obtain

$$a = 1 - t = \frac{n}{n+1} \text{ and } b = \frac{1-t}{\sqrt{1-2t}} = \frac{n}{\sqrt{n^2-1}}$$

To see the equation for b , observe that

$$b^2 = \frac{(1-t)^2}{1-2t} = \frac{(1-\frac{1}{n+1})^2}{1-\frac{2}{n+1}} = \frac{(\frac{n}{n+1})^2}{\frac{n-1}{n+1}}$$

The Easy Case

- ▶ We obtain the minimum for $t = \frac{1}{n+1}$.
- ▶ For this value we obtain

$$a = 1 - t = \frac{n}{n+1} \text{ and } b = \frac{1-t}{\sqrt{1-2t}} = \frac{n}{\sqrt{n^2-1}}$$

To see the equation for b , observe that

$$b^2 = \frac{(1-t)^2}{1-2t} = \frac{(1-\frac{1}{n+1})^2}{1-\frac{2}{n+1}} = \frac{(\frac{n}{n+1})^2}{\frac{n-1}{n+1}} = \frac{n^2}{n^2-1}$$

The Easy Case

Let $\gamma_n = \frac{\text{vol}(\hat{E}')}{\text{vol}(B(0,1))} = ab^{n-1}$ be the ratio by which the volume changes:

$$\gamma_n^2$$

The Easy Case

Let $\gamma_n = \frac{\text{vol}(\hat{E}')}{\text{vol}(B(0,1))} = ab^{n-1}$ be the ratio by which the volume changes:

$$\gamma_n^2 = \left(\frac{n}{n+1}\right)^2 \left(\frac{n^2}{n^2-1}\right)^{n-1}$$

The Easy Case

Let $y_n = \frac{\text{vol}(\hat{E}')}{\text{vol}(B(0,1))} = ab^{n-1}$ be the ratio by which the volume changes:

$$\begin{aligned} y_n^2 &= \left(\frac{n}{n+1}\right)^2 \left(\frac{n^2}{n^2-1}\right)^{n-1} \\ &= \left(1 - \frac{1}{n+1}\right)^2 \left(1 + \frac{1}{(n-1)(n+1)}\right)^{n-1} \end{aligned}$$

The Easy Case

Let $y_n = \frac{\text{vol}(\hat{E}')}{\text{vol}(B(0,1))} = ab^{n-1}$ be the ratio by which the volume changes:

$$\begin{aligned} y_n^2 &= \left(\frac{n}{n+1}\right)^2 \left(\frac{n^2}{n^2-1}\right)^{n-1} \\ &= \left(1 - \frac{1}{n+1}\right)^2 \left(1 + \frac{1}{(n-1)(n+1)}\right)^{n-1} \\ &\leq e^{-2\frac{1}{n+1}} \cdot e^{\frac{1}{n+1}} \end{aligned}$$

The Easy Case

Let $y_n = \frac{\text{vol}(\hat{E}')}{\text{vol}(B(0,1))} = ab^{n-1}$ be the ratio by which the volume changes:

$$\begin{aligned} y_n^2 &= \left(\frac{n}{n+1}\right)^2 \left(\frac{n^2}{n^2-1}\right)^{n-1} \\ &= \left(1 - \frac{1}{n+1}\right)^2 \left(1 + \frac{1}{(n-1)(n+1)}\right)^{n-1} \\ &\leq e^{-2\frac{1}{n+1}} \cdot e^{\frac{1}{n+1}} \\ &= e^{-\frac{1}{n+1}} \end{aligned}$$

The Easy Case

Let $\gamma_n = \frac{\text{vol}(\hat{E}')}{\text{vol}(B(0,1))} = ab^{n-1}$ be the ratio by which the volume changes:

$$\begin{aligned}\gamma_n^2 &= \left(\frac{n}{n+1}\right)^2 \left(\frac{n^2}{n^2-1}\right)^{n-1} \\ &= \left(1 - \frac{1}{n+1}\right)^2 \left(1 + \frac{1}{(n-1)(n+1)}\right)^{n-1} \\ &\leq e^{-2\frac{1}{n+1}} \cdot e^{\frac{1}{n+1}} \\ &= e^{-\frac{1}{n+1}}\end{aligned}$$

where we used $(1+x)^a \leq e^{ax}$ for $x \in \mathbb{R}$ and $a > 0$.

The Easy Case

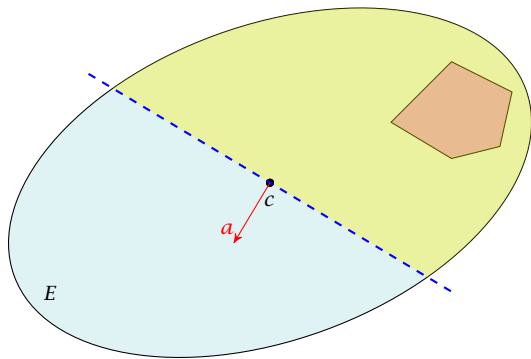
Let $y_n = \frac{\text{vol}(\hat{E}')}{\text{vol}(B(0,1))} = ab^{n-1}$ be the ratio by which the volume changes:

$$\begin{aligned} y_n^2 &= \left(\frac{n}{n+1}\right)^2 \left(\frac{n^2}{n^2-1}\right)^{n-1} \\ &= \left(1 - \frac{1}{n+1}\right)^2 \left(1 + \frac{1}{(n-1)(n+1)}\right)^{n-1} \\ &\leq e^{-2\frac{1}{n+1}} \cdot e^{\frac{1}{n+1}} \\ &= e^{-\frac{1}{n+1}} \end{aligned}$$

where we used $(1+x)^a \leq e^{ax}$ for $x \in \mathbb{R}$ and $a > 0$.

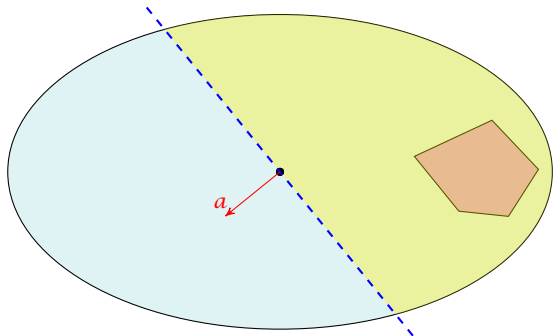
This gives $y_n \leq e^{-\frac{1}{2(n+1)}}$.

How to Compute the New Ellipsoid



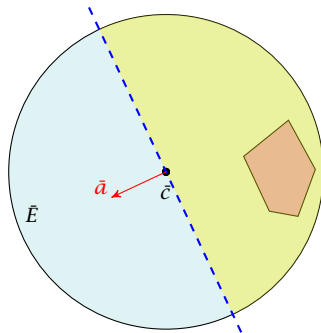
How to Compute the New Ellipsoid

- Use f^{-1} (recall that $f = Lx + t$ is the affine transformation of the unit ball) to translate/distort the ellipsoid (back) into the unit ball.



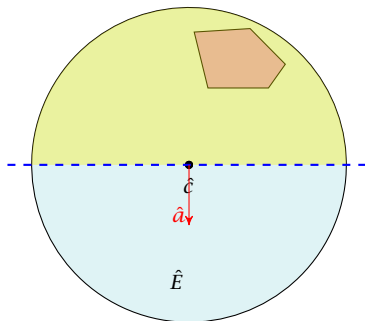
How to Compute the New Ellipsoid

- Use f^{-1} (recall that $f = Lx + t$ is the affine transformation of the unit ball) to translate/distort the ellipsoid (back) into the unit ball.



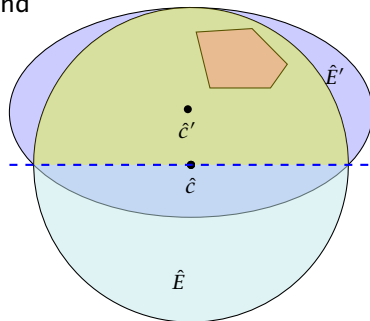
How to Compute the New Ellipsoid

- ▶ Use f^{-1} (recall that $f = Lx + t$ is the affine transformation of the unit ball) to translate/distort the ellipsoid (back) into the unit ball.
- ▶ Use a rotation R^{-1} to rotate the unit ball such that the normal vector of the halfspace is parallel to e_1 .



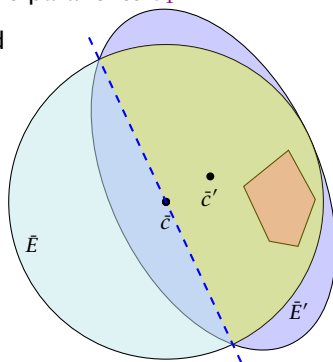
How to Compute the New Ellipsoid

- ▶ Use f^{-1} (recall that $f = Lx + t$ is the affine transformation of the unit ball) to translate/distort the ellipsoid (back) into the unit ball.
- ▶ Use a rotation R^{-1} to rotate the unit ball such that the normal vector of the halfspace is parallel to e_1 .
- ▶ Compute the new center $\hat{c}'$ and the new matrix $\hat{Q}'$ for this simplified setting.



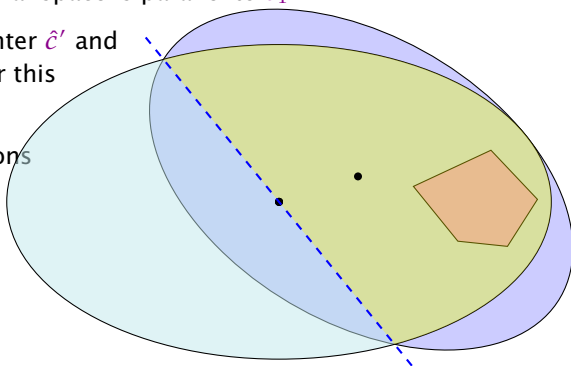
How to Compute the New Ellipsoid

- ▶ Use f^{-1} (recall that $f = Lx + t$ is the affine transformation of the unit ball) to translate/distort the ellipsoid (back) into the unit ball.
- ▶ Use a rotation R^{-1} to rotate the unit ball such that the normal vector of the halfspace is parallel to e_1 .
- ▶ Compute the new center $\tilde{c}'$ and the new matrix $\tilde{Q}'$ for this simplified setting.
- ▶ Use the transformations R and f to get the new center c' and the new matrix Q' for the original ellipsoid E .



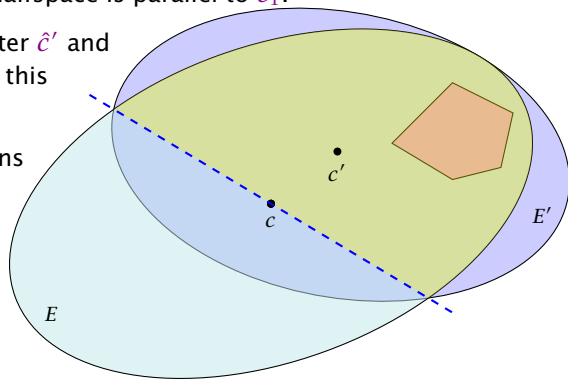
How to Compute the New Ellipsoid

- ▶ Use f^{-1} (recall that $f = Lx + t$ is the affine transformation of the unit ball) to translate/distort the ellipsoid (back) into the unit ball.
- ▶ Use a rotation R^{-1} to rotate the unit ball such that the normal vector of the halfspace is parallel to e_1 .
- ▶ Compute the new center $\hat{c}'$ and the new matrix $\hat{Q}'$ for this simplified setting.
- ▶ Use the transformations R and f to get the new center c' and the new matrix Q' for the original ellipsoid E .



How to Compute the New Ellipsoid

- ▶ Use f^{-1} (recall that $f = Lx + t$ is the affine transformation of the unit ball) to translate/distort the ellipsoid (back) into the unit ball.
- ▶ Use a rotation R^{-1} to rotate the unit ball such that the normal vector of the halfspace is parallel to e_1 .
- ▶ Compute the new center $\hat{c}'$ and the new matrix $\hat{Q}'$ for this simplified setting.
- ▶ Use the transformations R and f to get the new center c' and the new matrix Q' for the original ellipsoid E .



Our progress is the same:

$$e^{-\frac{1}{2(n+1)}}$$

Our progress is the same:

$$e^{-\frac{1}{2(n+1)}} \geq \frac{\text{vol}(\hat{E}')}{\text{vol}(B(0,1))}$$

Our progress is the same:

$$e^{-\frac{1}{2(n+1)}} \geq \frac{\text{vol}(\hat{E}')}{\text{vol}(B(0,1))} = \frac{\text{vol}(\hat{E}')}{\text{vol}(\hat{E})}$$

Our progress is the same:

$$e^{-\frac{1}{2(n+1)}} \geq \frac{\text{vol}(\hat{E}')}{\text{vol}(B(0,1))} = \frac{\text{vol}(\hat{E}')}{\text{vol}(\hat{E})} = \frac{\text{vol}(R(\hat{E}'))}{\text{vol}(R(\hat{E}))}$$

Our progress is the same:

$$\begin{aligned} e^{-\frac{1}{2(n+1)}} &\geq \frac{\text{vol}(\hat{E}')}{\text{vol}(B(0,1))} = \frac{\text{vol}(\hat{E}')}{\text{vol}(\hat{E})} = \frac{\text{vol}(R(\hat{E}'))}{\text{vol}(R(\hat{E}))} \\ &= \frac{\text{vol}(\tilde{E}')}{\text{vol}(\tilde{E})} \end{aligned}$$

Our progress is the same:

$$\begin{aligned} e^{-\frac{1}{2(n+1)}} &\geq \frac{\text{vol}(\hat{E}')}{\text{vol}(B(0,1))} = \frac{\text{vol}(\hat{E}')}{\text{vol}(\hat{E})} = \frac{\text{vol}(R(\hat{E}'))}{\text{vol}(R(\hat{E}))} \\ &= \frac{\text{vol}(\tilde{E}')}{\text{vol}(\tilde{E})} = \frac{\text{vol}(f(\tilde{E}'))}{\text{vol}(f(\tilde{E}))} \end{aligned}$$

Our progress is the same:

$$\begin{aligned} e^{-\frac{1}{2(n+1)}} &\geq \frac{\text{vol}(\hat{E}')}{\text{vol}(B(0,1))} = \frac{\text{vol}(\hat{E}')}{\text{vol}(\hat{E})} = \frac{\text{vol}(R(\hat{E}'))}{\text{vol}(R(\hat{E}))} \\ &= \frac{\text{vol}(\bar{E}')}{\text{vol}(\bar{E})} = \frac{\text{vol}(f(\bar{E}'))}{\text{vol}(f(\bar{E}))} = \frac{\text{vol}(E')}{\text{vol}(E)} \end{aligned}$$

Our progress is the same:

$$\begin{aligned} e^{-\frac{1}{2(n+1)}} &\geq \frac{\text{vol}(\hat{E}')}{\text{vol}(B(0,1))} = \frac{\text{vol}(\hat{E}')}{\text{vol}(\hat{E})} = \frac{\text{vol}(R(\hat{E}'))}{\text{vol}(R(\hat{E}))} \\ &= \frac{\text{vol}(\tilde{E}')}{\text{vol}(\tilde{E})} = \frac{\text{vol}(f(\tilde{E}'))}{\text{vol}(f(\tilde{E}))} = \frac{\text{vol}(E')}{\text{vol}(E)} \end{aligned}$$

Here it is important that mapping a set with affine function $f(x) = Lx + t$ changes the volume by factor $\det(L)$.

The Ellipsoid Algorithm

How to compute the new parameters?

The Ellipsoid Algorithm

How to compute the new parameters?

The transformation function of the (old) ellipsoid: $f(x) = Lx + c$;

The Ellipsoid Algorithm

How to compute the new parameters?

The transformation function of the (old) ellipsoid: $f(x) = Lx + c$;

The halfspace to be intersected: $H = \{x \mid a^T(x - c) \leq 0\}$;

The Ellipsoid Algorithm

How to compute the new parameters?

The transformation function of the (old) ellipsoid: $f(x) = Lx + c$;

The halfspace to be intersected: $H = \{x \mid a^T(x - c) \leq 0\}$;

$$f^{-1}(H) = \{f^{-1}(x) \mid a^T(x - c) \leq 0\}$$

The Ellipsoid Algorithm

How to compute the new parameters?

The transformation function of the (old) ellipsoid: $f(x) = Lx + c$;

The halfspace to be intersected: $H = \{x \mid a^T(x - c) \leq 0\}$;

$$\begin{aligned} f^{-1}(H) &= \{f^{-1}(x) \mid a^T(x - c) \leq 0\} \\ &= \{f^{-1}(f(y)) \mid a^T(f(y) - c) \leq 0\} \end{aligned}$$

The Ellipsoid Algorithm

How to compute the new parameters?

The transformation function of the (old) ellipsoid: $f(x) = Lx + c$;

The halfspace to be intersected: $H = \{x \mid a^T(x - c) \leq 0\}$;

$$\begin{aligned} f^{-1}(H) &= \{f^{-1}(x) \mid a^T(x - c) \leq 0\} \\ &= \{f^{-1}(f(y)) \mid a^T(f(y) - c) \leq 0\} \\ &= \{y \mid a^T(f(y) - c) \leq 0\} \end{aligned}$$

The Ellipsoid Algorithm

How to compute the new parameters?

The transformation function of the (old) ellipsoid: $f(x) = Lx + c$;

The halfspace to be intersected: $H = \{x \mid a^T(x - c) \leq 0\}$;

$$\begin{aligned} f^{-1}(H) &= \{f^{-1}(x) \mid a^T(x - c) \leq 0\} \\ &= \{f^{-1}(f(y)) \mid a^T(f(y) - c) \leq 0\} \\ &= \{y \mid a^T(f(y) - c) \leq 0\} \\ &= \{y \mid a^T(Ly + c - c) \leq 0\} \end{aligned}$$

The Ellipsoid Algorithm

How to compute the new parameters?

The transformation function of the (old) ellipsoid: $f(x) = Lx + c$;

The halfspace to be intersected: $H = \{x \mid a^T(x - c) \leq 0\}$;

$$\begin{aligned} f^{-1}(H) &= \{f^{-1}(x) \mid a^T(x - c) \leq 0\} \\ &= \{f^{-1}(f(y)) \mid a^T(f(y) - c) \leq 0\} \\ &= \{y \mid a^T(f(y) - c) \leq 0\} \\ &= \{y \mid a^T(Ly + c - c) \leq 0\} \\ &= \{y \mid (a^T L)y \leq 0\} \end{aligned}$$

The Ellipsoid Algorithm

How to compute the new parameters?

The transformation function of the (old) ellipsoid: $f(x) = Lx + c$;

The halfspace to be intersected: $H = \{x \mid a^T(x - c) \leq 0\}$;

$$\begin{aligned} f^{-1}(H) &= \{f^{-1}(x) \mid a^T(x - c) \leq 0\} \\ &= \{f^{-1}(f(y)) \mid a^T(f(y) - c) \leq 0\} \\ &= \{y \mid a^T(f(y) - c) \leq 0\} \\ &= \{y \mid a^T(Ly + c - c) \leq 0\} \\ &= \{y \mid (a^T L)y \leq 0\} \end{aligned}$$

This means $\bar{a} = L^T a$.

The Ellipsoid Algorithm

After rotating back (applying R^{-1}) the normal vector of the halfspace points in negative x_1 -direction. Hence,

$$R^{-1}\left(\frac{L^T a}{\|L^T a\|}\right) = -e_1 \quad \Rightarrow \quad -\frac{L^T a}{\|L^T a\|} = R \cdot e_1$$

The Ellipsoid Algorithm

After rotating back (applying R^{-1}) the normal vector of the halfspace points in negative x_1 -direction. Hence,

$$R^{-1} \left(\frac{L^T a}{\|L^T a\|} \right) = -e_1 \quad \Rightarrow \quad -\frac{L^T a}{\|L^T a\|} = R \cdot e_1$$

Hence,

$$\bar{c}'$$

The Ellipsoid Algorithm

After rotating back (applying R^{-1}) the normal vector of the halfspace points in negative x_1 -direction. Hence,

$$R^{-1} \left(\frac{L^T a}{\|L^T a\|} \right) = -e_1 \quad \Rightarrow \quad -\frac{L^T a}{\|L^T a\|} = R \cdot e_1$$

Hence,

$$\bar{c}' = R \cdot \hat{c}'$$

The Ellipsoid Algorithm

After rotating back (applying R^{-1}) the normal vector of the halfspace points in negative x_1 -direction. Hence,

$$R^{-1}\left(\frac{L^T a}{\|L^T a\|}\right) = -e_1 \quad \Rightarrow \quad -\frac{L^T a}{\|L^T a\|} = R \cdot e_1$$

Hence,

$$\bar{c}' = R \cdot \hat{c}' = R \cdot \frac{1}{n+1} e_1$$

The Ellipsoid Algorithm

After rotating back (applying R^{-1}) the normal vector of the halfspace points in negative x_1 -direction. Hence,

$$R^{-1}\left(\frac{L^T a}{\|L^T a\|}\right) = -e_1 \quad \Rightarrow \quad -\frac{L^T a}{\|L^T a\|} = R \cdot e_1$$

Hence,

$$\bar{c}' = R \cdot \hat{c}' = R \cdot \frac{1}{n+1} e_1 = -\frac{1}{n+1} \frac{L^T a}{\|L^T a\|}$$

The Ellipsoid Algorithm

After rotating back (applying R^{-1}) the normal vector of the halfspace points in negative x_1 -direction. Hence,

$$R^{-1}\left(\frac{L^T a}{\|L^T a\|}\right) = -e_1 \quad \Rightarrow \quad -\frac{L^T a}{\|L^T a\|} = R \cdot e_1$$

Hence,

$$\bar{c}' = R \cdot \hat{c}' = R \cdot \frac{1}{n+1} e_1 = -\frac{1}{n+1} \frac{L^T a}{\|L^T a\|}$$

$$c'$$

The Ellipsoid Algorithm

After rotating back (applying R^{-1}) the normal vector of the halfspace points in negative x_1 -direction. Hence,

$$R^{-1}\left(\frac{L^T a}{\|L^T a\|}\right) = -e_1 \quad \Rightarrow \quad -\frac{L^T a}{\|L^T a\|} = R \cdot e_1$$

Hence,

$$\tilde{c}' = R \cdot \hat{c}' = R \cdot \frac{1}{n+1} e_1 = -\frac{1}{n+1} \frac{L^T a}{\|L^T a\|}$$

$$c' = f(\tilde{c}')$$

The Ellipsoid Algorithm

After rotating back (applying R^{-1}) the normal vector of the halfspace points in negative x_1 -direction. Hence,

$$R^{-1}\left(\frac{L^T a}{\|L^T a\|}\right) = -e_1 \quad \Rightarrow \quad -\frac{L^T a}{\|L^T a\|} = R \cdot e_1$$

Hence,

$$\tilde{c}' = R \cdot \hat{c}' = R \cdot \frac{1}{n+1} e_1 = -\frac{1}{n+1} \frac{L^T a}{\|L^T a\|}$$

$$c' = f(\tilde{c}') = L \cdot \tilde{c}' + c$$

The Ellipsoid Algorithm

After rotating back (applying R^{-1}) the normal vector of the halfspace points in negative x_1 -direction. Hence,

$$R^{-1}\left(\frac{L^T a}{\|L^T a\|}\right) = -e_1 \quad \Rightarrow \quad -\frac{L^T a}{\|L^T a\|} = R \cdot e_1$$

Hence,

$$\tilde{c}' = R \cdot \hat{c}' = R \cdot \frac{1}{n+1} e_1 = -\frac{1}{n+1} \frac{L^T a}{\|L^T a\|}$$

$$\begin{aligned} c' &= f(\tilde{c}') = L \cdot \tilde{c}' + c \\ &= -\frac{1}{n+1} L \frac{L^T a}{\|L^T a\|} + c \end{aligned}$$

The Ellipsoid Algorithm

After rotating back (applying R^{-1}) the normal vector of the halfspace points in negative x_1 -direction. Hence,

$$R^{-1}\left(\frac{L^T a}{\|L^T a\|}\right) = -e_1 \quad \Rightarrow \quad -\frac{L^T a}{\|L^T a\|} = R \cdot e_1$$

Hence,

$$\tilde{c}' = R \cdot \hat{c}' = R \cdot \frac{1}{n+1} e_1 = -\frac{1}{n+1} \frac{L^T a}{\|L^T a\|}$$

$$\begin{aligned} c' &= f(\tilde{c}') = L \cdot \tilde{c}' + c \\ &= -\frac{1}{n+1} L \frac{L^T a}{\|L^T a\|} + c \\ &= c - \frac{1}{n+1} \frac{Qa}{\sqrt{a^T Qa}} \end{aligned}$$

For computing the matrix Q' of the new ellipsoid we assume in the following that $\hat{E}', \bar{E}'$ and E' refer to the ellipsoids centered in the origin.

Recall that

$$\hat{Q}' = \begin{pmatrix} a^2 & 0 & \dots & 0 \\ 0 & b^2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & b^2 \end{pmatrix}$$

This gives

$$\hat{Q}' = \frac{n^2}{n^2-1} \left(I - \frac{2}{n+1} e_1 e_1^T \right)$$

because for $a^2 = n^2/(n+1)^2$ and $b^2 = n^2/n^2-1$

Recall that

$$\hat{Q}' = \begin{pmatrix} a^2 & 0 & \dots & 0 \\ 0 & b^2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & b^2 \end{pmatrix}$$

This gives

$$\hat{Q}' = \frac{n^2}{n^2 - 1} \left(I - \frac{2}{n + 1} e_1 e_1^T \right)$$

because for $a^2 = n^2/(n+1)^2$ and $b^2 = n^2/n^2-1$

Recall that

$$\hat{Q}' = \begin{pmatrix} a^2 & 0 & \dots & 0 \\ 0 & b^2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & b^2 \end{pmatrix}$$

This gives

$$\hat{Q}' = \frac{n^2}{n^2 - 1} \left(I - \frac{2}{n + 1} e_1 e_1^T \right)$$

because for $a^2 = n^2/(n+1)^2$ and $b^2 = n^2/n^2 - 1$

Recall that

$$\hat{Q}' = \begin{pmatrix} a^2 & 0 & \dots & 0 \\ 0 & b^2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & b^2 \end{pmatrix}$$

This gives

$$\hat{Q}' = \frac{n^2}{n^2 - 1} \left(I - \frac{2}{n + 1} e_1 e_1^T \right)$$

because for $a^2 = n^2/(n+1)^2$ and $b^2 = n^2/n^2 - 1$

$$\begin{aligned} b^2 - b^2 \frac{2}{n + 1} &= \frac{n^2}{n^2 - 1} - \frac{2n^2}{(n - 1)(n + 1)^2} \\ &= \frac{n^2(n + 1) - 2n^2}{(n - 1)(n + 1)^2} = \frac{n^2(n - 1)}{(n - 1)(n + 1)^2} = a^2 \end{aligned}$$

Recall that

$$\hat{Q}' = \begin{pmatrix} a^2 & 0 & \dots & 0 \\ 0 & b^2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & b^2 \end{pmatrix}$$

This gives

$$\hat{Q}' = \frac{n^2}{n^2 - 1} \left(I - \frac{2}{n + 1} e_1 e_1^T \right)$$

because for $a^2 = n^2/(n+1)^2$ and $b^2 = n^2/n^2 - 1$

$$\begin{aligned} b^2 - b^2 \frac{2}{n + 1} &= \frac{n^2}{n^2 - 1} - \frac{2n^2}{(n - 1)(n + 1)^2} \\ &= \frac{n^2(n + 1) - 2n^2}{(n - 1)(n + 1)^2} = \frac{n^2(n - 1)}{(n - 1)(n + 1)^2} = a^2 \end{aligned}$$

Recall that

$$\hat{Q}' = \begin{pmatrix} a^2 & 0 & \dots & 0 \\ 0 & b^2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & b^2 \end{pmatrix}$$

This gives

$$\hat{Q}' = \frac{n^2}{n^2 - 1} \left(I - \frac{2}{n + 1} e_1 e_1^T \right)$$

because for $a^2 = n^2/(n+1)^2$ and $b^2 = n^2/n^2 - 1$

$$\begin{aligned} b^2 - b^2 \frac{2}{n + 1} &= \frac{n^2}{n^2 - 1} - \frac{2n^2}{(n - 1)(n + 1)^2} \\ &= \frac{n^2(n + 1) - 2n^2}{(n - 1)(n + 1)^2} = \frac{n^2(n - 1)}{(n - 1)(n + 1)^2} = a^2 \end{aligned}$$

Recall that

$$\hat{Q}' = \begin{pmatrix} a^2 & 0 & \dots & 0 \\ 0 & b^2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & b^2 \end{pmatrix}$$

This gives

$$\hat{Q}' = \frac{n^2}{n^2 - 1} \left(I - \frac{2}{n + 1} e_1 e_1^T \right)$$

because for $a^2 = n^2/(n+1)^2$ and $b^2 = n^2/n^2-1$

$$\begin{aligned} b^2 - b^2 \frac{2}{n+1} &= \frac{n^2}{n^2-1} - \frac{2n^2}{(n-1)(n+1)^2} \\ &= \frac{n^2(n+1) - 2n^2}{(n-1)(n+1)^2} = \frac{n^2(n-1)}{(n-1)(n+1)^2} = a^2 \end{aligned}$$

Recall that

$$\hat{Q}' = \begin{pmatrix} a^2 & 0 & \dots & 0 \\ 0 & b^2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & b^2 \end{pmatrix}$$

This gives

$$\hat{Q}' = \frac{n^2}{n^2 - 1} \left(I - \frac{2}{n + 1} e_1 e_1^T \right)$$

because for $a^2 = n^2/(n+1)^2$ and $b^2 = n^2/n^2-1$

$$\begin{aligned} b^2 - b^2 \frac{2}{n+1} &= \frac{n^2}{n^2-1} - \frac{2n^2}{(n-1)(n+1)^2} \\ &= \frac{n^2(n+1) - 2n^2}{(n-1)(n+1)^2} = \frac{n^2(n-1)}{(n-1)(n+1)^2} = a^2 \end{aligned}$$

9 The Ellipsoid Algorithm

$$\bar{E}'$$

9 The Ellipsoid Algorithm

$$\bar{E}' = R(\hat{E}')$$

9 The Ellipsoid Algorithm

$$\begin{aligned}\bar{E}' &= R(\hat{E}') \\ &= \{R(x) \mid x^T \hat{Q}'^{-1} x \leq 1\}\end{aligned}$$

9 The Ellipsoid Algorithm

$$\begin{aligned}\bar{E}' &= R(\hat{E}') \\ &= \{R(x) \mid x^T \hat{Q}'^{-1} x \leq 1\} \\ &= \{y \mid (R^{-1}y)^T \hat{Q}'^{-1} R^{-1}y \leq 1\}\end{aligned}$$

9 The Ellipsoid Algorithm

$$\begin{aligned}\bar{E}' &= R(\hat{E}') \\ &= \{R(x) \mid x^T \hat{Q}'^{-1} x \leq 1\} \\ &= \{y \mid (R^{-1}y)^T \hat{Q}'^{-1} R^{-1}y \leq 1\} \\ &= \{y \mid y^T (R^T)^{-1} \hat{Q}'^{-1} R^{-1}y \leq 1\}\end{aligned}$$

9 The Ellipsoid Algorithm

$$\begin{aligned}\bar{E}' &= R(\hat{E}') \\ &= \{R(x) \mid x^T \hat{Q}'^{-1} x \leq 1\} \\ &= \{y \mid (R^{-1}y)^T \hat{Q}'^{-1} R^{-1}y \leq 1\} \\ &= \{y \mid y^T (R^T)^{-1} \hat{Q}'^{-1} R^{-1}y \leq 1\} \\ &= \{y \mid y^T \underbrace{(R \hat{Q}' R^T)^{-1}}_{\hat{Q}'} y \leq 1\}\end{aligned}$$

9 The Ellipsoid Algorithm

Hence,

$$\bar{Q}'$$

9 The Ellipsoid Algorithm

Hence,

$$\bar{Q}' = R\hat{Q}'R^T$$

9 The Ellipsoid Algorithm

Hence,

$$\begin{aligned}\bar{Q}' &= R\hat{Q}'R^T \\ &= R \cdot \frac{n^2}{n^2 - 1} \left(I - \frac{2}{n + 1} e_1 e_1^T \right) \cdot R^T\end{aligned}$$

9 The Ellipsoid Algorithm

Hence,

$$\begin{aligned}\bar{Q}' &= R\hat{Q}'R^T \\ &= R \cdot \frac{n^2}{n^2 - 1} \left(I - \frac{2}{n + 1} e_1 e_1^T \right) \cdot R^T \\ &= \frac{n^2}{n^2 - 1} \left(R \cdot R^T - \frac{2}{n + 1} (Re_1)(Re_1)^T \right)\end{aligned}$$

9 The Ellipsoid Algorithm

Hence,

$$\begin{aligned}\bar{Q}' &= R\hat{Q}'R^T \\ &= R \cdot \frac{n^2}{n^2-1} \left(I - \frac{2}{n+1} e_1 e_1^T \right) \cdot R^T \\ &= \frac{n^2}{n^2-1} \left(R \cdot R^T - \frac{2}{n+1} (Re_1)(Re_1)^T \right) \\ &= \frac{n^2}{n^2-1} \left(I - \frac{2}{n+1} \frac{L^T a a^T L}{\|L^T a\|^2} \right)\end{aligned}$$

9 The Ellipsoid Algorithm

E'

9 The Ellipsoid Algorithm

$$E' = L(\tilde{E}')$$

9 The Ellipsoid Algorithm

$$\begin{aligned} E' &= L(\tilde{E}') \\ &= \{L(x) \mid x^T \tilde{Q}'^{-1} x \leq 1\} \end{aligned}$$

9 The Ellipsoid Algorithm

$$\begin{aligned} E' &= L(\tilde{E}') \\ &= \{L(x) \mid x^T \tilde{Q}'^{-1} x \leq 1\} \\ &= \{y \mid (L^{-1}y)^T \tilde{Q}'^{-1} L^{-1}y \leq 1\} \end{aligned}$$

9 The Ellipsoid Algorithm

$$\begin{aligned} E' &= L(\tilde{E}') \\ &= \{L(x) \mid x^T \tilde{Q}'^{-1} x \leq 1\} \\ &= \{y \mid (L^{-1}y)^T \tilde{Q}'^{-1} L^{-1}y \leq 1\} \\ &= \{y \mid y^T (L^T)^{-1} \tilde{Q}'^{-1} L^{-1}y \leq 1\} \end{aligned}$$

9 The Ellipsoid Algorithm

$$\begin{aligned} E' &= L(\tilde{E}') \\ &= \{L(x) \mid x^T \tilde{Q}'^{-1} x \leq 1\} \\ &= \{y \mid (L^{-1}y)^T \tilde{Q}'^{-1} L^{-1}y \leq 1\} \\ &= \{y \mid y^T (L^T)^{-1} \tilde{Q}'^{-1} L^{-1}y \leq 1\} \\ &= \{y \mid y^T \underbrace{(L\tilde{Q}'L^T)^{-1}}_{Q'} y \leq 1\} \end{aligned}$$

9 The Ellipsoid Algorithm

Hence,

Q'

9 The Ellipsoid Algorithm

Hence,

$$Q' = L\bar{Q}'L^T$$

9 The Ellipsoid Algorithm

Hence,

$$\begin{aligned} Q' &= L\bar{Q}'L^T \\ &= L \cdot \frac{n^2}{n^2 - 1} \left(I - \frac{2}{n + 1} \frac{L^T a a^T L}{a^T Q a} \right) \cdot L^T \end{aligned}$$

9 The Ellipsoid Algorithm

Hence,

$$\begin{aligned} Q' &= L\bar{Q}'L^T \\ &= L \cdot \frac{n^2}{n^2-1} \left(I - \frac{2}{n+1} \frac{L^T a a^T L}{a^T Q a} \right) \cdot L^T \\ &= \frac{n^2}{n^2-1} \left(Q - \frac{2}{n+1} \frac{Q a a^T Q}{a^T Q a} \right) \end{aligned}$$

Incomplete Algorithm

Algorithm 1 ellipsoid-algorithm

```
1: input: point  $c \in \mathbb{R}^n$ , convex set  $K \subseteq \mathbb{R}^n$ 
2: output: point  $x \in K$  or “ $K$  is empty”
3:  $Q \leftarrow ???$ 
4: repeat
5:     if  $c \in K$  then return  $c$ 
6:     else
7:         choose a violated hyperplane  $a$ 
8:         
$$c \leftarrow c - \frac{1}{n+1} \frac{Qa}{\sqrt{a^T Q a}}$$

9:         
$$Q \leftarrow \frac{n^2}{n^2 - 1} \left( Q - \frac{2}{n+1} \frac{Qaa^T Q}{a^T Q a} \right)$$

10:    endif
11: until  $???$ 
12: return “ $K$  is empty”
```

Repeat: Size of basic solutions

Lemma 52

Let $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$ be a bounded polyhedron. Let $L := 2\langle A \rangle + \langle b \rangle + 2n(1 + \log_2 n)$. Then every entry x_j in a basic solution fulfills $|x_j| = \frac{D_j}{D}$ with $D_j, D \leq 2^L$.

In the following we use $\delta := 2^L$.

Proof:

We can replace P by $P' := \{x \mid A'x \leq b; x \geq 0\}$ where $A' = \begin{bmatrix} A \\ -A \end{bmatrix}$. The lemma follows by applying Lemma 47, and observing that $\langle A' \rangle = 2\langle A \rangle$ and $n' = 2n$.

Repeat: Size of basic solutions

Lemma 52

Let $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$ be a bounded polyhedron. Let $L := 2\langle A \rangle + \langle b \rangle + 2n(1 + \log_2 n)$. Then every entry x_j in a basic solution fulfills $|x_j| = \frac{D_j}{D}$ with $D_j, D \leq 2^L$.

In the following we use $\delta := 2^L$.

Proof:

We can replace P by $P' := \{x \mid A'x \leq b; x \geq 0\}$ where $A' = \begin{bmatrix} A \\ -A \end{bmatrix}$. The lemma follows by applying Lemma 47, and observing that $\langle A' \rangle = 2\langle A \rangle$ and $n' = 2n$.

How do we find the first ellipsoid?

For feasibility checking we can assume that the polytop P is bounded; it is sufficient to consider basic solutions.

Every entry x_i in a basic solution fulfills $|x_i| \leq \delta$.

Hence, P is contained in the cube $-\delta \leq x_i \leq \delta$.

A vector in this cube has at most distance $R := \sqrt{n}\delta$ from the origin.

Starting with the ball $E_0 := B(0, R)$ ensures that P is completely contained in the initial ellipsoid. This ellipsoid has volume at most $R^n \text{vol}(B(0, 1)) \leq (n\delta)^n \text{vol}(B(0, 1))$.

How do we find the first ellipsoid?

For feasibility checking we can assume that the polytop P is bounded; it is sufficient to consider basic solutions.

Every entry x_i in a basic solution fulfills $|x_i| \leq \delta$.

Hence, P is contained in the cube $-\delta \leq x_i \leq \delta$.

A vector in this cube has at most distance $R := \sqrt{n}\delta$ from the origin.

Starting with the ball $E_0 := B(0, R)$ ensures that P is completely contained in the initial ellipsoid. This ellipsoid has volume at most $R^n \text{vol}(B(0, 1)) \leq (n\delta)^n \text{vol}(B(0, 1))$.

How do we find the first ellipsoid?

For feasibility checking we can assume that the polytop P is bounded; it is sufficient to consider basic solutions.

Every entry x_i in a basic solution fulfills $|x_i| \leq \delta$.

Hence, P is contained in the cube $-\delta \leq x_i \leq \delta$.

A vector in this cube has at most distance $R := \sqrt{n}\delta$ from the origin.

Starting with the ball $E_0 := B(0, R)$ ensures that P is completely contained in the initial ellipsoid. This ellipsoid has volume at most $R^n \text{vol}(B(0, 1)) \leq (n\delta)^n \text{vol}(B(0, 1))$.

How do we find the first ellipsoid?

For feasibility checking we can assume that the polytop P is bounded; it is sufficient to consider basic solutions.

Every entry x_i in a basic solution fulfills $|x_i| \leq \delta$.

Hence, P is contained in the cube $-\delta \leq x_i \leq \delta$.

A vector in this cube has at most distance $R := \sqrt{n}\delta$ from the origin.

Starting with the ball $E_0 := B(0, R)$ ensures that P is completely contained in the initial ellipsoid. This ellipsoid has volume at most $R^n \text{vol}(B(0, 1)) \leq (n\delta)^n \text{vol}(B(0, 1))$.

How do we find the first ellipsoid?

For feasibility checking we can assume that the polytop P is bounded; it is sufficient to consider basic solutions.

Every entry x_i in a basic solution fulfills $|x_i| \leq \delta$.

Hence, P is contained in the cube $-\delta \leq x_i \leq \delta$.

A vector in this cube has at most distance $R := \sqrt{n}\delta$ from the origin.

Starting with the ball $E_0 := B(0, R)$ ensures that P is completely contained in the initial ellipsoid. This ellipsoid has volume at most $R^n \text{vol}(B(0, 1)) \leq (n\delta)^n \text{vol}(B(0, 1))$.

How do we find the first ellipsoid?

For feasibility checking we can assume that the polytop P is bounded; it is sufficient to consider basic solutions.

Every entry x_i in a basic solution fulfills $|x_i| \leq \delta$.

Hence, P is contained in the cube $-\delta \leq x_i \leq \delta$.

A vector in this cube has at most distance $R := \sqrt{n}\delta$ from the origin.

Starting with the ball $E_0 := B(0, R)$ ensures that P is completely contained in the initial ellipsoid. This ellipsoid has volume at most $R^n \text{vol}(B(0, 1)) \leq (n\delta)^n \text{vol}(B(0, 1))$.

When can we terminate?

Let $P := \{x \mid Ax \leq b\}$ with $A \in \mathbb{Z}$ and $b \in \mathbb{Z}$ be a bounded polytop.

Consider the following polyhedron

$$P_\lambda := \left\{ x \mid Ax \leq b + \frac{1}{\lambda} \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \right\},$$

where $\lambda = \delta^2 + 1$.

Note that the volume of P_λ cannot be 0

When can we terminate?

Let $P := \{x \mid Ax \leq b\}$ with $A \in \mathbb{Z}$ and $b \in \mathbb{Z}$ be a bounded polytop.

Consider the following polyhedron

$$P_\lambda := \left\{ x \mid Ax \leq b + \frac{1}{\lambda} \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \right\},$$

where $\lambda = \delta^2 + 1$.

Note that the volume of P_λ cannot be 0

When can we terminate?

Let $P := \{x \mid Ax \leq b\}$ with $A \in \mathbb{Z}$ and $b \in \mathbb{Z}$ be a bounded polytop.

Consider the following polyhedron

$$P_\lambda := \left\{ x \mid Ax \leq b + \frac{1}{\lambda} \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \right\},$$

where $\lambda = \delta^2 + 1$.

Note that the volume of P_λ cannot be 0

Making P full-dimensional

Lemma 53

P_λ is feasible if and only if P is feasible.

$\Leftarrow$: obvious!

Making P full-dimensional

Lemma 53

P_λ is feasible if and only if P is feasible.

$\Leftarrow$: obvious!

Making P full-dimensional

$\Rightarrow$:

Consider the polyhedrons

$$\bar{P} = \{x \mid [A \ -A \ I_m]x = b; x \geq 0\}$$

and

$$\bar{P}_\lambda = \left\{x \mid [A \ -A \ I_m]x = b + \frac{1}{\lambda} \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}; x \geq 0\right\}.$$

P is feasible if and only if $\bar{P}$ is feasible, and P_λ feasible if and only if $\bar{P}_\lambda$ feasible.

$\bar{P}_\lambda$ is bounded since P_λ and P are bounded.

Making P full-dimensional

$\Rightarrow$:

Consider the polyhedrons

$$\bar{P} = \{x \mid [A \ -A \ I_m]x = b; x \geq 0\}$$

and

$$\bar{P}_\lambda = \left\{x \mid [A \ -A \ I_m]x = b + \frac{1}{\lambda} \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}; x \geq 0\right\} .$$

P is feasible if and only if $\bar{P}$ is feasible, and P_λ feasible if and only if $\bar{P}_\lambda$ feasible.

$\bar{P}_\lambda$ is bounded since P_λ and P are bounded.

Making P full-dimensional

$\Rightarrow$:

Consider the polyhedrons

$$\bar{P} = \{x \mid [A \ -A \ I_m]x = b; x \geq 0\}$$

and

$$\bar{P}_\lambda = \left\{x \mid [A \ -A \ I_m]x = b + \frac{1}{\lambda} \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}; x \geq 0\right\} .$$

P is feasible if and only if $\bar{P}$ is feasible, and P_λ feasible if and only if $\bar{P}_\lambda$ feasible.

$\bar{P}_\lambda$ is bounded since P_λ and P are bounded.

Making P full-dimensional

$\Rightarrow$:

Consider the polyhedrons

$$\bar{P} = \{x \mid [A \ -A \ I_m]x = b; x \geq 0\}$$

and

$$\bar{P}_\lambda = \left\{x \mid [A \ -A \ I_m]x = b + \frac{1}{\lambda} \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}; x \geq 0\right\} .$$

P is feasible if and only if $\bar{P}$ is feasible, and P_λ feasible if and only if $\bar{P}_\lambda$ feasible.

$\bar{P}_\lambda$ is bounded since P_λ and P are bounded.

Making P full-dimensional

Let $\bar{A} = \begin{bmatrix} A & -A & I_m \end{bmatrix}$.

$\bar{P}_\lambda$ feasible implies that there is a basic feasible solution represented by

$$x_B = \bar{A}_B^{-1}b + \frac{1}{\lambda}\bar{A}_B^{-1} \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$$

(The other x -values are zero)

The only reason that this basic feasible solution is not feasible for $\bar{P}$ is that one of the basic variables becomes negative.

Hence, there exists i with

$$(\bar{A}_B^{-1}b)_i < 0 \leq (\bar{A}_B^{-1}b)_i + \frac{1}{\lambda}(\bar{A}_B^{-1}\bar{1})_i$$

Making P full-dimensional

Let $\bar{A} = \begin{bmatrix} A & -A & I_m \end{bmatrix}$.

$\bar{P}_\lambda$ feasible implies that there is a basic feasible solution represented by

$$x_B = \bar{A}_B^{-1}b + \frac{1}{\lambda}\bar{A}_B^{-1} \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$$

(The other x -values are zero)

The only reason that this basic feasible solution is not feasible for $\bar{P}$ is that one of the basic variables becomes negative.

Hence, there exists i with

$$(\bar{A}_B^{-1}b)_i < 0 \leq (\bar{A}_B^{-1}b)_i + \frac{1}{\lambda}(\bar{A}_B^{-1}\bar{1})_i$$

Making P full-dimensional

Let $\bar{A} = \begin{bmatrix} A & -A & I_m \end{bmatrix}$.

$\bar{P}_\lambda$ feasible implies that there is a basic feasible solution represented by

$$x_B = \bar{A}_B^{-1}b + \frac{1}{\lambda}\bar{A}_B^{-1} \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$$

(The other x -values are zero)

The only reason that this basic feasible solution is not feasible for $\bar{P}$ is that one of the basic variables becomes negative.

Hence, there exists i with

$$(\bar{A}_B^{-1}b)_i < 0 \leq (\bar{A}_B^{-1}b)_i + \frac{1}{\lambda}(\bar{A}_B^{-1}\vec{1})_i$$

Making P full-dimensional

By Cramers rule we get

$$(\bar{A}_B^{-1}b)_i < 0 \quad \Rightarrow \quad (\bar{A}_B^{-1}b)_i \leq -\frac{1}{\det(\bar{A}_B)} \leq -1/\delta$$

and

$$(\bar{A}_B^{-1}\vec{1})_i \leq \det(\bar{A}_B^j) \leq \delta ,$$

where $\bar{A}_B^j$ is obtained by replacing the j -th column of $\bar{A}_B$ by $\vec{1}$.

But then

$$(\bar{A}_B^{-1}b)_i + \frac{1}{\lambda}(\bar{A}_B^{-1}\vec{1})_i \leq -1/\delta + \delta/\lambda < 0 ,$$

as we chose $\lambda = \delta^2 + 1$. Contradiction.

Making P full-dimensional

By Cramers rule we get

$$(\bar{A}_B^{-1}b)_i < 0 \quad \Rightarrow \quad (\bar{A}_B^{-1}b)_i \leq -\frac{1}{\det(\bar{A}_B)} \leq -1/\delta$$

and

$$(\bar{A}_B^{-1}\vec{1})_i \leq \det(\bar{A}_B^j) \leq \delta ,$$

where $\bar{A}_B^j$ is obtained by replacing the j -th column of $\bar{A}_B$ by $\vec{1}$.

But then

$$(\bar{A}_B^{-1}b)_i + \frac{1}{\lambda}(\bar{A}_B^{-1}\vec{1})_i \leq -1/\delta + \delta/\lambda < 0 ,$$

as we chose $\lambda = \delta^2 + 1$. Contradiction.

Lemma 54

If P_λ is feasible then it contains a ball of radius $r := 1/\delta^3$. This has a volume of at least $r^n \text{vol}(B(0, 1)) = \frac{1}{\delta^{3n}} \text{vol}(B(0, 1))$.

Lemma 54

If P_λ is feasible then it contains a ball of radius $r := 1/\delta^3$. This has a volume of at least $r^n \text{vol}(B(0, 1)) = \frac{1}{\delta^{3n}} \text{vol}(B(0, 1))$.

Proof:

If P_λ feasible then also P . Let x be feasible for P .

Lemma 54

If P_λ is feasible then it contains a ball of radius $r := 1/\delta^3$. This has a volume of at least $r^n \text{vol}(B(0, 1)) = \frac{1}{\delta^{3n}} \text{vol}(B(0, 1))$.

Proof:

If P_λ feasible then also P . Let x be feasible for P .

This means $Ax \leq b$.

Lemma 54

If P_λ is feasible then it contains a ball of radius $r := 1/\delta^3$. This has a volume of at least $r^n \text{vol}(B(0, 1)) = \frac{1}{\delta^{3n}} \text{vol}(B(0, 1))$.

Proof:

If P_λ feasible then also P . Let x be feasible for P .

This means $Ax \leq b$.

Let $\vec{\ell}$ with $\|\vec{\ell}\| \leq r$. Then

$$(A(x + \vec{\ell}))_i$$

Lemma 54

If P_λ is feasible then it contains a ball of radius $r := 1/\delta^3$. This has a volume of at least $r^n \text{vol}(B(0, 1)) = \frac{1}{\delta^{3n}} \text{vol}(B(0, 1))$.

Proof:

If P_λ feasible then also P . Let x be feasible for P .

This means $Ax \leq b$.

Let $\vec{\ell}$ with $\|\vec{\ell}\| \leq r$. Then

$$(A(x + \vec{\ell}))_i = (Ax)_i + (A\vec{\ell})_i$$

Lemma 54

If P_λ is feasible then it contains a ball of radius $r := 1/\delta^3$. This has a volume of at least $r^n \text{vol}(B(0, 1)) = \frac{1}{\delta^{3n}} \text{vol}(B(0, 1))$.

Proof:

If P_λ feasible then also P . Let x be feasible for P .

This means $Ax \leq b$.

Let $\vec{\ell}$ with $\|\vec{\ell}\| \leq r$. Then

$$(A(x + \vec{\ell}))_i = (Ax)_i + (A\vec{\ell})_i \leq b_i + \vec{a}_i^T \vec{\ell}$$

Lemma 54

If P_λ is feasible then it contains a ball of radius $r := 1/\delta^3$. This has a volume of at least $r^n \text{vol}(B(0, 1)) = \frac{1}{\delta^{3n}} \text{vol}(B(0, 1))$.

Proof:

If P_λ feasible then also P . Let x be feasible for P .

This means $Ax \leq b$.

Let $\vec{\ell}$ with $\|\vec{\ell}\| \leq r$. Then

$$\begin{aligned}(A(x + \vec{\ell}))_i &= (Ax)_i + (A\vec{\ell})_i \leq b_i + \vec{a}_i^T \vec{\ell} \\ &\leq b_i + \|\vec{a}_i\| \cdot \|\vec{\ell}\|\end{aligned}$$

Lemma 54

If P_λ is feasible then it contains a ball of radius $r := 1/\delta^3$. This has a volume of at least $r^n \text{vol}(B(0, 1)) = \frac{1}{\delta^{3n}} \text{vol}(B(0, 1))$.

Proof:

If P_λ feasible then also P . Let x be feasible for P .

This means $Ax \leq b$.

Let $\vec{\ell}$ with $\|\vec{\ell}\| \leq r$. Then

$$\begin{aligned}(A(x + \vec{\ell}))_i &= (Ax)_i + (A\vec{\ell})_i \leq b_i + \vec{a}_i^T \vec{\ell} \\ &\leq b_i + \|\vec{a}_i\| \cdot \|\vec{\ell}\| \leq b_i + \sqrt{n} \cdot 2^{\langle a_{\max} \rangle} \cdot r\end{aligned}$$

Lemma 54

If P_λ is feasible then it contains a ball of radius $r := 1/\delta^3$. This has a volume of at least $r^n \text{vol}(B(0, 1)) = \frac{1}{\delta^{3n}} \text{vol}(B(0, 1))$.

Proof:

If P_λ feasible then also P . Let x be feasible for P .

This means $Ax \leq b$.

Let $\vec{\ell}$ with $\|\vec{\ell}\| \leq r$. Then

$$\begin{aligned}(A(x + \vec{\ell}))_i &= (Ax)_i + (A\vec{\ell})_i \leq b_i + \vec{a}_i^T \vec{\ell} \\ &\leq b_i + \|\vec{a}_i\| \cdot \|\vec{\ell}\| \leq b_i + \sqrt{n} \cdot 2^{\langle a_{\max} \rangle} \cdot r \\ &\leq b_i + \frac{\sqrt{n} \cdot 2^{\langle a_{\max} \rangle}}{\delta^3}\end{aligned}$$

Lemma 54

If P_λ is feasible then it contains a ball of radius $r := 1/\delta^3$. This has a volume of at least $r^n \text{vol}(B(0, 1)) = \frac{1}{\delta^{3n}} \text{vol}(B(0, 1))$.

Proof:

If P_λ feasible then also P . Let x be feasible for P .

This means $Ax \leq b$.

Let $\vec{\ell}$ with $\|\vec{\ell}\| \leq r$. Then

$$\begin{aligned}(A(x + \vec{\ell}))_i &= (Ax)_i + (A\vec{\ell})_i \leq b_i + \vec{a}_i^T \vec{\ell} \\ &\leq b_i + \|\vec{a}_i\| \cdot \|\vec{\ell}\| \leq b_i + \sqrt{n} \cdot 2^{\langle a_{\max} \rangle} \cdot r \\ &\leq b_i + \frac{\sqrt{n} \cdot 2^{\langle a_{\max} \rangle}}{\delta^3} \leq b_i + \frac{1}{\delta^2 + 1} \leq b_i + \frac{1}{\lambda}\end{aligned}$$

Lemma 54

If P_λ is feasible then it contains a ball of radius $r := 1/\delta^3$. This has a volume of at least $r^n \text{vol}(B(0, 1)) = \frac{1}{\delta^{3n}} \text{vol}(B(0, 1))$.

Proof:

If P_λ feasible then also P . Let x be feasible for P .

This means $Ax \leq b$.

Let $\vec{\ell}$ with $\|\vec{\ell}\| \leq r$. Then

$$\begin{aligned}(A(x + \vec{\ell}))_i &= (Ax)_i + (A\vec{\ell})_i \leq b_i + \vec{a}_i^T \vec{\ell} \\ &\leq b_i + \|\vec{a}_i\| \cdot \|\vec{\ell}\| \leq b_i + \sqrt{n} \cdot 2^{\langle a_{\max} \rangle} \cdot r \\ &\leq b_i + \frac{\sqrt{n} \cdot 2^{\langle a_{\max} \rangle}}{\delta^3} \leq b_i + \frac{1}{\delta^2 + 1} \leq b_i + \frac{1}{\lambda}\end{aligned}$$

Hence, $x + \vec{\ell}$ is feasible for P_λ which proves the lemma.

How many iterations do we need until the volume becomes too small?

How many iterations do we need until the volume becomes too small?

$$e^{-\frac{i}{2(n+1)}} \cdot \text{vol}(B(0, R)) < \text{vol}(B(0, r))$$

How many iterations do we need until the volume becomes too small?

$$e^{-\frac{i}{2(n+1)}} \cdot \text{vol}(B(0, R)) < \text{vol}(B(0, r))$$

Hence,

i

How many iterations do we need until the volume becomes too small?

$$e^{-\frac{i}{2(n+1)}} \cdot \text{vol}(B(0, R)) < \text{vol}(B(0, r))$$

Hence,

$$i > 2(n+1) \ln \left(\frac{\text{vol}(B(0, R))}{\text{vol}(B(0, r))} \right)$$

How many iterations do we need until the volume becomes too small?

$$e^{-\frac{i}{2(n+1)}} \cdot \text{vol}(B(0, R)) < \text{vol}(B(0, r))$$

Hence,

$$\begin{aligned} i &> 2(n+1) \ln \left(\frac{\text{vol}(B(0, R))}{\text{vol}(B(0, r))} \right) \\ &= 2(n+1) \ln \left(n^n \delta^n \cdot \delta^{3n} \right) \end{aligned}$$

How many iterations do we need until the volume becomes too small?

$$e^{-\frac{i}{2(n+1)}} \cdot \text{vol}(B(0, R)) < \text{vol}(B(0, r))$$

Hence,

$$\begin{aligned} i &> 2(n+1) \ln \left(\frac{\text{vol}(B(0, R))}{\text{vol}(B(0, r))} \right) \\ &= 2(n+1) \ln \left(n^n \delta^n \cdot \delta^{3n} \right) \\ &= 8n(n+1) \ln(\delta) + 2(n+1)n \ln(n) \end{aligned}$$

How many iterations do we need until the volume becomes too small?

$$e^{-\frac{i}{2(n+1)}} \cdot \text{vol}(B(0, R)) < \text{vol}(B(0, r))$$

Hence,

$$\begin{aligned} i &> 2(n+1) \ln \left(\frac{\text{vol}(B(0, R))}{\text{vol}(B(0, r))} \right) \\ &= 2(n+1) \ln \left(n^n \delta^n \cdot \delta^{3n} \right) \\ &= 8n(n+1) \ln(\delta) + 2(n+1)n \ln(n) \\ &= \mathcal{O}(\text{poly}(n) \cdot L) \end{aligned}$$

Algorithm 1 ellipsoid-algorithm

```
1: input: point  $c \in \mathbb{R}^n$ , convex set  $K \subseteq \mathbb{R}^n$ , radii  $R$  and  $r$   
2:         with  $K \subseteq B(c, R)$ , and  $B(x, r) \subseteq K$  for some  $x$   
3: output: point  $x \in K$  or “ $K$  is empty”  
4:  $Q \leftarrow \text{diag}(R^2, \dots, R^2)$  // i.e.,  $L = \text{diag}(R, \dots, R)$   
5: repeat  
6:     if  $c \in K$  then return  $c$   
7:     else  
8:         choose a violated hyperplane  $a$   
9:         
$$c \leftarrow c - \frac{1}{n+1} \frac{Qa}{\sqrt{a^T Q a}}$$
  
10:        
$$Q \leftarrow \frac{n^2}{n^2 - 1} \left( Q - \frac{2}{n+1} \frac{Qa a^T Q}{a^T Q a} \right)$$
  
11:     endif  
12: until  $\det(Q) \leq r^{2n}$  // i.e.,  $\det(L) \leq r^n$   
13: return “ $K$  is empty”
```

Separation Oracle

Let $K \subseteq \mathbb{R}^n$ be a convex set. A separation oracle for K is an algorithm A that gets as input a point $x \in \mathbb{R}^n$ and either

- ▶ certifies that $x \in K$,
- ▶ or finds a hyperplane separating x from K .

We will usually assume that A is a polynomial-time algorithm.

In order to find a point in K we need

- ▶ a guarantee that a ball of radius r is contained in K , and
- ▶ an initial ball $B(x, R)$ with radius R that contains K , and a separation oracle for K .

The Ellipsoid algorithm requires $\mathcal{O}(\text{poly}(n) \cdot \log(R/r))$ iterations.
Each iteration is polytime for a polynomial-time Separation oracle.

Separation Oracle

Let $K \subseteq \mathbb{R}^n$ be a convex set. A separation oracle for K is an algorithm A that gets as input a point $x \in \mathbb{R}^n$ and either

- ▶ certifies that $x \in K$,
- ▶ or finds a hyperplane separating x from K .

We will usually assume that A is a polynomial-time algorithm.

In order to find a point in K we need

- ▶ a guarantee that a ball of radius r is contained in K ,
- ▶ an initial ball B of radius R that contains K ,
- ▶ a separation oracle.

The Ellipsoid algorithm requires $\mathcal{O}(\text{poly}(n) \cdot \log(R/r))$ iterations.
Each iteration is polytime for a polynomial-time Separation oracle.

Separation Oracle

Let $K \subseteq \mathbb{R}^n$ be a convex set. A separation oracle for K is an algorithm A that gets as input a point $x \in \mathbb{R}^n$ and either

- ▶ certifies that $x \in K$,
- ▶ or finds a hyperplane separating x from K .

We will usually assume that A is a polynomial-time algorithm.

In order to find a point in K we need

- ▶ to be guaranteed that a ball of radius r is contained in K (we call this an *inner ball*),
- ▶ an initial ball B of radius R that contains K (we call this a *separating oracle*).

The Ellipsoid algorithm requires $\mathcal{O}(\text{poly}(n) \cdot \log(R/r))$ iterations.
Each iteration is polytime for a polynomial-time Separation oracle.

Separation Oracle

Let $K \subseteq \mathbb{R}^n$ be a convex set. A separation oracle for K is an algorithm A that gets as input a point $x \in \mathbb{R}^n$ and either

- ▶ certifies that $x \in K$,
- ▶ or finds a hyperplane separating x from K .

We will usually assume that A is a polynomial-time algorithm.

In order to find a point in K we need

- ▶ a guarantee that a ball of radius r is contained in K ,
- ▶ an initial ball $B(c, R)$ with radius R that contains K ,
- ▶ a separation oracle for K .

The Ellipsoid algorithm requires $\mathcal{O}(\text{poly}(n) \cdot \log(R/r))$ iterations.
Each iteration is polytime for a polynomial-time Separation oracle.

Separation Oracle

Let $K \subseteq \mathbb{R}^n$ be a convex set. A separation oracle for K is an algorithm A that gets as input a point $x \in \mathbb{R}^n$ and either

- ▶ certifies that $x \in K$,
- ▶ or finds a hyperplane separating x from K .

We will usually assume that A is a polynomial-time algorithm.

In order to find a point in K we need

- ▶ a guarantee that a ball of radius r is contained in K ,
- ▶ an initial ball $B(c, R)$ with radius R that contains K ,
- ▶ a separation oracle for K .

The Ellipsoid algorithm requires $\mathcal{O}(\text{poly}(n) \cdot \log(R/r))$ iterations.
Each iteration is polytime for a polynomial-time Separation oracle.

Separation Oracle

Let $K \subseteq \mathbb{R}^n$ be a convex set. A separation oracle for K is an algorithm A that gets as input a point $x \in \mathbb{R}^n$ and either

- ▶ certifies that $x \in K$,
- ▶ or finds a hyperplane separating x from K .

We will usually assume that A is a polynomial-time algorithm.

In order to find a point in K we need

- ▶ a guarantee that a ball of radius r is contained in K ,
- ▶ an initial ball $B(c, R)$ with radius R that contains K ,
- ▶ a separation oracle for K .

The Ellipsoid algorithm requires $\mathcal{O}(\text{poly}(n) \cdot \log(R/r))$ iterations.
Each iteration is polytime for a polynomial-time Separation oracle.

Separation Oracle

Let $K \subseteq \mathbb{R}^n$ be a convex set. A separation oracle for K is an algorithm A that gets as input a point $x \in \mathbb{R}^n$ and either

- ▶ certifies that $x \in K$,
- ▶ or finds a hyperplane separating x from K .

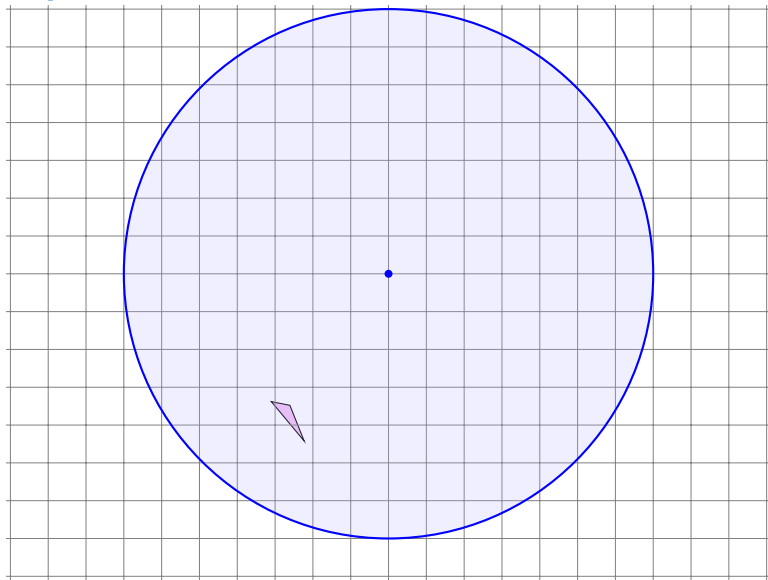
We will usually assume that A is a polynomial-time algorithm.

In order to find a point in K we need

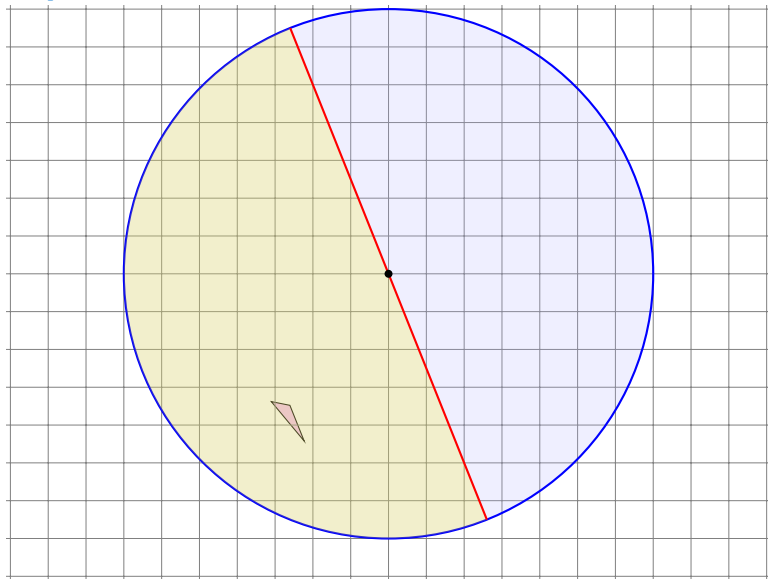
- ▶ a guarantee that a ball of radius r is contained in K ,
- ▶ an initial ball $B(c, R)$ with radius R that contains K ,
- ▶ a separation oracle for K .

The Ellipsoid algorithm requires $\mathcal{O}(\text{poly}(n) \cdot \log(R/r))$ iterations. Each iteration is polytime for a polynomial-time Separation oracle.

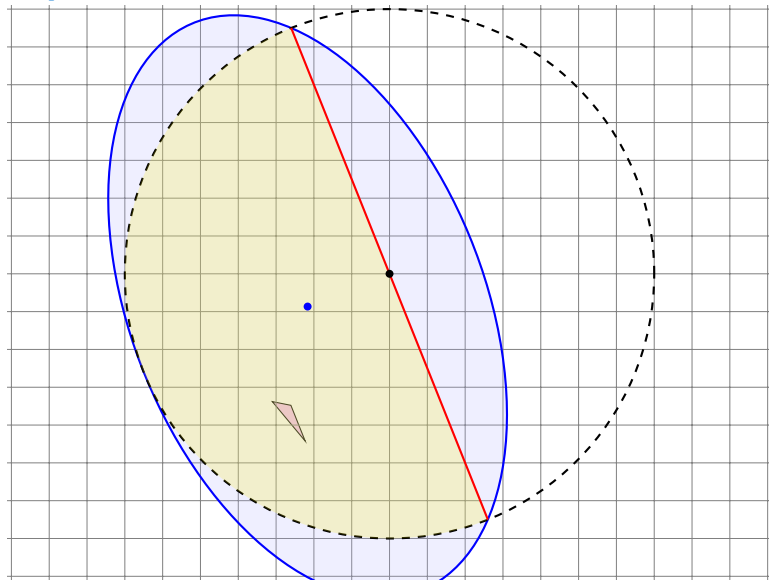
Example



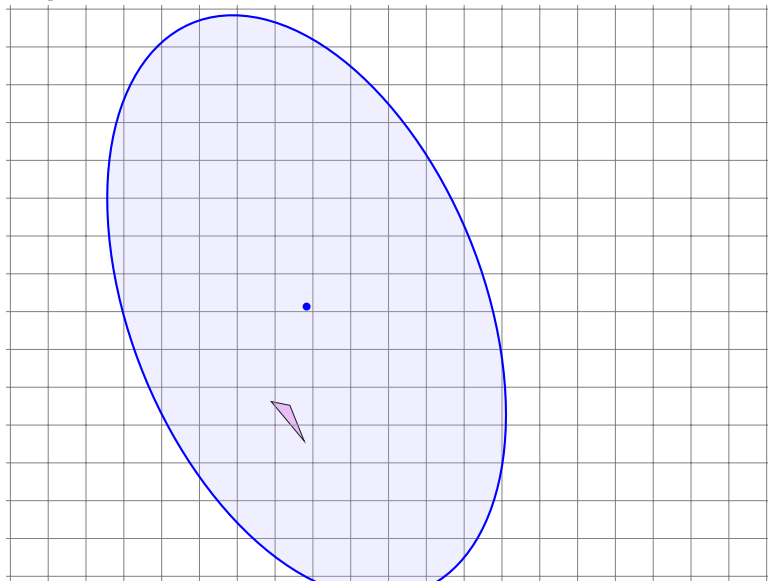
Example



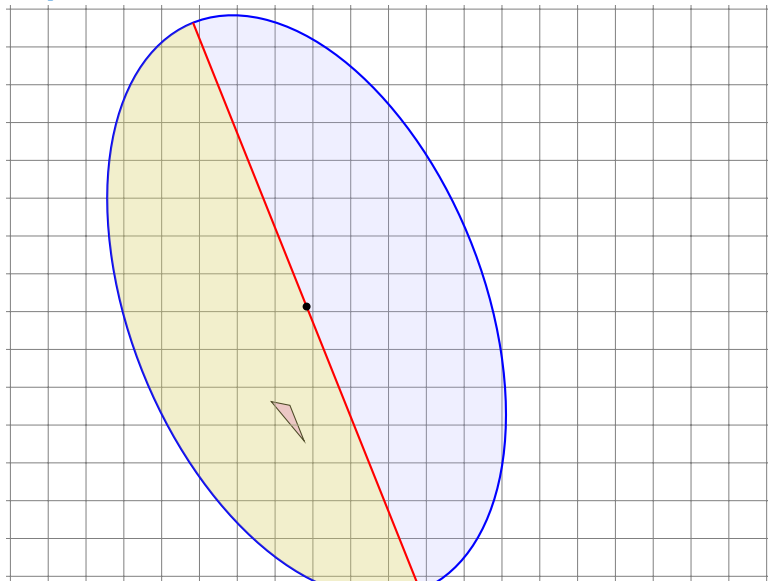
Example



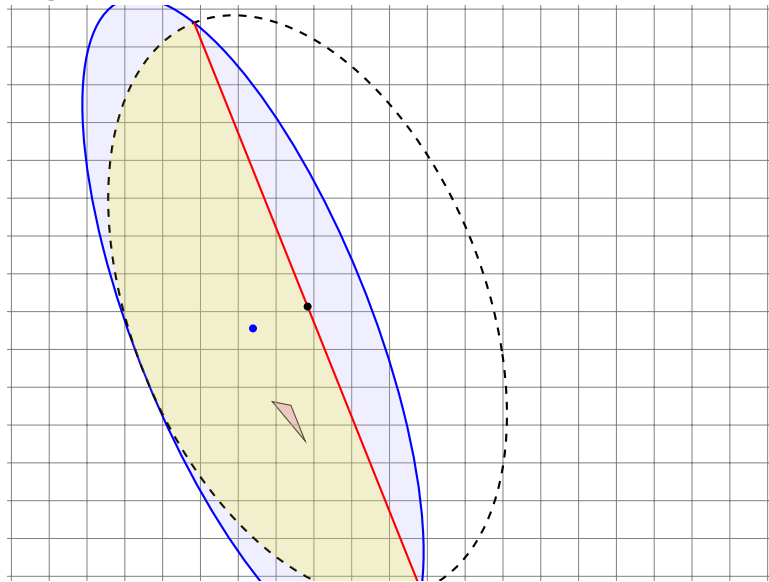
Example



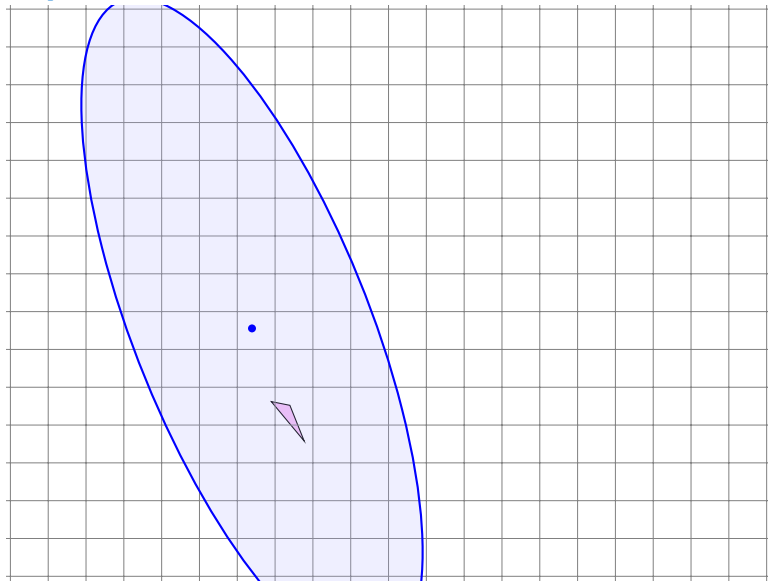
Example



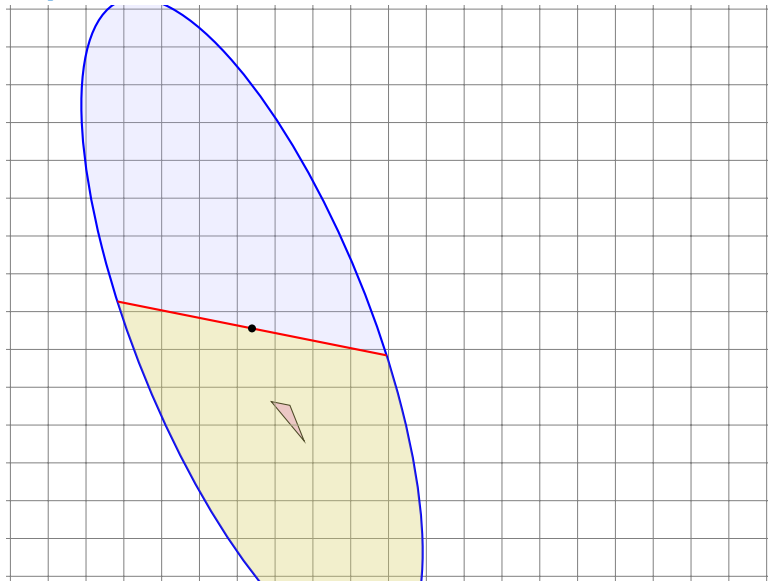
Example



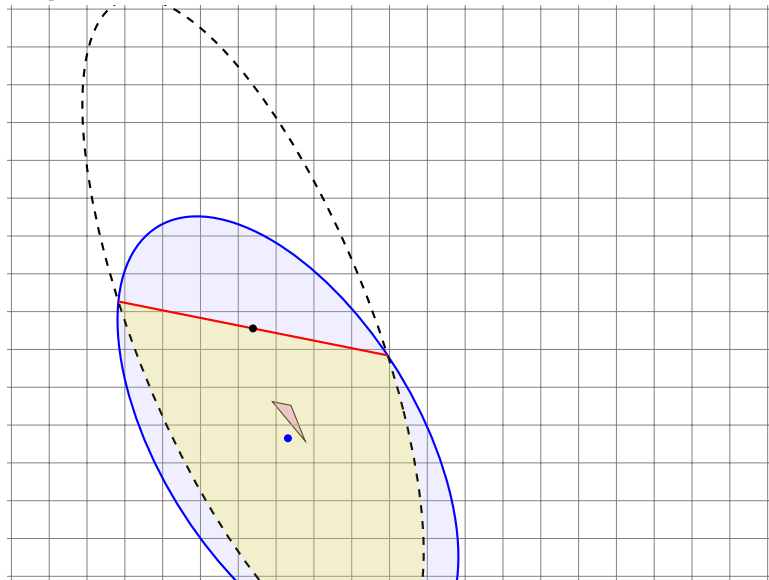
Example



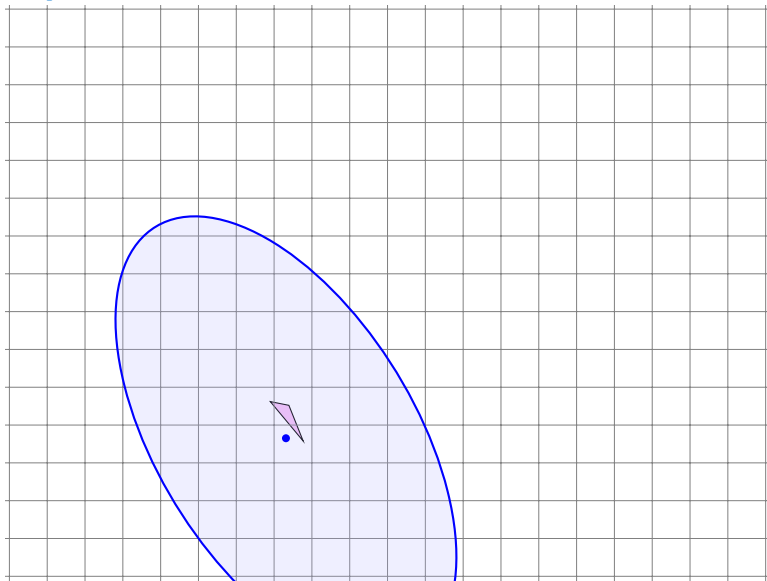
Example



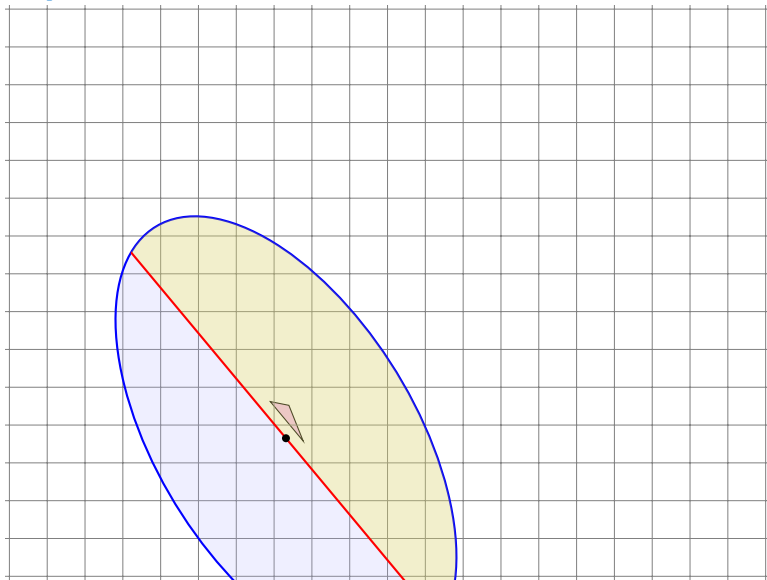
Example



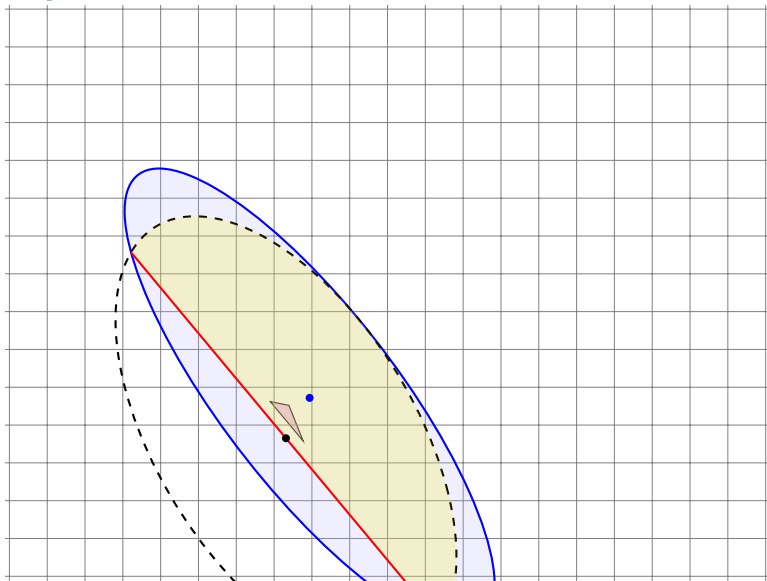
Example



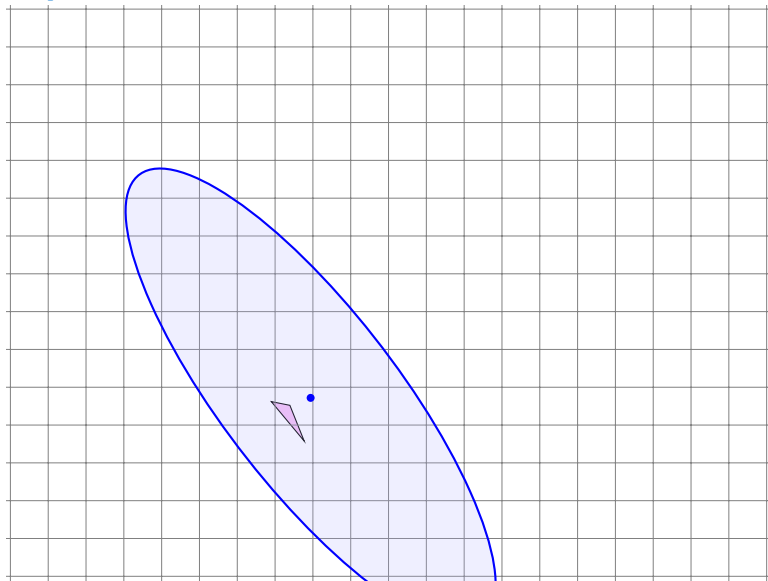
Example



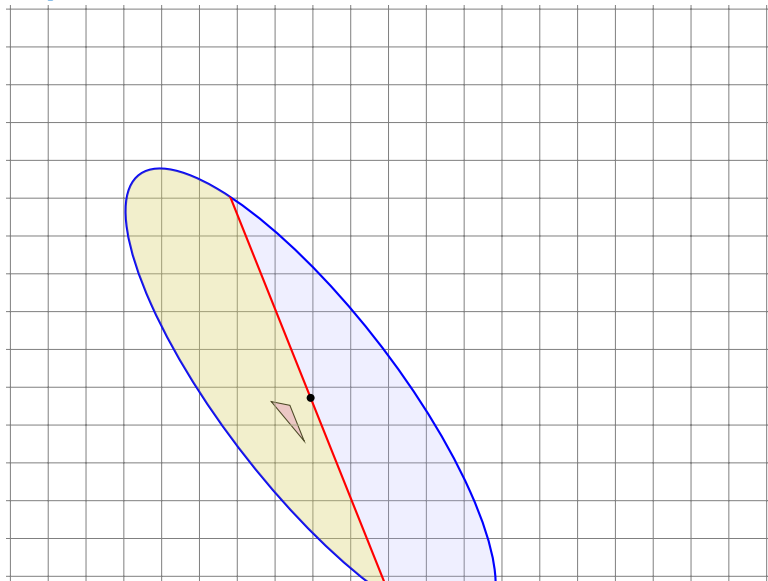
Example



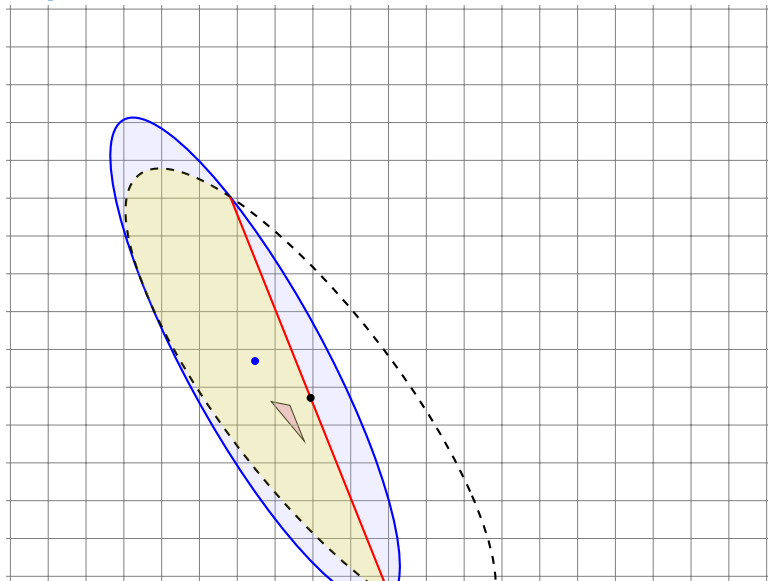
Example



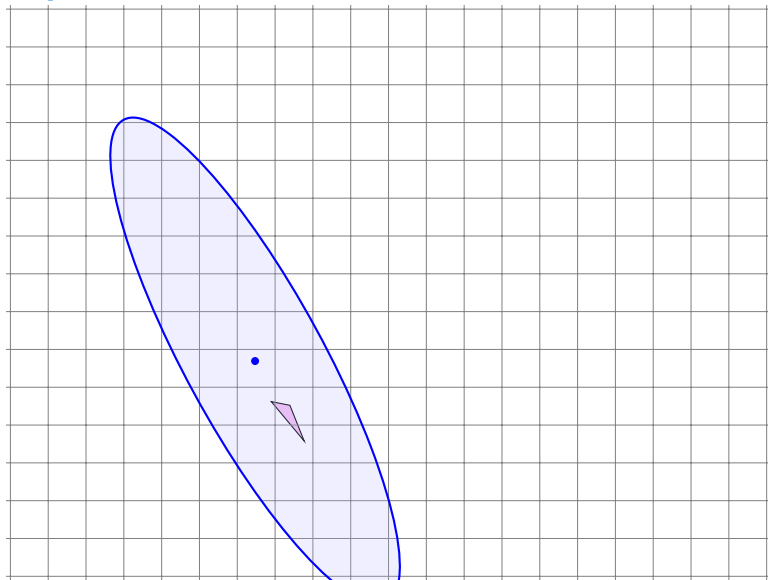
Example



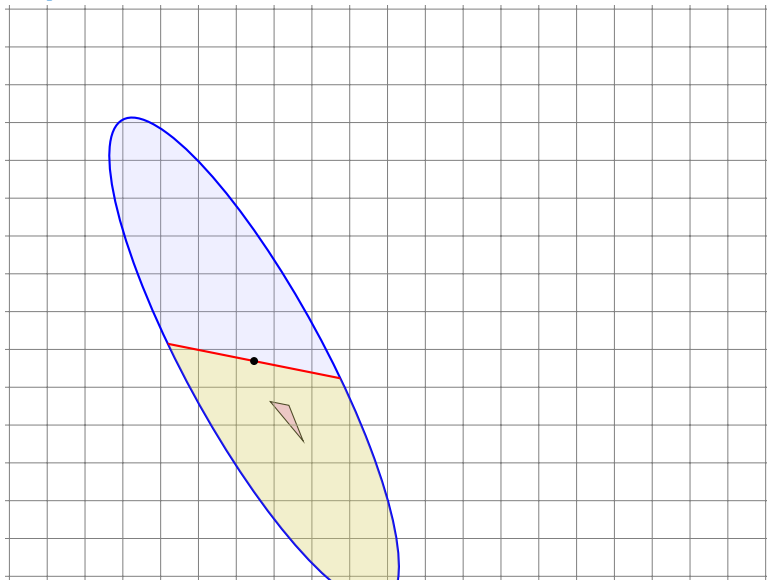
Example



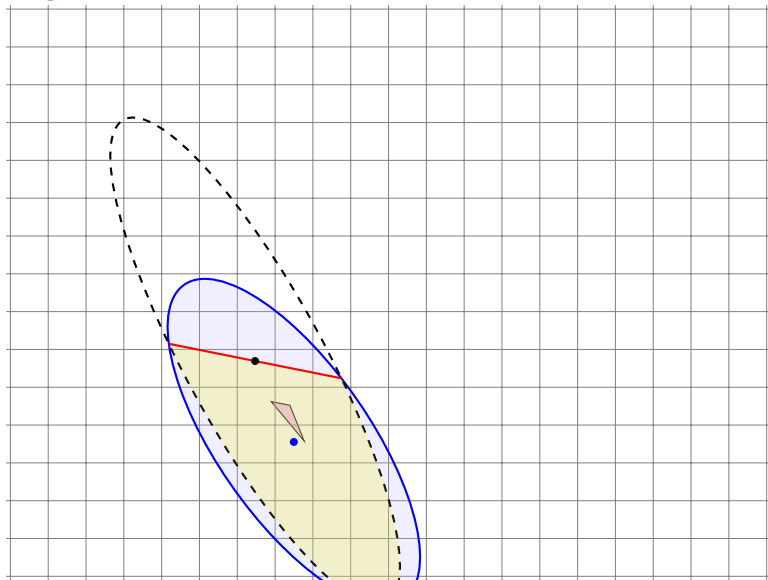
Example



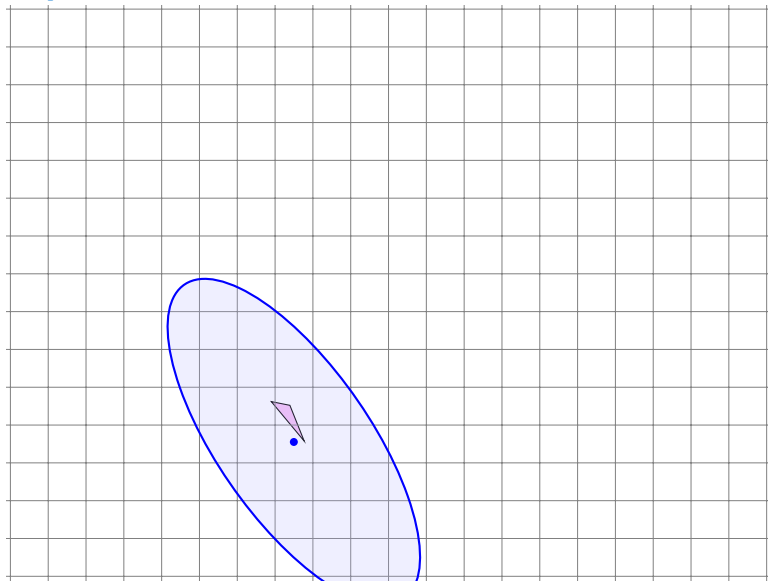
Example



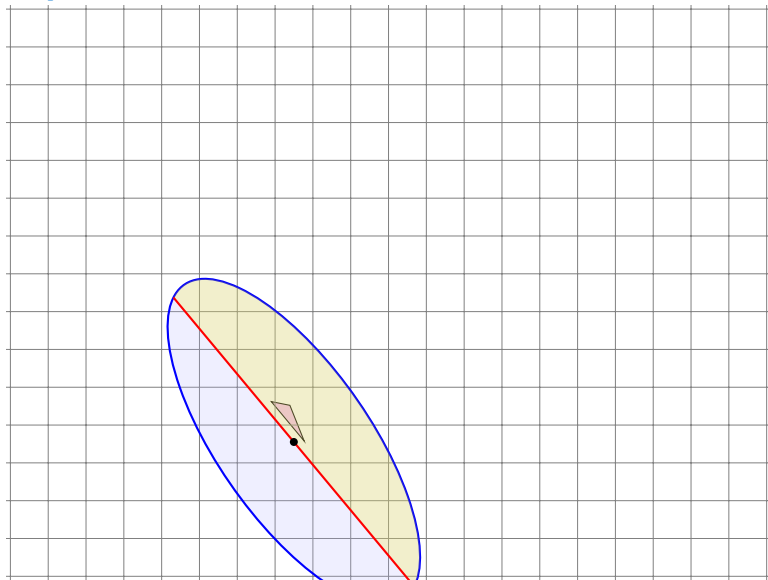
Example



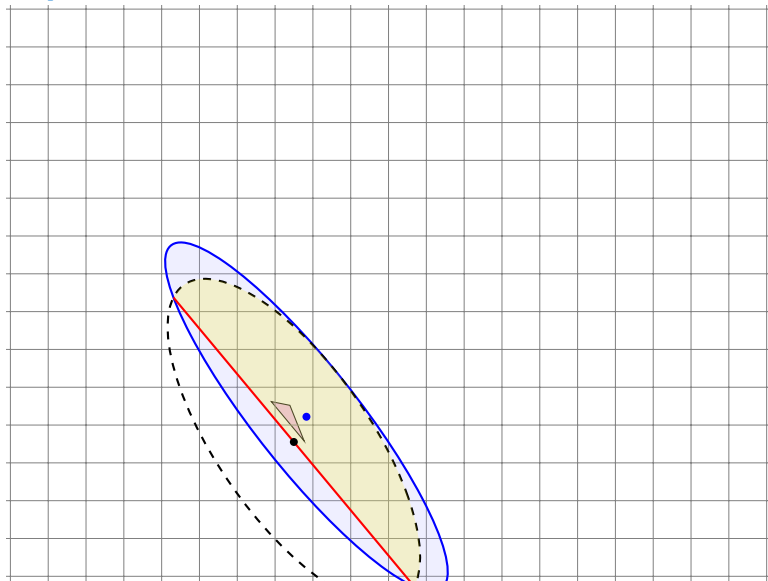
Example



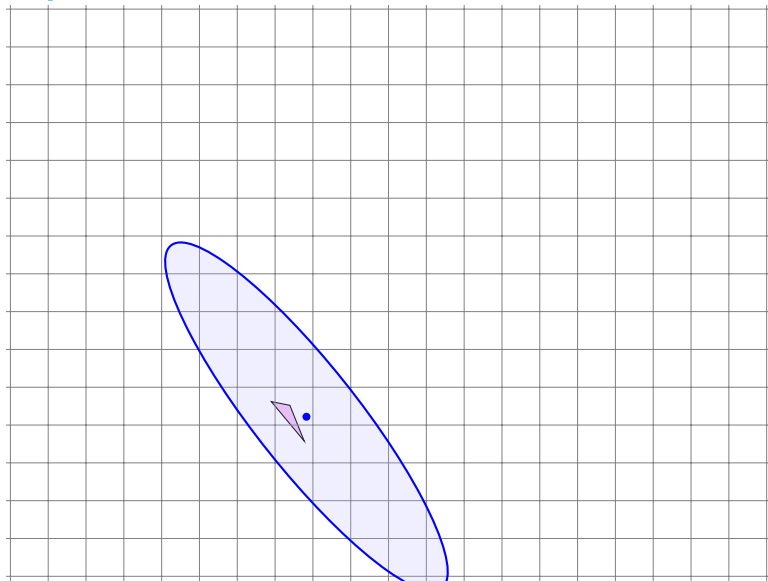
Example



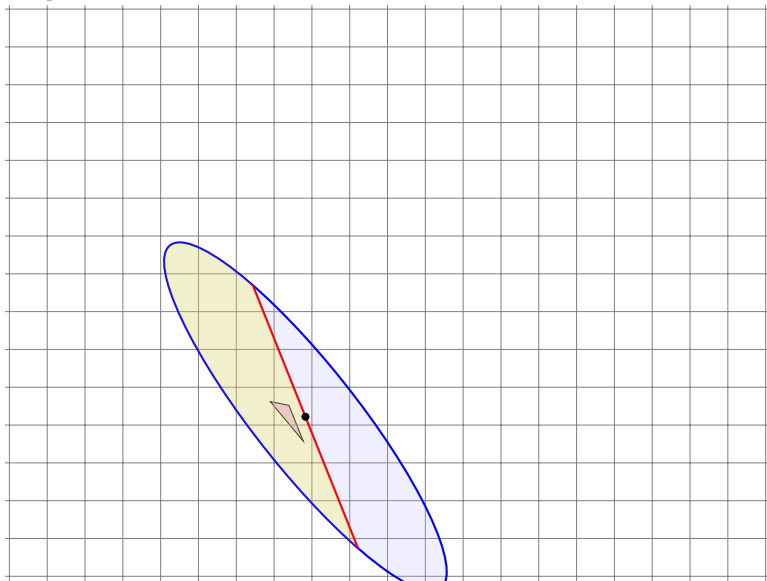
Example



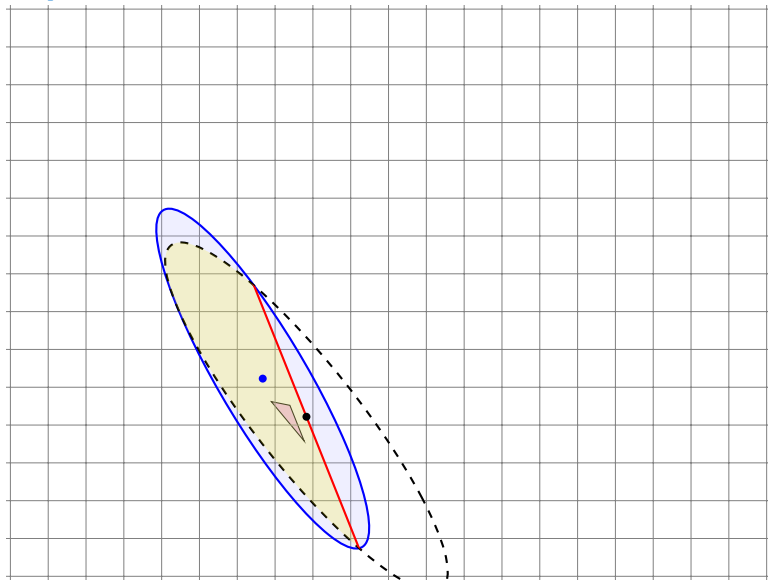
Example



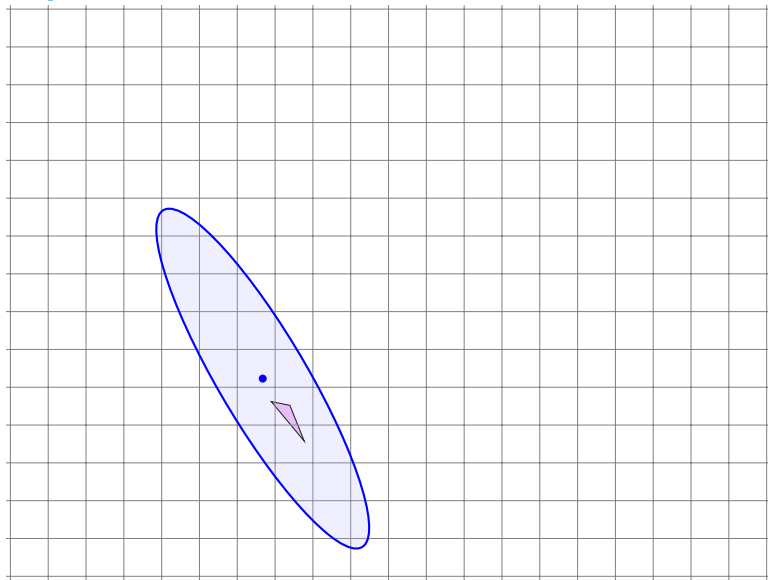
Example



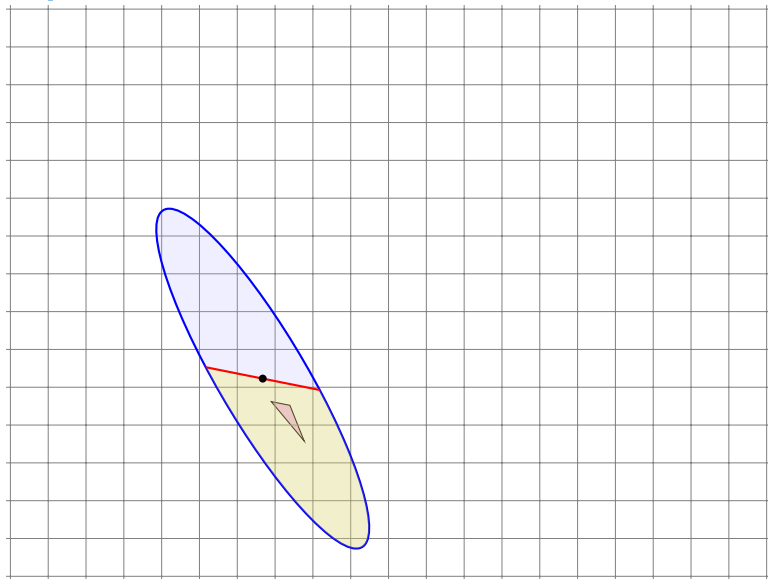
Example



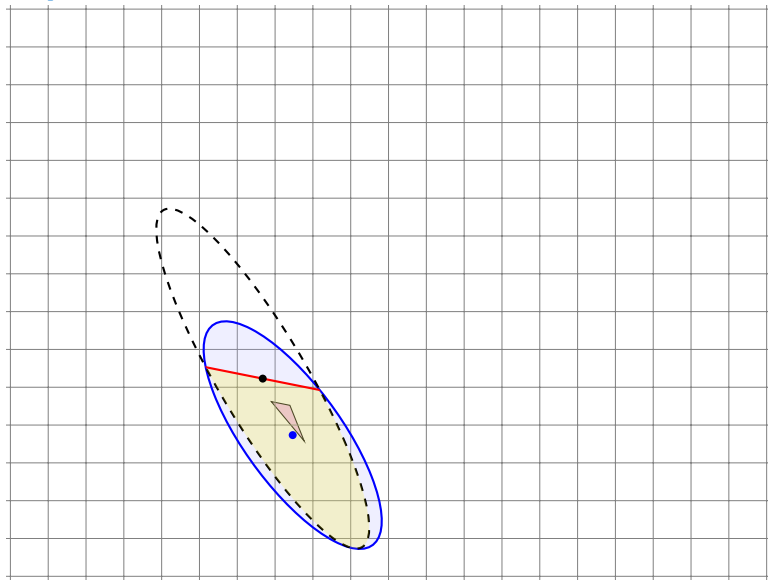
Example



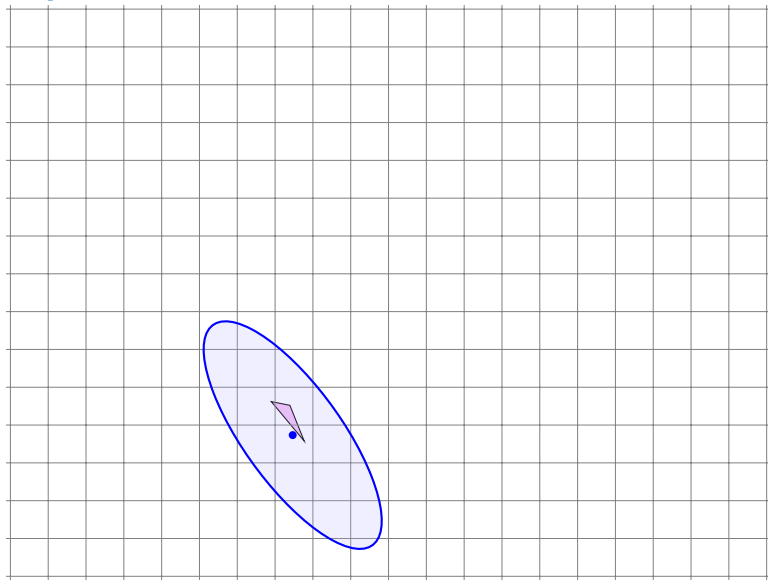
Example



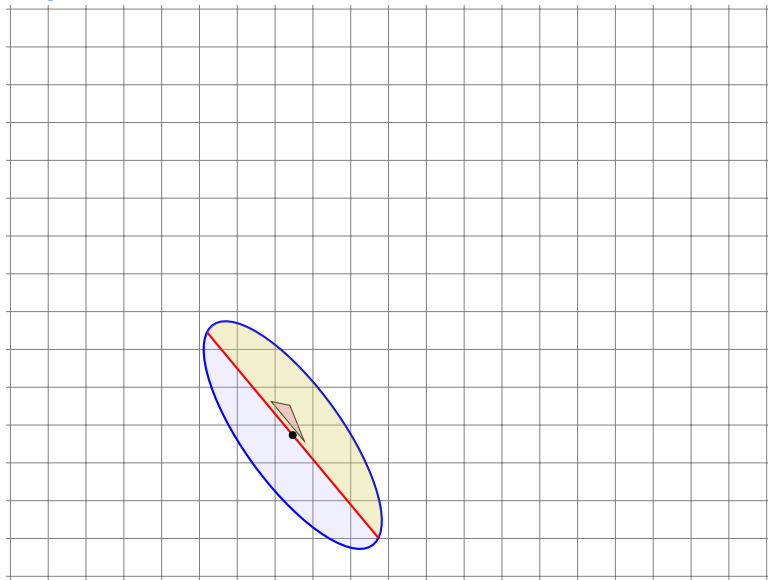
Example



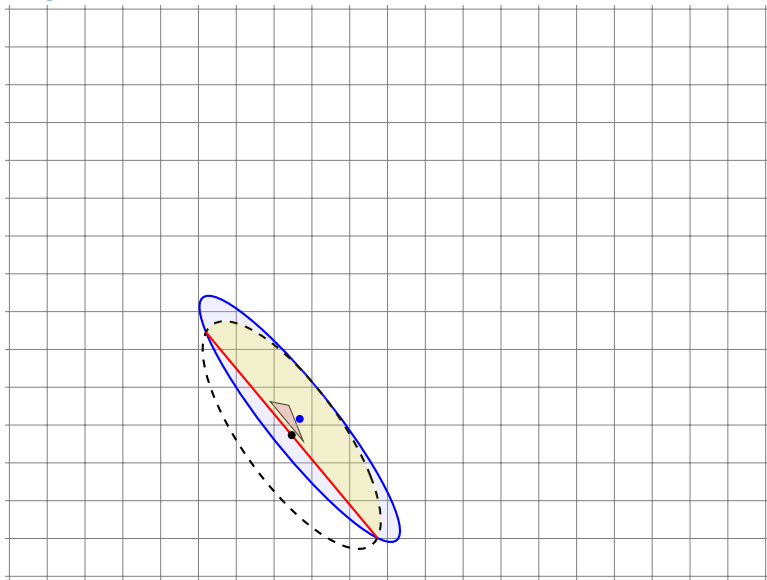
Example



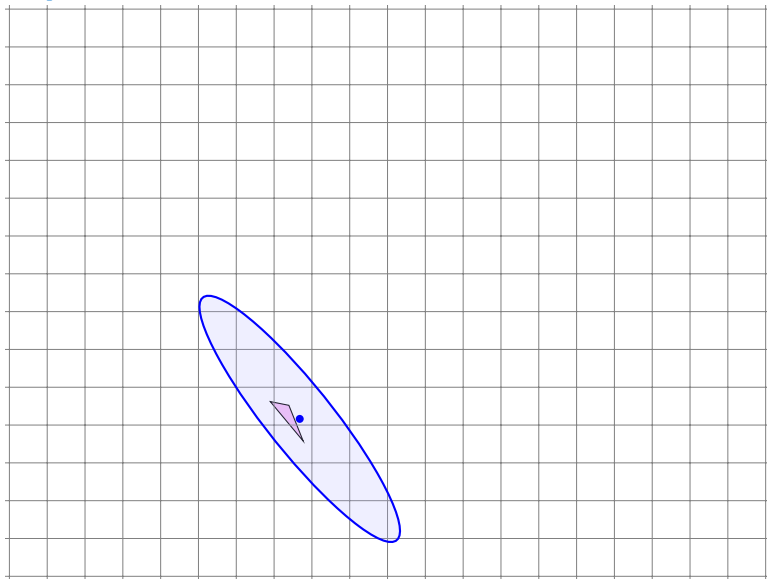
Example



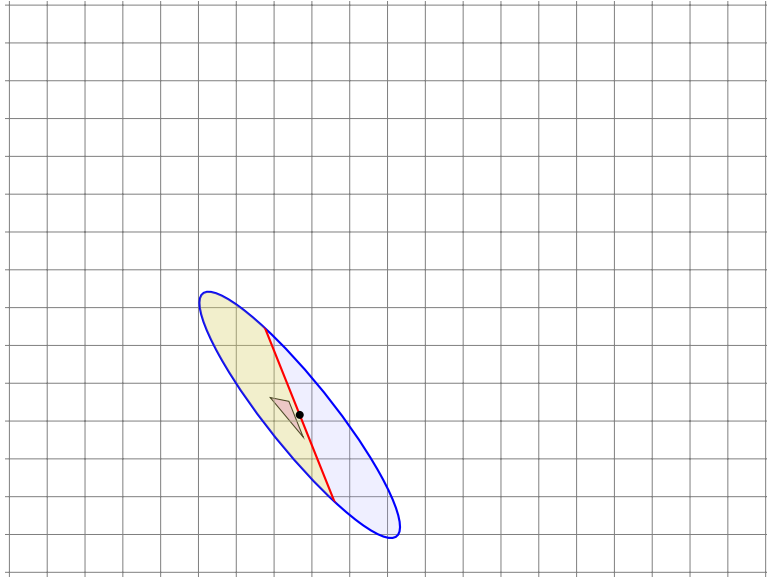
Example



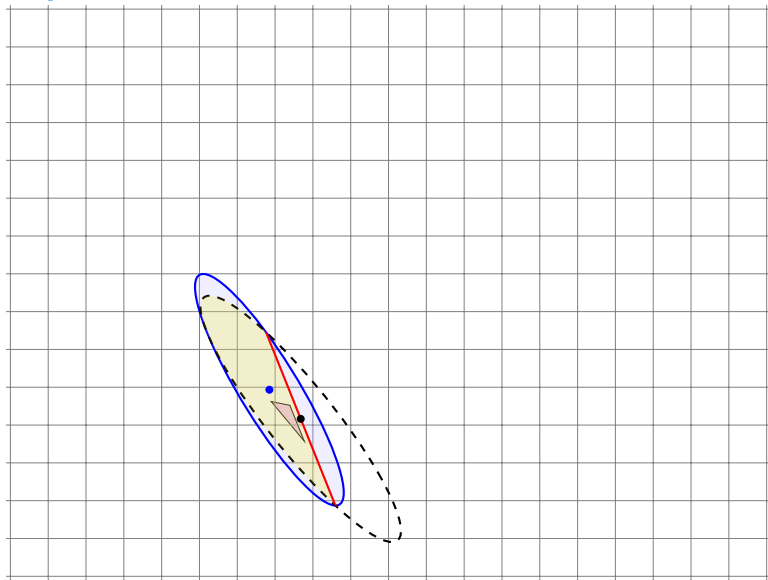
Example



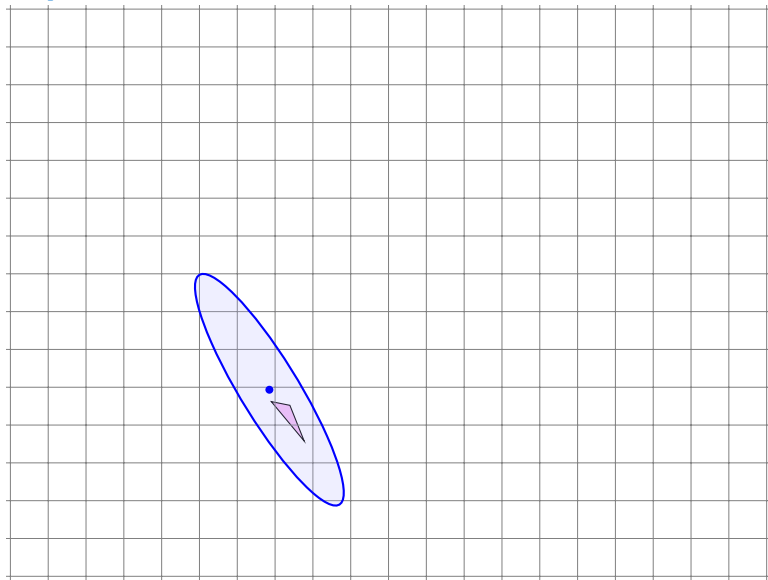
Example



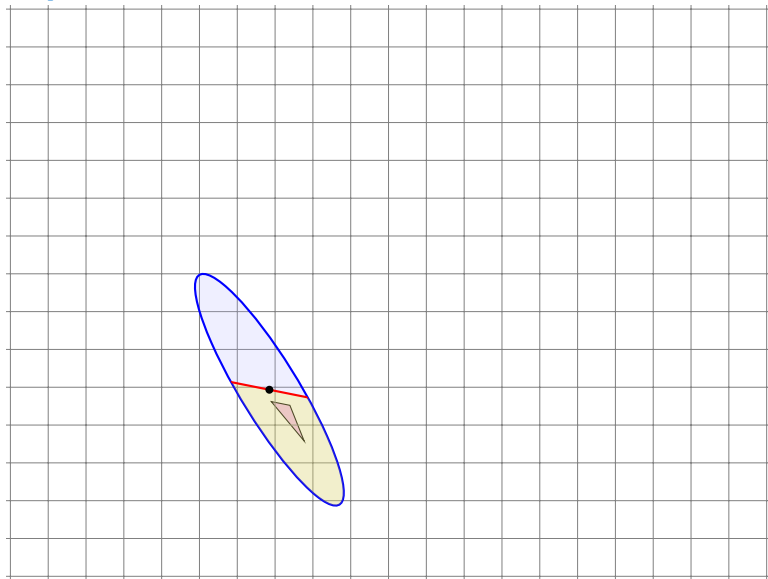
Example



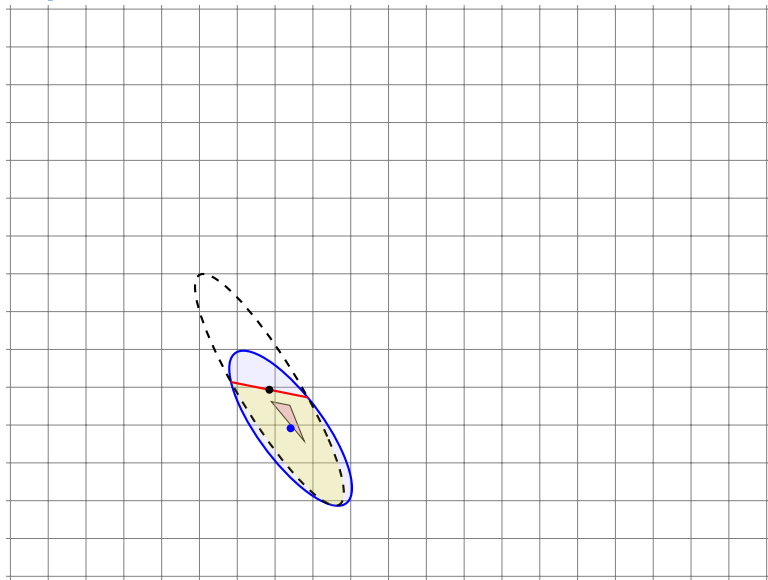
Example



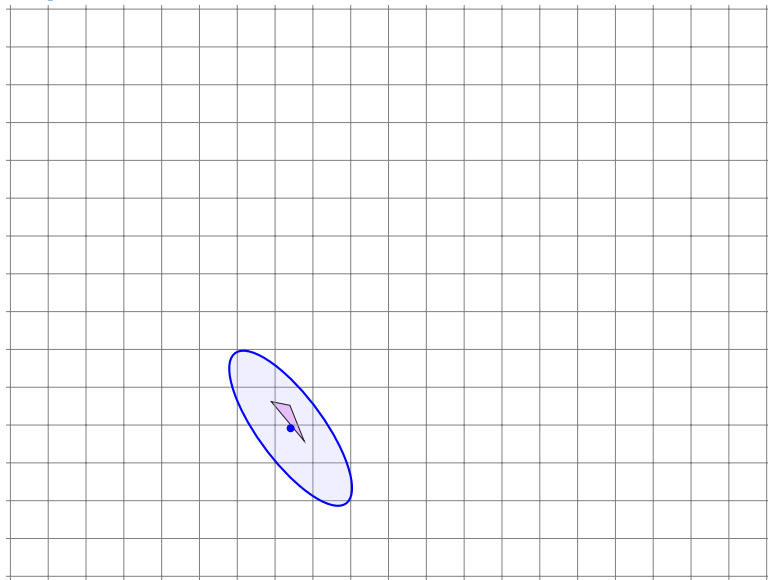
Example



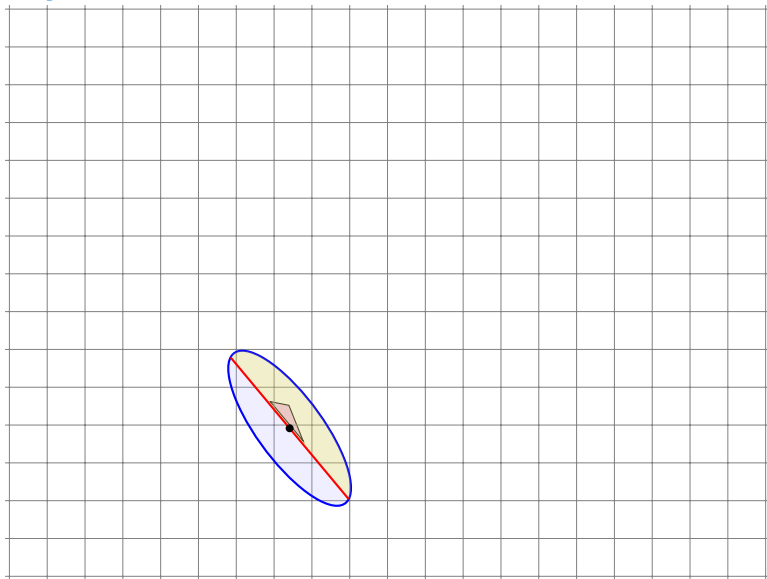
Example



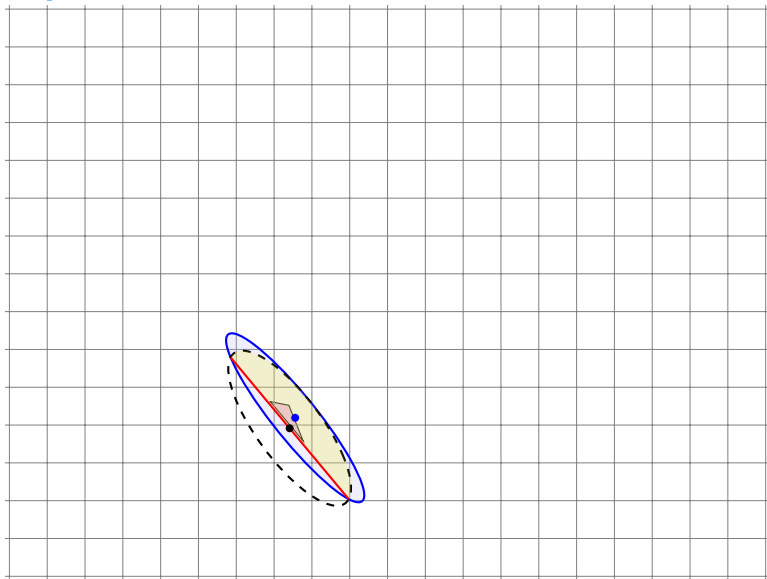
Example



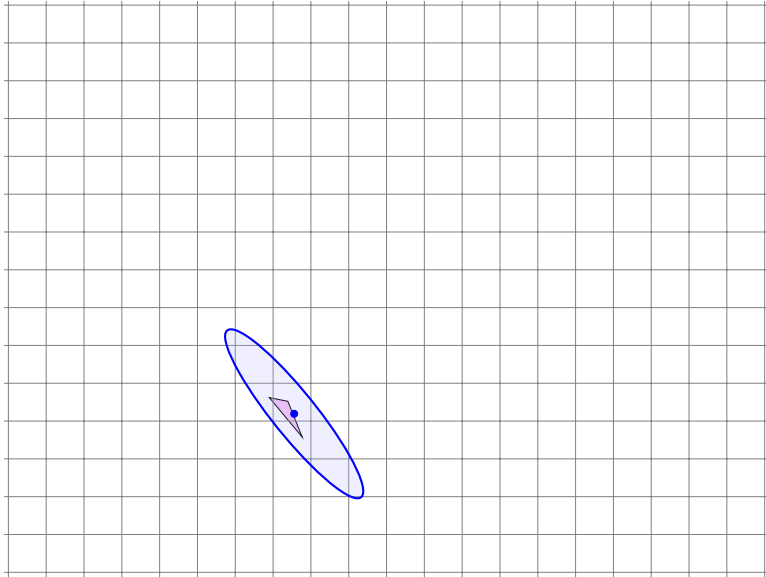
Example



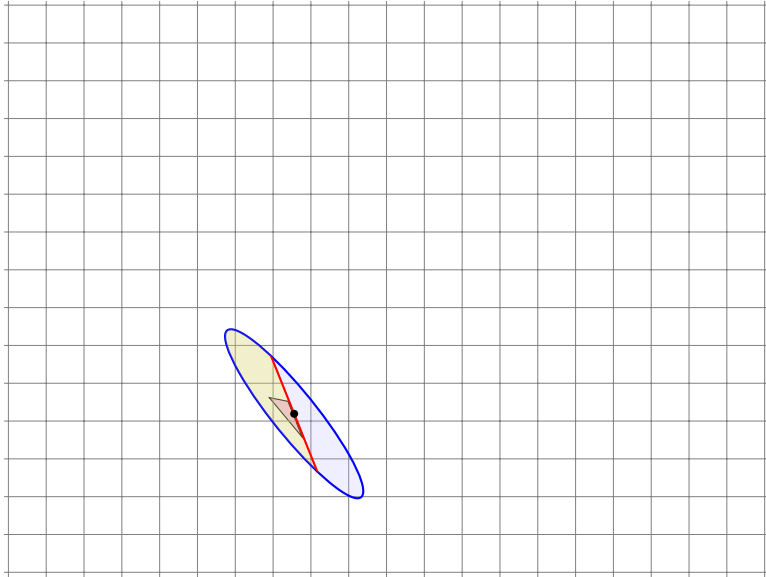
Example



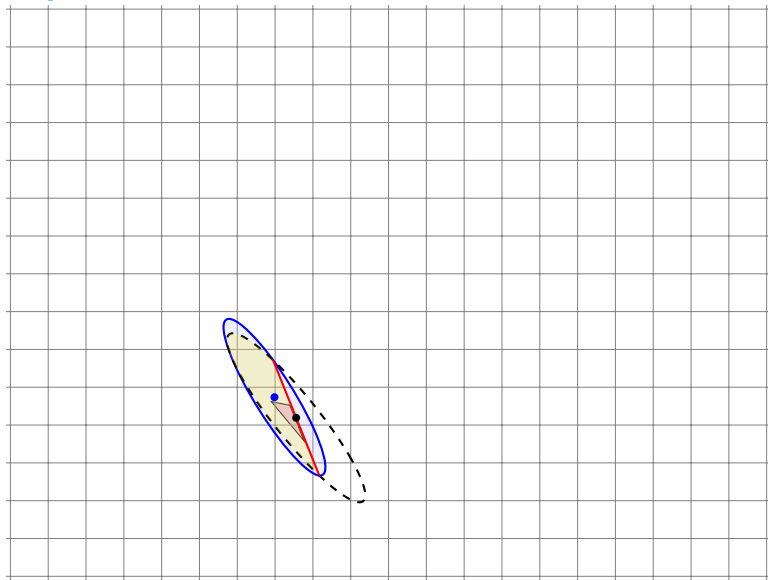
Example



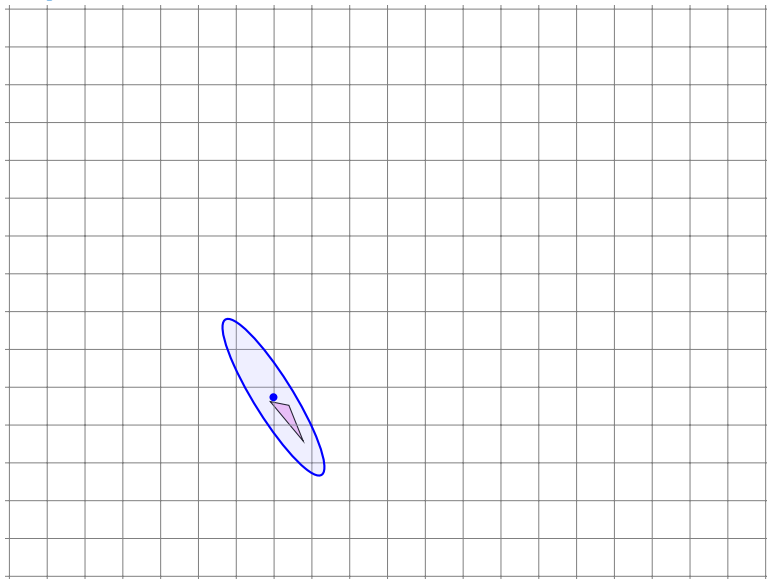
Example



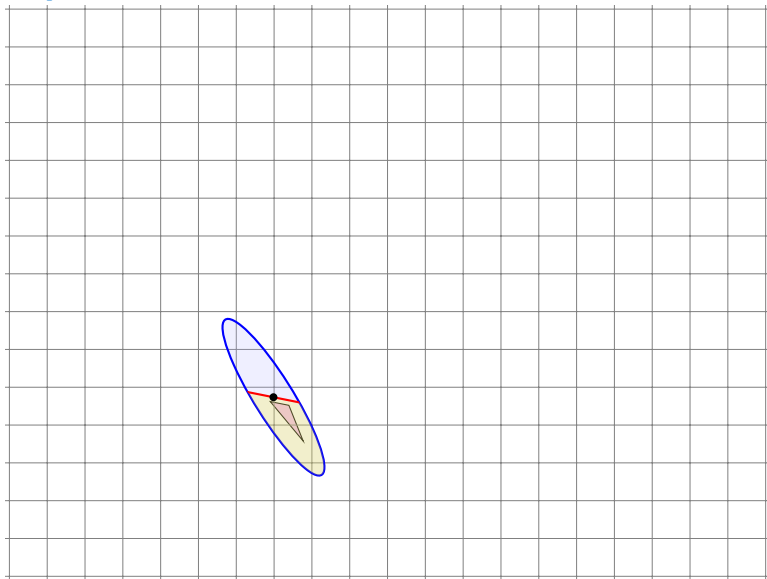
Example



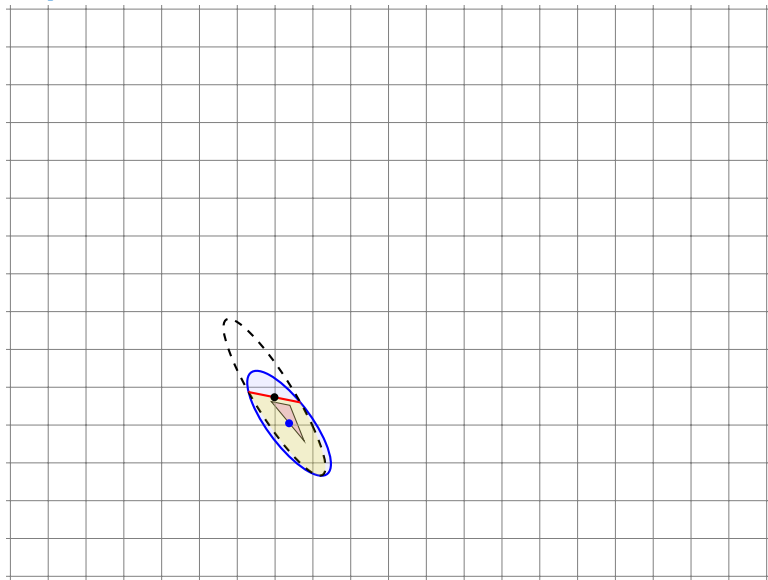
Example



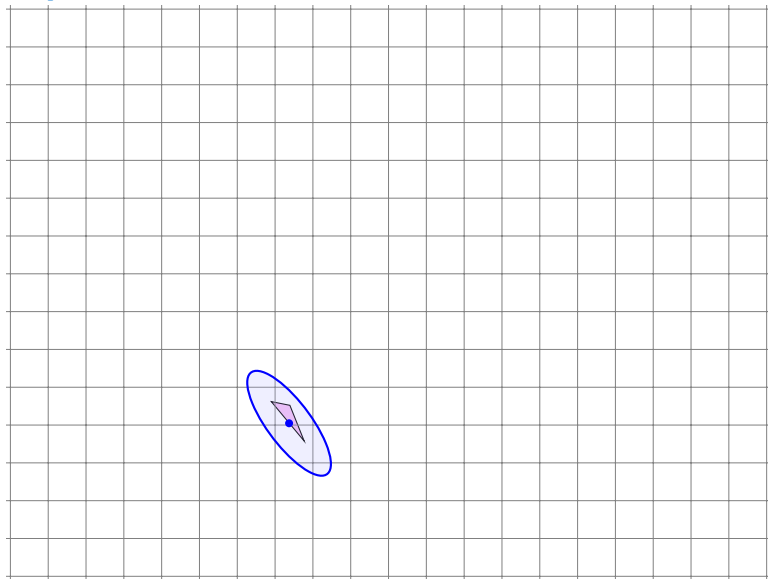
Example



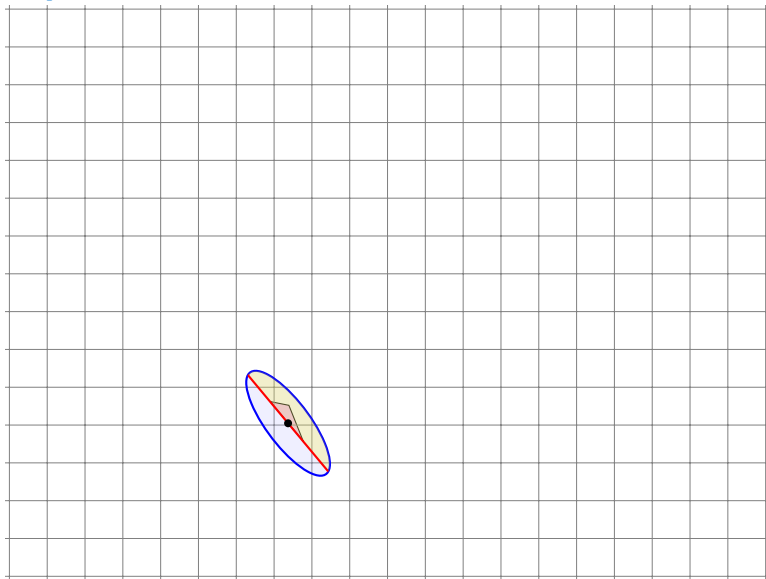
Example



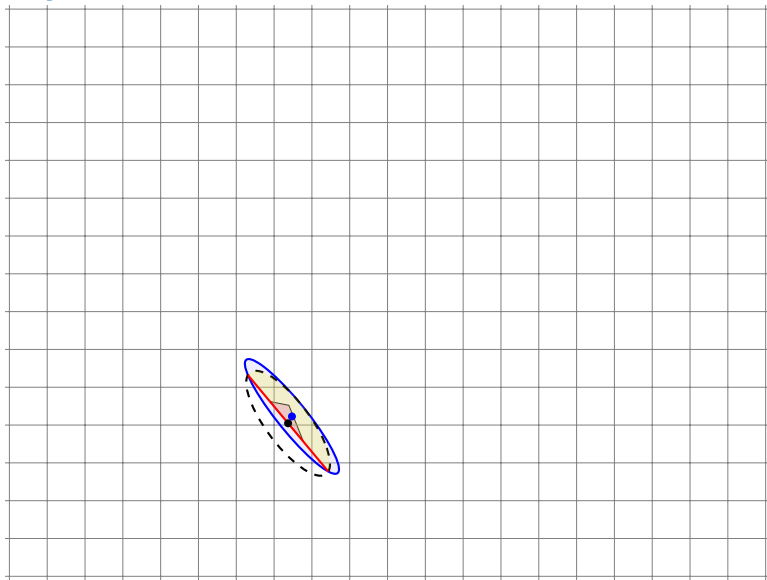
Example



Example



Example



Example

