MATH 652: Optimization II

Part 14

Linear programming 1

$$\begin{aligned} \text{minimize} \quad & c^T x \\ & Ax \geq b \end{aligned}$$

Linear Programming

Definition: A linear program is an optimization problem in which

- the objective function is linear (affine), i.e. has the form

$$f(x) = c^T x + d, \qquad c, x \in \mathbb{R}^n$$

(Note, however, that the existence of the constant d does not affect where the optimum lies.)

- all equality and inequality constraints are linear (affine), i.e. they either have the form

$$a_i^T x \geq b_i, \qquad a_i \in \mathbb{R}^n, b_i \in \mathbb{R}$$

or

$$a_i^T x = b_i, \qquad a_i \in \mathbb{R}^n, b_i \in \mathbb{R}$$

Note that an equality constraint is equivalent to two inequality constraints:

$$a_i^T x \geq b_i,$$
$$a_i^T x \leq b_i.$$

Linear Programming

Definition: A linear program is an optimization problem that can be written equivalently as

$$\min_{x \in \mathbb{R}^n} \quad c^T x$$
$$Ax \geq b$$

Linear Programming: Example 1

Production planning: A company wants to produce products $i=1...N$ which it can sell for a price of c_i. Product i needs a_{ji} units of resource $j=1...M$. We have b_j units of resource j available. How much of each product should it produce?

Mathematical formulation:

- Let x_i be the number of product i that is produced
- Revenue: $\sum_{i=1}^{P} c_i x_i = c^T x$
- We need $\sum_{i=1}^{P} a_{ji} x_i$ units of resource j
- The problem we need to solve is then

$$\max_{x \in \mathbb{R}^n} \; c^T x \qquad a_j^T x \le b_j, \quad j=1...M$$

or equivalently

$$\min_{x \in \mathbb{R}^n} \; -c^T x \qquad -Ax \ge -b$$

Linear Programming: Example 2

Future capacity planning: A power company foresees a demand of d_t gigawatt-hours in year t=*2010..2050*. It has existing capacity e_t and can choose to extend capacity using either coal (at a price of c_t per gigawatt-hour built) or wind power plants (at a cost of w_t built). Coal power plants last for 25 years, wind plants for 30. The company wants to build plants so that foreseen demand is met and at minimal cost.

Mathematical formulation:

- Let x_t be the capacity of coal power built in year t, and y_t be the capacity of wind power
- Total cost is then

$$\sum_{t=2010}^{2050} c_t x_t + w_t y_t$$

- The available capacity in year t is

$$e_t + \sum_{s=\max\{2010,t-25\}}^{t} x_s + \sum_{s=\max\{2010,t-30\}}^{t} y_s$$

Linear Programming: Example 2

Mathematical formulation:

$$\min_{x_t, y_t} \quad \sum_{t=2010}^{2050} c_t x_t + w_t y_t$$

$$\sum_{s=\max\{2010,\, t-25\}}^{t} x_s + \sum_{s=\max\{2010,\, t-30\}}^{t} y_s \geq d_t - e_t, \qquad t = 2010..2050$$

$$x_t \geq 0, \qquad t = 2010..2050$$

$$y_t \geq 0, \qquad t = 2010..2050$$

We can reformulate this in the usual form by denoting

$$X = (x_{2010}, x_{2011}, \ldots, x_{2050}, \quad y_{2010}, y_{2011}, \ldots, y_{2050})^T$$

$$C = (c_{2010}, c_{2011}, \ldots, c_{2050}, \quad w_{2010}, w_{2011}, \ldots, w_{2050})^T$$

$$b = (d_{2010}\text{-}e_{2010}, \; d_{2011}\text{-}e_{2011}, \; \ldots, \; d_{2050}\text{-}e_{2050})^T$$

$$\min_{X \in \mathbb{R}^{82}} \quad C^T X$$

$$A X \geq b$$

$$X \geq 0$$

Linear Programming: Example 3

Scheduling of resources to tasks: A hospital needs to schedule nurses to night shifts. Nurses work 5 days in a row on a 7-day schedule. On the *i*th day of the week, past history shows that n_i nurses are needed. How many nurses are needed in total, and on what schedules should they be?

Mathematical formulation:

- Let x_i be the number of nurses that start their 5-day run on day *i*
- The total number of nurses needed is then

$$\sum_{i=1}^{7} x_i$$

- On day 1, the number of nurses available is

$$x_4 + x_5 + x_6 + x_7 + x_1$$

- On day 2, the number of nurses available is

$$x_5 + x_6 + x_7 + x_1 + x_2$$

Linear Programming: Example 3

Mathematical formulation:

$$\min_{x \in \mathbb{R}^7} \quad \sum_{i=1}^{7} x_i$$

$$\sum_{k=i-4}^{i} x_{i \bmod 7} \geq d_i, \qquad i = 1 \ldots 7$$

$$x_i \geq 0, \qquad i = 1 \ldots 7$$

Note: More realistic formulations would also include preferences by employees, conflicts, tied schedules (e.g. advisor/trainee), contingencies/on-call duties, etc.

Linear Programming: Example 3

Mathematical formulation:

$$\min_{x \in \mathbb{R}^7} \sum_{i=1}^{7} x_i$$

$$\sum_{k=i-4}^{i} x_{i \bmod 7} \geq d_i, \qquad i = 1 \ldots 7$$

$$x_i \geq 0, \qquad i = 1 \ldots 7$$

However: In reality, we also need to consider that the number of nurses, x_i, must be an integer.

The problem is therefore not a common *linear programming* (LP) problem, but an *integer linear programming* (ILP) problem.

ILPs are much more difficult problems to solve!

Linear Programming: Example 4

Network flow: Food needs to get from a set of locations ("sources") of capacity s_i to a different set of locations ("sinks") of demand d_i. Sources have d_i=0, sinks s_i=0. Transportation happens on a network between nodes (i,j) of bandwidth b_{ij} and transporting one unit of food on this link costs c_{ij}.

How should food be transported most cost effective so that demand is

Linear Programming: Example 4

Mathematical formulation:

- Let x_{ij} be the amount of food that is transported on edge *(i,j)* of the network, i.e. from point *i* to point *j*

- We need to satisfy bandwidth constraints for each link:

$$x_{ij} \leq b_{ij}$$

(Note: If a link does not exist, then $b_{ij}=0$. Links are *directed*!)

- Sources can not deliver more than they have:

$$\sum_j x_{ij} \leq s_i \qquad \text{at sources}$$

- Sinks need to get their demand satisfied:

$$\sum_i x_{ij} \geq d_j \qquad \text{at sinks}$$

Linear Programming: Example 4

Mathematical formulation:

$$\begin{aligned} \min_{x \in \mathbb{R}^{n \times n}} \quad & \sum_{i,j} c_{ij} x_{ij} \\ & x_{ij} \leq b_{ij}, \\ & x_{ij} \geq 0, \\ & \sum_j x_{ij} \leq s_i, \\ & \sum_i x_{ij} \geq d_j. \end{aligned}$$

Note: Such problems appear in a wide variety of transportation problems. Examples are the shipping of books from amazon.com's distribution centers to customers, supplying goods from HEB's distribution centers to stores, etc.

A more complex version of the network flow problem would include a variety of products, rather than just one, or that origin and destination of each product matter (e.g. letters).

Linear Programming: Example 5

Network capacity: Data packets need to get from node A to node B along a network with given bandwidth b_{ij} on each edge.

What is the maximal data rate that can be transported on this network from A to B?

National LambdaRail network, part of NSF's TeraGrid.

Linear Programming: Example 5

Mathematical formulation:

- Let x_{ij} be the data rate transported on edge *(i,j)* of the network, i.e. from point *i* to point *j*

- We need to satisfy bandwidth constraints for each link:

$$x_{ij} \leq b_{ij}$$

(Note: If a link does not exist, then $b_{ij}=0$. Links are *directed*!)

- At location $i=A$, we have for the net inbound flux:

$$\sum_k x_{ki} - \sum_j x_{ij} = -s$$

- At location $i=B$, we have:

$$\sum_k x_{ki} - \sum_j x_{ij} = s$$

- At all other nodes, data is just transported through:

$$\sum_k x_{ki} - \sum_j x_{ij} = 0$$

Linear Programming: Example 5

Mathematical formulation:

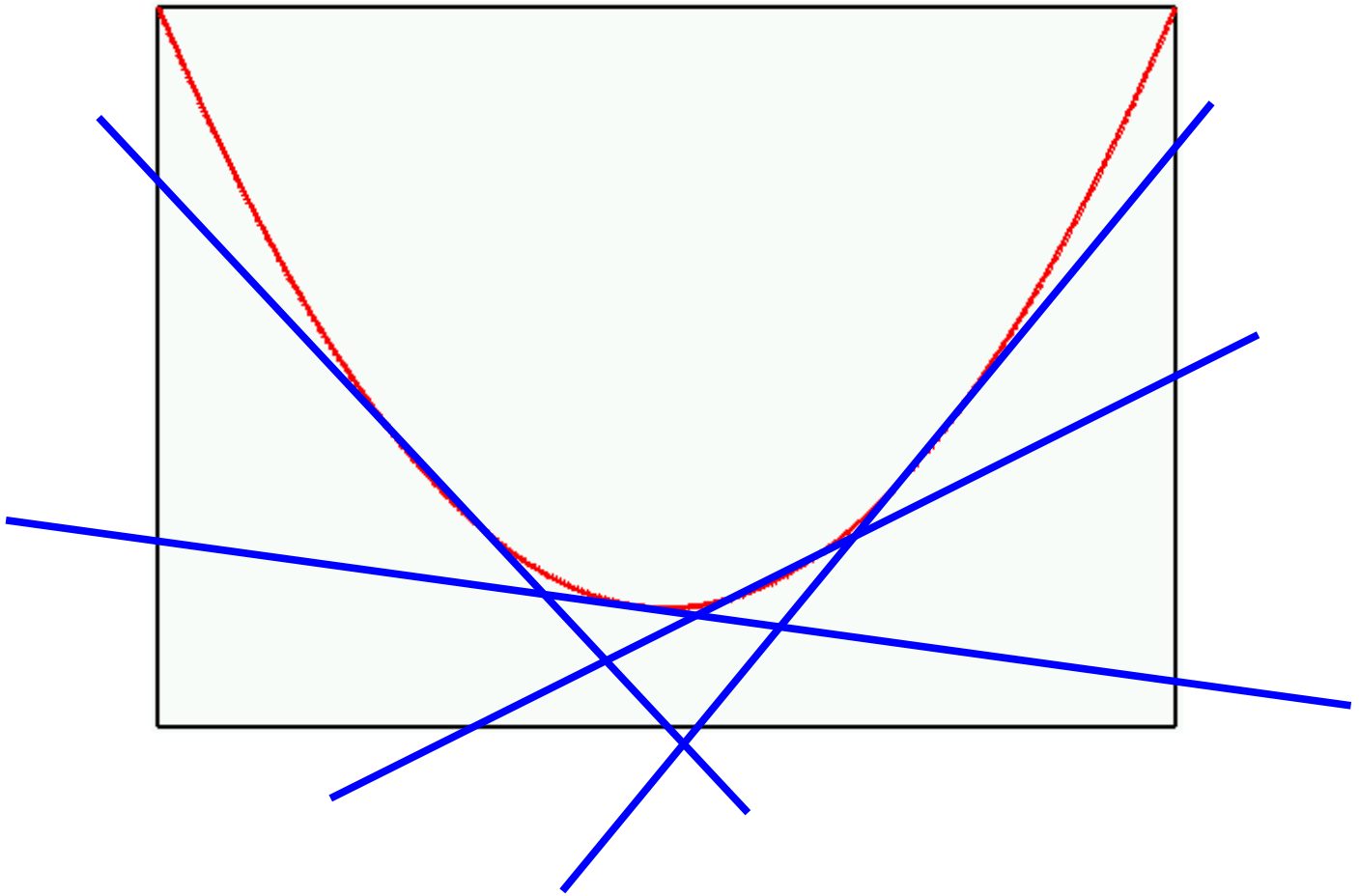
$$
\begin{aligned}
\min_{x \in \mathbb{R}^{n \times n},\, s \in \mathbb{R}} \quad & s \\
& x_{ij} \leq b_{ij}, \\
& x_{ij} \geq 0, \\
& \sum_k x_{ki} - \sum_j x_{ij} = s(-\delta_{iA} + \delta_{iB})
\end{aligned}
$$

Note: Extensions might consider that there can be multiple sources but only one destination (e.g. Google data centers, P2P/Napster servers), or that multiple products need to be provided.

Network capacity problems are ubiquitous in transportation planning such as highway networks, airplane scheduling, etc.

Linear Programming: Example 6

Convex optimization of a nonlinear function: Math 651 dealt almost exclusively with the question of finding the minimum of a function $f(x)$ that may be nonlinear. If it is at least convex, we may be able to approximate it with piecewise linears.



Linear Programming: Example 6

Mathematical formulation:

- The original problem was to find the solution of

$$\min_{x \in \mathbb{R}^n} \quad f(x)$$

- An equivalent formulation of this is:

$$\min_{x \in \mathbb{R}^n, z \in \mathbb{R}} \quad z$$
$$z \geq f(x)$$

- An approximation to this can be found if we choose points ξ_i and solve instead

$$\min_{x \in \mathbb{R}^n, z \in \mathbb{R}} \quad z$$
$$z \geq f(\xi_i) + \nabla f(\xi_i)^T (x - \xi_i), \qquad \forall i$$

Note: The solution of this problem could serve as a good starting point for the full nonlinear minimization. Constraints can easily be incorporated if they are linear, or a linearized as well.

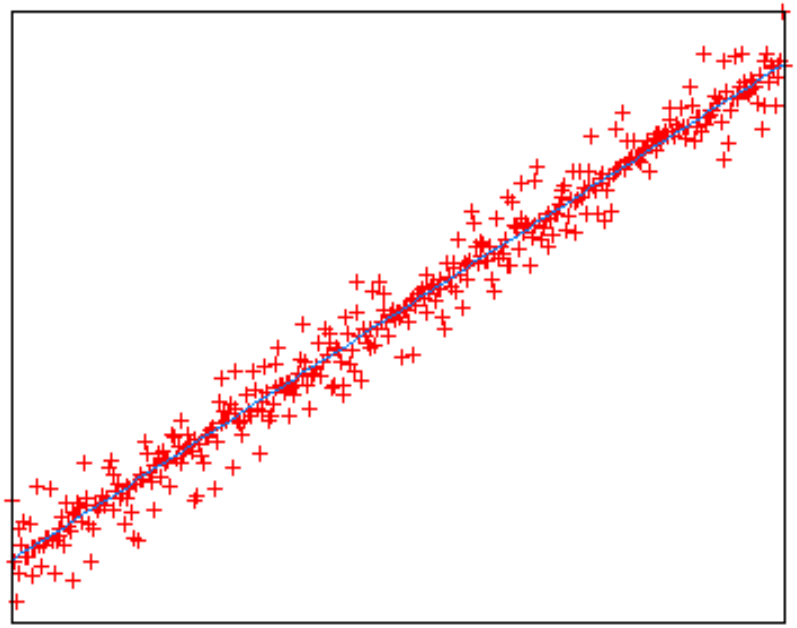
Linear Programming: Example 7

Data fitting: Find a linear relationship that best fits a set of data points. This can be formulated in a variety of ways:

$$\min_{a,b\in\mathbb{R}} \ \frac{1}{2}\sum_i (y_i-(at_i+b))^2$$

$$\min_{a,b\in\mathbb{R}} \ \sum_i |y_i-(at_i+b)|$$

$$\min_{a,b\in\mathbb{R}} \ \max_i \ |y_i-(at_i+b)|$$



The first one is a smooth, convex problem for which the techniques of Math 651 are well suited. The other two are non-smooth but convex problems that can be reformulated as linear programming problems.

Linear Programming: Example 7

Mathematical formulation:

- The original problem was to find the solution of

$$\min_{a,b\in\mathbb{R}} \quad \sum_{i=1}^{N} |y_i-(at_i+b)|$$

- An equivalent but still non-smooth formulation of this is:

$$\min_{a,b\in\mathbb{R},s\in\mathbb{R}^n} \quad \sum_{i=1}^{N} s_i$$
$$s_i=|y_i-(at_i+b)|, \qquad i=1\ldots N$$

- But we can re-formulate this as a smooth problem as follows:

$$\min_{a,b\in\mathbb{R},s\in\mathbb{R}^n} \quad \sum_{i=1}^{N} s_i$$
$$s_i\geq(y_i-(at_i+b)), \qquad i=1\ldots N$$
$$s_i\geq-(y_i-(at_i+b)), \qquad i=1\ldots N$$

Note: A similar techniques also works for the maximum-residual problem.

Formulating linear programs

Theorem:
Any linear optimization problem that is given in the form

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & c^T x \\ & A_1 x \geq b_1 \\ & A_2 x = b_2 \\ & A_3 x \leq b_3 \end{aligned}$$

can be restated in the form

$$\boxed{\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & c^T x \\ & Ax \geq b \end{aligned}}$$

where $A = \begin{pmatrix} A_1 \\ A_2 \\ -A_2 \\ -A_3 \end{pmatrix}$, $b = \begin{pmatrix} b_1 \\ b_2 \\ -b_2 \\ -b_3 \end{pmatrix}$,

Formulating linear programs

Theorem:
Any linear optimization problem that is given in the form

$$\min_{x \in \mathbb{R}^n} \quad c^T x$$
$$Ax \geq b$$

is equivalent to a problem written in *standard form of linear programming*:

$$\min_{\tilde{x} \in \mathbb{R}^{2n+m}} \quad \tilde{c}^T \tilde{x}$$
$$\tilde{A}\tilde{x} = b$$
$$\tilde{x} \geq 0$$

where

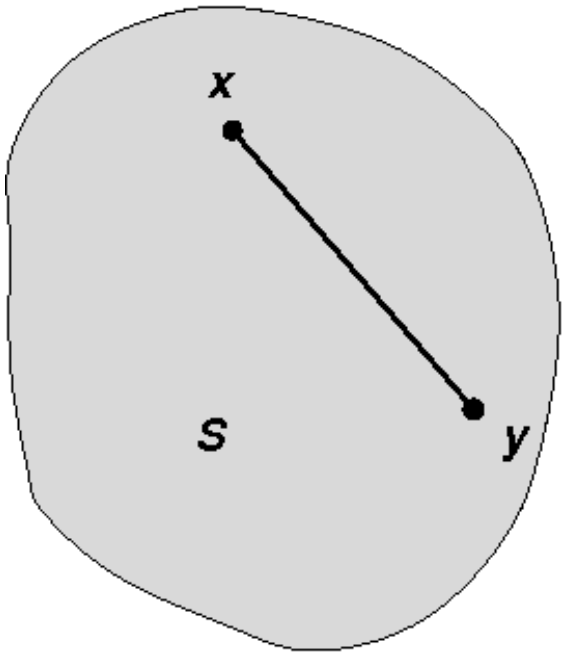
$$\tilde{A} = \begin{pmatrix} A & -A & -I \end{pmatrix}, \qquad \tilde{x} = \begin{pmatrix} x^+ \\ x^- \\ s \end{pmatrix}, \qquad \tilde{c} = \begin{pmatrix} c \\ -c \\ 0 \end{pmatrix}$$

Note: Standard form is more convenient for algorithm development and will therefore frequently be used.

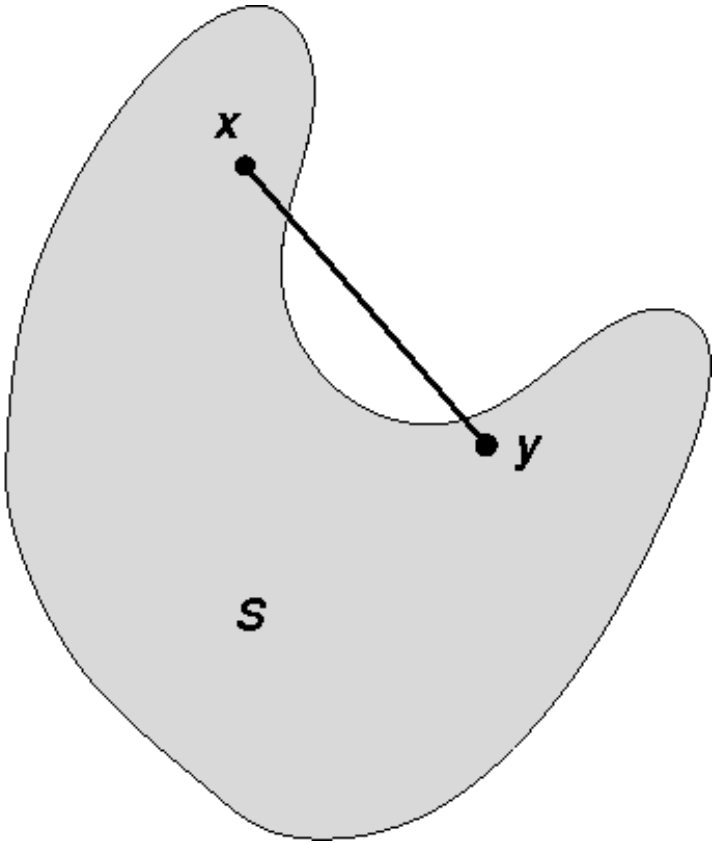
The geometry of feasible sets

Definition: A set S is called convex if

$$x, y \in S \quad \text{implies} \quad \lambda x + (1-\lambda) y \in S \quad \forall\, 0 \le \lambda \le 1.$$



A convex set S.



A nonconvex set S.

The geometry of feasible sets

Lemma: The set of points that satisfy a single constraint,

$$\left\{x \in \mathbb{R}^n : \quad a_i^T x \geq b_i\right\}$$

is a half-space and is convex.

Lemma: The intersection of finitely many convex sets is convex.

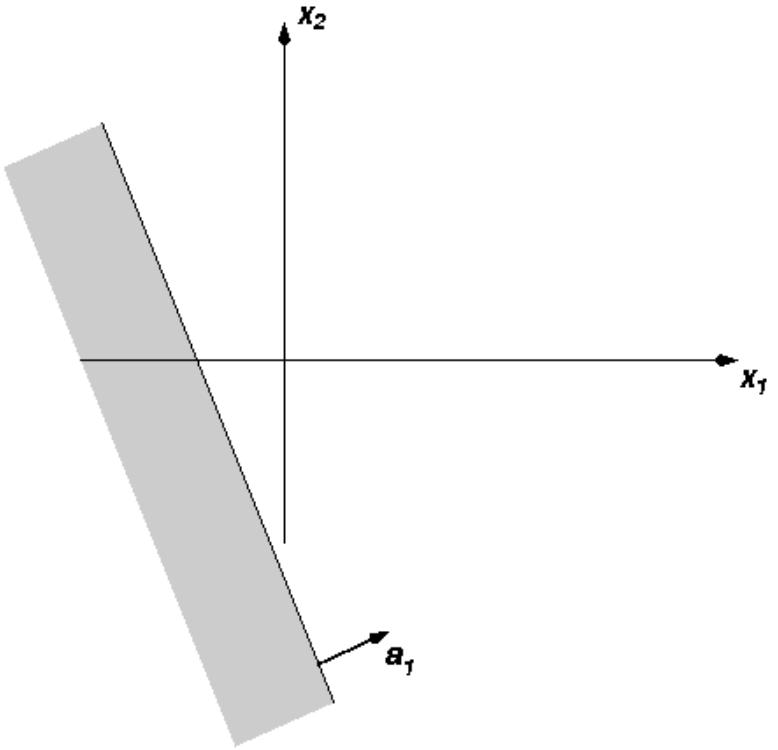
Theorem: The set of points described by the constraints,

$$\left\{x \in \mathbb{R}^n : \quad Ax \geq b\right\}$$

is convex.

The geometry of feasible sets

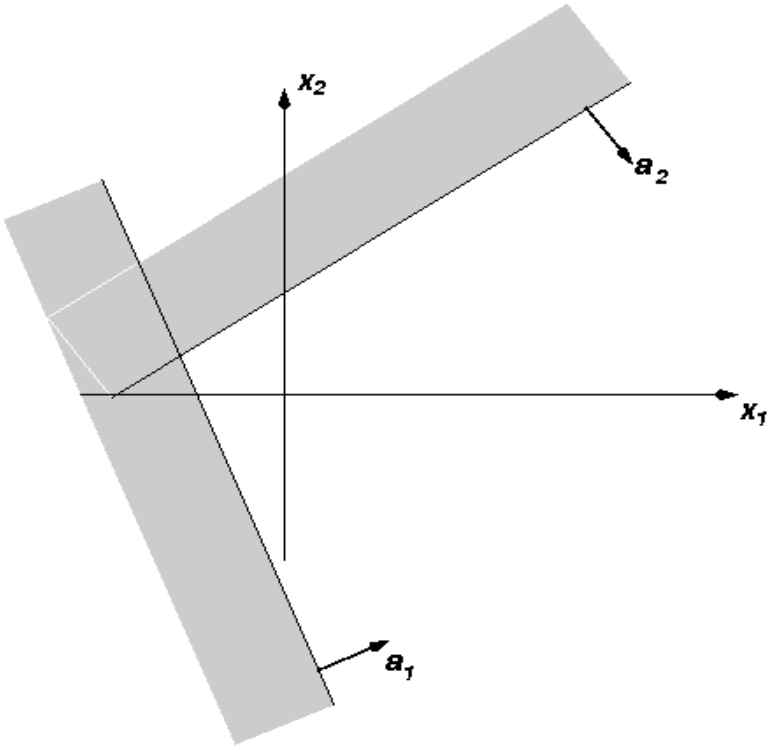
Example: The set of points described by four constraints.



$$\left\{x \in \mathbb{R}^2 : \; a_i^T x \geq b_i, i = 1\right\}$$

The geometry of feasible sets

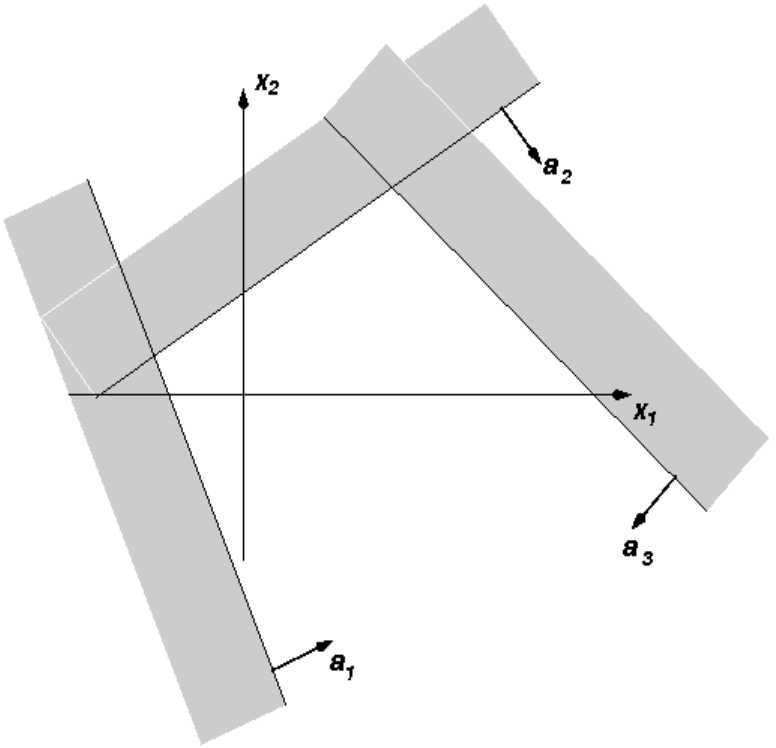
Example: The set of points described by four constraints.



$$\{x \in \mathbb{R}^2 : \; a_i^T x \geq b_i, i = 1,2\}$$

The geometry of feasible sets

Example: The set of points described by four constraints.



Note: The matrix

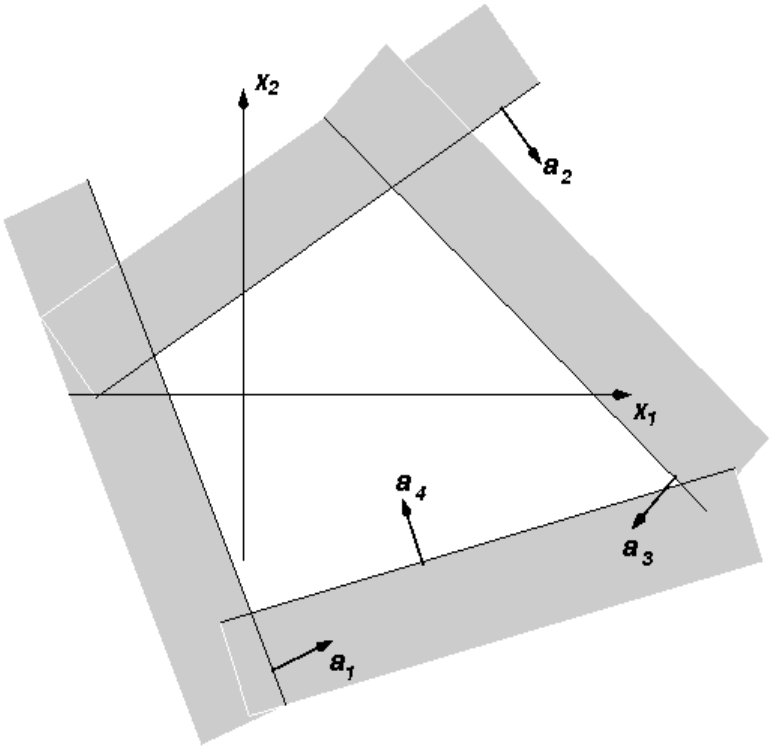
$$A = \begin{pmatrix} a_1^T \\ a_2^T \\ a_3^T \end{pmatrix} \in \mathbb{R}^{3\times 2}$$

does now no longer have full row rank!

$$\{x \in \mathbb{R}^2 : \; a_i^T x \geq b_i, i = 1..3\}$$

The geometry of feasible sets

Example: The set of points described by four constraints.

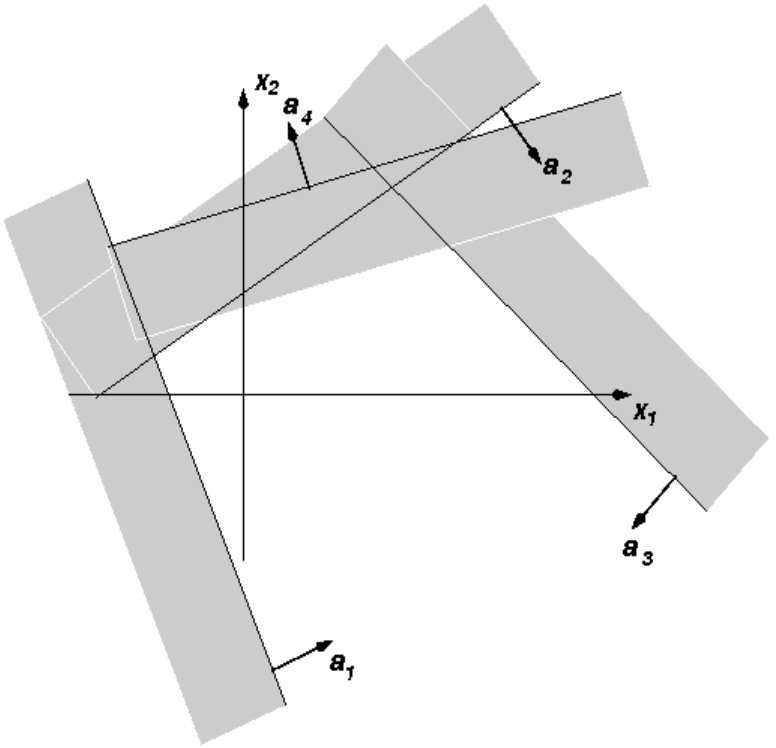


Note: The feasible set is a compact subset of R^2.

$$\{x \in \mathbb{R}^2 : \; a_i^T x \geq b_i, i = 1..4\}$$

The geometry of feasible sets – Pathologies 1

Example: The set of points described by four constraints.

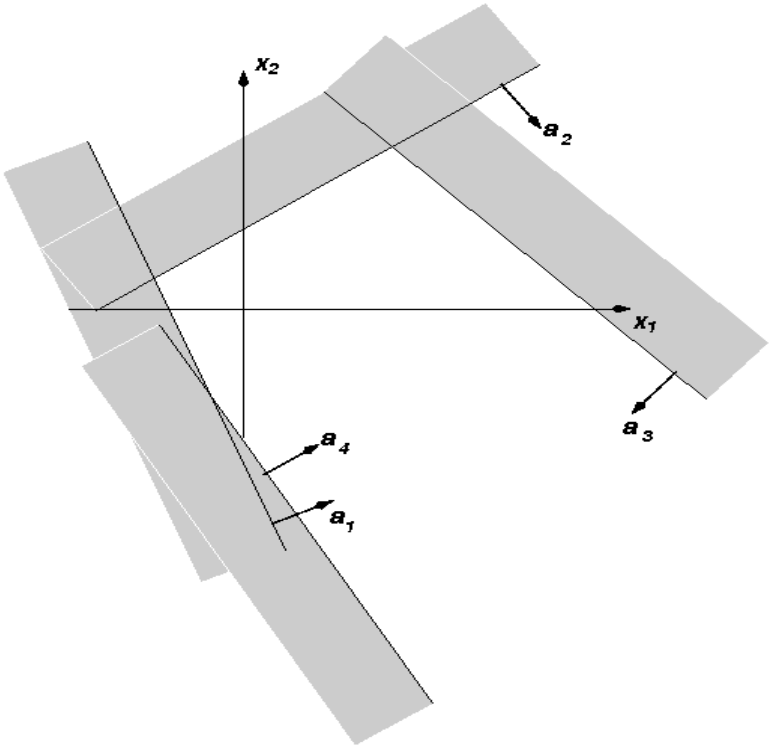


Note: The feasible set is empty. The constraints are said to be *mutually incompatible*!

$$\{x \in \mathbb{R}^2 : \; a_i^T x \geq b_i, i = 1..4\}$$

The geometry of feasible sets – Pathologies 2

Example: The set of points described by four constraints.

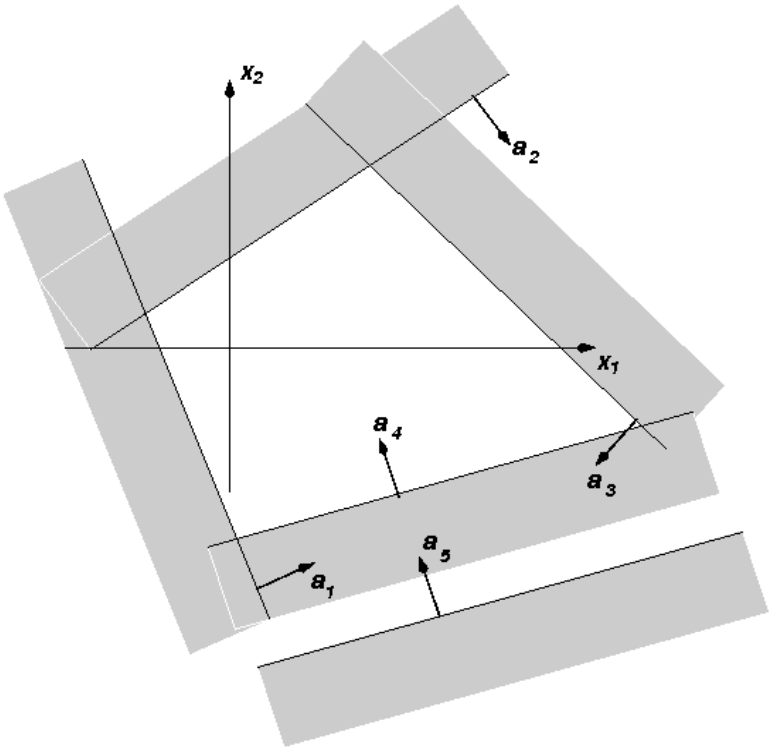


Note: The feasible set is unbounded and consequently not compact.

$$\{x \in \mathbb{R}^2 : \; a_i^T x \geq b_i, i = 1..4\}$$

The geometry of feasible sets – Pathologies 3

Example: The set of points described by five constraints.

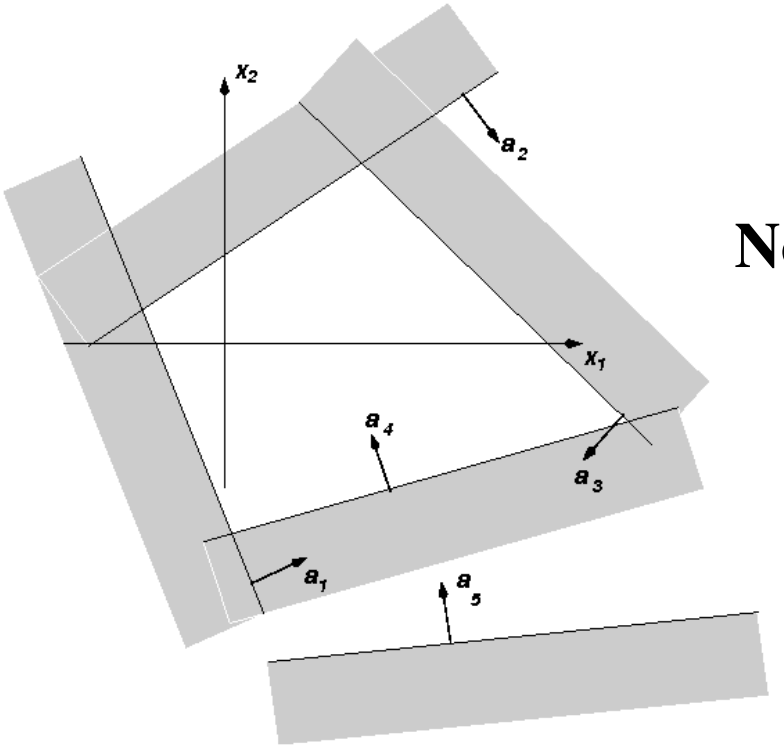


Note: Constraints 4 and 5 are linearly dependent but not mutually exclusive. Nothing bad happens.

$$\{x \in \mathbb{R}^2 : \; a_i^T x \geq b_i, i = 1..5\}$$

The geometry of feasible sets – Pathologies 4

Example: The set of points described by five constraints.

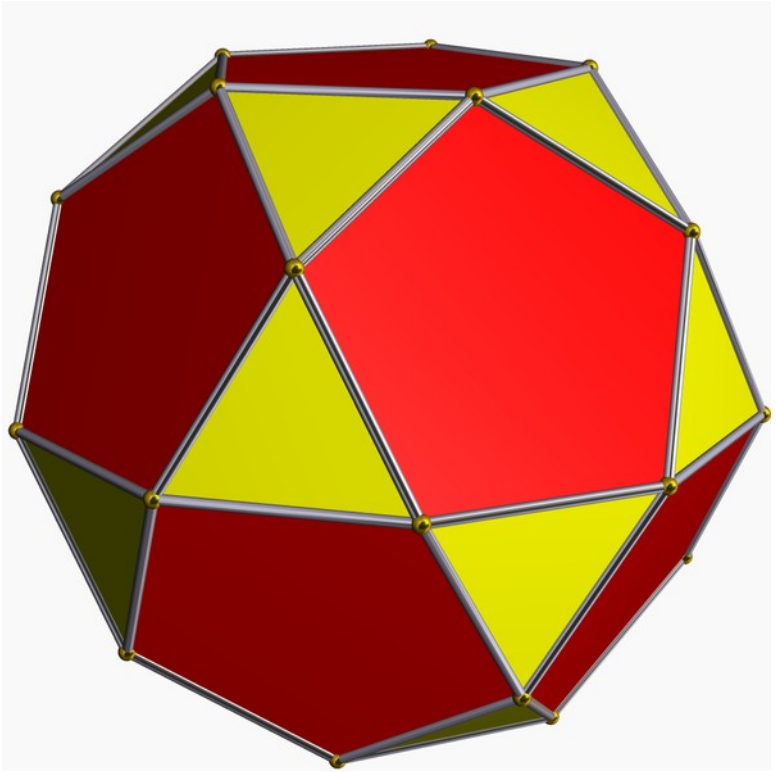


Note: Constraint 5 is not parallel to any of the other constraints but will never be active. Nothing bad happens.

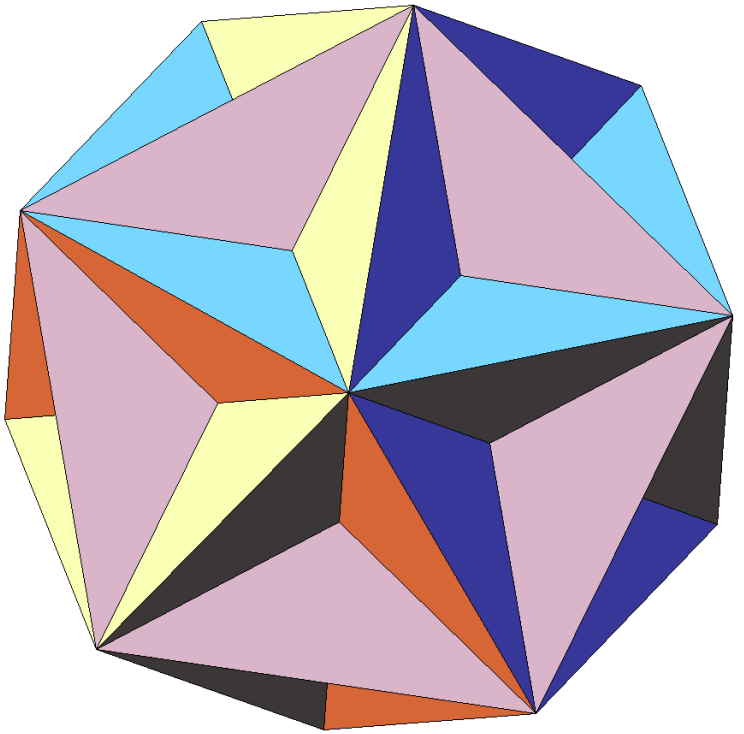
$$\{x \in \mathbb{R}^2 : \; a_i^T x \geq b_i, i = 1..5\}$$

The geometry of feasible sets – 3 and more dimensions

Feasible sets with three or more variables must still be convex. They are, in particular, convex polyhedra:



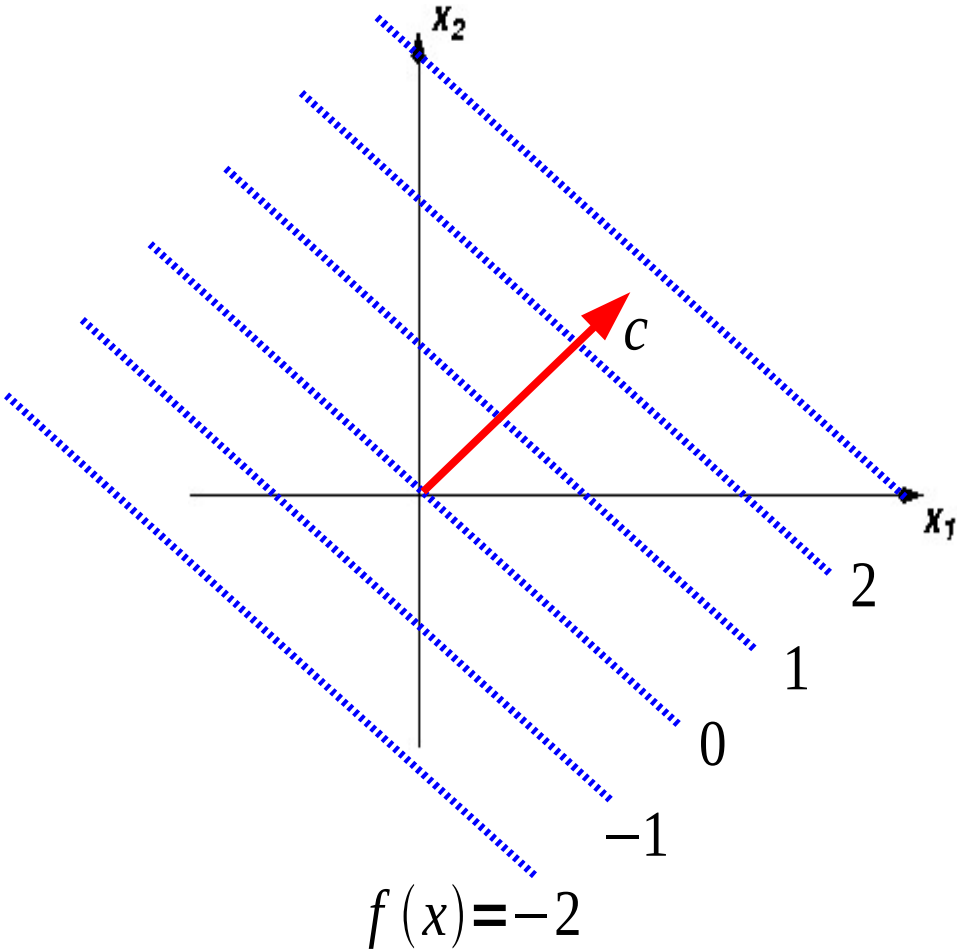
A convex polyhedron
(an icosidodecahedron)



A nonconvex polyhedron
(a stellation)

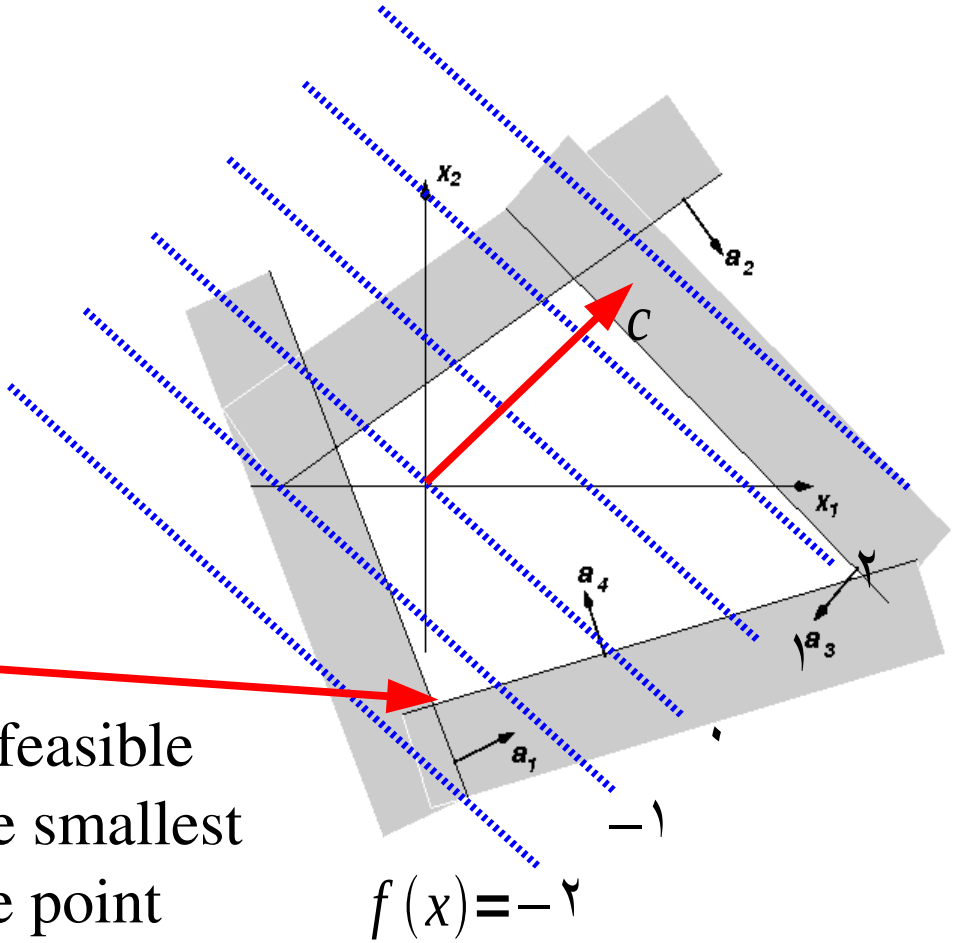
The geometry of the objective function

Example: The function $f(x) = x_1 + x_2 = c^T x$.



The geometry of linear problems

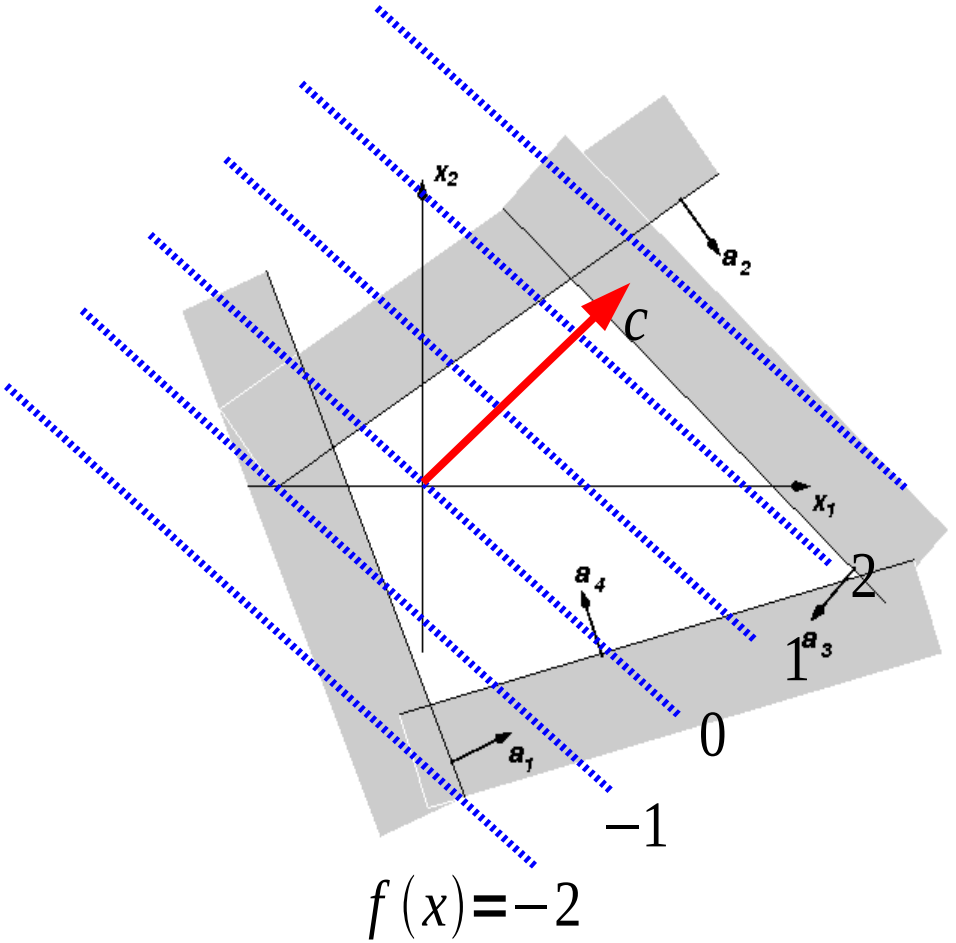
Example: The function $f(x) = x_1 + x_2 = c^T x$.



This is the location within the feasible set with the smallest $f(x)$, i.e. the point of solution.

The geometry of linear problems

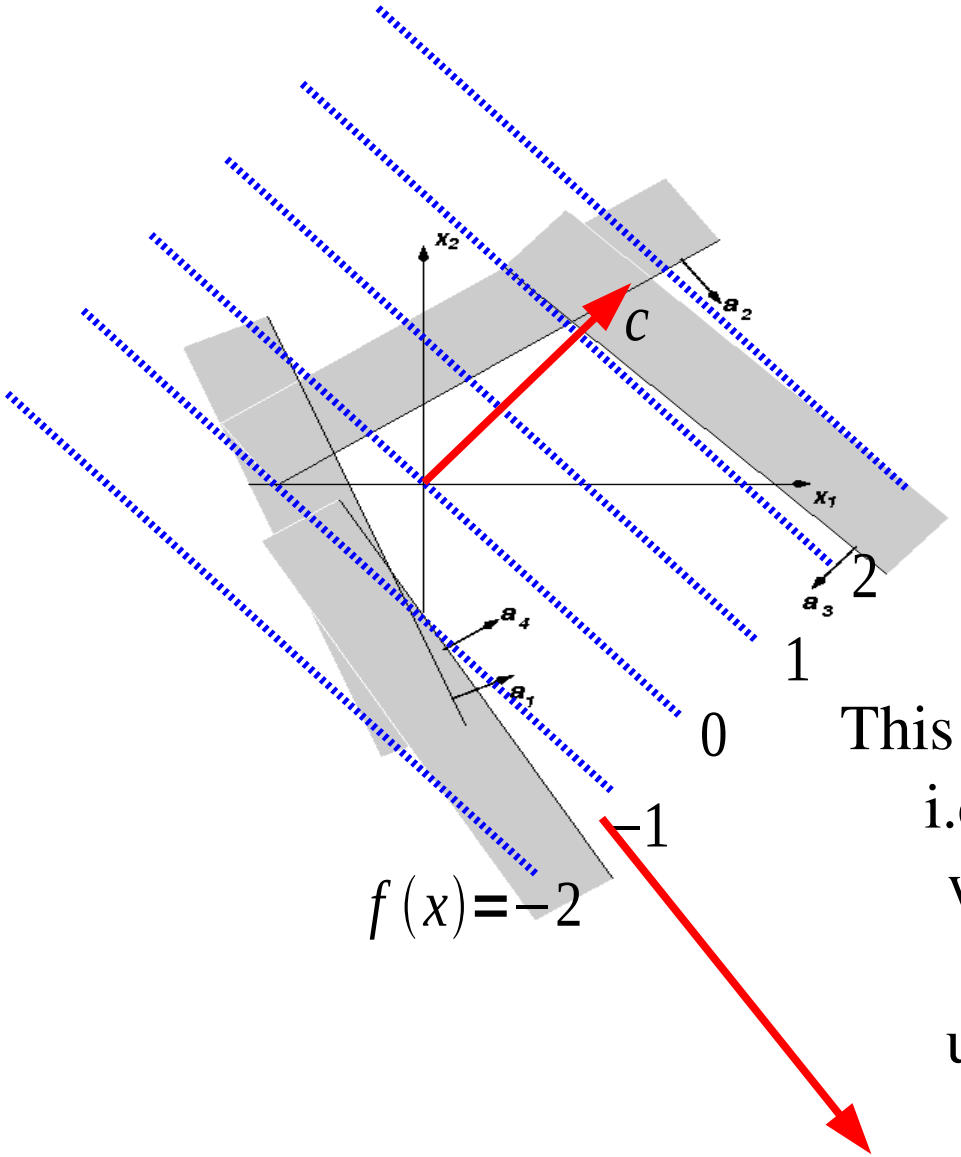
Example: The function $f(x) = x_1 + x_2 = c^T x$.



The geometry of such problem suggests an algorithm: Start at one of the vertices and keep trying to find an adjacent vertex with smaller $f(x)$. This is, in essence, Dantzig's *simplex algorithm*.

The geometry of linear problems

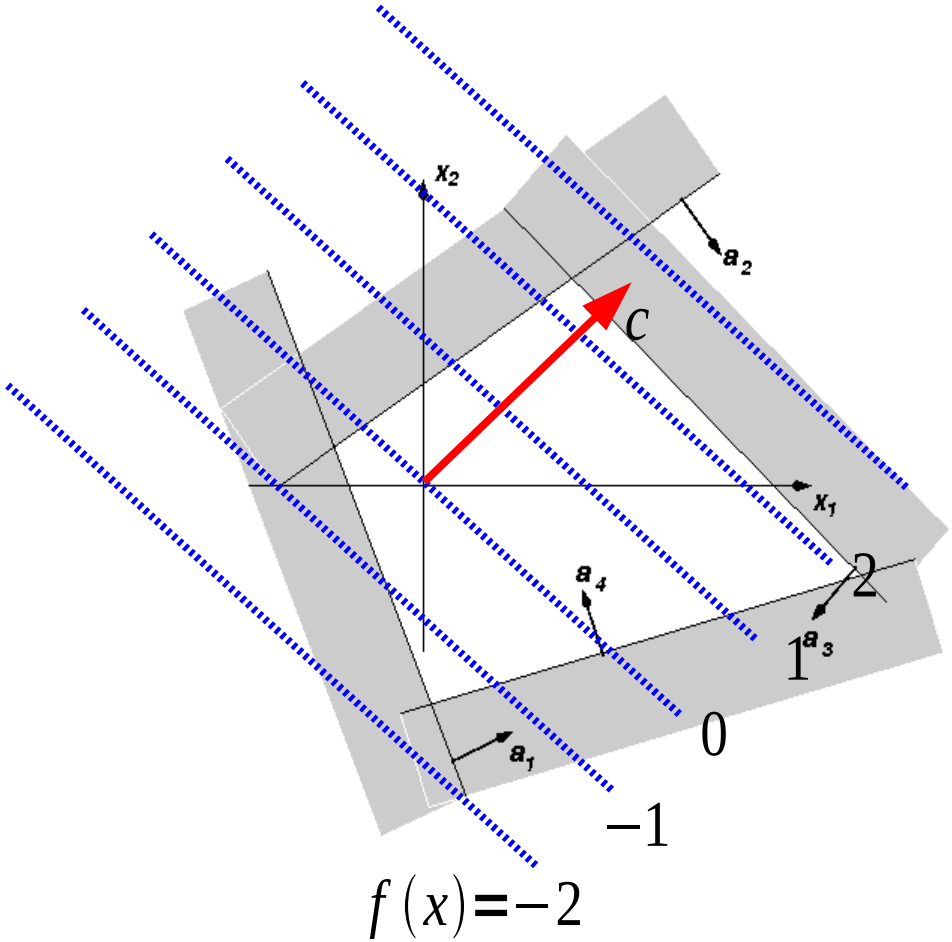
Example: The function $f(x) = x_1 + x_2 = c^T x$.



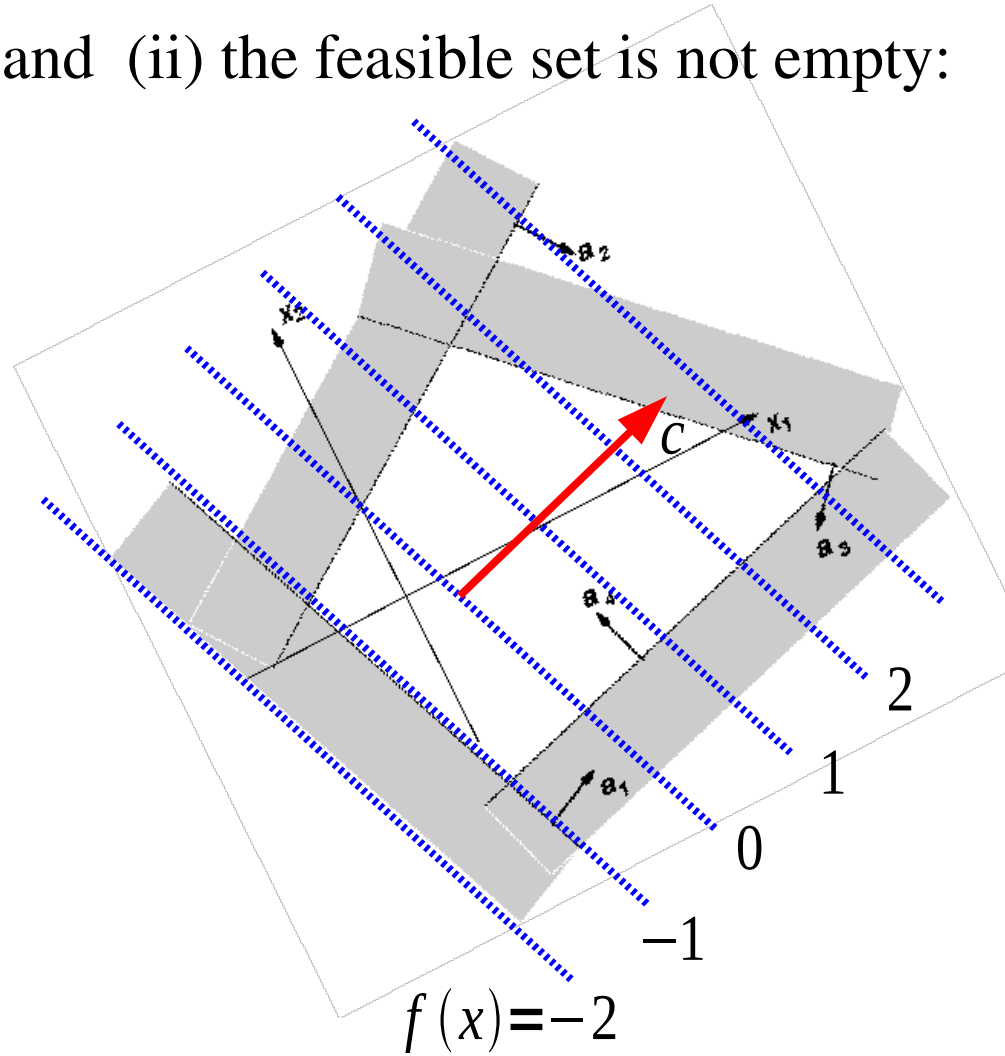
This problem is unbounded, i.e. there is a direction in which the feasible set is unbounded and $f(x)$ is unbounded from below.

Possible solutions of linear programs

If (i) the feasible set is bounded, and (ii) the feasible set is not empty:



A single vertex is the unique solution.



All points along a whole edge are solutions. In particular, the vertices of the edge are solutions.

The geometry of linear programs in standard form

Recall that any linear optimization problem that is given in the form

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & c^T x \\ & Ax \geq b \end{aligned}$$

is equivalent to a problem written in *standard form*:

$$\begin{aligned} \min_{\tilde{x} \in \mathbb{R}^{2n+m}} \quad & \tilde{c}^T \tilde{x} \\ & \tilde{A}\tilde{x} = b \\ & \tilde{x} \geq 0 \end{aligned}$$

where

$$\tilde{A} = \begin{pmatrix} A & -A & -I \end{pmatrix}, \qquad \tilde{x} = \begin{pmatrix} x^+ \\ x^- \\ s \end{pmatrix}, \qquad \tilde{c} = \begin{pmatrix} c \\ -c \\ 0 \end{pmatrix}$$

Note: If $A \in \mathbb{R}^{m \times n}$, then $\tilde{A} \in \mathbb{R}^{m \times (2n+m)}$. While the original matrix may not have fewer rows than columns, the second definitely does.

The geometry of linear programs in standard form

Problems written in *standard form*:

$$\begin{aligned} \min_{\tilde{x} \in \mathbb{R}^{2n+m}} \quad & \tilde{c}^T \tilde{x} \\ & \tilde{A}\tilde{x} = b \\ & \tilde{x} \geq 0 \end{aligned}$$

definitely have a matrix with fewer rows (*m)* than columns (*2n*+*m*).

Corollary: The feasible set is the intersection of a hyperplane (with dimension equal to at least *(2n+m)-m=2n*) with the first quadrant/ octant/etc.

In particular, the feasible set is also a polygon, just like before, except that this polygon now lies in a lower-dimensional subspace defined by the constraint.

The geometry of linear programs in standard form

Example: Consider $\min_{x \in \mathbb{R}} \; x$

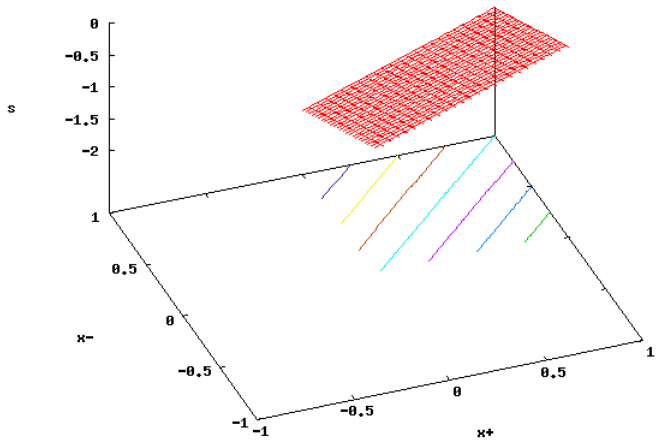
$$x \geq 1$$

The standard form of this problem is:

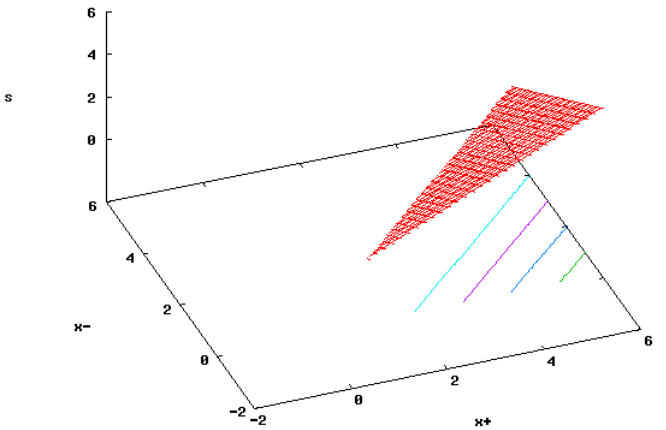
$$\min_{\tilde{x}=\{x^+, x^-, s\} \in \mathbb{R}^3} \; x^+ - x^-$$

$$x^+ - x^- - s = 1$$

$$\tilde{x} \geq 0$$



$x^+ - x^- - s = 1$ in the area $x^+ \geq 0, x^- \geq 0$



$x^+ - x^- - s = 1$ in the area $\tilde{x} \geq 0$
i.e. the feasible set

The geometry of linear programs in standard form

Example: Consider

$$\begin{aligned} \min_{x \in \mathbb{R}} \quad & x \\ & x \geq 1 \end{aligned}$$

The standard form of this problem is:

$$\begin{aligned} \min_{\tilde{x}=\{x^+, x^-, s\} \in \mathbb{R}^3} \quad & x^+ - x^- \\ & x^+ - x^- - s = 1 \\ & \tilde{x} \qquad\quad \geq 0 \end{aligned}$$

Note: The solution to this problem is not unique – any set of variables

$$x^+ = 1 + x^-, \quad x^- \geq 0, \quad s = 0$$

produces the optimal value 1 of the objective function. By unsubstituting variables we get the unique solution of the original problem:

$$x \;=\; x^+ - x^- \;=\; 1$$

The value of the objective function is of course the same.

Possible solutions of linear programs

One of the following cases must hold:

- A vertex of the feasible region is the unique solution
- All points of an edge or face of the feasible region are solutions; in particular, the vertices of the edge or face are solutions
- The feasible set is empty and there are no solutions
- The feasible set is unbounded and the objective function is unbounded from below in one of the directions in which the feasible set is unbounded; the problem then has no bounded solution.

In other words:

If bounded solutions exists,
the set of solutions must include at least one vertex!

Possible solutions of linear programs

Definition:
We call a point x^* a local solution of

$$\min_{x \in \mathbb{R}^n} \quad f(x)$$
$$g(x) = 0, \quad h(x) \geq 0$$

if there exists a neighborhood U of x^* so that

$$f(x^*) \leq f(x) \quad \forall\, x \in U \cap \{x : g(x) = 0,\ h(x) \geq 0\}$$

Definition:
We call a point x^* a global solution of

$$\min_{x \in \mathbb{R}^n} \quad f(x)$$
$$g(x) = 0, \quad h(x) \geq 0$$

if

$$f(x^*) \leq f(x) \quad \forall\, x \in \{x : g(x) = 0,\ h(x) \geq 0\}$$

Possible solutions of linear programs

Definition:
We call a function $f(x)$ convex if

$$f(\lambda x+(1-\lambda)y) \le \lambda f(x)+(1-\lambda)f(y) \quad \forall x, y \in D \subset \mathbb{R}^n, \lambda \in [0,1]$$

We call it concave if

$$f(\lambda x+(1-\lambda)y) \ge \lambda f(x)+(1-\lambda)f(y) \quad \forall x, y \in D \subset \mathbb{R}^n, \lambda \in [0,1]$$

Corollary:
Any linear (affine) function is both convex and concave.

Remark: In fact, affine functions are the only functions that are both convex and concave.

Possible solutions of linear programs

Theorem:
Any local solution of a linear program is also a global solution.

(Proof: Use convexity of the feasible set and of the objective function.)

Theorem:
The set of all global solutions of a linear program is convex (and consequently also singly connected).

(Proof: Use convexity of the feasible set and linearity of the objective function.)

Theorem:
Among all solutions of a linear program is always at least one vertex of the feasible set.

(Proof: Later, need to define precisely what a vertex is.)

Polyhedra in a formal language

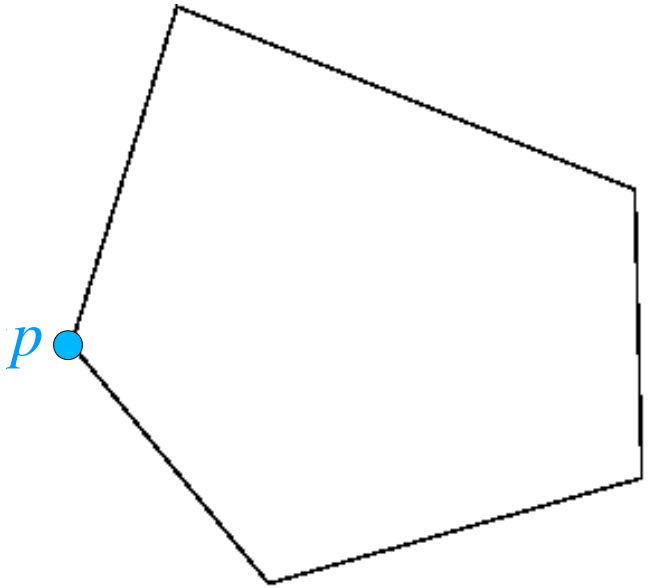
Definition:
We call the set of points $\{x \in \mathbb{R}^n : Ax \geq b\}$ a *polyhedron*.
If $n=2$, we also call it a *polygon*.

Corollary:
The set of points $\{x \in \mathbb{R}^n : Ax = b, x \geq 0\}$ is also a polyhedron.

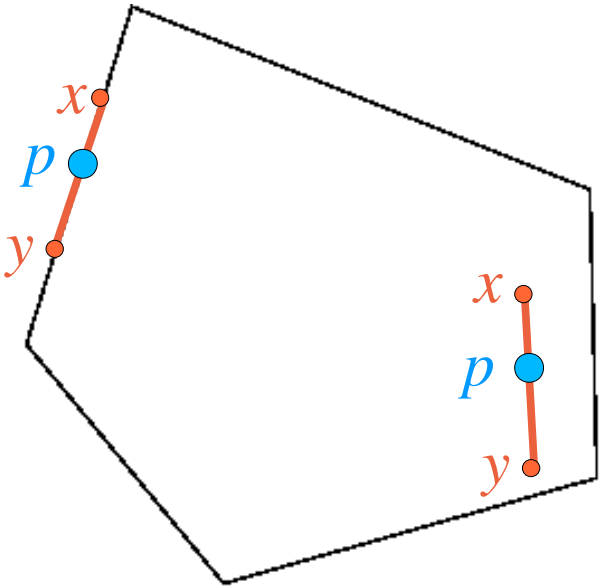
Polyhedra in a formal language

Definition:

Let $P \subset \mathbb{R}^n$ be a polyhedron. We call $p \in P$ an *extreme point* if there are no $x, y \in P, x \neq p, y \neq p$ so that $p = \lambda x + (1-\lambda) y$ for any $0 \leq \lambda \leq 1$.



An extreme point.



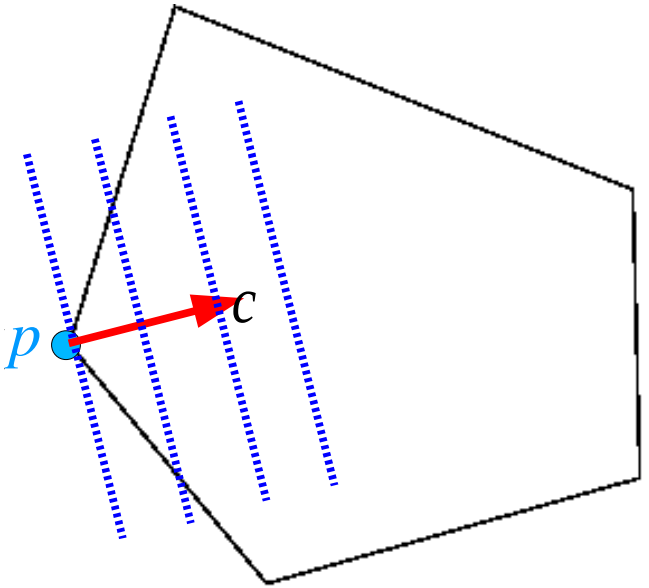
Two points that are not extreme.

Polyhedra in a formal language

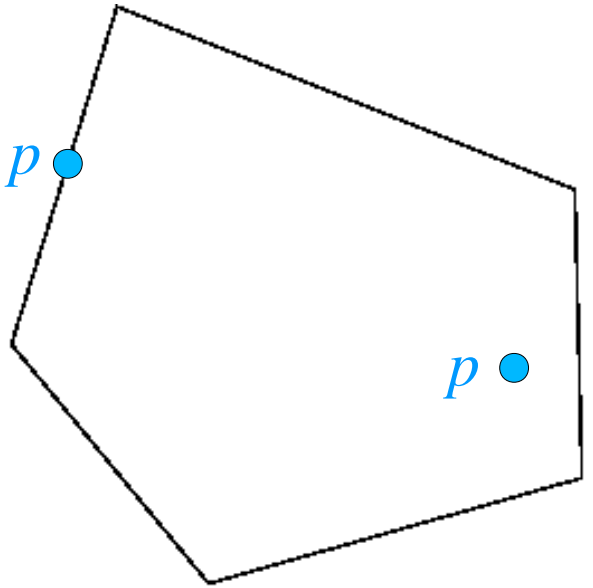
Definition:

Let $P \subset \mathbb{R}^n$ be a polyhedron. We call $p \in P$ a *vertex* of P if there is a vector c so that

$$c^T p < c^T x \quad \forall\, x \in P, x \neq p.$$



An extreme point.



Two points that are not extreme.

Polyhedra in a formal language

Definition:
Let $P \subset \mathbb{R}^n$ be a polyhedron defined by

$$P = \{x \in \mathbb{R}^n : a_i^T x \geq b_i \text{ for } i = 1 \ldots m_1, \quad a_i^T x = b_i \text{ for } i = m_1 + 1 \ldots m_2\}$$

The set of *active* or *binding* constraints at a arbitrary point $p \in \mathbb{R}^n$ is defined as

$$I(p) = \{i \in [1, m_1] : a_i^T p = b_i\} \cup \{i \in [m_1 + 1, m_2] : a_i^T p = b_i\} \subset [1, m_2]$$

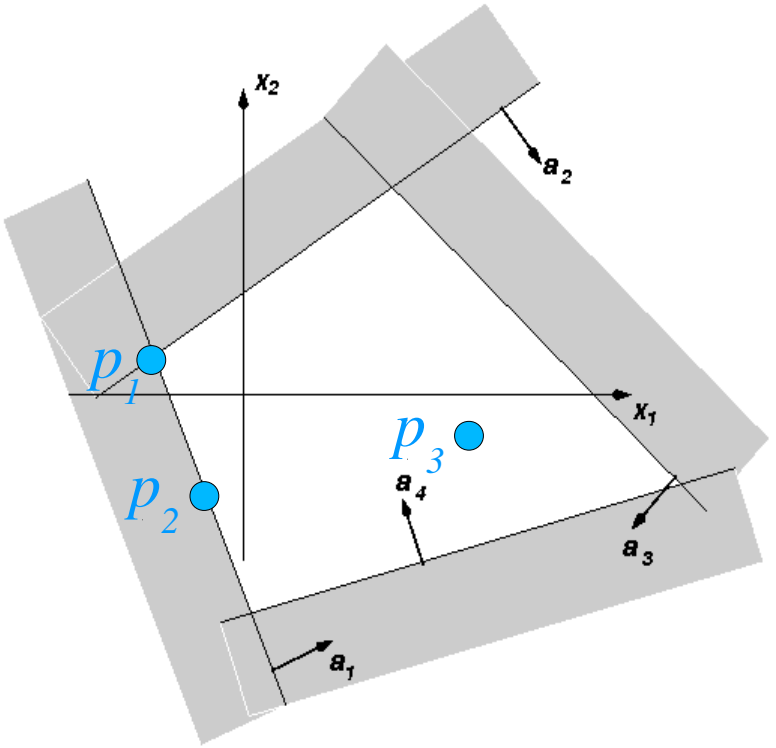
Note: If $p \in P$ then

$$I(p) = \{i \in [1, m_1] : a_i^T p = b_i\} \cup [m_1 + 1, m_2]$$

*b*ecause all equality constraints must be active.

Polyhedra in a formal language

Example:



$I(p_1)=\{1,2\}$

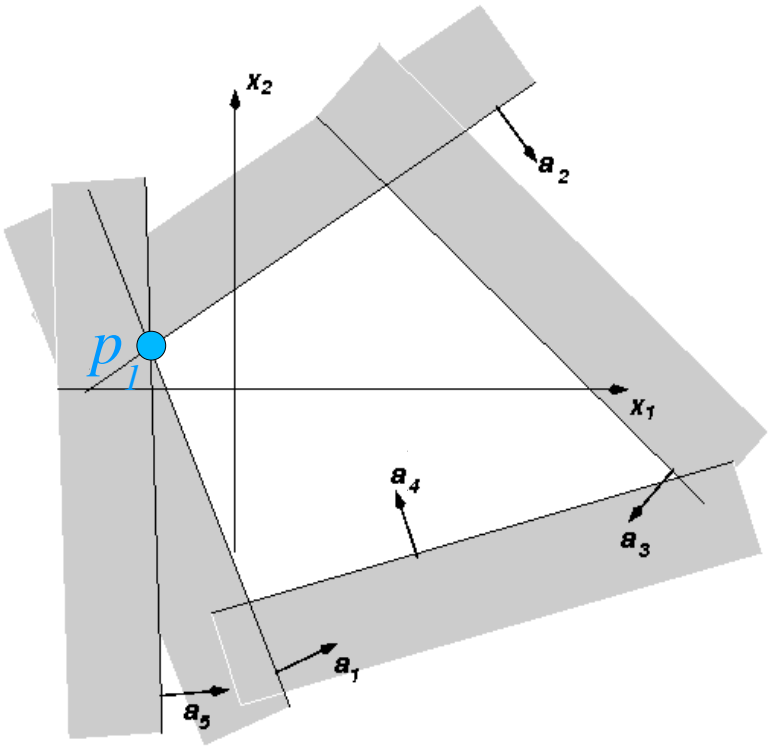
$I(p_1)=\{1\}$

$I(p_1)=\{\}$

Tentative conclusion: At a vertex of a polyhedron in *n* space dimension, *n* constraints are active, i.e. $\#I(p)=n$.

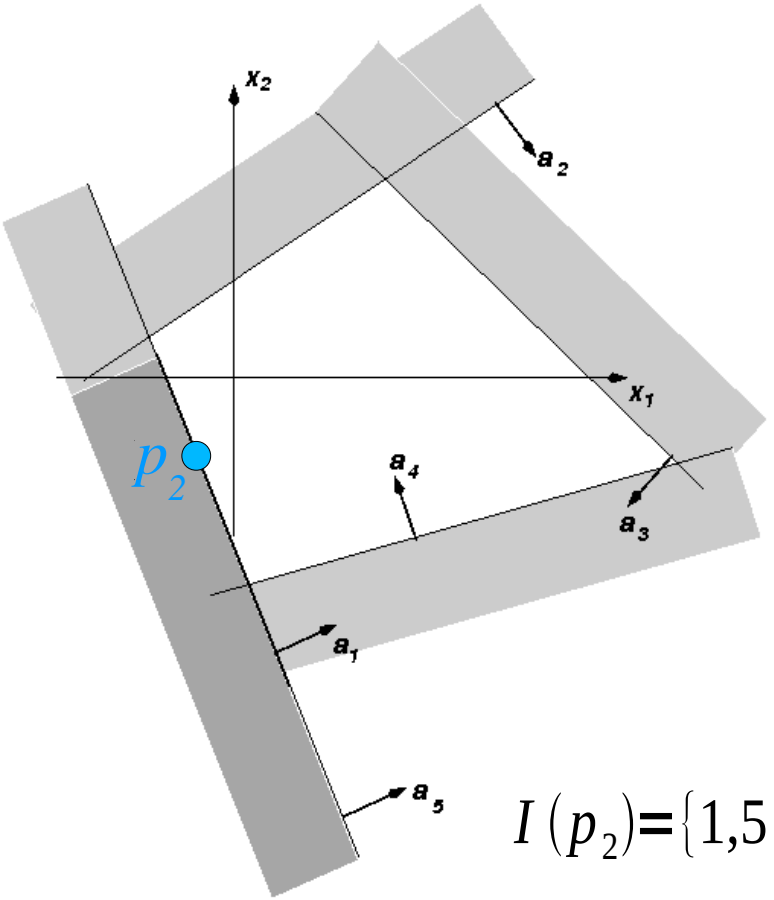
Polyhedra in a formal language

But careful:



$I(p_1)=\{1,2,5\}$

A vertex with $n+1$ active constraints.



$I(p_2)=\{1,5\}$

Not a vertex, but n constraints are active.

Polyhedra in a formal language

Definition:

Let $P \subset \mathbb{R}^n$ be a polyhedron defined by

$$P = \{x \in \mathbb{R}^n : a_i^T x \geq b_i \text{ for } i = 1 \ldots m_1, \quad a_i^T x = b_i \text{ for } i = m_1 + 1 \ldots m_2\}$$

We call $p \in \mathbb{R}^n$ a *basic solution of P* if:

- all equality constraints are satisfied at p
- the set

$$\{a_i : i \in I(p)\}$$

  contains n vectors that are linearly independent.

Polyhedra in a formal language

Note:
The condition that

$$\{a_i : i \in I(p)\}$$

contains n vectors that are linearly independent is equivalent to saying that

- The vectors a_i form a *basis* of R^n. This is why these points are called "basic" (i.e. "basic" as in "basis", not "fundamental").
- If we group the vectors and corresponding right hand sides into a linear system

$$a_i^T x = b_i, \qquad i \in I(p)$$

then the solution will be unique and will equal p. This is why these points are called "solutions".

Polyhedra in a formal language

Definition:

Let $P \subset \mathbb{R}^n$ be a polyhedron defined by

$$P = \{x \in \mathbb{R}^n : a_i^T x \geq b_i \text{ for } i = 1 \ldots m_1, \quad a_i^T x = b_i \text{ for } i = m_1 + 1 \ldots m_2\}$$

We call $p \in \mathbb{R}^n$ a *degenerate basic solution of P* if:

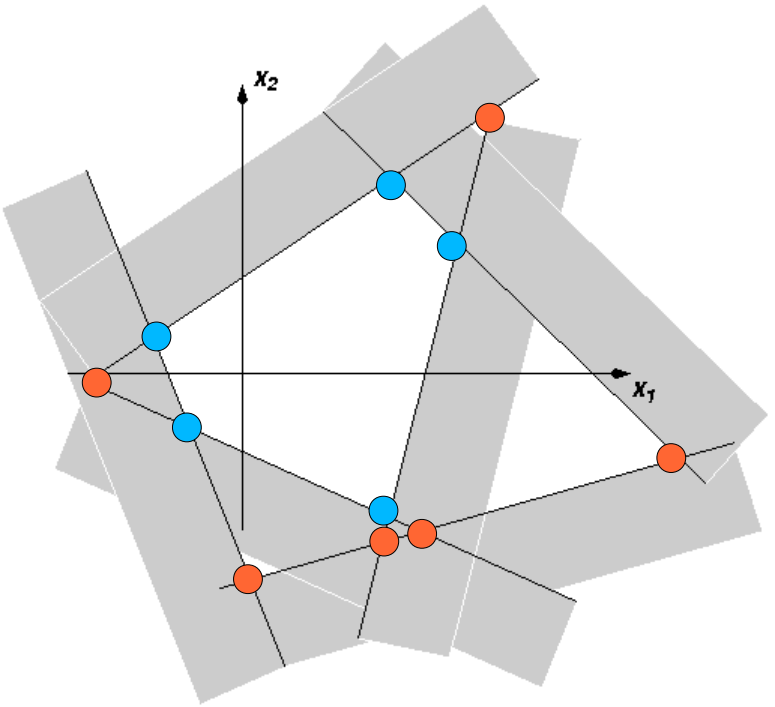
- all equality constraints are satisfied at p
- the set

  $$\{a_i : i \in I(p)\}$$

  contains n vectors that are linearly independent
- this set has more than n elements, i.e. more than n constraints are active at p.

Polyhedra in a formal language

Definition:
A basic solution *p* is called a *feasible basic solution* if in addition to the equality and active inequality constraints also the inactive inequality constraints are satisfied.



- The only five feasible basic solutions
- Some of the non-feasible basic solutions

Polyhedra in a formal language

Theorem:
A feasible basic solution is a vertex is an extreme point.

In other words: Let p be a point in a non-empty polyhedron P. Then it is either none or all of the following:

- p is a feasible basic solution
- p is a vertex of P
- p is an extreme point of P

Polyhedra in a formal language

Theorem:
A polyhedron can only have finitely many vertices.

Note: In fact, a polyhedron

$$P=\left\{x\in\mathbb{R}^n : a_i^T x\geq b_i \text{ for } i=1\ldots m',\quad a_i^T x=b_i \text{ for } i=m'+1\ldots m\right\}$$

can have at most

$$\binom{m}{n}=\frac{m!}{(m-n)!\,n!}$$

basic solutions. However, this can be a very large number!

Example: The unit cube in n dimensions has $2^n\approx 10^{0.3n}$ vertices.

Polyhedra in a formal language

Note: A polyhedron

$$P=\left\{x\in\mathbb{R}^n : a_i^T x\geq b_i \text{ for } i=1\ldots m',\quad a_i^T x=b_i \text{ for } i=m'+1\ldots m\right\}$$

can have at most

$$\binom{m}{n}=\frac{m!}{(m-n)!\,n!}$$

basic solutions. However, not all of them are feasible. In fact, at every feasible basic solution, the *m*-*m'* equality constraints need to all be active, so that there can be at most

$$\binom{m'}{n-(m-m')}=\frac{m'!}{(n-m+m')!\,(m'-(n-(m-m')))!}=\frac{m'!}{(n-(m-m'))!\,(m-n)!}$$

feasible basic solutions (vertices).

Possible solutions of linear programs

Theorem:
If the feasible set of a linear program is non-empty and has at least one vertex, then exactly one of the following is true:

- The minimum of the objective function over the feasible set is $-\infty$, or
- among all solutions of the linear program is always at least one vertex of the feasible set.

Part 15

Linear programming 2: A naïve solution algorithm

$$\text{minimize} \quad c^T x$$
$$Ax \geq b$$

A naïve algorithm

Theorem: A polyhedron can only have finitely many vertices.

Corollary: One (simplistic) way to find a solution to a linear program is the following procedure:

1. Convince ourselves that the linear program has a bounded solution
2. Find all *basic solutions*
3. Among these, identify all *feasible* basic solutions by testing which of the basic solutions satisfy all constraints. These are the vertices of the feasible set
4. Among these, find the vertex (feasible basic solution) or vertices that have the lowest value of the objective function. These are the solution(s) of the problem

A naïve algorithm

Practical implementation of step 2:
A basic solution of a problem with constraints

$$Ax \geq b, \quad \text{or equivalently} \quad a_i^T x \geq b_i, i = 1 \ldots m$$

is a point x at which n linearly independent constraints are active. (In addition to possibly more constraints that then need to be linearly dependent on the previous ones.)

One way to enumerate all basic solutions is by enumerating all subsets of n constraints among the total of m constraints:

- Take all possible selections I of n indices within the set $[1,m]$
- For each I see if the constraints are linearly independent. If so, find the (unique) point x at which

$$a_i^T x = b_i \quad \forall i \in I$$

  This is a basic solution.

A naïve algorithm

Practical implementation of step 2 – example:
If we have 3 variables $x=\{x_1,x_2,x_3\}$ and 8 constraints

$$Ax \geq b, \quad \text{or equivalently} \quad a_i^T x \geq b_i, i=1...8$$

then we need to

- try first the set $I=\{1,2,3\}$
- see if the 3x3 matrix $A_I=\begin{pmatrix} a_1^T \\ a_2^T \\ a_3^T \end{pmatrix}$ has full rank
- If so, then the equation $A_I x_I = b_I$ is unique and x_I is a basic solution
- Continue with the sets $I=\{1,2,4\}, \{1,2,5\}, ..., \{6,7,8\}$ and do the same steps

A naïve algorithm

Practical implementation of step 3:
Now that we have a basic solution x, we need to determine which of those are feasible.

By construction, we already know that

$$a_i^T x = b_i \quad \forall i \in I$$

but we also have to check the remaining m-n constraints:

- Go through all indices $i \notin I$
- If for any of these indices $a_i^T x < b_i$ then this basic solution is infeasible, i.e. it can not be a feasible basic solution and therefore not be a vertex. We can discard this basic solution
- If the basic solution turns out to be feasible with regards to all other constraints, then it must be a vertex

A naïve algorithm

Practical implementation of step 4:
Now that we have a feasible basic solution x, we need to determine which one is the best with regard to the objective function.

To do this:

- For every set of n indices I compute x_I as the basic solution
- If it turns out to be feasible, compute $f(x_I)=c^T x_I$
- If this value $f(x_I)$ is bigger than the previously smallest one seen, then forget about this feasible basic solution and move on to the next set of n indices
- If this value $f(x_I)$ is smaller than the previously smallest one seen, then save $f(x_I)$ and x_I for later comparison and move on to the next set of n indices

A naïve algorithm

Assessment of the algorithm:

- The algorithm works and finds the solution if there exists a bounded solution
- The algorithm is unaffected by degeneracy
- The algorithm is slow because it needs to test *every* vertex of the feasible region
- Since the number of vertices in general grows combinatorically with the number of variables and constraints, the run time of the algorithm grows exponentially as

$$\binom{m}{n}\left(n^3+(m-n)n\right)\approx(2.5\gamma)^n \qquad \text{if } m=\gamma n$$

- Such algorithms are not suited for practical, large-scale problems with thousands or millions of variables and constraints

Part 16

Linear programming 3: Dantzig's *simplex algorithm*

$$\begin{aligned} \text{minimize} \quad & c^T x \\ & Ax = b \\ & x \geq b \end{aligned}$$

The idea

Instead of enumerating and testing *all* vertices, we should:

- Start with a feasible basic solution (vertex)
- Tests its neighbors and go to one with a lower objective function value
- Since the objective function values are a decreasing sequence, cycling is not possible; since there are only finitely many vertices, the algorithm must terminate in a finite number of steps
- Since we only accept vertices with lower objective function values, we hope that we need to visit far fewer than all vertices

This is the basic idea of Dantzig's *simplex* algorithm

Preliminary considerations 1

Theorem:
Let the feasible set of a linear program in standard form be described by the equations

$$P=\{x\in\mathbb{R}^n:\quad Ax=b, A\in\mathbb{R}^{m\times n}, m\leq n,\quad x\geq 0\}$$

where the matrix A does not have full row rank (i.e. its rows are linearly dependent).
If P is not empty, then there exists a matrix with full row rank so that

$$Q=\{x\in\mathbb{R}^n:\quad \tilde{A}x=b, \tilde{A}\in\mathbb{R}^{m'\times n}, m'<m\leq n,\quad x\geq 0\}$$

and $Q=P$.

Due to this equivalence, we will in the following always assume that A has full row rank.

Preliminary considerations 2

The feasible sets of linear programs in standard form are also polyhedra and are described by the equations

$$P = \{x \in \mathbb{R}^n : \quad Ax = b, A \in \mathbb{R}^{m \times n}, m \leq n, \quad x \geq 0\}$$

Then at any feasible basic solution (vertex of P) the following holds true:

- all m equality constraints are active
- at least n-m variables x_i are zero
- if a basic solution is non-degenerate, exactly n-m variables are zero

Standard form is so convenient because we don't just know that *n-m* inequalities are active, but can associate them with vector components!

Preliminary considerations 3

Definition:

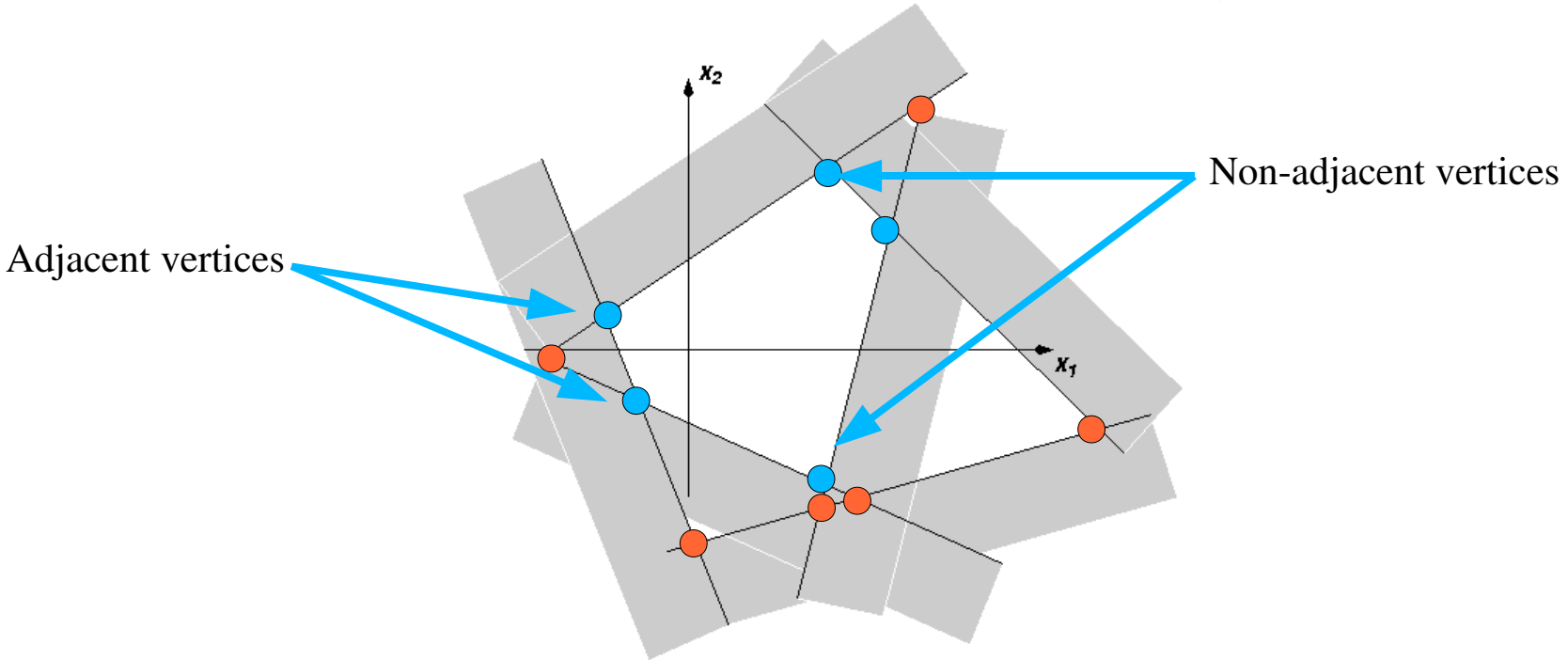
Let $P \subset \mathbb{R}^n$ be a polyhedron. Let $p_1, p_2 \in \mathbb{R}^n$ be two basic solutions of P. We call them *adjacent* if

$$\{a_i : i \in I(p_1)\}$$

and

$$\{a_i : i \in I(p_2)\}$$

contain a common set of *n-1* vectors that are linearly independent.



Preliminary considerations 3

In particular, for standard form:
Since equality constraints always have to be active, every feasible basic solutions of a polyhedron in standard form

$$P=\{x\in\mathbb{R}^n:\quad Ax=b, A\in\mathbb{R}^{m\times n}, m\leq n,\quad x\geq 0\}$$

must have *m* active equality constraints, and

- exactly *n*-*m* variables x_i that are zero (if the basic solution is not degenerate)
- or more than *n*-*m* variables x_i that are zero (if the basic solution is not degenerate).

In the non-degenerate case, two basic solutions are adjacent if they differ in exactly one pair of variables x_i that are zero/nonzero at one vertex and nonzero/zero at the other!

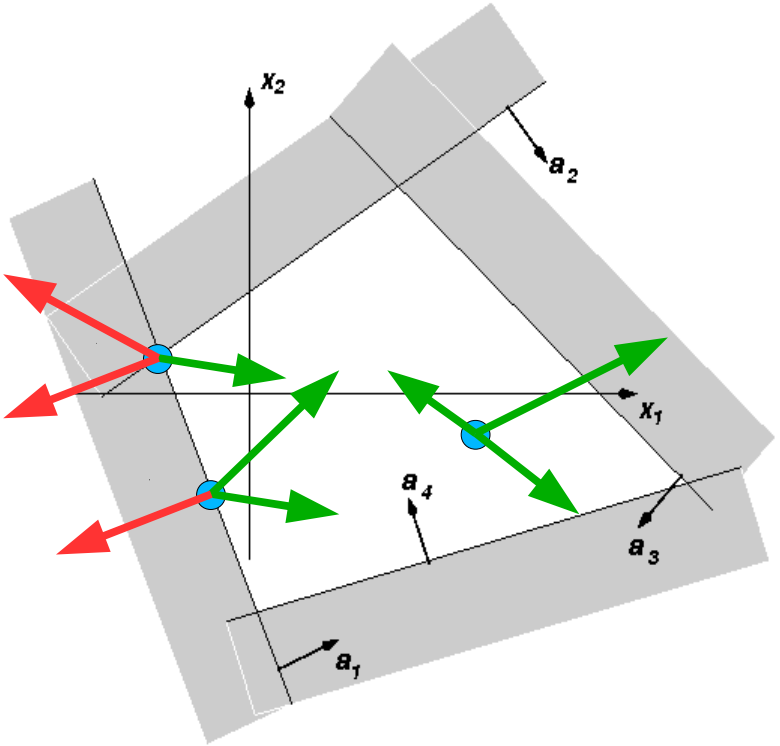
Preliminary considerations 4

Definition:

Let x be a point in a polyhedron P. Then we call a vector d a *feasible direction* if

$$\exists \theta > 0: \quad x + \theta d \in P$$

Example:



feasible directions

infeasible directions

Preliminary considerations 4

In particular:

Let p be a non-degenerate vertex of a polyhedron P described in standard form:

$$P=\left\{x\in\mathbb{R}^n:\ \ Ax=b, A\in\mathbb{R}^{m\times n}, m\leq n,\ \ x\geq 0\right\}$$

Let $I(p)$ be the active set of constraints at p. Then any *feasible direction* d needs to satisfy the following conditions:

$$\begin{aligned} Ad&=0\\ d_i&\geq 0 \quad \forall\, i+m\in I(p)\end{aligned}$$

Preliminary considerations 4

Conversely:
Let I, $\#I=n$ be a set of indices. Assume the associated constraints are linearly independent. Then I describes a vertex p of a polyhedron

$$P=\left\{x\in\mathbb{R}^n:\quad Ax=b, A\in\mathbb{R}^{m\times n}, m\leq n,\quad x\geq 0\right\}$$

If the vertex *is not* degenerate, then any direction that satisfies

$$\begin{aligned} Ad&=0\\ d_i&\geq 0 \quad \forall\, i+m\in I(p)\end{aligned}$$

is feasible. If the vertex *is* degenerate, then we have to require that

$$\begin{aligned} Ad&=0\\ d_i&\geq 0 \quad \forall\, i+m\in I(p)\\ d_i&\geq 0 \quad \forall\, i+m\notin I(p), x_i=0\end{aligned}$$

Note: These relations can be used to test whether a proposed direction is feasible or not.

The simplex algorithm, non-degenerate case

The simplex algorithm works on standard form:

$$P=\{x\in\mathbb{R}^n:\ \ Ax=b, A\in\mathbb{R}^{m\times n}, m\le n,\ \ x\ge 0\}$$

At every step of the simplex algorithm, the current state is described by the following pieces of information:

- A set of indices H, $\#H=m$, called the *basis*. H describes the variables that are *not* bound by the constraints and so is somewhat complementary to the set of active indices I.
- H defines a *basis matrix* $B=A_H$ that consists of the *columns* of A listed in H. B is the "interesting" part of the matrix A_I.
- H defines a basic solution x of a polyhedron (which in the algorithm will always be feasible) that satisfies

$$Bx_H=b$$
$$x_{H^c}=0$$

Due to non-degeneracy, $x_H>0$ for each vector element.

The simplex algorithm, non-degenerate case

Why bases instead of active sets:

Let the polyhedron be described by

$$P=\left\{x\in\mathbb{R}^n:\quad Ax=b, A\in\mathbb{R}^{m\times n}, m\le n,\quad x\ge 0\right\}$$

Then at every (non-degenerate) basic solution we have an active set I with exactly n elements. These are:

- The indices $1...m$ corresponding to equality constraints
- A subset of size $(n-m)$ of the indices $m+1...m+n$ corresponding to the positivity constraints

The linear system that describes the basic solution is therefore:

$$\begin{aligned} A\,x&=b\\ x_{I_i}&=0\quad i=m+1...n\end{aligned}$$

x therefore consists of two parts: components x_H that are not necessarily zero, and x_{H^c} that must be zero. Therefore, in the first equation, only columns listed in H participate, i.e. the basis matrix B.

The simplex algorithm, non-degenerate case

The main idea of the simplex algorithm:

Let H,B be the current basis and basis matrix, and x be a non-degenerate feasible basic solution defined by H.

- To move from x_H to another vertex, we need to release one non-basic variable j from its constraint $x_j=0$ and make it positive.

- Our search direction should therefore be

$$d_j=1$$
$$d_i=0 \quad i\notin H, i\neq j$$

- The basic components need to satisfy

$$A(x+d)=b \quad \rightarrow \quad Ad=0 \quad \rightarrow \quad Bd_H+A_j=0 \quad \rightarrow \quad d_H=-B^{-1}A_j$$

where A_j denotes the jth column of A.

- The vector d so defined is called the *jth basic direction.*

The simplex algorithm, non-degenerate case

Theorem:

Let H,B be the current basis and basis matrix, and x be a non-degenerate feasible basic solution defined by H.

Then for every $j \notin H$ the direction defined by

$$d_j = 1$$
$$d_i = 0 \quad i \notin H, i \neq j$$
$$d_H = -B^{-1} A_j$$

is feasible.

Note 1: If x is degenerate, then d is a feasible direction if and only if

$$(d_H)_i \geq 0 \quad \forall i \in H, x_i = 0$$

Note 2: Feasibility should not be a surprise – we still have *(n-1)* constraints that are active. d is constructed to lie in this 1d subspace.

The simplex algorithm, non-degenerate case

Theorem:

Let *H,B* be the current basis and basis matrix, and *x* be a feasible basic solution defined by *H*.

Then for every $j \notin H$ the *j*th basic direction is a direction of descent of the objective function if the *reduced cost* satisfies

$$\bar{c}_j = c_j - c_H^T B^{-1} A_j < 0$$

The simplex algorithm, non-degenerate case

Theorem:

Let H,B be the current basis and basis matrix, and x be a feasible basic solution defined by H. Then:

- If $\bar{c}_j = c_j - c_H^T B^{-1} A_j \geq 0 \quad \forall\, j \notin H$ then x is optimal

- If x is optimal and non-degenerate, then $\bar{c}_j \geq 0 \quad \forall\, j \notin H$

Note: The first condition can be used to test whether a vertex x is optimal – we only need to compute all reduced costs!

The simplex algorithm, non-degenerate case

Line search:

Let H,B be the current basis and basis matrix, and x be a non-degenerate feasible basic solution defined by H.

Let d be a feasible basic direction. Then:

- $x+\theta d$ satisfies $(n-1)$ constraints for sufficiently small step lengths θ.
- $x+\theta d$ is feasible for

$$\theta \leq \theta^* = \min_{i \in H,\, d_i < 0} \left(-\frac{x_i}{d_i} \right)$$

The simplex algorithm, non-degenerate case

Algorithm:

Let H,B be the current basis and basis matrix, and x be a non-degenerate feasible basic solution defined by H. Then perform the following steps:

Let $j=1...n$, $j \notin H$

Compute $d_H = -B^{-1}A_j$, $\bar{c}_j = c_j - c_H^T B^{-1} A_j$

If $\bar{c}_j < 0$ then

- compute $\theta^* = \min_{i \in H, d_i < 0} \left(-\frac{x_i}{d_i} \right)$ and let l be the index for which the minimum is attained
- set $x_j \leftarrow \theta^*$, $x_H \leftarrow x_H + \theta^* d_H$
  $H \leftarrow H \setminus \{l\} \cup \{j\}$
  $B_l \leftarrow A_j$
- start over

Try the next j. If no j allows has negative reduced costs, then we have a solution.

The simplex algorithm, non-degenerate case

Note:

If all components of *d* turn out to be positive, then we have a direction in which every point is feasible and we can choose $\theta^* = \infty$. This means that the problem has no bounded solution at the point where we compute the step length we have already determined that *d* is a direction of descent.

Theorem:

Assume the basic matrix $B = \left(A_{H_1} A_{H_2} \ldots A_{H_m} \right)$ at the beginning of the iteration has full rank. Then the new basic matrix

$$B = \left(A_{H_1} A_{H_2} \ldots A_j \ldots A_{H_m} \right)$$

also has full rank.

The simplex algorithm, non-degenerate case

Theorem:

The algorithm just outlined terminates after finitely many steps with one of the following results:

- If all reduced costs $\bar{c}_j$ are non-negative, then the current vertex is a solution of the minimization problem
- If at a vertex at least one of the reduced costs $\bar{c}_j$ is negative but the corresponding search direction satisfies $d>0$, then the linear problem is unbounded from below and has no bounded solution.

The simplex algorithm, non-degenerate case

Note: In our algorithm, we test

- Is one of the reduced costs $\bar{c}_j$ negative
- If so, let j enter the basis (i.e. release its constraint and make it a free variable)

Question:

What do we do if the reduced costs are negative for more than one index?

We could choose any variable with a negative reduced cost, but maybe some strategies will lead to algorithms that require fewer iterations than others.

The simplex algorithm, non-degenerate case

Question: What do we do if the reduced costs are negative for more than one index?

Answer: There are many *pivoting* strategies, for example we could

- Choose that index j for which the reduced cost is the most negative
- Choose that index j for which $\theta^* \bar{c}_j$ is the most negative
- Try to choose an index j that has not recently been chosen
- ....
- Bland's rule: Take the first index j for which the reduced cost is negative

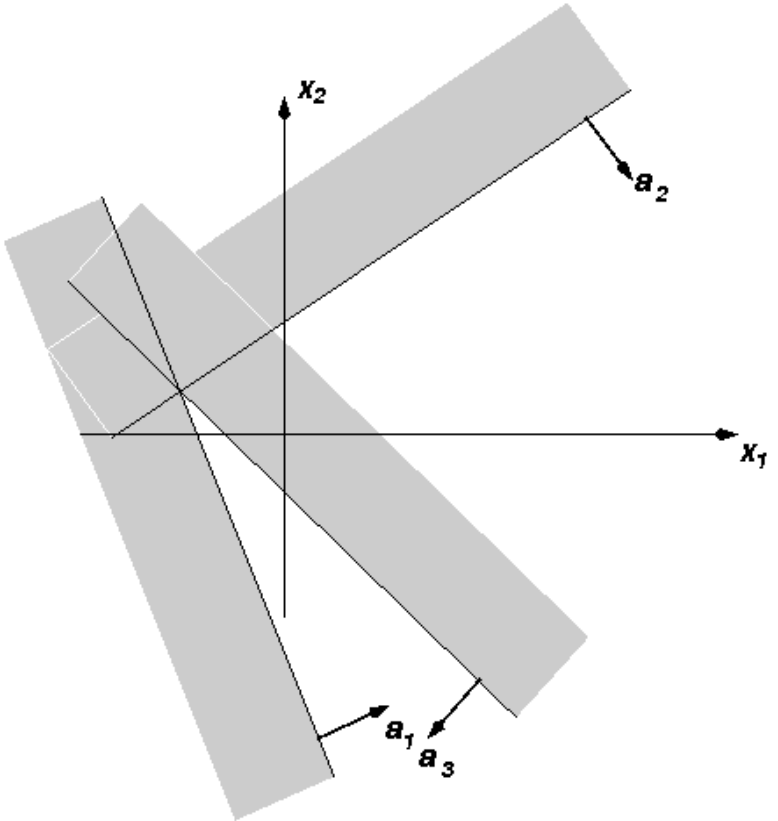
Note: More complex strategies often reduce the number of iterations at the cost of more expensive iterations. Bland's rule is a good choice to avoid *cycling* in the degenerate case.

The degenerate case

Two things can happen in the degenerate case:

- We want to release constraint j and let x_j enter the basis but we can't go into direction $d=-B^{-1}A_j$ because we immediately hit a previously active constraint that was not part of the basis H.

  In other words, we find that $\theta^*=0$. What do we do?



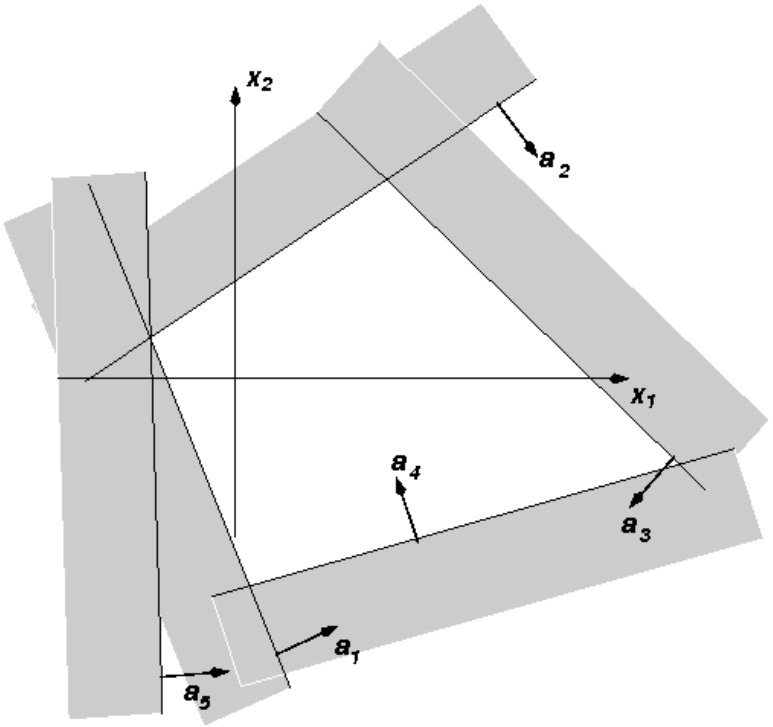
Example (not for standard form linear problems): Constraints 1 and 2 are active. We want to release constraint 1 but we can't move away from it.

The degenerate case

Two things can happen in the degenerate case:

- We let x_j enter the basis which then has $(n-1)$ active constraints. We move in direction $d=-B^{-1}A_j$ but at θ^* we find more than one new constraint.

  In this case, which of these constraints should exit the basis?



Example (not for standard form): Constraints 2 and 3 are active. We want to release constraint 3 and move along constraint 2 but then both constraints 1 and 5 become active.

The degenerate case

Case 1: We find that $\theta^* = 0$.

In this case, we know that more than *n* constraints are active at the current vertex, i.e. some of the basic ("free") variables in *H* are zero.

The question is then which of these variables to throw out of the basis in response to letting x_j enter the basis without taking a step that decreases the objective function?

The question is important because we want to avoid *cycling*, i.e. returning to the same basis after a number of steps without reducing the objective function.

Answer: There are a number of *pivoting* strategies. The simplest is Bland's rule – if there are multiple constraints that become active, take the one with the smallest index.

The degenerate case

Case 2: We find that $\theta^* > 0$ but that more than one constraint becomes active.

The question is then which of the variables whose constraints become active to throw out of the basis (i.e. let them "exit the basis") in response to letting x_j enter the basis?

Answer: There are a number of *pivoting* strategies. The simplest is Bland's rule – if there are multiple constraints that become active, take the one with the smallest index.

The degenerate case

Theorem:

Using Bland's rule for selecting

- which variable will enter the basis
- which variable will exit the basis if there are multiple that are eligible

avoids the problem of cycling and therefore guarantees that the algorithm terminates in finite time.

Starting the simplex method

Problem: The simplex algorithm needs to start from a feasible basic solution.

Unfortunately, finding any vertex is almost as expensive as finding the best one.

Typical strategy: The "two-phase simplex method"

Starting the simplex method

Starting point: Consider the problem

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & c^T x \\ & Ax = b \\ & x \geq 0 \end{aligned}$$

Without loss of generality, we can assume that $b \geq 0$. We seek a feasible vertex of the feasible set and a corresponding basis.

Consider now the auxiliary problem

$$\begin{aligned} \min_{x \in \mathbb{R}^n, y \in \mathbb{R}^m} \quad & (1,1,\ldots,1)^T y \\ & Ax + y = b \\ & x \geq 0 \\ & y \geq 0 \end{aligned}$$

Notes:

- For no (feasible) choice of y can the objective function of the auxiliary problem be negative.
- It is zero if x is a feasible point of the original problem.
- It is positive if there is no feasible point of the original problem.

Starting the simplex method

Consider the auxiliary problem

$$\begin{aligned} \min_{x\in\mathbb{R}^n, y\in\mathbb{R}^m} \quad & (1,1,\ldots,1)^T y \\ & Ax + y = b \\ & x \geq 0 \\ & y \geq 0 \end{aligned}$$

We can solve this problem using the simplex algorithm as discussed, starting with the feasible vertex *(x=0, y=b)*. The algorithm terminates with either of these outcomes:

- *The objective function is positive:* The original problem has no feasible point
- *The objective function is zero:* Then, *y=0* and *x* is a feasible vertex with respect to the original problem.
  Furthermore, at least *n* variables among the *x,y* are at their bounds, and at most *m* variables are greater than zero

Starting the simplex method

However, we need more to start the simplex algorithm on the original problem:

We need a basis H, which then implies the location of the vertex as well as the basis matrix B.

Can we use the final basis H_{aux} (consisting of m free variables) of the simplex algorithm applied to the auxiliary problem?

- If the vertex we find is non-degenerate, then all m entries in H_{aux} are components of x (because $y=0$) and we can use $H=H_{aux}$
- If the vertex is degenerate, then H_{aux} contains variables that are zero at the solution and could contain auxiliary variables y.
  We then need a procedure to "drive artificial variables out of the basis".

Starting the simplex method

Driving artificial variables out of the basis:

H_{aux} has m entries but some of them correspond to auxiliary variables and only $k<m$ non-artificial entries.

We need a basis H with m non-artificial entries. We can let currently non-basic variables x_i (for which $x_i=0$) enter this basis, but we need to make sure that the corresponding basis B retains full rank. The following procedure guarantees this:

While $k<m$:

- Let $l>k, l\leq m$ be an index in H_{aux} so that $(H_{aux})_l$ corresponds to an artificial variable
- Choose $j<m$ so that $\left(B_{aux}^{-1} A_j\right)_l \neq 0$
- Replace $H_{aux} \leftarrow H_{aux} \setminus \left\{\left(H_{aux}\right)_l\right\} \cup \{j\}$ and re-assemble B_{aux}.

The two-phase simplex algorithm

1. Remove linearly dependent constraints from the matrix A
2. Multiply constraints as necessary so that $b \geq$ ·
3. Introduce artificial variables $y \in \mathbb{R}^m$ and solve the auxiliary problem
4. If the objective function at the solution is positive the original problem does not have a feasible solution. Terminate.
5. Given H_{aux}, B_{aux}, drive artificial variables out of the basis until they only contain non-artificial variables
6. Set $H = H_{aux}$, $B = B_{aux}$

7. Solve the original problem using the simplex algorithm

Implementing the simplex method

Naïve implementation:

In a naïve implementation, in each iteration we have to

- Compute B^{-1} $O(m^3)$
- Compute reduced costs $O(m(n-m))$
- Compute a search direction $O(m^3)$
- Compute a step length $O(m)$

Thus, the total effort is $O(m^3+mn)$ per visited vertex.

Better implementations: We can achieve the same result using only $O(m^2+mn)$ operations per visited vertex. This uses, for example, updating rules to compute B^{-1} from the basis matrix used in the previous iteration.

Complexity of the simplex method

Overall complexity: In the best case, each iteration costs $O(m^2+mn)$. How many iterations does one need?

In practice: In most applications, the number of iterations appears to be a small multiple of the number of constraints m.

In theory: For most pivoting rules, examples are known where every single vertex is visited. These examples typically involve the 2^n vertices of the unit cube with n variables and $m=2n$ constraints.

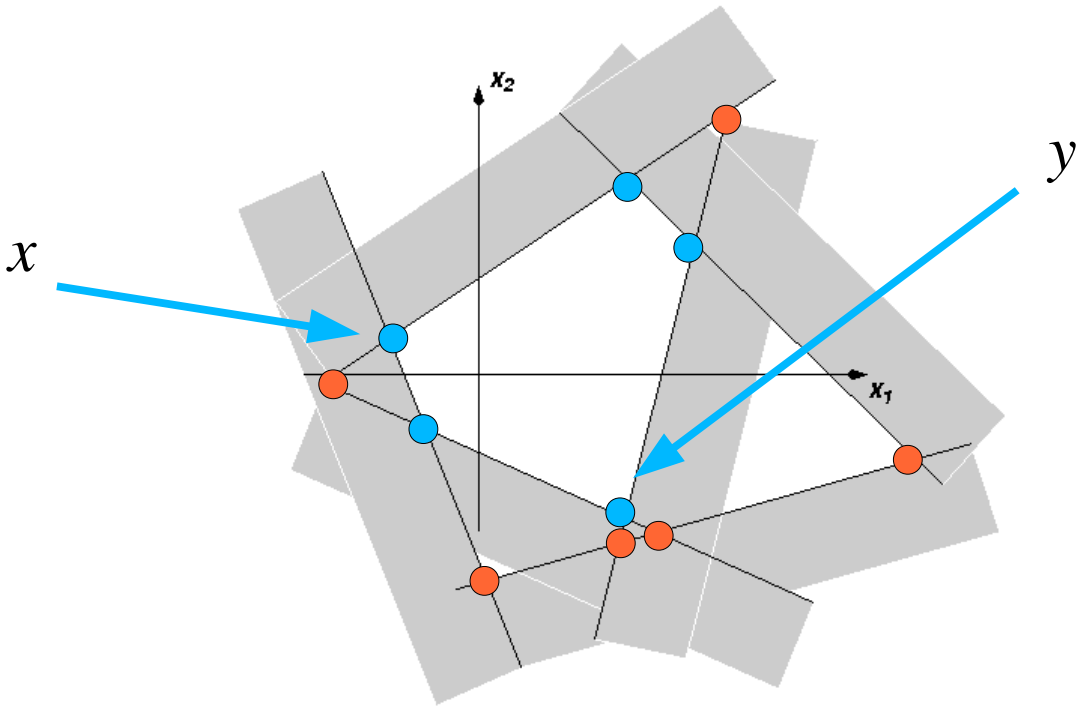
Questions:

- How often does this happen?
- Is this a property of individual algorithms/pivoting rules?
- Is this a property of linear problems?

Complexity of the simplex method

Definition: The distance $d(x,y)$ between two vertices x,y of a polyhedron P is the length of the shortest sequence of steps through intermediate vertices.

Definition: The diameter $diam(P)$ of a polyhedron P is the maximal distance between any two vertices of P.



$d(x,y)=2$

$diam(P)=2$

Complexity of the simplex method

Corollary: For any pivoting rule, for a bad choice of initial vertex, we always need to expect that we need at least *diam(P)* iterations.

Question: Do we know anything about *diam(P)* for given *n,m*?

Complexity of the simplex method

Definition:

Let $$\Delta(n,m) = \max_{\substack{A \in \mathbb{R}^{m \times n} \\ P = \{x \in \mathbb{R}^n : Ax \geq b\} \\ P \text{ is bounded}}} \operatorname{diam}(P)$$

$$\Delta_u(n,m) = \max_{\substack{A \in \mathbb{R}^{m \times n} \\ P = \{x \in \mathbb{R}^n : Ax \geq b\}}} \operatorname{diam}(P)$$

Corollary:

$$\Delta(n,m) \leq \Delta_u(n,m)$$

$$\Delta(2,m) = \lfloor \frac{m}{2} \rfloor$$

$$\Delta_u(2,m) = m-2$$

Complexity of the simplex method

Hirsch conjecture:

$$\Delta(n,m) \leq m-n$$

Consequence: This would imply that we could hope to find a pivoting rule that always terminates in *O(m)* iterations for bounded problems.

Theorem:

$$\Delta(n,m) \leq \Delta_u(n,m)$$

$$m-(n-\lfloor \frac{n}{5} \rfloor) \leq \Delta_u(n,m) \leq (2n)^{\log_2 m}$$

In other words, the best known upper bound for the diameter of unbounded polyhedra is not exponential, but worse than polynomial in *n,m*.

Complexity of the simplex method

Theorem (Borgwardt 1982):

Consider solving the following problem with the "shadow vertex" variant of the simplex algorithm

$$\min_{x \in \mathbb{R}^n} c^T x$$
$$a_i x \geq 1, \quad i = 1 \ldots m$$

where the vectors c, a_i are chosen randomly in $\mathbb{R}^n \setminus \{0\}$.

Then the *average* number of iterations necessary is less than

$$17 n^3 m^{\frac{1}{n-1}}$$

Note: This does not match practical experience.

Complexity of the simplex method

Theorem (Haimovich, Adler 1982):

Consider solving the following problem with the "shadow vertex" variant of the simplex algorithm

$$\min_{x \in \mathbb{R}^n} c^T x$$
$$a_i x \geq b_i, \quad i = 1 \ldots m$$

where the vectors c, a_i, b are chosen randomly with some assumptions.

Then the *average* number of iterations necessary is less than

$$n \frac{m-n+2}{m+1}$$

Under less stringent assumptions, the number of iterations is bounded by

$$C \min\left\{(m-n)^{2,} n^2\right\}$$

Part 17

Abstract duality

Weak duality

Theorem:

Consider any function $F(x,y): X \times Y \to \mathbb{R}$, then $\forall x \in X, y \in Y$

$$F_*(y) = \min_{x' \in X} F(x',y) \;\leq\; F(x,y) \;\leq\; \max_{y' \in Y} F(x,y') = F^*(x)$$

Corollary (weak duality):

$$\max_{y \in Y} \min_{x \in X} F(x,y) \;\leq\; \min_{x \in X} \max_{y \in Y} F(x,y)$$

MinMax vs MaxMin

Example:

Assume there are two players A,B. In a game, player A can play moves x from a set X, player B can play moves y from a set Y. $F(x,y)$ will be the payout from A to B.

If A gets to play first, she would like to play a move x^* so that the payout is minimal even if B responds with her best move:

$$\max_{y \in Y} F(x,y) = F^*(x) \rightarrow \min!$$

With A's best (defensive) move, the payout from A to B will be

$$\min_{x \in X} \max_{y \in Y} F(x,y)$$

On the other hand, if A has the advantage of going second and react to player B's move, the payout will be

$$\max_{y \in Y} \min_{x \in X} F(x,y) \leq \min_{x \in X} \max_{y \in Y} F(x,y)$$

MinMax vs MaxMin

Example:

Player A can play moves x_1, x_2, player B can play y_1, y_2. The payout from A to B is given by the following matrix:

$$\begin{array}{c} \\ x_1 \\ x_2 \end{array} \begin{array}{c} \begin{array}{cc} y_1 & y_2 \end{array} \\ \begin{pmatrix} -1 & 2 \\ 4 & 3 \end{pmatrix} \end{array}$$

If A gets to move first, she should play x_1, to which B will counter with y_2. As a result A will have to pay

$$\min_{x \in X} \max_{y \in Y} F(x, y) = 2$$

If B gets to move first, the moves will be y_2 and x_1, with result

$$\max_{y \in Y} \min_{x \in X} F(x, y) = 2$$

MinMax vs MaxMin

Example:

Player A can play moves x_1, x_2, player B can play y_1, y_2. The payout from A to B is given by the following matrix:

$$\begin{array}{c} \\ x_1 \\ x_2 \end{array} \begin{array}{c} \begin{array}{cc} y_1 & y_2 \end{array} \\ \begin{pmatrix} -1 & 2 \\ 4 & 1 \end{pmatrix} \end{array}$$

If A gets to move first, she should play x_1, to which B will counter with y_2. As a result A will have to pay

$$\min_{x \in X} \max_{y \in Y} F(x, y) = 2$$

If B gets to move first, the moves will be y_2 and x_2, with result

$$\max_{y \in Y} \min_{x \in X} F(x, y) = 1$$

Duality gap

Definition:

For a given function *F(x,y)* we have shown that

$$\max_{y \in Y} \min_{x \in X} F(x, y) \leq \min_{x \in X} \max_{y \in Y} F(x, y)$$

We call

$$G(F) = \min_{x \in X} \max_{y \in Y} F(x, y) - \max_{y \in Y} \min_{x \in X} F(x, y)$$

the *duality gap*.

Note:

Depending on *F(x,y)*, the duality gap can be zero, finite, or plus infinity.

Strong duality

Theorem (strong duality 1):

For a given function *F(x,y),* assume there exists a point *(x*,y*)* so that

$$F(x^*, y) \le F(x^*, y^*) \le F(x, y^*) \quad \forall x \in X, y \in Y$$

(such a point is called a *saddle point*). Then there holds

$$\max_{y \in Y} \min_{x \in X} F(x, y) = \min_{x \in X} \max_{y \in Y} F(x, y)$$

i.e. the duality gap *G(F)* is zero.

Theorem (strong duality 2):

The converse is also true: If the duality gap is zero, then there exists a saddle point.

Part 18

Lagrangian duality

Lagrangian optimization

Consider the following (possibly nonlinear) optimization problem:

$$\begin{aligned} &\text{minimize}_x && f(x) \\ &\text{subject to} && g(x) \geq 0 \\ & && x \in X \end{aligned}$$

Introduce the Lagrangian

$$L(x,\lambda) = f(x) - \lambda^T g(x)$$

$$L: X \times Y \to \mathbb{R}, \quad Y = \{\lambda \in \mathbb{R}^m : \lambda \geq 0\}$$

Define

$$L^*(x) = \max_{\lambda \in Y} L(x,\lambda)$$

Lagrangian optimization

Theorem:

The solution x^* of the optimization problem

$$\begin{aligned} &\text{minimize}_x && f(x) \\ &\text{subject to} && g(x) \geq 0 \\ & && x \in X \end{aligned}$$

equals the solution of the optimization problem

$$\text{minimize}_{x \in X} L^*(x) \quad = \quad \text{minimize}_{x \in X} \max_{\lambda \in Y} L(x, \lambda)$$

and $f(x^*)=L^*(x^*)$.

Note: This is trivially extended to problems with equality constraints. In this case the space Y places no restrictions on the sign of the Lagrange multiplier.

Lagrangian optimization

Definition:

We call

$$\begin{aligned} &\text{minimize}_x && f(x) \\ &\text{subject to} && g(x) \geq 0 \\ & && x \in X \end{aligned}$$

the *primal problem*. It corresponds to the min-max problem

$$\text{minimize}_{x \in X} L^*(x) \quad = \quad \text{minimize}_{x \in X} \max_{\lambda \in Y} L(x, \lambda)$$

Lagrangian optimization

Definition:

We call

$$\text{maximize}_{\lambda \in Y} L_*(\lambda) \quad = \quad \text{maximize}_{\lambda \in Y} \min_{x \in X} L(x, \lambda)$$

the *dual problem*. It can only be written in the form

$$\text{maximize}_{\lambda \in Y} \quad \hat{f}(\lambda)$$

if the minimizer x of $L(x,\lambda)$ for a given λ has a closed form solution $x(\lambda)$.

Lagrangian optimization

Theorem:

Consider an optimization problem

$$\begin{aligned} &\text{minimize}_x && f(x) \\ &\text{subject to} && g(x) \geq 0 \\ & && x \in X \end{aligned}$$

and its dual

$$\text{maximize}_{\lambda \in Y} L_*(\lambda) \quad = \quad \text{maximize}_{\lambda \in Y} \min_{x \in X} L(x, \lambda)$$

Let x^*, λ^* be their solutions. Then

$$L_*(\lambda^*) \quad \leq \quad f(x^*)$$

Proof: Follows from the weak duality theorem

Primal and dual problems

Example 1:

Consider $\text{minimize}_x \quad f(x)=x^2$

$\text{subject to} \quad g(x)=x-1\geq 0$

with solution $x^*=1$ and $f(x^*)=1$. Then

$$L(x,\lambda)=x^2-\lambda(x-1)$$

and the dual problem is

$$\text{maximize}_{\lambda\geq 0} L_*(\lambda), \qquad L_*(\lambda)=\min_x L(x,\lambda)=\lambda-\frac{1}{4}\lambda^2$$

The solution to the dual problem is

$$\lambda^*=2 \qquad L_*(\lambda^*)=1$$

Primal and dual problems

Example 2:

Consider

$$\text{minimize}_x \quad f(x)=e^x$$
$$\text{subject to} \quad g(x)=1-x^2\geq 0$$

with solution $x^*=-1$ and $f(x^*)=e^{-1}$. Then

$$L(x,\lambda)=e^x-\lambda(1-x^2)$$

and the dual problem is

$$\text{maximize}_{\lambda\geq 0}\, L_*(\lambda), \qquad L_*(\lambda)=\min_x L(x,\lambda)$$

Here, $L^*(\lambda)$ does not have a closed form expression since the minimizer of $L(x,\lambda)$ for a given λ does not have a closed form expression.

Note: This problem could have been avoided by writing the constraints as $-1\leq x\leq 1$.

Primal and dual problems

Example 3:

Consider

$$\begin{aligned} &\text{minimize}_x \quad f(x) = -x^2 \\ &\text{subject to} \quad x = 1 \\ &\qquad\qquad\quad x \in X = [0,2] \end{aligned}$$

with solution $x^*=1$ and $f(x^*)=-1$. Then

$$L(x,\lambda) = -x^2 - \lambda(x-1), \qquad x \in [0,1], \lambda \in \mathbb{R}$$

and the dual problem is

$$\text{maximize}_{\lambda \in \mathbb{R}} \, L_*(\lambda),$$

$$L_*(\lambda) = \min_{0 \le x \le 2} L(x,\lambda) = \begin{cases} \lambda & \text{for } \lambda < -2 \\ -4 - \lambda & \text{for } \lambda \ge -2 \end{cases}$$

Here, $L^*(\lambda^*)=-2$. Consequently, the *duality gap* is $G(L)=1$.

Primal and dual problems

Example 4:

Consider

$$\begin{aligned} &\text{minimize}_x \quad f(x) = -x^2 \\ &\text{subject to} \quad 0 \le x \le 1 \end{aligned}$$

with solution $x^*=1$ and $f(x^*)=-1$. Then

$$L(x,\lambda) = -x^2 - \lambda_1 x - \lambda_2 (1-x)$$

and the dual problem is

$$\begin{aligned} &\text{maximize}_{\lambda \ge 0} \, L_*(\lambda), \\ &\qquad L_*(\lambda) = \min_{x \in \mathbb{R}} L(x,\lambda) = -\infty \end{aligned}$$

Here, $L^*(\lambda^*)=-\infty$. Consequently, the *duality gap* is $G(L)=\infty$.

Primal and dual problems

Question:

Sometimes, it may be simpler to solve the dual rather than the primal problem.

In general, due to the duality gap, the solution of the dual problem only provides a *lower bound* on the optimal objective function value of the primal problem.

However, if we knew that the duality gap is zero, then we could get away by only solving the dual problem.

Convex duality

Theorem (Convex duality):

Let the primal problem be

$$\begin{aligned} &\text{minimize}_x \quad f(x) \\ &\text{subject to} \quad g_i(x) \geq 0, \quad i = 1 \ldots m \end{aligned}$$

where $f(x)$ is convex and each component $g_i(x)$ is concave (i.e. the feasible set is convex). Let x^* be a solution and assume that

$$\text{rank}(\nabla g(x^*)) = \min\{n, m\}$$

Then the duality gap for the Lagrangian is zero.

Convex duality

Note:

If the primal problem

$$\begin{aligned} &\text{minimize}_x \quad f(x) \\ &\text{subject to} \quad g_i(x) \geq 0, \quad i = 1 \ldots m \end{aligned}$$

does not have a feasible solution, then we say that

$$\min_x f(x) = \infty$$

and we will have

$$\max_\lambda L_*(\lambda) = \infty$$

Example:

Take $f(x)=x^2$, $g(x)=-1-x^2$.

Formulating dual problems: Wolfe dual

Example 2, revisited:

Consider $\text{minimize}_x \quad f(x)=e^x$

$\text{subject to} \quad g(x)=1-x^2\geq 0$

with solution $x^*=-1$ and $f(x^*)=e^{-1}$. Then $L(x,\lambda)=e^x-\lambda(1-x^2)$ and the dual problem is

$$\text{maximize}_{\lambda\geq 0} L_*(\lambda), \qquad L_*(\lambda)=\min_x L(x,\lambda)$$

$L^*(\lambda)$ does not have a closed form expression since the minimizer of $L(x,\lambda)$ for a given λ does not have a closed form expression.

However: Since f,g are differentiable, the minimizer $x^*(\lambda)$ must satisfy

$$\nabla_x L(x,\lambda)=0$$

Formulating dual problems: Wolfe dual

Example 2, revisited:

Consider $\begin{aligned} &\text{minimize}_x \quad f(x) \\ &\text{subject to} \quad g(x) \geq 0 \end{aligned}$ and assume $f, g \in C^1$

The Lagrangian is $L(x,\lambda) = f(x) - \lambda\, g(x)$ and the dual problem is

$$\text{maximize}_{\lambda \geq 0}\, L_*(\lambda), \qquad L_*(\lambda) = \min_x L(x,\lambda)$$

Alternatively: The dual problem can also be formulated as

$$\begin{aligned} &\text{maximize}_{x,\lambda \geq 0} \quad L(x,\lambda) \\ &\text{subject to} \quad \nabla_x L(x,\lambda) = 0 \end{aligned}$$

This formulation is called the *Wolfe dual*. The Wolfe dual is often simpler to formulate, since we do not need to eliminate x right away.

The dual of a linear program

Consider the following linear program:

$$\begin{aligned} \text{minimize}_x \quad & f(x) = c^T x \\ \text{subject to} \quad & Ax = b \\ & x \geq 0 \end{aligned}$$

The Lagrangian is $L(x, \{y, \lambda\}) = c^T x - y^T (Ax - b) - \lambda^T x$ and the Wolfe dual is

$$\begin{aligned} \text{maximize}_{x,\, y,\, \lambda \geq 0} \quad & L(x, \{y, \lambda\}) = (c^T - y^T A - \lambda^T) x + y^T b \\ \text{subject to} \quad & \nabla_x L(x, \{x, \lambda\}) = c^T - y^T A - \lambda^T = 0 \end{aligned}$$

We can use the constraint to eliminate x from the problem:

$$\begin{aligned} \text{maximize}_{y, \lambda \geq 0} \quad & b^T y \\ \text{subject to} \quad & c^T - y^T A - \lambda^T = 0 \end{aligned}$$

This is equivalent to the following linear problem:

$$\begin{aligned} \text{maximize}_y \quad & b^T y \\ \text{subject to} \quad & A^T y \leq c \end{aligned}$$

The dual of a quadratic program

Let Q be positive definite and consider

$$\begin{aligned} &\text{minimize}_x \quad f(x) = \frac{1}{2} x^T Q x + c^T x \\ &\text{subject to} \quad Ax \geq b \end{aligned}$$

The Lagrangian is $L(x,\lambda) = \frac{1}{2} x^T Q x + c^T x - \lambda^T (Ax - b)$ and the Wolfe dual is

$$\begin{aligned} &\text{maximize}_{x,\lambda \geq 0} \quad L(x,\lambda) = \frac{1}{2} x^T Q x + c^T x - \lambda^T (Ax - b) \\ &\text{subject to} \quad \nabla_x L(x,\lambda) = Qx + c - \lambda A = 0 \end{aligned}$$

We can again use the constraint to eliminate x from the problem:

$$\begin{aligned} &\text{maximize}_\lambda \quad -\lambda^T (A Q^{-1} A^T)\lambda + (AQ^{-1}c + b)^T \lambda - \frac{1}{2} c^T Q^{-1} c \\ &\text{subject to} \quad \lambda \geq 0 \end{aligned}$$

In other words: The dual of a linear quadratic program is a linear quadratic program! This pair of problems can not have a duality gap!

Part 19

Primal-dual methods for linear problems

Primal vs. dual problems

Theorem:

Let a primal linear program and its dual be described by

$$\begin{aligned} &\text{minimize}_x && c^T x \\ &\text{subject to} && Ax = b \\ & && x \geq 0 \end{aligned} \qquad \begin{aligned} &\text{maximize}_{y,s} && b^T y \\ &\text{subject to} && A^T y + s = c \\ & && s \geq 0 \end{aligned}$$

Let $\{x^*, y^*, s^*\}$ be their solutions and let $\{x, y, s\}$ satisfy the equalities and inequalities

$$\begin{aligned} Ax &= b \\ x &\geq 0 \\ A^T y + s &= c \\ s &\geq 0 \end{aligned}$$

Then we have the following lower bound on the optimal value of the objective function:

$$f(x^*) = c^T x^* \geq c^T x - s^T x$$

Primal vs. dual problems

Corollary:

Let a primal linear program and its dual be described by

$$\begin{aligned} &\text{minimize}_x && c^T x \\ &\text{subject to} && Ax = b \\ & && x \geq 0 \end{aligned} \qquad\qquad \begin{aligned} &\text{maximize}_{y,s} && b^T y \\ &\text{subject to} && A^T y + s = c \\ & && s \geq 0 \end{aligned}$$

Then any solution $\{x^*, y^*, s^*\}$ of the system of equations

$$\begin{aligned} Ax &= b \\ x &\geq 0 \\ A^T y + s &= c \\ s &\geq 0 \\ x^T s &= 0 \end{aligned}$$

is a solution to the primal and dual problems, respectively.

Note: Since x, s are non-negative, the last condition can also be written as $x_j s_j = 0$ for all j. This is called *strict complementarity*.

Primal-dual algorithms

Observation:

Any feasible point of the set of *nonlinear* equations

$$Ax = b$$
$$x \geq 0$$
$$A^T y + s = c$$
$$s \geq 0$$
$$x^T s = 0$$

generates a solution of the original linear program (though not necessarily a vertex).

If we had efficient algorithms for finding feasible points, this would also provide efficient methods for linear programming problems.

If we have points that satisfy the first four but violate the last equation, we can judge the quality of the current approximation using the previous theorem.

The primal-dual interior point method

We want to solve

$$\begin{aligned} Ax &= b \\ x &\geq 0 \\ A^T y + s &= c \\ s &\geq 0 \\ x_j s_j &= 0, \quad j = 1 \ldots n \end{aligned}$$

Consider a starting point $\{x_k, y_k, s_k\}$ that is feasible with respect to the first four (in)equalities and with

$$\mu_{kj} = (x_k)_j (s_k)_j \geq 0$$

If $\mu_{kj} = 0$ then we have a solution. Otherwise, find a better approximation $\{x_{k+1}, y_{k+1}, s_{k+1}\}$ that is still feasible with respect to the first four (in)equalities and with

$$(x_{k+1})_j (s_{k+1})_j = \mu_{k+1} < \mu_{kj}, \quad \text{for all } j = 1 \ldots n$$

The primal-dual interior point method

Algorithm:

Given $\{x_k, y_k, s_k\}$, find $\{\Delta x_k, \Delta y_k, \Delta s_k\}$ where

$$A(x_k + \Delta x_k) = b$$
$$A^T(y_k + \Delta y_k) + (s_k + \Delta s_k) = c$$
$$(x_k)_j \Delta s_j + (s_k)_j \Delta x_j = \mu_{k+1} - (x_k)_j (s_k)_j$$

In general, $\{x_k + \Delta x_k, y_k + \Delta y_k, s_k + \Delta s_k\}$ will not satisfy the inequalities

$$x_k + \Delta x_k > 0, \qquad s_k + \Delta s_k > 0$$

Consequently, set

$$x_{k+1} = x_k + \alpha^* \Delta x_k, \qquad y_{k+1} = y_k + \alpha^* \Delta y_k, \qquad s_{k+1} = s_k + \alpha^* \Delta s_k$$

where

$$\alpha^* = (1 - 10^{-5}) \max_\alpha \{x_k + \alpha \Delta x_k \geq 0, s_k + \alpha \Delta s_k \geq 0\}$$

The primal-dual interior point method

Theorem:

If

$$0<\mu_{k+1}=\theta\mu_k, \qquad \text{for } \theta=1-\frac{7}{2\sqrt{n}}$$

then the algorithm converges to a solution of the primal and dual problems (not necessarily a vertex), and the overall algorithm needs at most

$$O(|\log\epsilon|\sqrt{n}L)$$

iterations to achieve an accuracy of ϵ . Here, L is the number of bits needed to encode the coefficients of the problem.

Note: Each iteration requires $O(n^{2.5})$ operations. The algorithm is therefore of *polynomial complexity*.

Note: In practice, one can choose significantly smaller values of theta and still obtain convergence.

The primal-dual interior point method

Remark:

The primal-dual interior point method can be derived in many different ways. In particular:

- It is equivalent to a barrier method approach
- It is similar to the Karmarkar algorithm of 1984 which described for the first time a practical algorithm that is (i) of polynomial complexity, and (ii) competitive with the simplex algorithm on many practical problems.

Interior point methods are often faster than the simplex algorithm if

- The problem is degenerate
- No good initial guess is available
- The number of constraints is very large
- The matrix A is not sparse

Part 20

Integer programming problems

Integer programming

See distributed chapters of the book by Bertsimas and Tsitsiklis.

Part 21

Optimal Control: Examples

Definition of optimal control problems

Commonly understood definition of optimal control problems:

Let

- X be a space of time-dependent functions
- Q be a space of control parameters, time dependent or not
- $f: X \times Q \to \mathbb{R}$ be a continuous *functional* on X and Q
- $L: X \times Q \to Y$ be a continuous operator on X mapping into a space Y
- $g: X \to Z_x$ be a continuous operator on X mapping into a space Z_x
- $h: Q \to Z_q$ be a continuous operator on Q mapping into a space Z_q

Then the problem

$$\begin{array}{lll} \min_{x=x(t)\in X,\, q\in Q} & f(x(t),q) & \\ \text{such that} & L(x(t),q)=0 & \forall t\in[t_i,t_f] \\ & g(x(t)) \geq 0 & \forall t\in[t_i,t_f] \\ & h(q) \geq 0 & \end{array}$$

is called an *optimal control problem*.

Definition of optimal control problems

Remark:

For existence and uniqueness of solutions of the problem

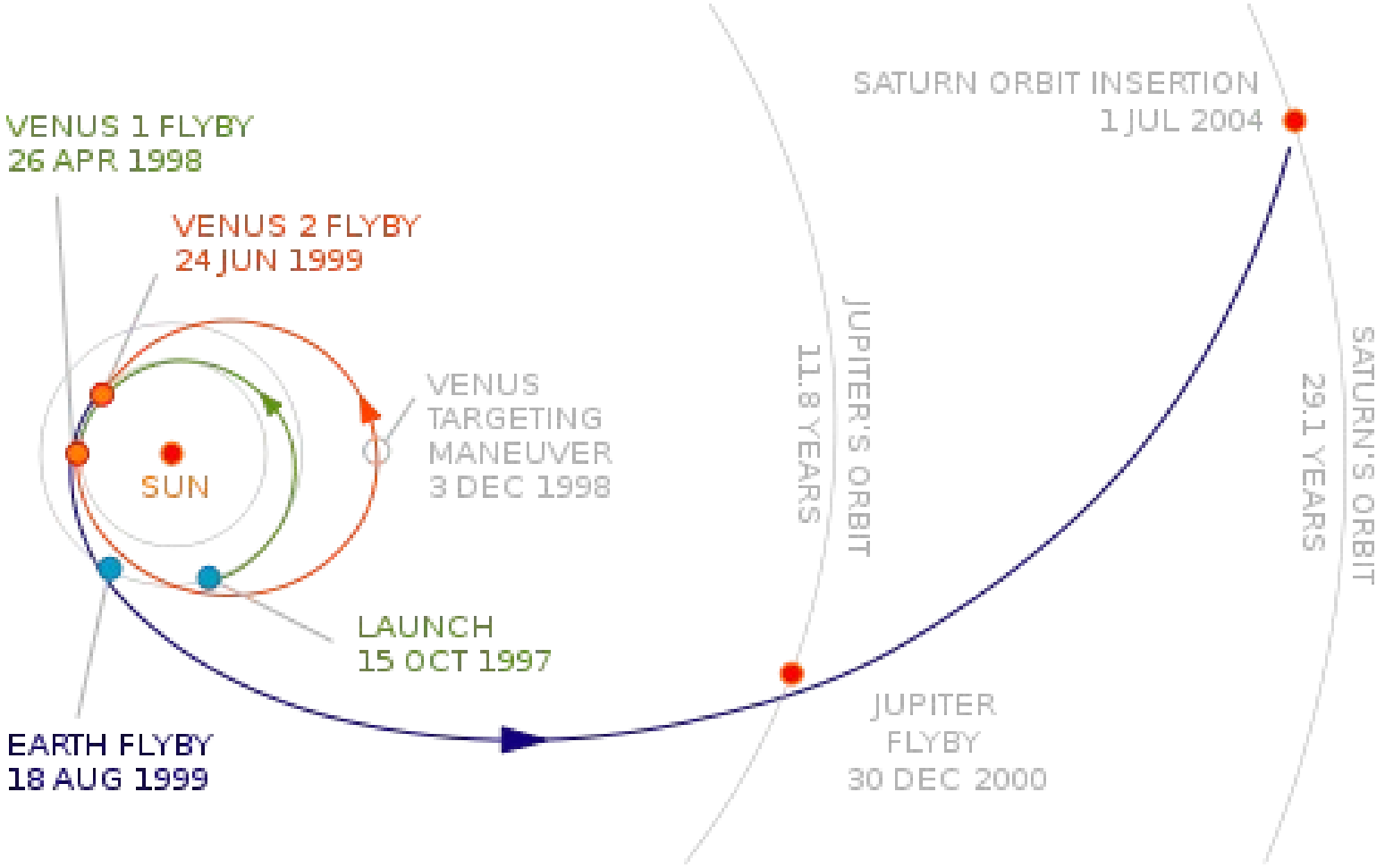
$$\begin{array}{lll} \min_{x=x(t)\in X,\, q\in Q} & f(x(t),q) & \\ \text{such that} & L(x(t),q)=0 & \forall\, t\in[t_i,t_f] \\ & g(x(t)) \geq 0 & \forall\, t\in[t_i,t_f] \\ & h(q) \geq 0 & \end{array}$$

one will need convexity properties of f, L, g, h.

In order to state optimality conditions, we will in general also require certain differentiability properties.

Example 1: Trajectory planning

The trajectory of the Cassini space probe from Earth to Saturn:



Goal: We want to get from A to B using the least amount of fuel, in the least amount of time, ..., subject to Newton's law.

Example 1: Trajectory planning

Version 1: Minimal energy trajectory

- $X = \{x(t) : x \in H^1([0,T])^3\} = \{x(t) : x(t) \in L^2([0,T])^3, \dot{x}(t) \in L^2((0,T))^3\}$
- $Q = \{u(t) : u \in L^\infty([0,T])^3\} \subset L^2([0,T])^3$
- $f : Q \to \mathbb{R}$
- $L : X \times Q \to Y, \quad Y = H^{-1}([0,T])^3 = \left(H^1([0,T])^3\right)^*$
- $g : X \to Z_x = \mathbb{R}^3 \times \mathbb{R}^3$
- $h : Q \to Z_q = L^\infty([0,T])^3$

Then the problem is as follows:

$$
\begin{array}{lll}
\min_{x=x(t)\in X,\, q\in Q} & \int_0^T |u(t)| & \\
\text{such that} & m\ddot{x}(t) - ku(t) = 0 & \forall\, t \in [0,T] \\
& x(0) = \text{Earth}, \quad x(T) = \text{Saturn} & \\
& u_{\max} - |u(t)| \quad \geq 0 & \forall\, t \in [0,T]
\end{array}
$$

Example 1: Trajectory planning

Remark 1:

A more realistic formulation would take into account that the mass of the space ship diminishes as fuel is burnt:

$$m = m(t) = m(\cdot) - \int_{\cdot}^{t} |u(t)|$$

Remark 2:

The formulation on the previous page is nonlinear because of the absolute values $|u(t)|$. The objective function can be made linear by using the following reparameterisation:

$$u(t) = \hat{u}(t)\theta(t), \qquad \hat{u}(t) \in \mathbb{R}_{\cdot}^{+}, \quad \theta \in S^{\cdot}$$

On the other hand, the ODE constraint will then be nonlinear (a complication that is usually easier to handle).

Example 1: Trajectory planning

Version 2: Minimal time trajectory

- $X = H^1([0,T])^3$
- $Q = [u(t), T] = L^\infty([0,T])^3 \times \mathbb{R}_0^+$
- $f : Q \to \mathbb{R}$
- $L : X \times Q \to Y$
- $g : X \to Z_x = \mathbb{R}^3 \times \mathbb{R}^3$
- $h : Q \to Z_q = L^\infty([0,T])^3$

Then the problem is as follows:

$$
\begin{array}{lll}
\min_{x=x(t)\in X,\, q\in Q} & T & \\
\text{such that} & m\ddot{x}(t) - ku(t) = 0 & \forall\, t \in [0,T] \\
& x(0) = \text{Earth}, \quad x(T) = \text{Saturn} & \\
& u_{\max} - |u(t)| \quad \geq 0 & \forall\, t \in [0,T]
\end{array}
$$

Example 1: Trajectory planning

Version 3: Minimal thrust requirement trajectory

- $X = H^1([0,T])^3$
- $Q = [u(t), u_{\max}] = L^\infty([0,T])^3 \times \mathbb{R}_0^+$
- $f : Q \to \mathbb{R}$
- $L : X \times Q \to Y$
- $g : X \to Z_x = \mathbb{R}^3 \times \mathbb{R}^3$
- $h : Q \to Z_q = L^\infty([0,T])^3$

Then the problem is as follows:

$$
\begin{array}{lll}
\min_{x=x(t)\in X,\, q\in Q} & u_{\max} & \\
\text{such that} & m\ddot{x}(t) - ku(t) = 0 & \forall\, t \in [0,T] \\
& x(0) = \text{Earth}, \quad x(T) = \text{Saturn} & \\
& u_{\max} - |u(t)| \quad \geq 0 & \forall\, t \in [0,T]
\end{array}
$$

Example 1: Trajectory planning

Remark 1:

Problems with a similar formulation appear in planning the paths of

- mobile robots
- air planes, manned or unmanned
- the arms of stationary robots (e.g. welding robots on assembly lines)
- braking a car without exceeding the maximal force the tires can transmit to the road

Remark 2:

For some problems, $T=\infty$. These are called *infinite horizon* problems. An example is the problem of keeping a satellite or airship stationary at a given point above earth.

Example 2: Chemical reactors

State:

Concentrations $x_i(t)$ of chemical species $i=1...N$.

Controls:

Pressure $p(t)$, temperature $T(t)$.

Goals:

- Maximize output of a particular species
- Maximize purity
- Minimize cost
- Minimize time

Example 2: Chemical reactors

Version 1: Maximize yield of species N

$$
\begin{array}{lll}
\min_{x(t),\,p(t),T(t)} & -x_N(T) & \\
\text{such that} & \dot{x}(t) - f(x(t), p(t), T(t)) = 0 & \forall t \in [0, T] \\
& x(0) = x_0 & \\
& p_0 \le p(t) \le p_1, \quad T_0 \le T(t) \le T_1 & \forall t \in [0, T]
\end{array}
$$

Example 2: Chemical reactors

Version 2: Minimize reaction time, subject to minimum yield constraints:

$$
\begin{aligned}
&\min_{x(t),\,p(t),T(t)} && T \\
&\text{such that} && \dot{x}(t) - f(x(t), p(t), T(t)) = 0 && \forall t \in [0, T] \\
& && x(0) = x_0 \\
& && p_0 \le p(t) \le p_1, \quad T_0 \le T(t) \le T_1 && \forall t \in [0, T] \\
& && x_N \ge x_{N,\min}
\end{aligned}
$$

Example 2: Chemical reactors

Version 3: Minimize cost due to heat losses (with a heat loss factor alpha) and due to the cost of changing the temperature by cooling and heating (with a cost factor beta), subject to minimum yield constraints:

$$
\begin{aligned}
\min_{x(t),\,p(t),T(t)} \quad & \int_0^T \alpha T(t) + \beta \left|\dot{T}(t)\right| \\
\text{such that} \quad & \dot{x}(t) - f(x(t), p(t), T(t)) = 0 \qquad \forall t \in [0, T] \\
& x(0) = x_0 \\
& p_0 \le p(t) \le p_1, \quad T_0 \le T(t) \le T_1 \qquad \forall t \in [0, T] \\
& x_N \ge x_{N,\min}
\end{aligned}
$$

Part 22

Optimal control:
The shooting method

The solution operator

Definition:
Let the ordinary differential equation of the state and control variables be given by

$$\dot{x}(t) - f(x(t), q) = 0 \qquad \forall t \in [t_i, t_f]$$
$$x(t_i) = g(x_0, q)$$

Let *x(t)* solve this set of equations for a given set of control variables *q*. Then define

$$S(q, x_0, t_i, t) := x(t)$$

In other words: *S* is the operator that given controls and initial data provides the value of the corresponding solution of the ODE at time *t*. We call *S* the *solution operator*.

Note: If the ODE is complicated, then *S* is a purely theoretical construct, though it can be approximated numerically.

The solution operator

Corollary:
Consider the optimal control problem

$$\min_{x(t),q} \ \frac{1}{2}\left(x(t_f)-x_{\text{desired}}\right)^2$$
$$\dot{x}(t)-f(x(t),q)=0 \qquad \forall\, t\in[t_i,t_f]$$
$$x(t_i)=g(x_0,q)$$

It is equivalent to the problem

$$\min_{q} \ \frac{1}{2}\left(S(q,x_0,t_i,t_f)-x_{\text{desired}}\right)^2$$

Note 1: Similar reformulations are trivially available if the objective function has a different form or if there are constraints.

Note 2: If we can represent *S* and its derivatives, then we can apply Newton's method (or any other optimization method) to the reformulated problem.

The shooting method

Algorithm:

Starting from the formulation

$$\min_q \ \frac{1}{2}\left(S(q, x_0, t_i, t_f) - x_{\text{desired}}\right)^2$$

we can think of the shooting method as an iterative procedure that does the following steps:

- Start with a certain control value q
- See where the trajectory $S(q,...)$ takes us for this control value
- If we "overshot" the goal, then do the same again with a smaller value of q
- If we "undershot" the goal, then try a larger value of q
- Iterate until we have the solution we were looking for

The shooting method: An example

Example: Charged particles in a magnetic field

Charged particles moving in a magnetic field follow the Lorentz force:

$$m\ddot{x}(t) = e\dot{x}(t) \times B(x(t), t)$$

Here, e is the charge of the particle and $B(x(t),t)$ is the magnetic field at the location of the particle at time t. Assume the magnetic is constant but that the magnitude is adjustable.

Given initial position and velocity, find the value of B for which the particle passes through location x_{desired}.

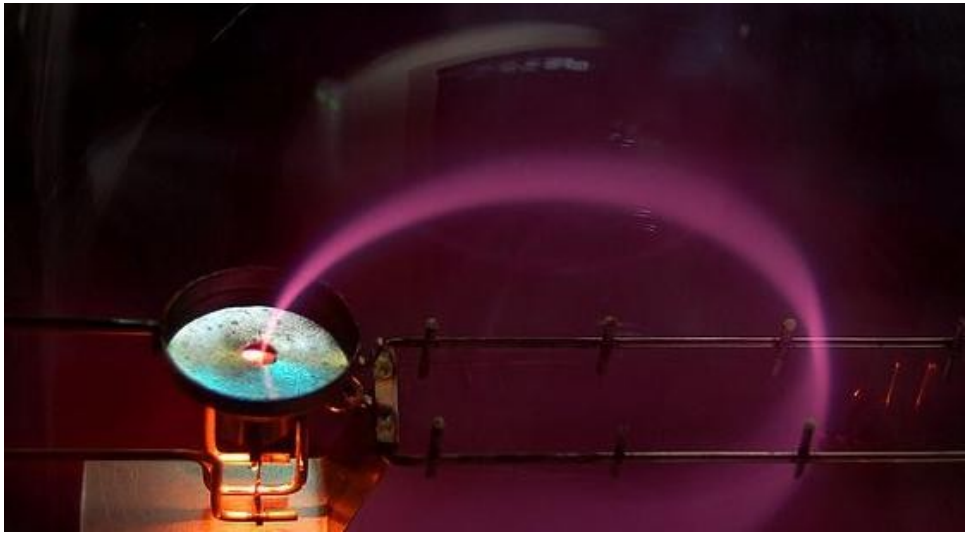
Formulation:

$$\min_{x(t),B,T} \ \frac{1}{2}\left(x(T) - x_{\text{desired}}\right)^2$$

$$m\ddot{x}(t) - e\,x(t) \times B = 0$$

$$x(0) = x_0$$

$$\dot{x}(0) = v_0$$



The shooting method: An example

Example: Charged particles in a magnetic field

For the equation of motion

$$m\ddot{x}(t) = e\,\dot{x}(t) \times B, \qquad x(0) = 0, \quad \dot{x}(0) = \begin{pmatrix} 0 \\ v_0 \end{pmatrix}$$

and if B is perpendicular to the x-y plane, then we can write down the exact trajectory:

$$x(t) = r \begin{pmatrix} 1 - \cos \omega t \\ \sin \omega t \end{pmatrix}$$

where

$$r = \frac{m v_0}{e \|B\|}, \quad \omega = \frac{v_0}{r} = \frac{e \|B\|}{m}$$

Then we know the solution operator in closed form:

$$S(B, 0, t) = r \begin{pmatrix} 1 - \cos \omega t \\ \sin \omega t \end{pmatrix}$$

The shooting method: An example

Example: Charged particles in a magnetic field

We can now restate the original problem

$$\min_{x(t),B,T} \frac{1}{2}\left(x(T)-x_{\text{desired}}\right)^2$$

$$m\ddot{x}(t)-e\,x(t)\times B=0$$

$$x(0)=x_0$$

$$\dot{x}(0)=v_0$$

as follows:

$$\min_{B,T} \frac{1}{2}\left(S(B,0,T)-x_{\text{desired}}\right)^2 = \frac{1}{2}\left(r\begin{pmatrix}1-\cos\omega T\\ \sin\omega T\end{pmatrix}-x_{\text{desired}}\right)^2$$

Note: This is a nonlinear optimization problem in two variables (B,T) that we can solve with any of the usual methods.

The shooting method: Practical implementation

Consider the optimal control problem with control constraints:

$$\begin{aligned} \min_{x(t),q} \quad & F(x(t),q) \\ & \dot{x}(t) - f(x(t),q) = 0 \qquad \forall\, t \in [t_i, t_f] \\ & x(t_i) = g(x_0, q) \\ & h(q) \geq 0 \end{aligned}$$

It is equivalent to the problem

$$\begin{aligned} \min_{q} \quad & F(S(q, x_0, t_i, t), q) \\ & h(q) \geq 0 \end{aligned}$$

Using the techniques we learned last semester (e.g. the active set method, barrier methods, etc), we can solve this problem. However, we will also need first and second *derivatives* of F with respect to q!

The shooting method: Computing derivatives

By the chain rule, we have

$$\frac{d}{dq_i} F(S(q, x_0, t_i, t), q)$$

$$= \nabla_s F(S(q, x_0, t_i, t), q) \frac{d}{dq_i} S(q, x_0, t_i, t) + \frac{\partial}{\partial q_i} F(S(q, x_0, t_i, t), q)$$

In other words, in order to compute derivatives of F, we need derivatives of S. To compute these, remember that

$$S(q, x_0, t_i, t) = x(t)$$

where $x(t) = x_q(t)$ solves the following ODE for the given value of q:

$$\dot{x}(t) - f(x(t), q) = 0 \qquad \forall t \in [t_i, t_f]$$
$$x(t_i) = g(x_0, q)$$

The shooting method: Computing derivatives

By definition:

$$\frac{d}{dq_i} S(q, x_0, t_i, t) = \lim_{\epsilon \to 0} \frac{S(q+\epsilon e_i, x_0, t_i, t) - S(q, x_0, t_i, t)}{\epsilon}$$

Consequently, we can approximate derivatives using the formula

$$\frac{d}{dq_i} S(q, x_0, t_i, t) \approx \frac{S(q+\delta e_i, x_0, t_i, t) - S(q, x_0, t_i, t)}{\delta} = \frac{x_{q+\delta e_i}(t) - x_q(t)}{\delta}$$

for a finite $\delta > 0$. Note that $x_q(t)$ and $x_{q+\delta ei}(t)$ solve the ODEs

$$\dot{x}_q(t) - f(x_q(t), q) = 0 \qquad x_q(t_i) = g(x_0, q)$$

$$\dot{x}_{q+\delta e_i}(t) - f(x_{q+\delta e_i}(t), q+\delta e_i) = 0 \qquad x_{q+\delta e_i}(t_i) = g(x_0, q+\delta e_i)$$

The shooting method: Computing derivatives

Corollary:

To compute $\nabla_q F(S(q, x_0, t_i, t), q)$ we need to compute

$$\nabla_q S(q, x_0, t_i, t)$$

For $q \in \mathbb{R}^n$, this requires the solution of $n+1$ ordinary differential equations:

- For the given q:

$$\dot{x}_q(t) - f(x_q(t), q) = 0$$
$$x_q(t_i) = g(x_0, q)$$

- Perturbed in directions $i = 1 \ldots n$:

$$\dot{x}_{q+\delta e_i}(t) - f(x_{q+\delta e_i}(t), q+\delta e_i) = 0$$
$$x_{q+\delta e_i}(t_i) = g(x_0, q+\delta e_i)$$

The shooting method: Computing derivatives

Practical considerations 1:

When computing finite difference approximations

$$\frac{d}{dq_i} S(q, x_0, t_i, t) \approx \frac{S(q+\delta e_i, x_0, t_i, t) - S(q, x_0, t_i, t)}{\delta} = \frac{x_{q+\delta e_i}(t) - x_q(t)}{\delta}$$

how should we choose the step length δ?

δ must be small enough to yield a good approximation to the exact derivative but large enough so that floating point roundoff does not affect the accuracy!

Rule of thumb: If

- ϵ is the precision of the machine's floating point number format
- $\hat{q}_i$ is a typical size of the *i*th control variable q_i

then choose $\delta = \sqrt{\epsilon}\,\hat{q}_i$.

The shooting method: Computing derivatives

Practical considerations 2:

The one-sided finite difference quotient

$$\frac{d}{dq_i} S(q, x_0, t_i, t) \approx \frac{S(q+\delta e_i, x_0, t_i, t) - S(q, x_0, t_i, t)}{\delta} = \frac{x_{q+\delta e_i}(t) - x_q(t)}{\delta}$$

is only first order accurate in δ, i.e.

$$\left| \frac{d}{dq_i} S(q, x_0, t_i, t) - \frac{S(q+\delta e_i, x_0, t_i, t) - S(q, x_0, t_i, t)}{\delta} \right| = O(\delta)$$

The shooting method: Computing derivatives

Practical considerations 2:

We can improve on this by using the two-sided finite difference quotient

$$\frac{d}{dq_i} S(q, x_0, t_i, t) \approx \frac{S(q+\delta e_i, x_0, t_i, t) - S(q-\delta e_i, x_0, t_i, t)}{2\delta} = \frac{x_{q+\delta e_i}(t) - x_{q-\delta e_i}(t)}{2\delta}$$

which is second order accurate in δ, i.e.

$$\left| \frac{d}{dq_i} S(q, x_0, t_i, t) - \frac{S(q+\delta e_i, x_0, t_i, t) - S(q-\delta e_i, x_0, t_i, t)}{2\delta} \right| = O(\delta^2)$$

Note:

The price to pay for this higher accuracy is that we now have to solve $2n+1$ ordinary differential equations!

The shooting method: Computing derivatives

Practical considerations 3:

To approximate derivatives (using, for example, the one-sided finite difference quotient), we have to solve the ODEs

$$\dot{x}_q(t) - f(x_q(t), q) = 0$$
$$x_q(t_i) = g(x_0, q)$$

$$\dot{x}_{q+\delta e_i}(t) - f(x_{q+\delta e_i}(t), q+\delta e_i) = 0 \qquad i = 1 \ldots n$$
$$x_{q+\delta e_i}(t_i) = g(x_0, q+\delta e_i)$$

If we can do that analytically, then good.

If we do this numerically, then numerical approximation introduces systematic errors related to

- the numerical method used
- the time mesh (i.e. the collection of time step sizes) chosen

The shooting method: Computing derivatives

Practical considerations 3:

We know that we can gain the highest accuracy in the numerical solution of equations like

$$\dot{x}_q(t) - f(x_q(t), q) = 0$$
$$x_q(t_i) = g(x_0, q)$$

by choosing highly sophisticated adaptive time step, extrapolating multistep ODE integrators (e.g. RK45).

On the other hand, to get the best accuracy in evaluating

$$\frac{d}{dq_i} S(q, x_0, t_i, t) \approx \frac{x_{q+\delta e_i}(t) - x_q(t)}{\delta}$$

experience has shown that we should use *predictable* integrators for all ODE solvers that for all variables $x_q(t), x_{q+\delta e_i}(t)$ use

- the same numerical method
- the same time steps
- no extrapolation

The shooting method: Computing derivatives

Practical considerations 3:

To solve the collection ODEs

$$\dot{x}_q(t) - f(x_q(t), q) = 0 \qquad x_q(t_i) = g(x_0, q)$$

$$\dot{x}_{q+\delta e_i}(t) - f(x_{q+\delta e_i}(t), q+\delta e_i) = 0 \qquad x_{q+\delta e_i}(t_i) = g(x_0, q+\delta e_i)$$

it therefore turns out to be useful to not solve them individually but instead solve them all at once as

$$\frac{d}{dt}\begin{pmatrix} x_q(t) \\ x_{q+\delta_1 e_1}(t) \\ \vdots \\ x_{q+\delta_n e_n}(t) \end{pmatrix} - \begin{pmatrix} f(x_q(t), q) \\ f(x_{q+\delta_1 e_1}(t), q+\delta_1 e_1) \\ \vdots \\ f(x_{q+\delta_n e_n}(t), q+\delta_n e_n) \end{pmatrix} = 0$$

$$\begin{pmatrix} x_q(t_i) \\ x_{q+\delta_1 e_1}(t_i) \\ \vdots \\ x_{q+\delta_n e_n}(t_i) \end{pmatrix} = \begin{pmatrix} g(x_0, q) \\ g(x_0, q+\delta_1 e_1) \\ \vdots \\ g(x_0, q+\delta_n e_n) \end{pmatrix}$$

The shooting method: Computing derivatives

Practical considerations 4:

If we use the BFGS method, we only need first order derivatives of *F(S(q),q)*, but if we want to use a full Newton method we also need

$$\frac{d^2}{dq_i^2} S(q, x_0, t_i, t), \frac{d^2}{dq_i\, dq_j} S(q, x_0, t_i, t)$$

These can also be computed using finite difference methods:

$$\frac{d}{dq_i^2} S(q, x_0, t_i, t) \approx \frac{\frac{x_{q+\delta e_i}(t) - x_q(t)}{\delta} - \frac{x_q(t) - x_{q-\delta e_i}(t)}{\delta}}{\delta} = \frac{x_{q+\delta e_i}(t) - 2x_q(t) + x_{q-\delta e_i}(t)}{\delta^2}$$

$$\frac{d}{dq_i\, dq_j} S(q, x_0, t_i, t) \approx \frac{\frac{x_{q+\delta e_i+\delta e_j}(t) - x_{q-\delta e_i+\delta e_j}(t)}{2\delta} - \frac{x_{q+\delta e_i-\delta e_j}(t) - x_{q-\delta e_i-\delta e_j}(t)}{2\delta}}{2\delta}$$

Note: The cost for this operation is 3^n ODE solves.

The shooting method: Practical implementation

Algorithm:

To solve the optimal control problem with control constraints

$$\begin{aligned} \min_{x(t),q} \quad & F(x(t),q) \\ & \dot{x}(t) - f(x(t),q) = 0 \qquad \forall\, t \in [t_i, t_f] \\ & x(t_i) = g(x_0, q) \\ & h(q) \geq 0 \end{aligned}$$

reformulate it as

$$\begin{aligned} \min_{q} \quad & F(S(q, x_0, t_i, t), q) \\ & h(q) \geq 0 \end{aligned}$$

Solve it using your favorite nonlinear optimization technique where

- by the chain rule $\nabla_q F(S,q) = F_S(S,q)\nabla_q S(q, x_0, t_i, t) + F_q(S,q)$ and similarly for second derivatives
- the quantities $\nabla_q S(q, x_0, t_i, t), \nabla_q^2 S(q, x_0, t_i, t)$ are approximated by finite difference quotients by solving multiple ODEs for different values of the control variable q

The shooting method: Practical implementation

Implementation (Newton method without line search; no attempt to compute ODE and its derivatives in synch):

```
function f(double[N] q) →double;
function grad_f(double[N] q) →double[N];
function grad_grad_f(double[N] q) →double[N][N];

function newton(double[N] q) →double[N]
{
    do {
        double[N] dq = - invert(grad_grad_f(q)) * grad_f(q);
        q = q + dq;
    } while (norm(grad_f(x)) > 1e-12);        // for example
    return x;
}
```

The shooting method: Practical implementation

Implementation (objective function only depends on *x(T)*):

```
function S(double[N] q) →double[M]
{
    double[M] x = x0;
    double t = ti;
    while (t<tf) {
        x = x + dt * rhs(x,q);        // explicit Euler method with fixed dt
        t = t + dt;
    }
    return x;
}

function f(double[N] q) →double
{
    return objective_function(S(q),q);
}
```

The shooting method: Practical implementation

Implementation (one-sided finite difference quotient):

```
function grad_f(double[N] q) →double[N]
{
    double[N] df = 0;
    for (i=1...N) {
        eps = 1e-8 * typical_q[i];
        double[N] q_plus = q;
        q_plus[i] = q[i] + eps;
        df[i] = (f(q_plus) – f(q)) / eps;
    }
    return df;
}
```

Part 23

Optimal control:
The multiple shooting method

Motivation

In the shooting method, we need to evaluate and differentiate the function

$$S(q, x_0, t_i, t) = x_q(t)$$

where $x_q(t)$ solves the ODE

$$\dot{x}(t) - f(x(t), q) = 0 \qquad \forall t \in [t_i, t_f]$$
$$x(t_i) = g(x_0, q)$$

Observation:

If the time interval $[t_i, t_f]$ is "long", then S is often a strongly nonlinear function of q.

Consequence:

It is difficult to approximate S and its derivatives numerically since constants in error terms typically depend on higher derivatives of S times terms like e^{LT} where L is a Lipschitz constant and $T = t_f - t_i$.

Idea

Observation:

If the time interval $[t_i,t_f]$ is "long", then S is often a strongly nonlinear function of q because of the exponential term.

But then S should be less nonlinear on smaller intervals!

Idea:

While $S(q,x_0,t_i,t_f)$ is a strongly nonlinear function of q, we could introduce

$$t_i = t_0 < t_1 < \ldots < t_k < \ldots < t_K = t_f$$

and the functions $S(q,x_k,t_k,t_{k+1})$ should be less nonlinear and therefore simpler to approximate or differentiate numerically!

Multiple shooting

Outline:

To solve

$$\min_{x(t),q} \; F(x(t),q)$$
$$\dot{x}(t) - f(x(t),q) = 0 \qquad \forall\, t \in [t_i, t_f]$$
$$x(t_i) = g(x_0, q)$$
$$h(q) \geq 0$$

replace this problem by the following:

$$\min_{x^1(t), x^2(t), \ldots, x^K(t), q} \; F(x(t), q)$$
$$\text{where } x(t) := x^k(t) \qquad \forall\, t \in [t_{k-1}, t_k]$$
$$\text{such that } \dot{x}^1(t) - f(x^1(t), q) = 0 \qquad \forall\, t \in [t_0, t_1]$$
$$x^1(t_i) = g(x_0, q)$$

$$\dot{x}^k(t) - f(x^k(t), q) = 0 \qquad \forall\, t \in [t_{k-1}, t_k], k = 2 \ldots K$$
$$x^k(t_{k-1}) = x^{k-1}(t_{k-1})$$

$$h(q) \geq 0$$

Multiple shooting

Outline:

In the formulation on the previous slide, every x^k depended explicitly on x^{k-1}. We can decouple this as follows:

$$\min_{x^1(t),x^2(t),\ldots,x^K(t),\hat{x}^1_{0,},\ldots,\hat{x}^K_{0,}q} \; F(x(t),q)$$

$$\text{where } x(t) := x^k(t) \qquad \forall t \in [t_{k-1}, t_k]$$

$$\text{such that } \dot{x}^k(t) - f(x^k(t),q) = 0 \qquad \forall t \in [t_{k-1}, t_k], k = 1\ldots K$$

$$x^k(t_{k-1}) = \hat{x}^k_0$$

$$\hat{x}^1_0 - g(x_{0,}q) = 0$$

$$\hat{x}^k_0 - x^{k-1}(t_{k-1}) = 0 \qquad \forall k = 2\ldots K$$

$$h(q) \geq 0$$

Note: The "defect constraints" $\hat{x}^k_{\cdot} - x^{k-1}(t_{k-1}) = 0$ do not need to be satisfied in intermediate iterations of Newton's method. They will only be satisfied at the solution, forcing *x(t)* to then be continuous.

Multiple shooting

Outline with the solution operator:

By introducing the solution operator as before, the problem can be written as

$$\min_{\hat{x}_{0,}^{1}\ldots,\hat{x}_{0,}^{K}\,q} \; F\left(S\left(q,x_0,t_i,t\right),q\right)$$

$$\text{where} \quad S\left(q,x_0,t_i,t\right) := S\left(q,\hat{x}_0^k,t_{k-1},t\right) \qquad \forall\, t\in\left[t_{k-1},t_k\right]$$

$$\text{such that} \quad \hat{x}_0^1 - g\left(x_{0,}q\right) = 0$$

$$\hat{x}_0^k - x^{k-1}\left(t_k\right) = 0 \qquad \forall\, k = 2\ldots K$$

$$h\left(q\right) \geq 0$$

Note: Through this reformulation, we now only ever have to differentiate

$$S\left(q,\hat{x}_0^k,t_{k-1},t\right)$$

which integrates the ODE on the much shorter time intervals $[t_{k-1},t_k]$ and consequently is much less nonlinear.

Part 24

Optimal control: Introduction to the Theory

Preliminaries

Definition: A vector space is a set X of objects so that the following holds:

$$\forall x, y \in X: \quad x+y \in X$$
$$\forall x \in X, \alpha \in \mathbb{R}: \quad \alpha x \in X$$

In addition, associativity, distributivity and commutativity of addition has to hold. There also need to be identity and null elements of addition and scalar multiplication.

Examples:

$$X = \mathbb{R}^N$$
$$X = C^0(0,T) = \{x(t) : x(t) \text{ is continuous on } (0,T)\}$$
$$X = C^1(0,T) = \{x(t) \in C^0(0,T) : x(t) \text{ is continuously differentiable on } (0,T)\}$$
$$X = L^2(0,T) = \left\{x(t) : \int_0^T |x(t)|^2 \, dt < \infty\right\}$$

Preliminaries

Definition: A scalar product is a mapping

$$\langle \cdot, \cdot \rangle : X \times Y \to \mathbb{R}$$

of a pair of vectors from (real) vector spaces X, Y into the real numbers. It needs to be linear. If $X = Y$ and x=y, then it also needs to be positive or zero.

Examples:

$X = Y = \mathbb{R}^N$

$$\langle x, y \rangle = \sum_{i=1}^{N} x_i y_i$$

$$\langle x, y \rangle = \sum_{i=1}^{N} \alpha_i x_i y_i \quad \text{with weights } 0 < \alpha_i < \infty$$

$X = Y = l_2$

$$\langle x, y \rangle = \sum_{i=1}^{\infty} x_i y_i$$

$X = Y = L^2(0,T)$

$$\langle x, y \rangle = \int_0^T x(t)\, y(t)\, dt$$

Preliminaries

Definition: Given a space X and a scalar product

$$\langle \cdot, \cdot \rangle : X \times Y \to \mathbb{R}$$

we call $Y = X'$ the *dual space of* X if Y is the largest space for which the scalar product above "makes sense".

Examples:

$X = \mathbb{R}^N$	$\langle x, y \rangle = \sum_{i=1}^{N} x_i y_i$	$Y = \mathbb{R}^N$
$X = C^0(0,T)$	$\langle x, y \rangle = \int_0^T x(t)\, y(t)\, dt$	$Y = S(0,T)$
$X = L^2(0,T)$	$\langle x, y \rangle = \int_0^T x(t)\, y(t)\, dt$	$Y = L^2(0,T)$
$X = L^p(0,T), 1 < p < \infty$	$\langle x, y \rangle = \int_0^T x(t)\, y(t)\, dt$	$Y = L^q(0,T), \frac{1}{p} + \frac{1}{q} = 1$

Lagrange multipliers for finite dimensional problems

Consider the following finite dimensional problem:

$$\begin{array}{ll} \min_{x \in \mathbb{R}^n} & f(x) \\ \text{such that} & g_1(x)=0 \\ & g_2(x)=0 \\ & \vdots \\ & g_K(x)=0 \end{array}$$

Definition: Let the Lagrangian be $L(x,\lambda)=f(x)-\sum_{i=1}^{K}\lambda_i g_i(x).$

Theorem: Under certain conditions on f,g the solution of above problem satisfies

$$\frac{\partial L}{\partial x_i}(x^*,\lambda^*)=0, \qquad i=1,\ldots,N$$

$$\frac{\partial L}{\partial \lambda_i}(x^*,\lambda^*)=0, \qquad i=1,\ldots,K$$

Lagrange multipliers for optimal control problems

Consider the following optimal control problem:

$$\begin{array}{lll} \min_{x(t)} & f(x(t),t) & \\ \text{such that} & g(x(t),t)=0 & \forall\, t\in[0,T] \end{array}$$

Questions:

- What would be the corresponding Lagrange multiplier for such a problem?
- What would be the corresponding Lagrangian function?
- What are optimality conditions in this case?

Lagrange multipliers for optimal control problems

Formal approach: Given

$$\begin{array}{lll} \min_{x(t)} & f(x(t),t) & \\ \text{such that} & g(x(t),t)=0 & \forall t\in[0,T] \end{array}$$

Note: Formally, we now just have infinitely many constraints, one constraint for each possible time instant!

Following this idea, we would then have to replace

$$L(x,\lambda)=f(x)-\sum_{i=1}^{K}\lambda_i g_i(x).$$

by

$$L(x(t),\lambda(t))=f(x(t),t)-\int_0^T \lambda(t)g(x(t),t)\ dt$$

where now we have one Lagrange multiplier for every time: $\lambda(t)$

Lagrange multipliers for optimal control problems

The "correct" approach: If we have a set of equations like

$$\begin{aligned} g_1(x) &= 0 \\ g_2(x) &= 0 \\ &\vdots \\ g_K(x) &= 0 \end{aligned}$$

then we can write this as

$$\vec{g}(x) = 0$$

which we can interpret as saying

$$\langle \vec{g}(x), h \rangle = 0 \qquad \forall h \in \mathbb{R}^K$$

Lagrange multipliers for optimal control problems

The "correct" approach: Likewise, if we have

$$g(x(t),t)=0$$

then we can interpret this in different ways:

- At *every possible time t* we want that *g(x(t),t)* equals zero
- The measure of the set *{t: g(x(t),t)≠0}* is zero ("almost all *t*")
- The integral $\int_0^T |g(x(t),t)|^2\,dt$ is zero
- If $g: X\times[0,T]\to V$ then *g(x(t),t)* is zero in *V*, i.e.

$$\langle g(x(t),t),h\rangle=\int_0^T g(x(t),t)h(t)\,dt=0 \qquad \forall h\in V'$$

Notes:

- The first and fourth statement are the same if $V=C^0([0,T])$
- The second and fourth statement are the same if $V=L^1([0,T])$
- The third and fourth statement are the same if $V=L^2([0,T])$

Lagrange multipliers for optimal control problems

In either case: Given

$$\begin{aligned} &\min_{x(t)\in X} && f(x(t),t) \\ &\text{such that} && g(x(t),t)=0 \end{aligned}$$

the Lagrangian is now

$$\begin{aligned} L(x(t),\lambda(t)) &= f(x(t),t)-\langle \lambda, g(x(t),t)\rangle \\ &= f(x(t),t)-\int_0^T \lambda(t)g(x(t),t)\ dt \end{aligned}$$

and

$$L: X\times V' \to \mathbb{R}$$

Optimality conditions for finite dimensional problems

Corollary: In view of the definition

$$\left\langle \nabla_x f(x), \xi \right\rangle = \lim_{\epsilon \to 0} \frac{f(x+\epsilon\xi) - f(x)}{\epsilon}$$

we can say that the gradient of a function $f : \mathbb{R}^K \to \mathbb{R}$ is a functional

$$\nabla_x f \; : \; \mathbb{R}^K \to \left(\mathbb{R}^K\right)'$$

In other words: The gradient of a function is an element in the dual space of its argument.

Note: For finite dimensional spaces, we can identify space and dual space. Alternatively, we can consider $\mathbb{R}^K$ as the space of column vectors with K elements and $\left(\mathbb{R}^K\right)'$ as the space of row vectors with K elements.

In either case, the dual product is well defined.

Optimality conditions for finite dimensional problems

Corollary: From above considerations it follows that for

$$\begin{array}{ll} \min_{x\in\mathbb{R}^n} & f(x) \\ \text{such that} & g_1(x)=0 \\ & g_2(x)=0 \\ & \vdots \\ & g_K(x)=0 \end{array}$$

we define

$$L(x,\lambda)=f(x)-\sum_{i=1}^{K}\lambda_i g_i(x)$$

where

$$L:\ \mathbb{R}^N\times\mathbb{R}^K\to\mathbb{R}$$

and

$$\nabla_x L:\ \mathbb{R}^N\times\mathbb{R}^K\to(\mathbb{R}^N)'$$
$$\nabla_\lambda L:\ \mathbb{R}^N\times\mathbb{R}^K\to(\mathbb{R}^K)'$$

Optimality conditions for finite dimensional problems

Summary: For the problem

$$\begin{array}{ll} \min_{x\in\mathbb{R}^n} & f(x) \\ \text{such that} & g_1(x)=0 \\ & g_2(x)=0 \\ & \vdots \\ & g_K(x)=0 \end{array}$$

we define

$$L(x,\lambda)=f(x)-\sum_{i=1}^{K}\lambda_i g_i(x).$$

The optimality conditions are then

$$\begin{aligned} \nabla_x L(x^*,\lambda^*)&=0 \quad \text{in } \mathbb{R}^N \\ \nabla_\lambda L(x^*,\lambda^*)&=0 \quad \text{in } \mathbb{R}^K \end{aligned}$$

or equivalently:

$$\begin{aligned} \left\langle \nabla_x L(x^*,\lambda^*),\xi \right\rangle &=0 \qquad \forall\, \xi\in\mathbb{R}^N \\ \left\langle \nabla_\lambda L(x^*,\lambda^*),\eta \right\rangle &=0 \qquad \forall\, \eta\in\mathbb{R}^K \end{aligned}$$

Optimality conditions for finite dimensional problems

Theorem: Under certain conditions on f, g the solution satisfies

$$\frac{\partial L}{\partial x_i}(x^*, \lambda^*) = 0, \qquad i = 1, \ldots, N$$

$$\frac{\partial L}{\partial \lambda_i}(x^*, \lambda^*) = 0, \qquad i = 1, \ldots, K$$

Note 1: These conditions can also be written as

$$\left\langle \nabla_x L(x^*, \lambda^*), \xi \right\rangle = 0, \qquad \forall \xi \in \mathbb{R}^N$$

$$\left\langle \nabla_\lambda L(x^*, \lambda^*), \eta \right\rangle = 0, \qquad \forall \eta \in \mathbb{R}^K$$

Note 2: This, in turn, can be written as follows:

$$\left\langle \nabla_x L(x^*, \lambda^*), \xi \right\rangle = \lim_{\epsilon \to 0} \frac{L(x^* + \epsilon \xi, \lambda^*) - L(x^*, \lambda^*)}{\epsilon} = 0, \qquad \forall \xi \in \mathbb{R}^N$$

$$\left\langle \nabla_\lambda L(x^*, \lambda^*), \eta \right\rangle = \lim_{\epsilon \to 0} \frac{L(x^*, \lambda^* + \epsilon \eta) - L(x^*, \lambda^*)}{\epsilon} = 0, \qquad \forall \eta \in \mathbb{R}^K$$

Optimality conditions for optimal control problems

Recall: For an optimal control problem

$$\begin{aligned} &\min_{x(t)\in X} && f(x(t),t) \\ &\text{such that} && g(x(t),t)=0 \end{aligned}$$

with

$$g: X\times\mathbb{R}\to V$$

we have defined the Lagrangian as

$$L(x(t),\lambda(t))=f(x(t),t)-\langle\lambda, g(x(t),t)\rangle$$

$$L: X\times V'\to\mathbb{R}$$

Optimality conditions for optimal control problems

Theorem: Under certain conditions on f, g the solution satisfies

$$\left\langle \nabla_x L(x^*, \lambda^*), \xi \right\rangle = 0, \qquad \forall \xi \in X$$
$$\left\langle \nabla_\lambda L(x^*, \lambda^*), \eta \right\rangle = 0, \qquad \forall \eta \in V$$

or equivalently

$$\int_0^T \nabla_x L(x^*(t), \lambda^*(t))\, \xi(t)\ dt = 0, \qquad \forall \xi \in X$$
$$\int_0^T \nabla_\lambda L(x^*(t), \lambda^*(t))\, \eta(t)\ dt = 0, \qquad \forall \eta \in V$$

Note: The derivative of the Lagrangian is defined as usual:

$$\left\langle \nabla_x L(x^*(t), \lambda^*(t)), \xi(t) \right\rangle = \lim_{\epsilon \to 0} \frac{L(x^*(t) + \epsilon \xi(t), \lambda^*(t)) - L(x^*(t), \lambda^*(t))}{\epsilon}$$
$$\left\langle \nabla_\lambda L(x^*(t), \lambda^*(t)), \eta(t) \right\rangle = \lim_{\epsilon \to 0} \frac{L(x^*(t), \lambda^*(t) + \epsilon \eta(t)) - L(x^*(t), \lambda^*(t))}{\epsilon}$$

Optimality conditions: Example 1

Example: Consider the rather boring problem

$$\begin{aligned} \min_{x(t)\in X} \quad & f(x(t),t)=\int_0^T x(t)\,dt \\ \text{such that} \quad & g(x(t),t)=x(t)-\psi(t)=0 \end{aligned}$$

for a given function $\psi(t)$. The solution is obviously $x(t)=\psi(t)$. Then the Lagrangian is defined as

$$\begin{aligned} L(x(t),\lambda(t)) &= \int_0^T x(t)\,dt-\langle\lambda(t),x(t)-\psi(t)\rangle \\ &= \int_0^T x(t)-\lambda(t)[x(t)-\psi(t)]\,dt \end{aligned}$$

and we can compute optimality conditions in the next step.

Optimality conditions: Example 1

Given

$$L(x(t),\lambda(t)) = \int_0^T x(t) - \lambda(t)[x(t) - \psi(t)]\,dt$$

we can compute derivatives of the Lagrangian:

$$\begin{aligned}
&\langle \nabla_x L(x(t),\lambda(t)),\xi \rangle \\
&= \lim_{\epsilon\to 0} \frac{1}{\epsilon} \left\{ \begin{array}{r} \int_0^T (x(t)+\epsilon\xi(t)) - \lambda(t)[(x(t)+\epsilon\xi(t)) - \psi(t)]\,dt \\ - \int_0^T x(t) - \lambda(t)[x(t) - \psi(t)]\,dt \end{array} \right\} \\
&= \lim_{\epsilon\to 0} \frac{\int_0^T \epsilon\xi(t) - \lambda(t)[\epsilon\xi(t)]\,dt}{\epsilon} \\
&= \int_0^T \xi(t) - \lambda(t)\xi(t)\,dt \\
&= \int_0^T [1-\lambda(t)]\xi(t)\,dt
\end{aligned}$$

$$\langle \nabla_\lambda L(x(t),\lambda(t)),\eta \rangle = \int_0^T -[x(t)-\psi(t)]\eta(t)\,dt$$

Optimality conditions: Example 1

Example: Consider the rather boring problem

$$\begin{aligned} \min_{x(t)\in X} \quad & f(x(t),t)=\int_0^T x(t)\,dt \\ \text{such that} \quad & g(x(t),t)=x(t)-\psi(t)=0 \end{aligned}$$

The optimality conditions are now

$$\left\langle \nabla_x L(x(t),\lambda(t)),\xi \right\rangle = \int_0^T [1-\lambda(t)]\xi(t)\,dt = 0 \qquad \forall\, \xi(t)$$

$$\left\langle \nabla_\lambda L(x(t),\lambda(t)),\eta \right\rangle = \int_0^T -[x(t)-\psi(t)]\eta(t)\,dt = 0 \qquad \forall\, \eta(t)$$

These can only be satisfied for

$$1-\lambda(t)=0, \qquad x(t)-\psi(t)=0, \qquad \forall\, 0\le t\le T$$

Optimality conditions: Example 2

Example: Consider the slightly more interesting problem

$$\begin{aligned} \min_{x(t)\in X} \quad & f(x(t),t)=\int_0^T x(t)^2\,dt \\ \text{such that} \quad & g(x(t),t)=\dot{x}(t)-t=0 \end{aligned}$$

The constraint allows all functions of the form $x(t)=a+\frac{1}{2}t^2$ for all constants a. Then the Lagrangian is defined as

$$\begin{aligned} L(x(t),\lambda(t)) &= \int_0^T x(t)^2\,dt-\langle\lambda(t),\dot{x}(t)-t\rangle \\ &= \int_0^T x(t)^2-\lambda(t)[\dot{x}(t)-t]\,dt \end{aligned}$$

Note: For $x(t)=a+\frac{1}{2}t^2$ the objective function has the value

$$\int_0^T x(t)^2dt=\int_0^T\left[a+\frac{1}{2}t^2\right]^2=\frac{1}{20}T^5+\frac{1}{3}aT^3+a^2T$$

which takes on its minimal value for $a=-\frac{1}{6}T^2$

Optimality conditions: Example 2

Given

$$L(x(t),\lambda(t)) = \int_0^T x(t)^2 - \lambda(t)[\dot{x}(t) - t]\,dt$$

we can compute derivatives of the Lagrangian:

$$\begin{aligned}
&\langle \nabla_x L(x(t),\lambda(t)),\xi \rangle \\
&= \lim_{\epsilon\to 0} \frac{1}{\epsilon} \left\{ \begin{array}{r} \int_0^T (x(t)+\epsilon\xi(t))^2 - \lambda(t)[\dot{x}(t)+\epsilon\dot{\xi}(t) - t]\,dt \\ - \int_0^T x(t)^2 - \lambda(t)[\dot{x}(t) - t]\,dt \end{array} \right\} \\
&= \lim_{\epsilon\to 0} \frac{\int_0^T 2\epsilon\, x(t)\xi(t) + \epsilon^2 \xi(t)^2 - \lambda(t)[\epsilon\dot{\xi}(t)]\,dt}{\epsilon} \\
&= \int_0^T 2x(t)\xi(t) - \lambda(t)\dot{\xi}(t)\,dt \\
&= \int_0^T [2x(t) + \dot{\lambda}(t)]\xi(t)\,dt - \left[\lambda(t)\xi(t)\right]_{t=0}^T
\end{aligned}$$

$$\langle \nabla_\lambda L(x(t),\lambda(t)),\eta \rangle = \int_0^T -[\dot{x}(t) - t]\eta(t)\,dt$$

Optimality conditions: Example 2

The optimality conditions are now

$$\left\langle \nabla_x L(x(t),\lambda(t)),\xi \right\rangle = \int_0^T [2x(t)+\dot{\lambda}(t)]\xi(t)\,dt - \left[\lambda(t)\xi(t)\right]_{t=0}^{T} = 0 \quad \forall \xi(t)$$

$$\left\langle \nabla_\lambda L(x(t),\lambda(t)),\eta \right\rangle = \int_0^T -[\dot{x}(t)-t]\eta(t)\,dt = 0 \quad \forall \eta(t)$$

From the second equation we can conclude that

$$\dot{x}(t)-t=0 \quad \rightarrow \quad x(t)=a+\frac{1}{2}t^2$$

On the other hand, the first equation yields

$$2x(t)+\dot{\lambda}(t)=0,\ \lambda(0)=0,\ \lambda(T)=0$$

Given the form of *x(t)*, the first of these three conditions can be integrated:

$$\lambda(t)=-2at-\frac{1}{3}t^3+b$$

Enforcing boundary conditions then yields $b=0, a=-\frac{1}{6}T^2$

Optimality conditions: Example 3 – initial conditions

Theorem: Let $x \in C^1, f \in C^0$. If *x(t)* satisfies the initial value problem

$$\dot{x}(t) = f(x(t), t)$$
$$x(0) = x_0$$

then it also satisfies the "variational" equality

$$\int_0^T \left[\dot{x}(t) - f(x(t), t)\right] \lambda(t)\, dt + \left[x(0) - x_0\right] \lambda(0) = 0 \qquad \forall\, \lambda(t) \in C^0([0, T])$$

and vice versa.

Optimality conditions: Example 3 – initial conditions

Example: Consider the (again slightly boring) problem

$$\begin{aligned} \min_{x(t)\in X} \quad & f(x(t),t)=\int_0^T x(t)\,dt \\ \text{such that} \quad & \dot{x}(t)-t=0 \\ & x(0)=1 \end{aligned}$$

The constraint allows for only a single feasible point, $x(t)=1+\frac{1}{2}t^2$

The Lagrangian is now defined as

$$\begin{aligned} L(x(t),\lambda(t)) &= \int_0^T x(t)\,dt-\langle\lambda(t),\dot{x}(t)-t\rangle-[x(0)-1]\lambda(0) \\ &= \int_0^T x(t)-\lambda(t)[\dot{x}(t)-t]\,dt-\lambda(0)[x(0)-1] \end{aligned}$$

Optimality conditions: Example 3 – initial conditions

Given $L(x(t),\lambda(t)) = \int_0^T x(t) - \lambda(t)[\dot{x}(t) - t]\,dt - \lambda(0)[x(0) - 1]$

we can compute derivatives of the Lagrangian:

$$\langle \nabla_x L(x(t),\lambda(t)),\xi \rangle$$

$$= \lim_{\epsilon \to 0} \frac{1}{\epsilon} \left\{ \begin{array}{r} \int_0^T (x(t) + \epsilon\xi(t)) - \lambda(t)[\dot{x}(t) + \epsilon\dot{\xi}(t) - t]\,dt - \lambda(0)[x(0) + \epsilon\xi(0) - 1] \\ - \int_0^T x(t) - \lambda(t)[\dot{x}(t) - t]\,dt + \lambda(0)[x(0) - 1] \end{array} \right\}$$

$$= \lim_{\epsilon \to 0} \frac{\int_0^T \epsilon\xi(t) - \lambda(t)[\epsilon\dot{\xi}(t)]\,dt - \epsilon\lambda(0)\xi(0)}{\epsilon}$$

$$= \int_0^T \xi(t) - \lambda(t)\dot{\xi}(t)\,dt - \lambda(0)\xi(0)$$

$$= \int_0^T [1 + \dot{\lambda}(t)]\xi(t)\,dt - \left[\lambda(t)\xi(t)\right]_{t=0}^{T} - \lambda(0)\xi(0)$$

$$= \int_0^T [1 + \dot{\lambda}(t)]\xi(t)\,dt - \lambda(T)\xi(T)$$

$$\langle \nabla_\lambda L(x(t),\lambda(t)),\eta \rangle = \int_0^T -[\dot{x}(t) - t]\eta(t)\,dt - \eta(0)[x(0) - 1]$$

Optimality conditions: Example 3 – initial conditions

The optimality conditions are now

$$\left\langle \nabla_x L(x(t),\lambda(t)),\xi \right\rangle = \int_0^T [1+\dot{\lambda}(t)]\xi(t)\,dt - \lambda(T)\xi(T) \qquad = 0 \quad \forall \xi(t)$$

$$\left\langle \nabla_\lambda L(x(t),\lambda(t)),\eta \right\rangle = \int_0^T -[\dot{x}(t)-t]\eta(t)\,dt - [x(0)-1]\eta(0) \; = 0 \quad \forall \eta(t)$$

From the second equation we can conclude that

$$\dot{x}(t)-t=0$$
$$x(0)=1$$

In other words: Taking the derivative of the Lagrangian with respect to the Lagrange multiplier gives us back the (initial value problem) constraint, just like in the finite dimensional case.

Note: The only feasible point of this constraint is of course

$$x(t)=1+\frac{1}{2}t^2$$

Optimality conditions: Example 3 – initial conditions

The optimality conditions are now

$$\left\langle \nabla_x L(x(t),\lambda(t)),\xi \right\rangle = \int_0^T [1+\dot{\lambda}(t)]\xi(t)\,dt - \lambda(T)\xi(T) \qquad = 0 \quad \forall \xi(t)$$

$$\left\langle \nabla_\lambda L(x(t),\lambda(t)),\eta \right\rangle = \int_0^T -[\dot{x}(t)-t]\eta(t)\,dt - [x(0)-1]\eta(0) \; = 0 \quad \forall \eta(t)$$

From the first equation we can conclude that

$$1+\dot{\lambda}(t)=0$$
$$\lambda(T)=0$$

in much the same way as we could obtain the initial value problem for *x(t)*.

Note: This is a *final value problem* for the Lagrange multiplier! Its solution is

$$\lambda(t)=T-t$$

Optimality conditions: Example 4 – initial conditions

Note: If the objective function had been nonlinear, then the equation for $\lambda(t)$ would contain $x(t)$ but still be linear in $\lambda(t)$.

Example: Consider the (again slightly boring) variant of the same problem

$$\begin{aligned} \min_{x(t)\in X} \quad & f(x(t),t) = \int_0^T \frac{1}{2} x(t)^2 \, dt \\ \text{such that} \quad & \dot{x}(t) - t = 0 \\ & x(0) = 1 \end{aligned}$$

The constraint allows for only a single feasible point, $x(t) = 1 + \frac{1}{2} t^2$

The Lagrangian is now defined as

$$L(x(t), \lambda(t)) = \int_0^T \frac{1}{2} x(t)^2 - \lambda(t)[\dot{x}(t) - t] \, dt - \lambda(0)[x(0) - 1]$$

Optimality conditions: Example 4 – initial conditions

Given

$$L(x(t),\lambda(t)) = \int_0^T \frac{1}{2}x(t)^2 - \lambda(t)[\dot{x}(t) - t]\,dt - \lambda(0)[x(0) - 1]$$

the derivatives of the Lagrangian are now:

$$\left\langle \nabla_x L(x(t),\lambda(t)), \xi \right\rangle = \int_0^T [x(t) + \dot{\lambda}(t)]\xi(t)\,dt - \lambda(T)\xi(T)$$

$$\left\langle \nabla_\lambda L(x(t),\lambda(t)), \eta \right\rangle = \int_0^T -[\dot{x}(t) - t]\eta(t)\,dt - \eta(0)[x(0) - 1]$$

Optimality conditions: Example 4 – initial conditions

The optimality conditions are now

$$\left\langle \nabla_x L(x(t),\lambda(t)),\xi \right\rangle = \int_0^T [x(t)+\dot{\lambda}(t)]\xi(t)\,dt - \lambda(T)\xi(T) \quad = 0 \quad \forall \xi(t)$$

$$\left\langle \nabla_\lambda L(x(t),\lambda(t)),\eta \right\rangle = \int_0^T -[\dot{x}(t)-t]\eta(t)\,dt - [x(0)-1]\eta(0) \quad = 0 \quad \forall \eta(t)$$

From the second equation we can again conclude that

$$\dot{x}(t)-t=0$$
$$x(0)=1$$

with solution

$$x(t)=1+\frac{1}{2}t^2$$

Optimality conditions: Example 4 – initial conditions

The optimality conditions are now

$$\left\langle \nabla_x L(x(t),\lambda(t)),\xi \right\rangle = \int_0^T [x(t)+\dot{\lambda}(t)]\xi(t)\,dt - \lambda(T)\xi(T) \quad = 0 \quad \forall \xi(t)$$

$$\left\langle \nabla_\lambda L(x(t),\lambda(t)),\eta \right\rangle = \int_0^T -[\dot{x}(t)-t]\eta(t)\,dt - [x(0)-1]\eta(0) \quad = 0 \quad \forall \eta(t)$$

From the first equation we can now conclude that

$$x(t)+\dot{\lambda}(t)=0$$
$$\lambda(T)=0$$

Note: This is a *linear final value problem* for the Lagrange multiplier.

Given the form of *x(t)*, we can integrate the first of these equations:

$$\lambda(t) = -t - \frac{1}{6}t^3 + a$$

Together with the final condition, we obtain

$$\lambda(t) = -t - \frac{1}{6}t^3 + T + \frac{1}{6}T^3$$

Optimality conditions: Preliminary summary

Summary so far: Consider the (not very interesting) case where the constraints completely determine the solution, i.e. without any control variables:

$$\begin{aligned} \min_{x(t)\in X} \quad & f(x(t),t) = \int_0^T F(x(t),t)\,dt \\ \text{such that} \quad & \dot{x}(t) - g(x(t),t) = 0 \\ & x(0) = x_0 \end{aligned}$$

Then the optimality conditions read in "variational form":

$$\left\langle \nabla_x L(x(t),\lambda(t)),\xi \right\rangle = \int_0^T [F_x(x(t),t) + g_x(x(t),t) + \dot{\lambda}(t)]\xi(t)\,dt - \lambda(T)\xi(T) = 0$$

$$\left\langle \nabla_\lambda L(x(t),\lambda(t)),\eta \right\rangle = \int_0^T -[\dot{x}(t) - g(x(t),t)]\eta(t)\,dt - [x(0) - x_0]\eta(0) = 0$$

$$\forall\, \xi(t), \eta(t)$$

Optimality conditions: Preliminary summary

Summary so far: Consider the (not very interesting) case where the constraints completely determine the solution, i.e. without any control variables:

$$\begin{aligned} \min_{x(t)\in X} \quad & f(x(t),t)=\int_0^T F(x(t),t)\,dt \\ \text{such that} \quad & \dot{x}(t)-g(x(t),t)=0 \\ & x(0)=x_0 \end{aligned}$$

Then the optimality conditions read in "strong" form:

$$\dot{x}(t)-g(x(t),t)=0 \qquad \dot{\lambda}(t)=-F_x(x(t),t)-g_x(x(t),t)$$
$$x(0)=x_0 \qquad \lambda(T)=0$$

Note: Because $x(t)$ does not depend on the Lagrange multiplier, the optimality conditions can be solved by first solving for $x(t)$ as an initial value problem from 0 to T and in a second step solving the final value problem for $\lambda(t)$ backward from T to 0.

Part 25

Optimal control: Theory

Optimality conditions for optimal control problems

Recap:

Let

- X be a space of time-dependent functions
- Q be a space of control parameters, time dependent or not
- $f: X \times Q \to \mathbb{R}$ be a continuous *functional* on X and Q
- $L: X \times Q \to Y$ be a continuous operator on X mapping into a space Y
- $g: X \to Z_x$ be a continuous operator on X mapping into a space Z_x
- $h: Q \to Z_q$ be a continuous operator on Q mapping into a space Z_q

Then the problem

$$\begin{aligned} &\min_{x=x(t)\in X,\, q\in Q} && f(x(t),q) && \\ &\text{such that} && L(x(t),q) = 0 && \forall t \in [t_i, t_f] \\ & && g(x(t)) \geq 0 && \forall t \in [t_i, t_f] \\ & && h(q) \geq 0 && \end{aligned}$$

is called an *optimal control problem*.

Optimality conditions for optimal control problems

There are two important cases:

- The space of control parameters, Q, is a finite dimensional set

$$\begin{array}{lll} \min_{x=x(t)\in X,\, q\in Q=\mathbb{R}^n} & f(x(t),q) & \\ \text{such that} & L(x(t),q)=0 & \forall t\in[t_i,t_f] \\ & g(x(t)) \geq 0 & \forall t\in[t_i,t_f] \\ & h(q) \geq 0 & \end{array}$$

- The space of control parameters, Q, consists of time dependent functions

$$\begin{array}{lll} \min_{x=x(t)\in X,\, q\in Q} & f(x(t),q(t)) & \\ \text{such that} & L(x(t),q(t))=0 & \forall t\in[t_i,t_f] \\ & g(x(t),q(t)) \geq 0 & \forall t\in[t_i,t_f] \\ & h(q(t)) \geq 0 & \end{array}$$

The finite dimensional case

Consider the case of a finite dimensional set of control variables q:

$$\begin{aligned} &\min_{x(t)\in X,\, q\in\mathbb{R}^n} && f(x(t),t,q)=\int_0^T F(x(t),t,q)\,dt \\ &\text{such that} && \dot{x}(t)-g(x(t),t,q)=0 \\ & && x(0)=x_0(q) \end{aligned}$$

with

$$g: X\times\mathbb{R}\times\mathbb{R}^n\to V$$

Because the differential equation now depends on q, the feasible set is no longer just a single point. Rather, for every q there is a feasible $x(t)$ if the ODE is solvable.

In this case, we have (all products are understood to be dot products):

$$L(x(t),q,\lambda(t))=\int_0^T F(x(t),t,q)\,dt-\langle\lambda,\dot{x}(t)-g(x(t),t,q)\rangle-\lambda(0)[x(0)-x_0(q)]$$

$$L: X\times\mathbb{R}^n\times V'\to\mathbb{R}$$

The finite dimensional case

Theorem: Under certain conditions on f, g the solution satisfies

$$\left\langle \nabla_x L(x^*, q^*, \lambda^*), \xi \right\rangle = 0, \qquad \forall \xi \in X$$
$$\left\langle \nabla_\lambda L(x^*, q^*, \lambda^*), \eta \right\rangle = 0, \qquad \forall \eta \in V$$
$$\left\langle \nabla_q L(x^*, q^*, \lambda^*), \rho \right\rangle = 0, \qquad \forall \rho \in (\mathbb{R}^n)' = \mathbb{R}^n$$

The first two conditions can equivalently be written as

$$\int_0^T \nabla_x L(x^*(t), q, \lambda^*(t))\, \xi(t)\ dt = 0, \qquad \forall \xi \in X$$
$$\int_0^T \nabla_\lambda L(x^*(t), q, \lambda^*(t))\, \eta(t)\ dt = 0, \qquad \forall \eta \in V$$

Note: Since q is finite dimensional, the following conditions are equivalent:

$$\left\langle \nabla_q L(x^*, q, \lambda^*), \rho \right\rangle = 0, \qquad \forall \rho \in (\mathbb{R}^n)' = \mathbb{R}^n$$
$$\nabla_q L(x^*, q, \lambda^*) = 0$$

The finite dimensional case

Corollary: Given the form of the Lagrangian,

$$L(x(t),q,\lambda(t))=\int_0^T F(x(t),t,q)-\lambda(t)[\dot{x}(t)-g(x(t),t,q)]dt \\ -\lambda(0)[x(0)-x_0(q)]$$

the optimality conditions are equivalent to the following three sets of equations:

$$\dot{x}(t)=g(x(t),t,q), \qquad x(0)=x_0(q)$$

$$\dot{\lambda}(t)=-F_x(x(t),t,q)-g_x(x(t),t,q), \qquad \lambda(T)=0$$

$$\int_0^T F_q(x(t),t,q)+\lambda(t)\,g_q(x(t),t,q)\,dt+\lambda(0)\frac{\partial x_0(q)}{\partial q}=0$$

Remark: These are called the *primal*, *dual* and *control* equations, respectively.

The finite dimensional case

The optimality conditions for the finite dimensional case are

$$\dot{x}(t)=g(x(t),t,q), \qquad x(0)=x_0(q)$$

$$\dot{\lambda}(t)=-F_x(x(t),t,q)-g_x(x(t),t,q), \qquad \lambda(T)=0$$

$$\int_0^T F_q(x(t),t,q)+\lambda(t)\,g_q(x(t),t,q)\,dt+\lambda(0)\frac{\partial x_0(q)}{\partial q}=0$$

Note: The primal and dual equations are differential equations, whereas the control equation is a (in general nonlinear) algebraic equation. This should be enough to identify the two time-dependent functions and the finite dimensional parameter.

However: Since the control equation determines q for given primal and dual variables, we can no longer integrate the first equation forward and the second backward to solve the problem. *Everything is coupled now!*

The finite dimensional case: An example

Example: Throw a ball from height h with horizontal velocity v_x so that it lands as close as possible from $x=(1,0)$ after one time unit:

$$\min_{\{x(t),v(t)\}\in X,\, q=\{h,v_x\}\in\mathbb{R}^2} \quad \frac{1}{2}\left\|x(t)-\begin{pmatrix}1\\0\end{pmatrix}\right\|^2 = \frac{1}{2}\int_0^T \left\|x(t)-\begin{pmatrix}1\\0\end{pmatrix}\right\|^2 \delta(t-1)\,dt$$

such that

$$\dot{x}(t)=v(t) \qquad x(0)=\begin{pmatrix}0\\h\end{pmatrix}$$

$$\dot{v}(t)=\begin{pmatrix}0\\-1\end{pmatrix} \qquad v(0)=\begin{pmatrix}v_x\\0\end{pmatrix}$$

Then:

$$L(\{x(t),v(t)\},q,\{\lambda_x(t),\lambda_v(t)\})$$

$$=\frac{1}{2}\int_0^T \left\|x(t)-\begin{pmatrix}1\\0\end{pmatrix}\right\|^2 \delta(t-1)\,dt-\left\langle \lambda_x,\dot{x}(t)-v(t)\right\rangle-\left\langle \lambda_v,\dot{v}(t)-\begin{pmatrix}0\\-1\end{pmatrix}\right\rangle$$

$$-\lambda_x(0)\left[x(0)-\begin{pmatrix}0\\h\end{pmatrix}\right]-\lambda_v(0)\left[v(0)-\begin{pmatrix}v_x\\0\end{pmatrix}\right]$$

The finite dimensional case: An example

From the Lagrangian

$$L(\{x(t),v(t)\},q,\{\lambda_x(t),\lambda_v(t)\})$$
$$=\frac{1}{2}\int_0^T\left\|x(t)-\begin{pmatrix}1\\0\end{pmatrix}\right\|^2\delta(t-1)\,dt-\langle\lambda_x,\dot{x}(t)-v(t)\rangle-\left\langle\lambda_v,\dot{v}(t)-\begin{pmatrix}0\\-1\end{pmatrix}\right\rangle$$
$$-\lambda_x(0)\left[x(0)-\begin{pmatrix}0\\h\end{pmatrix}\right]-\lambda_v(0)\left[v(0)-\begin{pmatrix}v_x\\0\end{pmatrix}\right]$$

we get the optimality conditions:

- Derivative with respect to *x(t)*:

$$\int_0^T\left(x(t)-\begin{pmatrix}1\\0\end{pmatrix}\right)\xi_x(t)\delta(t-1)\,dt-\int_0^T\lambda_x(t)\dot{\xi}_x(t)\,dt-\lambda_x(0)\,\xi_x(0)=0\qquad\forall\,\xi_x(t)$$

After integration by parts, we see that this is equivalent to

$$\left(x(t)-\begin{pmatrix}1\\0\end{pmatrix}\right)\delta(t-1)+\dot{\lambda}_x(t)=0\qquad\lambda_x(T)=0$$

The finite dimensional case: An example

From the Lagrangian

$$L(\{x(t),v(t)\},q,\{\lambda_x(t),\lambda_v(t)\})$$
$$=\frac{1}{2}\int_0^T\left\|x(t)-\begin{pmatrix}1\\0\end{pmatrix}\right\|^2\delta(t-1)\,dt-\langle\lambda_x,\dot{x}(t)-v(t)\rangle-\left\langle\lambda_v,\dot{v}(t)-\begin{pmatrix}0\\-1\end{pmatrix}\right\rangle$$
$$-\lambda_x(0)\left[x(0)-\begin{pmatrix}0\\h\end{pmatrix}\right]-\lambda_v(0)\left[v(0)-\begin{pmatrix}v_x\\0\end{pmatrix}\right]$$

we get the optimality conditions:

- Derivative with respect to *v(t)*:

$$\int_0^T\lambda_x(t)\xi_v(t)\,dt-\int_0^T\lambda_v(t)\dot{\xi}_v(t)dt-\lambda_v(0)\xi_v(0)=0\qquad\forall\,\xi_v(t)$$

After integration by parts, we see that this is equivalent to

$$\lambda_x(t)+\dot{\lambda}_v(t)=0\qquad\lambda_v(T)=0$$

The finite dimensional case: An example

From the Lagrangian

$$L(\{x(t),v(t)\},q,\{\lambda_x(t),\lambda_v(t)\})$$
$$=\frac{1}{2}\int_0^T\left(x(t)-\begin{pmatrix}1\\0\end{pmatrix}\right)^2\delta(t-1)\,dt-\langle\lambda_x,\dot{x}(t)-v(t)\rangle-\left\langle\lambda_v,\dot{v}(t)-\begin{pmatrix}0\\-1\end{pmatrix}\right\rangle$$
$$-\lambda_x(0)\left[x(0)-\begin{pmatrix}0\\h\end{pmatrix}\right]-\lambda_v(0)\left[v(0)-\begin{pmatrix}v_x\\0\end{pmatrix}\right]$$

we get the optimality conditions:

- Derivative with respect to $\lambda_x(t)$:

$$\int_0^T\eta_x(t)[\dot{x}(t)-v(t)]\,dt-\eta_x(0)\left[x(0)-\begin{pmatrix}0\\h\end{pmatrix}\right]=0\qquad\forall\,\eta_x(t)$$

This is equivalent to

$$\dot{x}(t)-v(t)=0\qquad x(0)-\begin{pmatrix}0\\h\end{pmatrix}=0$$

The finite dimensional case: An example

From the Lagrangian

$$L(\{x(t),v(t)\},q,\{\lambda_x(t),\lambda_v(t)\})$$
$$=\frac{1}{2}\int_0^T\left(x(t)-\begin{pmatrix}1\\0\end{pmatrix}\right)^2\delta(t-1)\,dt-\langle\lambda_x,\dot{x}(t)-v(t)\rangle-\left\langle\lambda_v,\dot{v}(t)-\begin{pmatrix}0\\-1\end{pmatrix}\right\rangle$$
$$-\lambda_x(0)\left[x(0)-\begin{pmatrix}0\\h\end{pmatrix}\right]-\lambda_v(0)\left[v(0)-\begin{pmatrix}v_x\\0\end{pmatrix}\right]$$

we get the optimality conditions:

- Derivative with respect to $\lambda_v(t)$:

$$\int_0^T\eta_v(t)\left[\dot{v}(t)-\begin{pmatrix}0\\-1\end{pmatrix}\right]dt-\eta_v(0)\left[v(0)-\begin{pmatrix}v_x\\0\end{pmatrix}\right]=0\qquad\forall\,\eta_v(t)$$

This is equivalent to

$$\dot{v}(t)-\begin{pmatrix}0\\-1\end{pmatrix}=0\qquad v(0)-\begin{pmatrix}v_x\\0\end{pmatrix}=0$$

The finite dimensional case: An example

From the Lagrangian

$$L(\{x(t),v(t)\},q,\{\lambda_x(t),\lambda_v(t)\})$$
$$= \frac{1}{2}\int_0^T \left\| x(t) - \begin{pmatrix} 1 \\ 0 \end{pmatrix} \right\|^2 \delta(t-1)\,dt - \left\langle \lambda_x, \dot{x}(t) - v(t) \right\rangle - \left\langle \lambda_v, \dot{v}(t) - \begin{pmatrix} 0 \\ -1 \end{pmatrix} \right\rangle$$
$$-\lambda_x(0)\left[x(0) - \begin{pmatrix} 0 \\ h \end{pmatrix} \right] - \lambda_v(0)\left[v(0) - \begin{pmatrix} v_x \\ 0 \end{pmatrix} \right]$$

we get the optimality conditions:

- Derivative with respect to the first control parameter h:

$$\lambda_{x,2}(0) = 0$$

- Derivative with respect to the second control parameter v_x:

$$\lambda_{v,1}(0) = 0$$

The finite dimensional case: An example

The complete set of optimality conditions is now as follows:

State equations:
(initial value problem)

$$\dot{x}(t) - v(t) = 0 \qquad x(0) - \begin{pmatrix} 0 \\ h \end{pmatrix} = 0$$

$$\dot{v}(t) - \begin{pmatrix} 0 \\ -1 \end{pmatrix} = 0 \qquad v(0) - \begin{pmatrix} v_x \\ 0 \end{pmatrix} = 0$$

Adjoint equations:
(final value problem)

$$\left(x(t) - \begin{pmatrix} 1 \\ 0 \end{pmatrix} \right) \delta(t-1) + \dot{\lambda}_x(t) = 0 \qquad \lambda_x(T) = 0$$

$$\lambda_x(t) + \dot{\lambda}_v(t) = 0 \qquad \lambda_v(T) = 0$$

Control equations:
(algebraic)

$$\lambda_{x,2}(0) = 0$$

$$\lambda_{v,1}(0) = 0$$

The finite dimensional case: An example

In this simple example, we can integrate the optimality conditions in time:

State equations:
(initial value problem)

$$\dot{x}(t) - v(t) = 0 \qquad x(0) - \begin{pmatrix} 0 \\ h \end{pmatrix} = 0$$

$$\dot{v}(t) - \begin{pmatrix} 0 \\ -1 \end{pmatrix} = 0 \qquad v(0) - \begin{pmatrix} v_x \\ 0 \end{pmatrix} = 0$$

Solution:

$$v(t) = \begin{pmatrix} v_x \\ -t \end{pmatrix}$$

$$x(t) = \begin{pmatrix} v_x t \\ h - \frac{1}{2} t^2 \end{pmatrix}$$

The finite dimensional case: An example

In this simple example, we can integrate the optimality conditions in time:

Adjoint equations: $\left(x(t)-\begin{pmatrix}1\\0\end{pmatrix}\right)\delta(t-1)+\dot{\lambda}_x(t)=0 \qquad \lambda_x(T)=0$

(final value problem)

$$\lambda_x(t)+\dot{\lambda}_v(t)=0 \qquad \lambda_v(T)=0$$

Solution: $\lambda_x(t)=-\left[x(1)-\begin{pmatrix}1\\0\end{pmatrix}\right] \quad \text{for } t<1 \qquad \lambda_v(t)=\left[x(1)-\begin{pmatrix}1\\0\end{pmatrix}\right]t \quad \text{for } t<1$

$$\lambda_x(t)=0 \quad \text{for } t>1 \qquad \lambda_v(t)=0 \quad \text{for } t>1$$

Using what we found for x(1) previously:

$$\lambda_x(t)=-\begin{pmatrix}v_x-1\\h-\frac{1}{2}\end{pmatrix} \quad \text{for } t<1 \qquad \lambda_v(t)=\begin{pmatrix}v_x-1\\h-\frac{1}{2}\end{pmatrix}(t-1) \quad \text{for } t<1$$

$$\lambda_x(t)=0 \quad \text{for } t>1 \qquad \lambda_v(t)=0 \quad \text{for } t>1$$

The finite dimensional case: An example

In the final step, we use the control equations:

$$\lambda_{x,2}(0)=0$$

$$\lambda_{v,1}(0)=0$$

But we know that

$$\lambda_x(t)=-\begin{pmatrix} v_x-1 \\ h-\frac{1}{2} \end{pmatrix} \quad \text{for } t<1 \qquad\qquad \lambda_v(t)=\begin{pmatrix} v_x-1 \\ h-\frac{1}{2} \end{pmatrix}(t-1) \quad \text{for } t<1$$

Consequently, the solution is given by

$$h=\frac{1}{2}$$

$$v_x=1$$

The infinite dimensional case

Consider the case of a control variable $q(t)$ that is a function (here, for example, a function in L^2):

$$\begin{aligned} &\min_{x(t)\in X,\, q(t)\in L^2([0,T])} && f(x(t),t,q(t)) = \int_0^T F(x(t),t,q(t))\,dt \\ &\text{such that} && \dot{x}(t) - g(x(t),t,q(t)) = 0 \\ & && x(0) = x_0(q(0)) \end{aligned}$$

with

$$g : X \times \mathbb{R} \times \mathbb{R}^n \to V$$

In this case, we have

$$\begin{aligned} L(x(t),q(t),\lambda(t)) = \int_0^T F(x(t),t,q(t))\,dt - \langle \lambda, \dot{x}(t) - g(x(t),t,q(t)) \rangle \\ - \lambda(0)[x(0) - x_0(q(0))] \end{aligned}$$

$$L : X \times L^2([0,T]) \times V' \to \mathbb{R}$$

The infinite dimensional case

Theorem: Under certain conditions on f, g the solution satisfies

$$\left\langle \nabla_x L(x^*, q^*, \lambda^*), \xi \right\rangle = 0, \qquad \forall \xi \in X$$
$$\left\langle \nabla_\lambda L(x^*, q^*, \lambda^*), \eta \right\rangle = 0, \qquad \forall \eta \in V$$
$$\left\langle \nabla_q L(x^*, q^*, \lambda^*), \rho \right\rangle = 0, \qquad \forall \rho \in L^2([0,T])' = L^2([0,T])$$

The first two conditions can equivalently be written as

$$\int_0^T \nabla_x L(x^*(t), q, \lambda^*(t))\, \xi(t)\ dt = 0, \qquad \forall \xi \in X$$
$$\int_0^T \nabla_\lambda L(x^*(t), q, \lambda^*(t))\, \eta(t)\ dt = 0, \qquad \forall \eta \in V$$

Note: Since q is is now a function, the third optimality condition is:

$$\int_0^T \nabla_q L(x^*(t), q, \lambda^*(t))\, \rho(t)\ dt = 0, \qquad \forall \rho \in L^2([0,T])$$

The infinite dimensional case

Corollary: Given the form of the Lagrangian,

$$L(x(t),q(t),\lambda(t))=\int_0^T F(x(t),t,q(t))\,dt-\langle\lambda,\dot{x}(t)-g(x(t),t,q(t))\rangle$$
$$-\lambda(0)[x(0)-x_0(q(0))]$$

the optimality conditions are equivalent to the following three sets of equations:

$$\dot{x}(t)=g(x(t),t,q(t)), \qquad x(0)=x_0(q(0))$$

$$\dot{\lambda}(t)=-F_x(x(t),t,q(t))-g_x(x(t),t,q(t)), \qquad \lambda(T)=0$$

$$F_q(x(t),t,q)+\lambda(t)g_q(x(t),t,q)=0, \qquad \lambda(0)\frac{\partial x_0(q)}{\partial q}=0$$

Remark: These are again called the *primal*, *dual* and *control* equations, respectively.

The infinite dimensional case

The optimality conditions for the infinite dimensional case are

$$\dot{x}(t) = g(x(t), t, q(t)), \qquad x(0) = x_0(q(0))$$

$$\dot{\lambda}(t) = -F_x(x(t), t, q(t)) - g_x(x(t), t, q(t)), \qquad \lambda(T) = 0$$

$$F_q(x(t), t, q) + \lambda(t) g_q(x(t), t, q) = 0, \qquad \lambda(0) \frac{\partial x_0(q)}{\partial q} = 0$$

Note 1: The primal and dual equations are differential equations, whereas the control equation is a (in general nonlinear) algebraic equation that has to hold for all times between 0 and T. This should be enough to identify the three time-dependent functions.

Note 2: Like for the finite dimensional case, all three equations are coupled and can not be solved one after the other.

The infinite dimensional case: An example

Example: Throw a ball from height 1. Use vertical thrusters so that the altitude follows the path $1+t^2$:

$$\min_{\{x(t),v(t)\}\in X,q(t)\in L^2([0,T])} \quad \frac{1}{2}\int_0^T \left(x(t)-(1+t^2)\right)^2 dt$$

$$\text{such that} \quad \dot{x}(t)=v(t) \qquad x(0)=1$$

$$\dot{v}(t)=-1+q(t) \qquad v(0)=0$$

Then:

$$L(\{x(t),v(t)\},q(t),\{\lambda_x(t),\lambda_v(t)\})$$

$$=\frac{1}{2}\int_0^T \left(x(t)-(1+t^2)\right)^2 dt-\langle\lambda_x,\dot{x}(t)-v(t)\rangle-\langle\lambda_v,\dot{v}(t)-[-1+q(t)]\rangle$$

$$-\lambda_x(0)[x(0)-1]-\lambda_v(0)[v(0)-0]$$

The infinite dimensional case: An example

From the Lagrangian

$$L(\{x(t),v(t)\},q(t),\{\lambda_x(t),\lambda_v(t)\})$$
$$=\frac{1}{2}\int_0^T\left(x(t)-(1+t^2)\right)^2dt-\left\langle\lambda_x,\dot{x}(t)-v(t)\right\rangle-\left\langle\lambda_v,\dot{v}(t)-[-1+q(t)]\right\rangle$$
$$-\lambda_x(0)\left[x(0)-1\right]-\lambda_v(0)\left[v(0)-0\right]$$

we get the optimality conditions:

- Derivative with respect to *x(t)*:

$$\int_0^T\left(x(t)-(1+t^2)\right)\xi_x(t)\,dt-\int_0^T\lambda_x(t)\dot{\xi}_x(t)\,dt-\lambda_x(0)\xi_x(0)=0\qquad\forall\,\xi_x(t)$$

After integration by parts, we see that this is equivalent to

$$\left(x(t)-(1+t^2)\right)+\dot{\lambda}_x(t)=0\qquad\lambda_x(T)=0$$

The infinite dimensional case: An example

From the Lagrangian

$$L(\{x(t),v(t)\},q(t),\{\lambda_x(t),\lambda_v(t)\})$$
$$=\frac{1}{2}\int_0^T \left(x(t)-(1+t^2)\right)^2 dt-\langle\lambda_x,\dot{x}(t)-v(t)\rangle-\langle\lambda_v,\dot{v}(t)-[-1+q(t)]\rangle$$
$$-\lambda_x(0)[x(0)-1]-\lambda_v(0)[v(0)-0]$$

we get the optimality conditions:

- Derivative with respect to *v(t)*:

$$\int_0^T \lambda_x(t)\xi_v(t)dt-\int_0^T \lambda_v(t)\dot{\xi}_v(t)dt-\lambda_v(0)\xi_v(0)=0 \qquad \forall\,\xi_v(t)$$

After integration by parts, we see that this is equivalent to

$$\lambda_x(t)+\dot{\lambda}_v(t)=0 \qquad \lambda_v(T)=0$$

The infinite dimensional case: An example

From the Lagrangian

$$L(\{x(t),v(t)\},q(t),\{\lambda_x(t),\lambda_v(t)\})$$
$$=\frac{1}{2}\int_0^T\left(x(t)-(1+t^2)\right)^2dt-\left\langle\lambda_x,\dot{x}(t)-v(t)\right\rangle-\left\langle\lambda_v,\dot{v}(t)-[-1+q(t)]\right\rangle$$
$$-\lambda_x(0)\left[x(0)-1\right]-\lambda_v(0)\left[v(0)-0\right]$$

we get the optimality conditions:

- Derivative with respect to $\lambda_x(t)$:

$$\int_0^T\eta_x(t)\left[\dot{x}(t)-v(t)\right]dt-\eta_x(0)\left[x(0)-1\right]=0\qquad\forall\,\eta_x(t)$$

This is equivalent to

$$\dot{x}(t)-v(t)=0\qquad x(0)-1=0$$

The infinite dimensional case: An example

From the Lagrangian

$$L(\{x(t),v(t)\},q(t),\{\lambda_x(t),\lambda_v(t)\})$$
$$=\frac{1}{2}\int_0^T\left(x(t)-(1+t^2)\right)^2dt-\left\langle\lambda_x,\dot{x}(t)-v(t)\right\rangle-\left\langle\lambda_v,\dot{v}(t)-[-1+q(t)]\right\rangle$$
$$-\lambda_x(0)\left[x(0)-1\right]-\lambda_v(0)\left[v(0)-0\right]$$

we get the optimality conditions:

- Derivative with respect to $\lambda_v(t)$:

$$\int_0^T\eta_v(t)\left[\dot{v}(t)-(-1+q(t))\right]dt-\eta_v(0)\left[v(0)-0\right]=0\qquad\forall\,\eta_v(t)$$

This is equivalent to

$$\dot{v}(t)-(-1+q(t))=0\qquad v(0)=0$$

The infinite dimensional case: An example

From the Lagrangian

$$L(\{x(t),v(t)\},q(t),\{\lambda_x(t),\lambda_v(t)\})$$
$$=\frac{1}{2}\int_0^T\left(x(t)-(1+t^2)\right)^2dt-\left\langle\lambda_x,\dot{x}(t)-v(t)\right\rangle-\left\langle\lambda_v,\dot{v}(t)-[-1+q(t)]\right\rangle$$
$$-\lambda_x(0)\left[x(0)-1\right]-\lambda_v(0)\left[v(0)-0\right]$$

we get the optimality conditions:

- Derivative with respect to the control function *q(t)*:

$$\int_0^T\lambda_v(t)\rho(t)dt=0\qquad\forall\,\rho(t)$$

This is equivalent to

$$\lambda_v(t)=0$$

The infinite dimensional case: An example

The complete set of optimality conditions is now as follows:

State equations: $\dot{x}(t)-v(t)=0 \qquad x(0)-1=0$

(initial value problem)

$\dot{v}(t)-(-1+q(t))=0 \qquad v(0)=0$

Adjoint equations: $\left(x(t)-(1+t^2)\right)+\dot{\lambda}_x(t)=0 \qquad \lambda_x(T)=0$

(final value problem)

$\lambda_x(t)+\dot{\lambda}_v(t)=0 \qquad \lambda_v(T)=0$

Control equation: $\lambda_v(t)=0$

(algebraic, time dependent)

The infinite dimensional case: An example

Let us use all these equations in turn:

Control equation: $\lambda_v(t)=0$

Adjoint equations: $\left(x(t)-(1+t^2)\right)+\dot{\lambda}_x(t)=0 \qquad \lambda_x(T)=0$

$\lambda_x(t)+\dot{\lambda}_v(t)=0 \qquad \lambda_v(T)=0$

Solution:

$$\lambda_v(t)=0$$
$$\lambda_x(t)=0$$
$$x(t)=1+t^2$$

Remark: This already implies that we can follow the desired trajectory *exactly*!

The infinite dimensional case: An example

Let us use all these equations in turn:

Now known: $x(t)=1+t^2$

State equation: $\dot{x}(t)-v(t)=0 \qquad x(0)-1=0$

$\dot{v}(t)-(-1+q(t))=0 \qquad v(0)=0$

Solution:

$v(t)=2t$

$q(t)=\dot{v}(t)+1=2+1=3$

Conclusion: We need a vertical thrust of 3 to offset gravity and achieve the desired trajectory!

Part 26

Optimal control with equality constraints: Theory

Equality constrained optimal control problems

Previously: So far, we have considered optimal control problems where the only constraints were the ODE and initial conditions.

Now: Consider a problem where we also have equality constraints on the state. Specifically, consider final time constraints:

$$\begin{aligned} \min_{x(t)\in X, q(t)\in L^2([0,T])} \quad & f(x(t),t,q(t)) = \int_0^T F(x(t),t,q(t))\,dt \\ \text{such that} \quad & \dot{x}(t) - g(x(t),t,q(t)) = 0 \\ & x(0) = x_0(q(0)) \\ & \psi(x(T),q(T),T) = 0 \end{aligned}$$

Constraints of this form typically occur if we want to be in a certain state (e.g. location) at the end time and seek the minimal energy/ minimal cost path to get there.

Equality constrained optimal control problems

Consider a problem where we also have equality constraints on the state. Specifically, consider final time constraints:

$$\begin{aligned} \min_{x(t)\in X, q(t)\in L^2([0,T])} \quad & f(x(t),t,q(t)) = \int_0^T F(x(t),t,q(t))\,dt \\ \text{such that} \quad & \dot{x}(t) - g(x(t),t,q(t)) = 0 \\ & x(0) = x_0(q(0)) \\ & \psi(x(T),q(T),T) = 0 \end{aligned}$$

Then:

$$\begin{aligned} L(x(t),q(t),\lambda(t),\nu) = \int_0^T F(x(t),t,q(t))\,dt - \langle \lambda, \dot{x}(t) - g(x(t),t,q(t)) \rangle \\ - \lambda(0)[x(0) - x_0(q(0))] - \nu\psi(x(T),q(T),T) \end{aligned}$$

$$L: X \times L^2([0,T]) \times V' \times \mathbb{R} \to \mathbb{R}$$

Equality constrained optimal control problems

Theorem: Under certain conditions the solution satisfies

$$\begin{aligned}
&\left\langle \nabla_x L(x^*,q^*,\lambda^*,\nu^*),\xi\right\rangle = 0, \qquad \forall \xi \in X\\
&\left\langle \nabla_\lambda L(x^*,q^*,\lambda^*,\nu^*),\eta\right\rangle = 0, \qquad \forall \eta \in V\\
&\left\langle \nabla_q L(x^*,q^*,\lambda^*,\nu^*),\rho\right\rangle = 0, \qquad \forall \rho \in L^2([0,T])' = L^2([0,T])\\
&\left\langle \nabla_\nu L(x^*,q^*,\lambda^*,\nu^*),\mu\right\rangle = 0, \qquad \forall \mu \in \mathbb{R}
\end{aligned}$$

Note 1: The last of these equations is simply

$$\psi(x(T),q(T),T) = 0$$

Note 2: The first equation is now

$$\begin{aligned}
\int_0^T F_x(x(t),t,q(t))\xi(t)dt - \left\langle \lambda, \dot{\xi}(t) - g_x(x(t),t,q(t))\xi(t)\right\rangle \\
-\lambda(0)\xi(0) - \nu\,\psi_x(x(T),q(T),T)\xi(T) = 0 \qquad \forall \xi(t)
\end{aligned}$$

Equality constrained optimal control problems

Corollary: Given the form of the Lagrangian,

$$L(x(t),q(t),\lambda(t),\nu)=\int_0^T F(x(t),t,q(t))\,dt-\langle\lambda,\dot{x}(t)-g(x(t),t,q(t))\rangle$$
$$-\lambda(0)[x(0)-x_0(q(0))]-\nu\psi(x(T),q(T),T)$$

the optimality conditions are equivalent to the following four sets of equations:

$$\dot{x}(t)=g(x(t),t,q(t)), \qquad x(0)=x_0(q(0))$$

$$\dot{\lambda}(t)=-F_x(x(t),t,q(t))-g_x(x(t),t,q(t)), \qquad \lambda(T)=-\nu\psi_x(x(T),q(T),T)$$

$$F_q(x(t),t,q)+\lambda(t)g_q(x(t),t,q)=0, \qquad \lambda(0)\frac{\partial x_0(q)}{\partial q}=\nu\psi_q(x(T),q(T),T)$$

$$\psi(x(T),q(T),T)=0$$

These are now called *state equations*, *adjoint equation*, *control equation*, and *transversality equation.*

Equality constrained optimal control problems

Example ("geodesics"): Consider a mars rover. Given a force vector *q(t)* then it will move with a velocity

$$\dot{x}(t) = \phi(x(t))\, q(t)$$

where the function $\phi(x)$ indicates how "rough/smooth" the terrain is at position *x*: if the terrain is smooth, then $\phi(x)$ is large; if it is rough, then $\phi(x)$ is small.

The goal is then to find a path from x_A to x_B with minimal energy. Let's assume that the power necessary to create a force *q(t)* is equal to $|q(t)|^2$. Then the problem is:

$$\begin{aligned} \min_{x(t)\in X,\, q(t)\in L^2([0,T])} \quad & \frac{1}{2}\int_0^T \left(q(t)\right)^2 dt \\ \text{such that} \quad & \dot{x}(t) = \phi(x(t))\, q(t) \\ & x(0) = x_A \\ & x(T) = x_B \end{aligned}$$

Equality constrained optimal control problems

Example ("geodesics"): For the problem

$$\begin{aligned} &\min_{x(t)\in X,\, q(t)\in L^2([0,T])} && \frac{1}{2}\int_0^T \left|q(t)\right|^2 dt \\ &\text{such that} && \dot{x}(t) = \phi(x(t))q(t) \\ & && x(0) = x_A \\ & && x(T) = x_B \end{aligned}$$

the Lagrangian is given by

$$\begin{aligned} L(x(t),q(t),\lambda(t),\nu) = \frac{1}{2}\int_0^T \left|q(t)\right|^2 dt - \left\langle \lambda, \dot{x}(t) - \phi(x(t))q(t)\right\rangle \\ -\lambda(0)[x(0)-x_A] - \nu[x(T)-x_B] \end{aligned}$$

Equality constrained optimal control problems

Example ("geodesics"): The Lagrangian is given by

$$L(x(t),q(t),\lambda(t),\nu) = \frac{1}{2}\int_0^T |q(t)|^2 dt - \langle \lambda, \dot{x}(t) - \phi(x(t))q(t) \rangle$$
$$- \lambda(0)[x(0) - x_A] - \nu[x(T) - x_B]$$

The optimality conditions are then:

$$\dot{\lambda}(t) + \nabla\phi(x(t))[q(t)\cdot\lambda(t)] = 0 \quad \lambda(T) = -\nu$$

$$\dot{x}(t) - \phi(x(t))q(t) = 0 \qquad x(0) = x_A$$

$$q(t) + \lambda\phi(x(t)) = 0$$

$$x(T) = x_B$$

In general, there is no trivial solution to this system.

Equality constrained optimal control problems

Example ("geodesics"): Consider the simplest case, $\phi(x)=1$
Then the optimality conditions are:

$$\dot{\lambda}(t)=0 \qquad \lambda(T)=-\nu$$

$$\dot{x}(t)-q(t)=0 \qquad x(0)=x_A$$

$$q(t)+\lambda=0$$

$$x(T)=x_B$$

This system is solved by

$$\lambda(t)=-\nu \qquad q(t)=\nu$$

$$x(t)=\nu t+x_A \qquad \nu=(x_B-x_A)/T$$

That is, the rover moves at constant speed on a straight line and the optimal value of the objective function is $\frac{1}{2}\nu^2 T=\frac{1}{2}\|x_B-x_A\|^2/T$

Equality constrained optimal control problems

Example ("geodesics"): Consider the more difficult case where the rover can move twice as fast in the lower half plane than in the upper half plane:

$$\phi(x)=\begin{Bmatrix}1 & \text{if } x_2>0\\ 2 & \text{if } x_2\le 0\end{Bmatrix}=2-H(x_2)$$

with $H(y)=0$ for $y=0$. Let $x_A=(-2,1)^T$, $x_B=(2,1)^T$.

Then the optimality conditions are:

$$\dot{\lambda}(t)-\begin{pmatrix}0\\ \delta(x_2(t))\end{pmatrix}[q(t)\cdot\lambda(t)]=0 \qquad \lambda(T)=-\nu$$

$$\dot{x}(t)-(2-H(x_2(t)))q(t)=0 \qquad x(0)=x_A$$

$$q(t)+(2-H(x_2(t)))\lambda=0$$

$$x(T)=x_B$$

Equality constrained optimal control problems

Example ("geodesics"): Consider this difficult case. The conditions

$$\dot{\lambda}(t) - \begin{pmatrix} 0 \\ \delta(x_2(t)) \end{pmatrix} [q(t) \cdot \lambda(t)] = 0 \qquad \lambda(T) = -\nu$$

$$\dot{x}(t) - (2 - H(x_2(t)))\, q(t) = 0 \qquad x(0) = x_A$$

$$q(t) + (2 - H(x_2(t)))\, \lambda = 0$$

$$x(T) = x_B$$

have the following solution (note: the path is entirely in the upper half):

$$\lambda(t) = -\nu \qquad q(t) = \nu$$

$$x(t) = \nu t + x_A \qquad \nu = \frac{1}{T} \begin{pmatrix} 4 \\ 0 \end{pmatrix}$$

The optimal objective function value is then

$$\frac{1}{2} \nu^2 T = \frac{8}{T}$$

Equality constrained optimal control problems

But careful: The conditions

$$\dot{\lambda}(t) - \begin{pmatrix} 0 \\ \delta(x_2(t)) \end{pmatrix} [q(t) \cdot \lambda(t)] = 0 \qquad \lambda(T) = -\nu$$

$$\dot{x}(t) - (2 - H(x_2(t)))q(t) = 0 \qquad x(0) = x_A$$

$$q(t) + (2 - H(x_2(t)))\lambda = 0$$

$$x(T) = x_B$$

also have a solution of the form

$$x(t) = \begin{pmatrix} -2 \\ 1 \end{pmatrix} \to \begin{pmatrix} -\alpha \\ 0 \end{pmatrix} \to \begin{pmatrix} \alpha \\ 0 \end{pmatrix} \to \begin{pmatrix} 2 \\ 1 \end{pmatrix}$$

$$q(t) = \text{const}$$

(Details to be determined. We also have to specify in more detail what it means if we move along the line $x_2 = 0$, c.f. the first equation above.)

Part 27

Direct vs. indirect methods

Direct vs. indirect methods

How do we solve general optimal control problems:

- *Direct methods* are based on the original problem formulation. We can think of them as "discretize first, then optimize".
- *Indirect methods* attempt to solve the optimality conditions. We can think of them as "optimize first, then discretize"

Example: To find a minimum of $f(x)$,

- Direct methods would find a sequence $x_1, x_2, \ldots$ and would only have to ensure that $f(x_1) > f(x_2), \ldots$

  I.e. it would only have to *compare* function values.

- Indirect methods would try to find a solution of the equation $f'(x)=0$.

  I.e. we would have to compute *derivatives* of the objective function.

Direct vs. indirect methods

In practice, all methods in actual use are *direct*:

- For many realistic problems, the user-defined function $F, g, \ldots$ are complicated and providing derivatives for the necessary conditions is not practical
- Good initial estimates for the Lagrange multipliers are typically not available
- Without good initial estimates, indirect methods often just wander off into lala-land unless the problem is exceptionally stable
- If state inequalities are present, one needs to provide an a-priori guess when the inequalities will be active. This is not practical.

Consequently: The optimality conditions derived so far are of mostly theoretical interest in optimal control. They are of importance in PDE-constrained optimization, however.

Part 28

Numerical solution of optimal control problems with direct methods

The shooting method for realistic optimal control

Consider a problem with equality constraints on the state: Specifically, consider final time constraints:

$$\begin{aligned} &\min_{x(t)\in X,\, q(t)\in L^2([0,T])} && f(x(t),t,q(t)) = \int_0^T F(x(t),t,q(t))\,dt \\ &\text{such that} && \dot{x}(t) - g(x(t),t,q(t)) = 0 \\ & && x(0) = x_0(q(0)) \\ & && \psi(x(T),q(T),T) = 0 \end{aligned}$$

Approach: We want to apply a (single) shooting method to it. To this end, introduce a time mesh

$$0 = t_0 < t_1 < t_2 < \ldots < t_N = T$$

and a time step size $k_n = t_n - t_{n-1}$.

We then apply one of the common time stepping methods to the optimal control problem. (This step is called "discretization".)

The shooting method for realistic optimal control

Consider a problem with equality constraints on the state: Specifically, consider final time constraints:

$$\begin{aligned} &\min_{x(t)\in X,\, q(t)\in L^2([0,T])} && f(x(t),t,q(t)) = \int_0^T F(x(t),t,q(t))\,dt \\ &\text{such that} && \dot{x}(t) - g(x(t),t,q(t)) = 0 \\ & && x(0) = x_0(q(0)) \\ & && \psi(x(T),q(T),T) = 0 \end{aligned}$$

Example: Using the (overly trivial and low-order) forward Euler method, we replace the original problem with the discretized form

$$\min_{x^n,q^n,\, n=0,\ldots,N} \quad f(x^0,\ldots,x^N,q^0,\ldots,q^N) = \sum_{n=1}^{N} k_n F\left(\frac{x^n + x^{n-1}}{2}, t_n, \frac{q^n + q^{n-1}}{2}\right)$$

$$\begin{aligned} &\text{such that} && \frac{x^n - x^{n-1}}{k_n} - g(x^{n-1}(t), t_{n-1}, q^{n-1}(t)) = 0 \\ & && x^0 = x_0(q^0) \\ & && \psi(x^N, q^N, T) = 0 \end{aligned}$$

The shooting method for realistic optimal control

The discretized problem now reads as:

$$\min_{x^n, q^n,\, n=0,\ldots,N} \quad f(x^0,\ldots,x^N,q^0,\ldots,q^N)=\sum_{n=1}^{N} k_n F\left(\frac{x^n+x^{n-1}}{2}, t_n, \frac{q^n+q^{n-1}}{2}\right)$$

$$\text{such that} \quad \frac{x^n-x^{n-1}}{k_n}-g\left(x^{n-1}(t),t_{n-1},q^{n-1}(t)\right)=0$$

$$x^0=x_0(q^0)$$

$$\psi(x^N,q^N,T)=0$$

Note: Introducing $y=(x^0,q^0,x^1,q^1,\ldots x^N,q^N)^T$ this has the form

$$\min_y \quad f(y)$$
$$\text{such that} \quad c(y)=0$$

If $x(t)$ has n_x components and $q(t)$ has n_q components, then

$$y\in\mathbb{R}^{(N+1)(n_x+n_q)}, \qquad c\in\mathbb{R}^{Nn_x+n_x+n_\psi}$$

The shooting method for realistic optimal control

The discretized problem now reads is equivalent to a large, nonlinear optimization problem:

$$\begin{aligned} \min_y \quad & f(y) \\ \text{such that} \quad & c(y) = 0 \end{aligned}$$

Its solution has to satisfy

$$\begin{aligned} \frac{\partial L}{\partial y} &= \frac{\partial f(y)}{\partial y} - \lambda^T \frac{\partial c(y)}{\partial y} = 0 \\ \frac{\partial L}{\partial \lambda} &= c(y) = 0 \end{aligned}$$

where $L(y,\lambda) = f(y) - \lambda^T c(y)$.

Note: We have one Lagrange multiplier for each time step, but these are all independent. Conversely, in the indirect approach, we would have had Lagrange multipliers for each time step that satisfy a discrete ODE and are therefore all coupled.

This is what makes the direct method more practical.

The shooting method for realistic optimal control

We can solve this problem using, for example, the SQP method:

$$\begin{pmatrix} \nabla_y^2 f(y_k) - \lambda_k^T \nabla_y^2 c(y_k) & -\nabla_y c(y_k) \\ -\nabla_y c(y_k)^T & 0 \end{pmatrix} \begin{pmatrix} p_k^y \\ p_k^\lambda \end{pmatrix} =$$

$$= -\begin{pmatrix} \nabla_y f(y_k) - \lambda_k^T \nabla_y c(y_k) \\ -g(y_k) \end{pmatrix}$$

We will abbreviate this as

$$\begin{pmatrix} W_k & -A_k \\ -A_k^T & 0 \end{pmatrix} \begin{pmatrix} p_k^y \\ p_k^\lambda \end{pmatrix} = -\begin{pmatrix} \nabla_y f(y_k) - \lambda_k^T \nabla c(y_k) \\ -c(y_k) \end{pmatrix}$$

where

$$W_k = \nabla_y^2 L(y_k, \lambda_k)$$

$$A_k = \nabla_y c(y_k) = -\nabla_x \nabla_\lambda L(y_k, \lambda_k)$$

The shooting method for realistic optimal control

In each iteration, we have to solve the linear system

$$\begin{pmatrix} W_k & -A_k \\ -A_k^T & \cdot \end{pmatrix} \begin{pmatrix} p_k^y \\ p_k^\lambda \end{pmatrix} = -\begin{pmatrix} \nabla_y f(y_k) - \lambda_k^T \nabla c(y_k) \\ -c(y_k) \end{pmatrix}$$

The matrix on the left has dimensions

$$\begin{aligned} &\left[(N+1)(n_x+n_q)+Nn_x+n_x+n_\psi\right]\times\left[(N+1)(n_x+n_q)+Nn_x+n_x+n_\psi\right] \\ &=\left[(N+1)(n_x+1+n_q)+n_\psi\right]\times\left[(N+1)(n_x+1+n_q)+n_\psi\right] \end{aligned}$$

Note: It is not uncommon to have 10-100 state variables, 1-10 control variables, and 1,000-10,000 time steps. That means the matrix on the left can easily be of size $10{,}000^2$ to $1{,}000{,}000^2$!

That would be a very large and awkward system to solve in each iteration!

The shooting method for realistic optimal control

Conclusion so far: The SQP system

$$\begin{pmatrix} W_k & -A_k \\ -A_k^T & 0 \end{pmatrix} \begin{pmatrix} p_k^y \\ p_k^\lambda \end{pmatrix} = -\begin{pmatrix} \nabla_y f(y_k) - \lambda_k^T \nabla c(y_k) \\ -c(y_k) \end{pmatrix}$$

is very large.

However: The matrix on the left is also almost completely empty. Remember that

$$W_k = \nabla_y^2 L(y_k, \lambda_k) = \nabla_y^2 f(y_k) - \sum_i (\lambda_{k,i}) \nabla_y^2 c_i(y_k)$$

$$A_k = \nabla_y c(y_k)$$

and that

$$f(y) = \sum_{n=1}^{N} k_n F\left(\frac{x^n + x^{n-1}}{2}, t_n, \frac{q^n + q^{n-1}}{2} \right)$$

$$c(y) = \begin{pmatrix} \frac{x^n - x^{n-1}}{k_n} - g(x^{n-1}(t), t_{n-1}, q^{n-1}(t)) \\ x^0 - x_0(q^0) \\ \psi(x^N, q^N, T) \end{pmatrix}$$

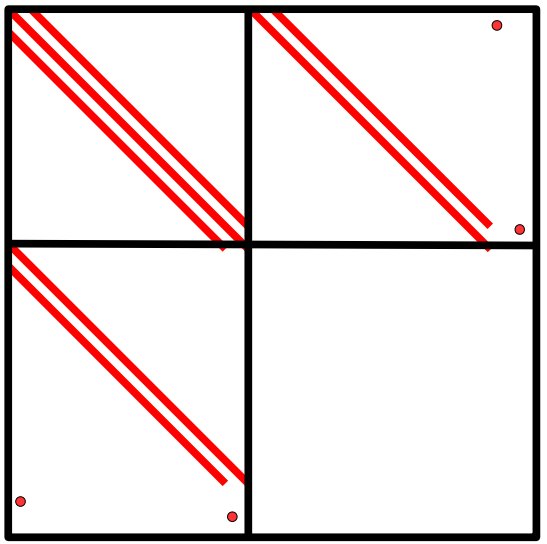
The shooting method for realistic optimal control

Conclusion so far: The SQP system

$$\begin{pmatrix} W_k & -A_k \\ -A_k^T & 0 \end{pmatrix} \begin{pmatrix} p_k^y \\ p_k^\lambda \end{pmatrix} = -\begin{pmatrix} \nabla_y f(y_k) - \lambda_k^T \nabla c(y_k) \\ -c(y_k) \end{pmatrix}$$

is very large.

However: The matrix on the left is also almost completely empty. It typically has a (block) structure of the form



Note: Such systems are not overly complicated to solve.

The multiple shooting method

Instead of using the single shooting method,

$$\min_{x^n,q^n,n=0,\dots,N} \quad f(x^0,\dots,x^N,q^0,\dots,q^N)=\sum_{n=1}^{N} k_n F\left(\frac{x^n+x^{n-1}}{2},t_n,\frac{q^n+q^{n-1}}{2}\right)$$

$$\text{such that} \qquad \frac{x^n-x^{n-1}}{k_n}-g(x^{n-1}(t),t_{n-1},q^{n-1}(t))=0$$

$$x^0=x_0(q^0)$$

$$\psi(x^N,q^N,T)=0$$

we can relax the formulation to obtain the multiple shooting method:

$$\min_{x^{s,n},q^{s,n},n=0,\dots,N_s,s=1\dots S} \quad \sum_{s=1}^{S}\sum_{n=1}^{N_s} k_{s,n} F\left(\frac{x^{s,n}+x^{s,n-1}}{2},t_{s,n},\frac{q^{s,n}+q^{s,n-1}}{2}\right)$$

$$\text{such that} \qquad \frac{x^{s,n}-x^{s,n-1}}{k_{s,n}}-g(x^{s,n-1}(t),t_{s,n-1},q^{s,n-1}(t))=0, \quad s=2\dots S$$

$$x^{1,0}=x_0(q^{1,0})$$

$$x^{s,0}=x^{s-1,N_{s-1}}, \qquad s=2\dots S$$

$$\psi(x^{S,N_S},q^{S,N_S},T)=0$$

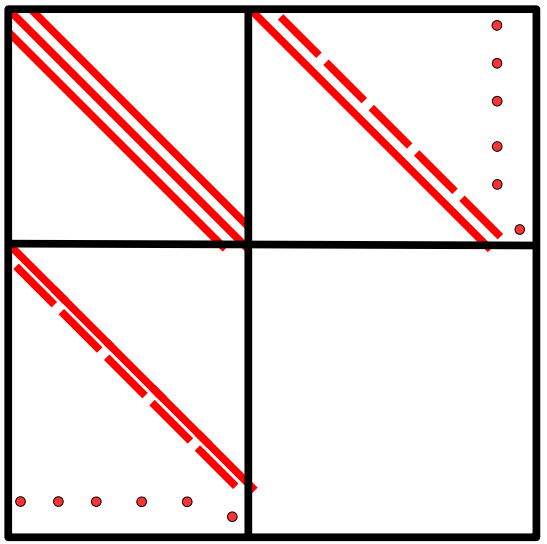
The multiple shooting method

The multiple shooting method: The SQP system has again the form

$$\begin{pmatrix} W_k & -A_k \\ -A_k^T & 0 \end{pmatrix} \begin{pmatrix} p_k^y \\ p_k^\lambda \end{pmatrix} = -\begin{pmatrix} \nabla_y f(y_k) - \lambda_k^T \nabla c(y_k) \\ -c(y_k) \end{pmatrix}$$

with now even more variables.

However: The matrix on the left is again also almost completely empty. It typically has a (block) structure of the form



Note: Again, such systems are not overly complicated to solve. In particular, this system can now also be solved in parallel.

Time stepping vs. SQP

Remark: A typical strategy of coupling time discretization and nonlinear optimization is

- to start with a relatively small number of time steps
- do one or more SQP steps
- interpolate the current solution variables x^n, q^n as well as the Lagrange multipliers to a finer time mesh
- do some more SQP iterations and iterate this procedure

Advantages:

- While we are far away from the solution, the number of variables is small and so every SQP step is fast
- Only as we get closer to the solution do iterations get expensive
- The degree of ill-posedness of problems typically increases with smaller time steps. We can work with well-posed problems while we need to take large steps, stabilizing the process.

Part 29

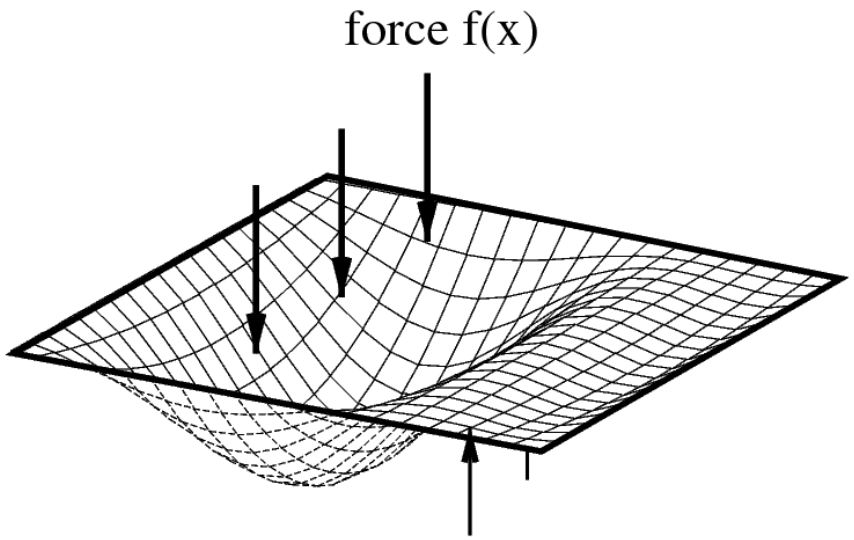
Optimization with Partial Differential Equations

(PDE-constrained optimization)

An example

Consider the following simple example:

The vertical deflection of a thin sheet (e.g. a membrane) clamped at the boundary is described by the following Poisson equation:



$$-\Delta u(x) = f(x) \qquad \text{in } \Omega$$
$$u(x) = g(x) \qquad \text{on } \partial\Omega$$

Goal: Find the force *f(x)* so that the deflection *u(x)* matches a desired deflection *z(x)*!

An example

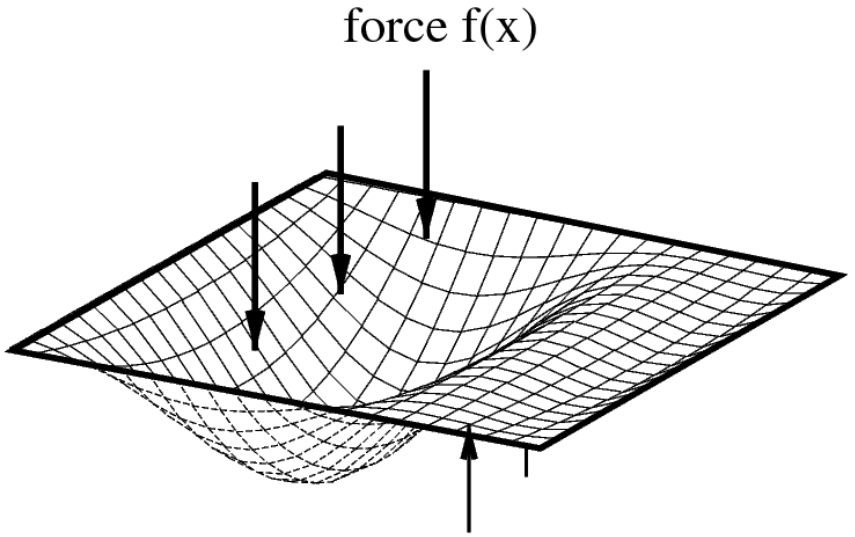
Mathematical description:

For simplicity, let us measure the distance between the actual state $u(x)$ and the desired state $z(x)$ in the L^2 norm:

$$\min_{u(x),\,f(x)} \quad \frac{1}{2}\|u(x)-z(x)\|^2 = \frac{1}{2}\int_\Omega [u(x)-z(x)]^2 \, dx$$

$$\text{subject to} \quad -\Delta u(x) = f(x) \qquad \text{in } \Omega$$

$$u(x) = g(x) \qquad \text{on } \partial\Omega$$



Note: We could have measured the distance in other norms as well.

An example

Mathematical description:

For problems with partial differential equations, one typically needs to be careful with how exactly the various function spaces are chosen. Here:

$$\min_{u(x),\,f(x)} \quad \frac{1}{2}\|u(x)-z(x)\|^2 = \frac{1}{2}\int_\Omega [u(x)-z(x)]^2\, dx$$

$$\text{subject to} \quad -\Delta u(x) = f(x) \qquad \text{in } \Omega$$

$$u(x) = g(x) \qquad \text{on } \partial\Omega$$

Consequently:

- $u(x)$ must be from H^1
- $f(x)$ must be from H^{-1}
- $z(x)$ must be from L^2

Note: If $u(x)$ is in H^1 then the restriction of $u(x)$ to the boundary is in $H^{1/2}$.

An example

Mathematical description:

In analogy to how we did this for the ODE case, we could then define the Lagrangian corresponding to

$$\min_{u(x)} \frac{1}{2}\|u(x)-z(x)\|^2 = \frac{1}{2}\int_\Omega [u(x)-z(x)]^2\, dx$$

$$\text{subject to } -\Delta u(x) = f(x) \quad \text{in } \Omega \quad (\text{to hold in } H^{-1})$$

$$u(x) = g(x) \quad \text{on } \partial\Omega \quad (\text{to hold in } H^{1/2})$$

by setting

$$L(u(x), f(x), \lambda(x)) = \frac{1}{2}\|u(x)-z(x)\|^2 - \left\langle -\Delta u(x) - f(x), \lambda(x)\right\rangle_{H^{-1}\times H^1} - \left\langle u(x)-g(x), \lambda(x)\right\rangle_{H^{1/2}\times H^{-1/2}}$$

An example

Note:

By integrating by parts, we see that

$$L(u(x),f(x),\lambda(x)) = \frac{1}{2}\|u(x)-z(x)\|^2 - \langle -\Delta u(x)-f(x),\lambda(x)\rangle_{H^{-1}\times H^1} - \langle u(x)-g(x),\lambda(x)\rangle_{H^{1/2}\times H^{-1/2}}$$

$$= \int_\Omega \frac{1}{2}[u(x)-z(x)]^2 - [-\Delta u(x)-f(x)]\lambda(x) - \int_{\partial\Omega}[u(x)-g(x)]\lambda(x)$$

$$= \int_\Omega \frac{1}{2}[u(x)-z(x)]^2 - \nabla u(x)\cdot\nabla\lambda(x) + f(x)\lambda(x) - \int_{\partial\Omega} n\cdot\nabla u(x)\lambda(x) + [u(x)-g(x)]\lambda(x)$$

An example

Problem:

The optimality conditions that result from the Lagrangian

$$L(u(x),f(x),\lambda(x)) = \int_\Omega \frac{1}{2}[u(x)-z(x)]^2-\nabla u(x)\cdot\nabla\lambda(x)+f(x)\lambda(x) - \int_{\partial\Omega} n\cdot\nabla u(x)\lambda(x)+[u(x)-g(x)]\lambda(x)$$

read:

$$\int_\Omega -\nabla u(x)\cdot\nabla\xi(x)+f(x)\xi(x) - \int_{\partial\Omega} n\cdot\nabla u(x)\xi(x)+[u(x)-g(x)]\xi(x)=0 \quad \forall\,\xi(x)\in H^1(\Omega)$$

$$\int_\Omega [u(x)-z(x)]\eta(x)-\nabla\eta(x)\cdot\nabla\lambda(x) - \int_{\partial\Omega} n\cdot\nabla\eta(x)\lambda(x)+\eta(x)\lambda(x)=0 \qquad \forall\,\eta(x)\in H^1(\Omega)$$

$$\int_\Omega \rho(x)\lambda(x)=0 \qquad \forall\,\rho(x)\in L^2(\Omega)$$

An example

Problem:

Consider just the first optimality condition:

$$\int_\Omega -\nabla u(x)\cdot\nabla\xi(x)+f(x)\xi(x) - \int_{\partial\Omega} n\cdot\nabla u(x)\xi(x)+[u(x)-g(x)]\xi(x)=0 \quad \forall\,\xi(x)\in H^1(\Omega)$$

We would like to choose test functions in such a way that we can conclude that

$$\begin{aligned} -\Delta u(x) &= f(x) && \text{in } \Omega \\ u(x) &= g(x) && \text{on } \partial\Omega \end{aligned}$$

However: We can't do this in the same way as we did when we analyzed the weak form of an ODE because our test functions are in too small a space!

Remember: We wanted that $u(x)=g(x)$ in $H^{1/2}$, so we would need to test with functions in $H^{-1/2}$, but we can only test in $H^{1/2}$!

An example

Alternative mathematical description:

Define the (affine) function space

$$H_g^1 = \{u(x) \in H^1(\Omega) : u(x)|_{\partial\Omega} = g(x)\}$$
$$= \{u(x) \in L^2(\Omega) : \nabla u(x) \in L^2(\Omega), u(x)|_{\partial\Omega} = g(x)\}$$

and rewrite the problem as

$$\min_{u(x) \in H_g^1} \frac{1}{2} \|u(x) - z(x)\|^2$$

subject to $-\Delta u(x) = f(x)$ in Ω (to hold in H^{-1})

Then we can define the Lagrangian as follows:

$$L(u(x), f(x), \lambda(x)) = \frac{1}{2} \|u(x) - z(x)\|^2 - \langle -\Delta u(x) - f(x), \lambda(x) \rangle_{H^{-1} \times H_0^1}$$

$$L : H_g^1 \times L^2 \times H_0^1 \to \mathbb{R}$$

An example

Alternative mathematical description:

The optimality conditions that result from this formulation are

$$\int_\Omega -\nabla u(x)\cdot\nabla \xi(x)+f(x)\xi(x) - \int_{\partial\Omega} n\cdot\nabla u(x)\,\xi(x)=0 \qquad \forall \xi(x)\in H_0^1(\Omega)$$

$$\int_\Omega [u(x)-z(x)]\eta(x)-\nabla \eta(x)\cdot\nabla \lambda(x) - \int_{\partial\Omega} n\cdot\nabla \eta(x)\lambda(x)=0 \qquad \forall \eta(x)\in H_0^1(\Omega)$$

$$\int_\Omega \rho(x)\lambda(x)=0 \qquad \forall \rho(x)\in L^2(\Omega)$$

Note: The test functions in the first and second optimality condition now have *zero* boundary values. This cancels the remaining boundary integrals.

An example

Alternative mathematical description:

In "strong form" these optimality conditions are equivalent to

$$-\Delta u = f$$

$$-\Delta \lambda = u(x) - z(x)$$

$$\int_\Omega \rho(x)\lambda(x) = 0 \qquad \forall \rho(x) \in L^2(\Omega)$$

Note 1: The last condition does not necessarily imply that the Lagrange multiplier is zero because we only test with functions in L^2, not H^{-1}.

Note 2: Consequently, we can't be sure that the optimal u equals z. This makes sense because z is in L^2 but u in H^1. Furthermore, we have required f to be from L^2, making u even smoother than H^1.

Note 3: We could have avoided this problem by allowing f to be in H^{-1}.

Solving PDE-constrained problems

The optimality conditions then read:

Find functions $u(x), f(x), \lambda(x)$ so that

$$\int_\Omega -\nabla u(x)\cdot\nabla \xi(x) + f(x)\xi(x) = 0 \qquad \forall\, \xi(x) \in H_0^1(\Omega)$$

$$\int_\Omega [u(x) - z(x)]\eta(x) - \nabla \eta(x)\cdot\nabla \lambda(x) = 0 \qquad \forall\, \eta(x) \in H_0^1(\Omega)$$

$$\int_\Omega \rho(x)\lambda(x) = 0 \qquad \forall\, \rho(x) \in L^2(\Omega)$$

We can approximate solutions by seeking $u_h(x), f_h(x), \lambda_h(x)$ in *finite dimensional* subspaces. For example:

- $u_h \in V_{h,g} = \{v_h \in P_1(T) : v_h|_{\partial\Omega} = g\} \subset V_g = H_g^1$
- $f_h \in H_h = \{p_h \in DP_0(T)\} \subset H = L^2$
- $\lambda_h \in V_{h,0} = \{v_h \in P_1(T) : v_h|_{\partial\Omega} = 0\} \subset V_0 = H_0^1$

Solving PDE-constrained problems

The discrete optimality conditions in weak form read:

Find functions $u_h(x), f_h(x), \lambda_h(x)$ so that

$$\int_\Omega -\nabla u_h(x)\cdot\nabla \xi_h(x)+f_h(x)\,\xi_h(x)=0 \qquad \forall\, \xi_h(x)\in V_{h,0}$$

$$\int_\Omega [u_h(x)-z(x)]\eta_h(x)-\nabla \eta_h(x)\cdot\nabla \lambda_h(x)=0 \qquad \forall\, \eta_h(x)\in V_{h,0}$$

$$\int_\Omega \rho_h(x)\lambda_h(x)=0 \qquad \forall\, \rho_h(x)\in H_h$$

Note: As before, the last condition does not imply that $\lambda_h(x)=0$. Rather, it only means that on every cell, the *average* of $\lambda_h(x)$ has to be zero!

Solving PDE-constrained problems

The discrete optimality conditions in weak form read:

Find functions $u_h(x), f_h(x), \lambda_h(x)$ so that

$$\int_\Omega -\nabla u_h(x)\cdot\nabla \xi_h(x)+f_h(x)\,\xi_h(x)=0 \qquad \forall\, \xi_h(x)\in V_{h,0}$$

$$\int_\Omega [u_h(x)-z(x)]\eta_h(x)-\nabla \eta_h(x)\cdot\nabla \lambda_h(x)=0 \qquad \forall\, \eta_h(x)\in V_{h,0}$$

$$\int_\Omega \rho_h(x)\lambda_h(x)=0 \qquad \forall\, \rho_h(x)\in H_h$$

By choosing a basis for the finite dimensional spaces, we arrive at the linear system

$$\begin{pmatrix} A & Q & 0 \\ M & 0 & A^T \\ 0 & 0 & Q^T \end{pmatrix}\begin{pmatrix} U \\ F \\ \Lambda \end{pmatrix}=\begin{pmatrix} 0 \\ Z \\ 0 \end{pmatrix} \quad \Leftrightarrow \quad \begin{pmatrix} M & 0 & A^T \\ 0 & 0 & Q^T \\ A & Q & 0 \end{pmatrix}\begin{pmatrix} U \\ F \\ \Lambda \end{pmatrix}=\begin{pmatrix} Z \\ 0 \\ 0 \end{pmatrix}$$

Solving PDE-constrained problems

We now have to solve the following linear system:

$$\begin{pmatrix} M & 0 & A^T \\ 0 & 0 & Q^T \\ A & Q & 0 \end{pmatrix} \begin{pmatrix} U \\ F \\ \Lambda \end{pmatrix} = \begin{pmatrix} Z \\ 0 \\ 0 \end{pmatrix}$$

- This system may be very large: On a 100x100x100 grid in 3d, the system has 3 million unknowns
- The matrix is symmetric
- The matrix is indefinite
- The matrix is sparse

Problem 1: Large linear systems with these properties are typically difficult to solve unless we have good preconditioners.

Problem 2: Good preconditioners are typically only available for discretized versions of the underlying PDE (i.e. for the matrices A, A^T).

Solving PDE-constrained problems

We now have to solve the following linear system:

$$\begin{pmatrix} M & 0 & A^T \\ 0 & 0 & Q^T \\ A & Q & 0 \end{pmatrix} \begin{pmatrix} U \\ F \\ \Lambda \end{pmatrix} = \begin{pmatrix} Z \\ 0 \\ 0 \end{pmatrix}$$

One solution: Use block elimination

$$AU + QF = 0 \quad \Rightarrow \quad U = -A^{-1} QF$$

$$MU + A^T \Lambda = Z \quad \Rightarrow \quad \Lambda = A^{-T}(Z - MU) = A^{-T}(Z + MA^{-1} QF)$$

$$Q^T \Lambda = 0 \quad \Rightarrow \quad \underbrace{Q^T A^{-T} M A^{-1} Q}_{S} F = -Q^T A^{-T} Z$$

Definition: S is called the *Schur complement* of the system matrix with respect to the FF block.

Strategy: Solve the last equation for F and the solve the other two equations in turn.

Solving PDE-constrained problems

We now have to solve the following sequence of linear systems:

$$\underbrace{Q^T A^{-T} M A^{-1} Q}_{S} F = -Q^T A^{-T} Z$$

$$AU = -QF$$

$$A^T \Lambda = Z - MU$$

Properties:

- S is symmetric and positive definite.
  We can thus apply the Conjugate Gradient (CG) method
- The entries of S are typically not known explicitly
- The matrix S is dense

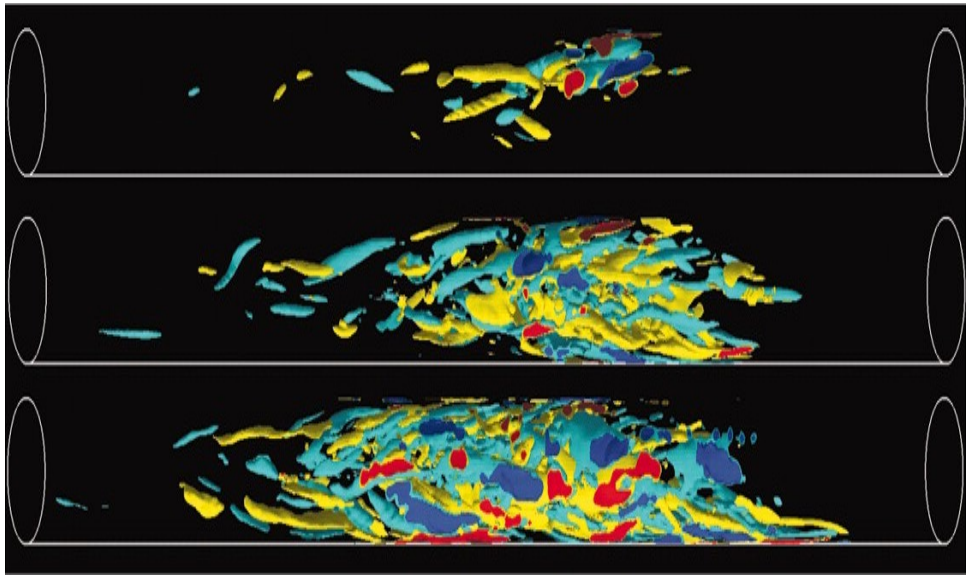
- By applying matrices term-by-term, the only solvers we need are ones for the underlying PDE. Good solvers for these are available.

Remaining problem: We do not have good preconditioners for S.

An nonlinear example

Consider the following not-quite-so-simple example:

At low velocities, fluid flow is reasonably accurately described by the time independent Navier-Stokes equations



$$
\begin{aligned}
u\cdot\nabla u-\nu\,\Delta u+\nabla p &= 0 && \text{in } \Omega\\
\nabla\cdot u &= 0 && \text{in } \Omega\\
u(x) &= g(x) && \text{on } \Gamma_1\subset\partial\Omega
\end{aligned}
$$

Observation: For certain "electro/magnetostrictive" materials, the viscosity depends on the electric/magnetic field. For example

$$\nu(x)=\nu_0+\nu_1 I(x), \qquad \text{where } I(x)=|E(x)|^2 \text{ or } I(x)=|B(x)|^2$$

Goal: Control flow properties by controlling the viscosity!

A nonlinear example

Mathematical description:

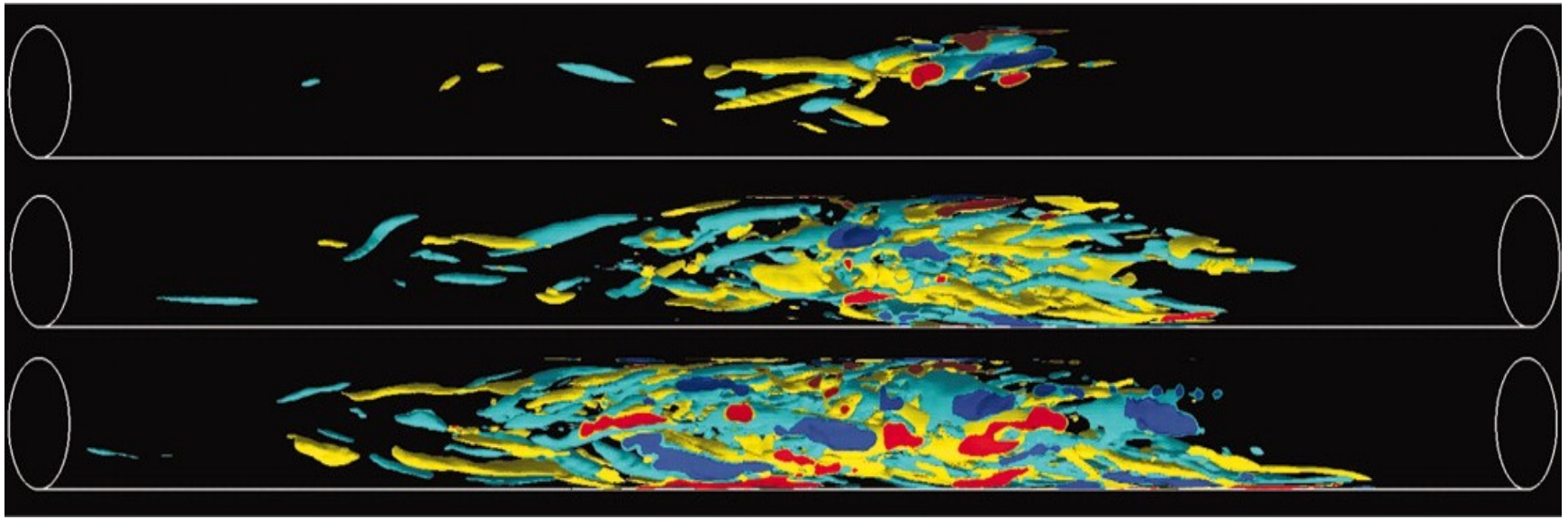
Let us assume that the goal is to control the velocity at the outflow and that we can control the intensity *I(x)* in a part $\omega \subset \Omega$ of the domain:

$$\min_{u(x)\in H_g^1(\Omega),\ p(x)\in L^2(\Omega),\ I(x)\in L^\infty(\omega)} \frac{1}{2}\|u(x)-z(x)\|_{\Gamma_2}^2$$

$$\text{subject to } u\cdot\nabla u-\nabla\cdot(\nu_0+\nu_1 I(x))\nabla u+\nabla p=0 \quad \text{in } \Omega$$

$$\nabla\cdot u=0 \quad \text{in } \Omega$$

Note: Boundary conditions for *u(x)* have been put into the function space again!



A nonlinear example

Mathematical description:

For this problem,

$$\min_{u(x)\in H_g^1(\Omega),\ p(x)\in L^2(\Omega),\ I(x)\in L^\infty(\omega)} \frac{1}{2}\|u(x)-z(x)\|_{\Gamma_2}^2$$

$$\text{subject to}\quad u\cdot\nabla u-\nabla\cdot(\nu_0+\nu_1 I(x))\nabla u+\nabla p=0 \quad \text{in } \Omega$$

$$\nabla\cdot u=0 \quad \text{in } \Omega$$

the Lagrangian would be chosen as

$$\begin{aligned} L(u(x),p(x),I(x),\lambda_u(x),\lambda_p(x)) &= \frac{1}{2}\|u(x)-z(x)\|_{\Gamma_2}^2 \\ &\quad -(u\cdot\nabla u,\lambda_u)-((\nu_0+\nu_1 I(x))\nabla u,\nabla\lambda_u)-(p,\nabla\cdot\lambda_u) \\ &\quad -(\nabla\cdot u,\lambda_p) \end{aligned}$$

A nonlinear example

Mathematical description: From the Lagrangian

$$L(u(x), p(x), I(x), \lambda_u(x), \lambda_p(x))$$
$$= \frac{1}{2}\|u(x) - z(x)\|^2_{\Gamma_2}$$
$$-(u \cdot \nabla u, \lambda_u) - ((\nu_0 + \nu_1 I(x))\nabla u, \nabla \lambda_u) - (p, \nabla \cdot \lambda_u) - (\nabla \cdot u, \lambda_p)$$

we get the optimality conditions. Taking derivatives with respect to the dual (adjoint) variables $\lambda_u(x)$, $\lambda_p(x)$ yields

$$u \cdot \nabla u - \nabla \cdot (\nu_0 + \nu_1 I(x))\nabla u + \nabla p = 0$$
$$\nabla \cdot u = 0$$

A nonlinear example

Mathematical description: From the Lagrangian

$$L(u(x), p(x), I(x), \lambda_u(x), \lambda_p(x))$$
$$= \frac{1}{2}\|u(x) - z(x)\|^2_{\Gamma_2}$$
$$-(u \cdot \nabla u, \lambda_u) - ((\nu_0 + \nu_1 I(x))\nabla u, \nabla \lambda_u) - (p, \nabla \cdot \lambda_u) - (\nabla \cdot u, \lambda_p)$$

we get the optimality conditions. Taking derivatives with respect to the primal (state) variables *u(x), p(x)* yields

$$(\nabla u)^T \lambda_u - \nabla \cdot (u \otimes \lambda_u) - \nabla \cdot (\nu_0 + \nu_1 I(x))\nabla \lambda_u - \nabla \lambda_p = (u - z)\,\delta_{\Gamma_2}(x)$$
$$\nabla \cdot \lambda_u = 0$$

Using the *u(x)* is divergence free, we can rewrite this as

$$(-u) \cdot \nabla \lambda_u - \nabla \cdot (\nu_0 + \nu_1 I(x))\nabla \lambda_u + (\nabla u)^T \lambda_u - \nabla \lambda_p = (u - z)\,\delta_{\Gamma_2}(x)$$
$$\nabla \cdot \lambda_u = 0$$

A nonlinear example

Mathematical description: From the Lagrangian

$$L(u(x),p(x),I(x),\lambda_u(x),\lambda_p(x))$$
$$= \frac{1}{2}\|u(x)-z(x)\|^2_{\Gamma_2}$$
$$-(u\cdot\nabla u,\lambda_u)-((\nu_0+\nu_1 I(x))\nabla u,\nabla\lambda_u)-(p,\nabla\cdot\lambda_u)-(\nabla\cdot u,\lambda_p)$$

we get the optimality conditions. Taking derivatives with respect to the control variables yields the control equation:

$$\int_\omega \nabla u\cdot\nabla\lambda_u\rho = 0 \qquad \forall\rho(x)\in L^\infty$$

A nonlinear example

The optimality conditions all together are now:

$$(-u)\cdot\nabla\lambda_u-\nabla\cdot(\nu_0+\nu_1 I(x))\nabla\lambda_u+(\nabla u)^T\lambda_u-\nabla\lambda_p=(u-z)\delta_{\Gamma_2}(x)$$
$$\nabla\cdot\lambda_u=0$$

$$\nabla u\cdot\nabla\lambda_u|_\omega=0$$

$$u\cdot\nabla u-\nabla\cdot(\nu_0+\nu_1 I(x))\nabla u+\nabla p=0$$
$$\nabla\cdot u=0$$

Note 1: The advection term in the state equation transports momentum *along* with the velocity *u(x)*. On the other hand, advection in the *adjoint* equation goes in the *opposite direction, -u(x)*!

This is just like the *time* direction for ODEs!

Note 2: This is still a nonlinear system of coupled PDEs.

A nonlinear example

Solving the optimality conditions:

The optimality conditions form a set of 5 partial differential equations that we can write as follows:

$$F(\{u, p, I, \lambda_u, \lambda_p\}) = 0$$

More specifically:

$$F(\{u, p, I, \lambda_u, \lambda_p\}) = \begin{pmatrix} L_u(\{u, p, I, \lambda_u, \lambda_p\}) \\ L_p(\{u, p, I, \lambda_u, \lambda_p\}) \\ L_I(\{u, p, I, \lambda_u, \lambda_p\}) \\ L_{\lambda_u}(\{u, p, I, \lambda_u, \lambda_p\}) \\ L_{\lambda_p}(\{u, p, I, \lambda_u, \lambda_p\}) \end{pmatrix}$$

A nonlinear example

Solving the optimality conditions:

The optimality conditions form a set of 5 partial differential equations that we can write as follows:

$$F(\{u, p, I, \lambda_u, \lambda_p\}) = 0$$

We can apply Newton's method to solve this system:

$$F'(\{u^{(k)}, p^{(k)}, I^{(k)}, \lambda_u^{(k)}, \lambda_p^{(k)}\}) \begin{Vmatrix} \delta u^{(k)} \\ \delta p^{(k)} \\ \delta I^{(k)} \\ \delta \lambda_u^{(k)} \\ \delta \lambda_p^{(k)} \end{Vmatrix} = -F(\{u^{(k)}, p^{(k)}, I^{(k)}, \lambda_u^{(k)}, \lambda_p^{(k)}\})$$

Note: Because $F=L'$, the matrix operator $F'=L''$ is symmetric.

A nonlinear example

The Newton system that we need to solve in each step has the form:

$$
\begin{aligned}
&(-u^{(k)})\cdot\nabla\,\delta\lambda_u^{(k)}+(-\delta u^{(k)})\cdot\nabla\,\lambda_u^{(k)}\\
&\qquad -\nabla\cdot(\nu_1\,\delta I^{(k)})\nabla\,\lambda_u^{(k)}-\nabla\cdot(\nu_0+\nu_1 I^{(k)})\nabla\,\delta\lambda_u^{(k)}\\
&\qquad +(\nabla u^{(k)})^T\,\delta\lambda_u^{(k)}+(\nabla\,\delta u^{(k)})^T\,\lambda_u^{(k)}-\nabla\,\delta\lambda_p^{(k)}=r_{\lambda_u}^{(k)}\\
&\nabla\cdot\delta\lambda_u^{(k)}=-\nabla\cdot\lambda_u^{(k)}
\end{aligned}
$$

$$
\nabla u^{(k)}\cdot\nabla\,\delta\lambda_u^{(k)}|_\omega+\nabla\,\delta u^{(k)}\cdot\nabla\,\lambda_u^{(k)}|_\omega=-\nabla u^{(k)}\cdot\nabla\,\lambda_u^{(k)}|_\omega
$$

$$
\begin{aligned}
&u^{(k)}\cdot\nabla\,\delta u^{(k)}+\delta u^{(k)}\cdot\nabla u^{(k)}\\
&\qquad -\nabla\cdot(\nu_1\,\delta I^{(k)})\nabla u^{(k)}-\nabla\cdot(\nu_0+\nu_1 I^{(k)})\nabla\,\delta u^{(k)}+\nabla\,\delta\,p^{(k)}=r_u^{(k)}\\
&\nabla\cdot\delta u^{(k)}=-\nabla\cdot u^{(k)}
\end{aligned}
$$

Note: This nasty system really is linear in the updates and symmetric. Trust me! ☺

Solving PDE-constrained problems

Practical aspects of PDE-constrained problems:

- They are *very* expensive
  - the number of variables is often very large
  - the degree of coupling is often much higher than for optimal control problems
- PDE-constrained problems are almost always ill-posed by themselves
- Discretized PDE-constrained problems are well-posed but ill-conditioned

- Strategies to deal with many difficulties (e.g. complex state equations, solver issues, inequalities) remain the subject of current research

Part 30

Further Applications I:

Parameter Estimation

Motivation

Simulation in engineering and sciences follows the following sequence:

- We model the process, i.e. we derive equations that describe how the system behaves in response to external forces.
  The equations may be algebraic, differential, integral, partial-differential, ..., and typically contain system parameters

- Given the model and parameter values, we can numerically simulate the system on a computer

- If we know how to simulate, we can optimize

Problem: Someone will have to give us values for these system/ material parameters at one point. Determining these values is done in the field of *parameter estimation*.

If the model is a PDE, parameter estimation problems are often called *inverse problems*.

Idea

The basic idea of parameter estimation:

Let us assume that

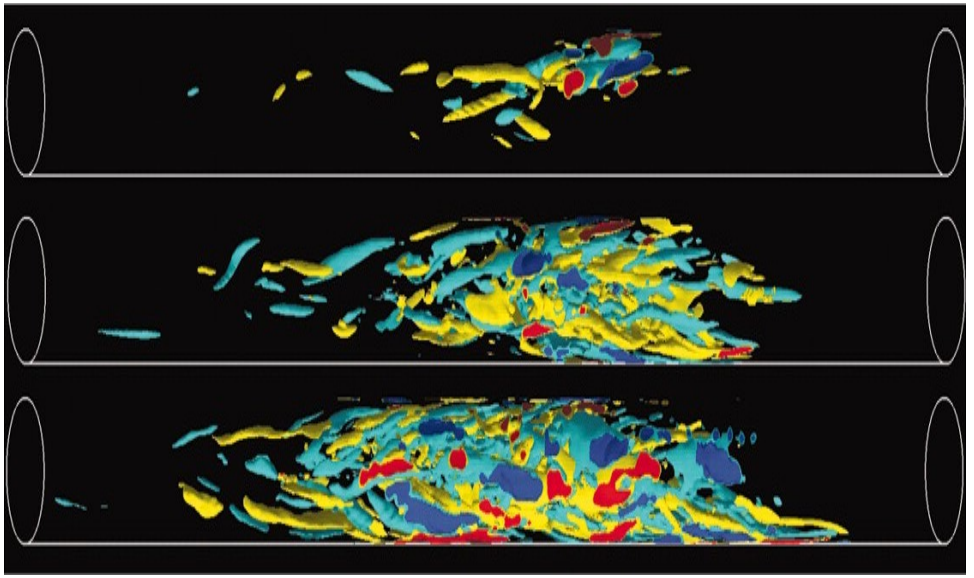
- the state of a system is described by a variable y
- there are a number of system parameters p
- we have a model that can *predict* the state y if we know p:

$$f(y;p)=0$$

- a function $h(y;p)$ that extracts from state and parameters the part we can measure

Examples of parameter estimation problems

Example 1: Determining the viscosity of a fluid.



State variable: $y=\{u(x),p(x)\}$
$\in Y=H^1\times L^2$

Parameters: $p=\{\nu\}\in P=\mathbb{R}$

Model: The Navier-Stokes equations

$$f(y;p)=\begin{pmatrix} u\cdot\nabla u-\nu\Delta u+\nabla p \\ \nabla\cdot u \\ u(x)-g(x) \end{pmatrix}=0$$

Measurements: For example velocities or pressures at individual points

$$h(y;p)=\left(u(x_1),\ldots,u(x_{N_u}),p(x_1),\ldots,p(x_{N_p})\right)^T:Y\times P\to\mathbb{R}^{dN_u+N_p}$$

Examples of parameter estimation problems

Example 2: Determining earth gravity and the friction coefficient for a body falling in air

State variable: $y=\{z(t)\}\in Y=C^1([0,T])$

Parameters: $p=\{g,C\}\in P=\mathbb{R}^2$

Model: Newton's second law with a model for air friction

$$f(y;p)=\begin{pmatrix}\ddot{x}(t)-\left(g-C\dot{x}(t)^2\right)\\ x(0)\\ \dot{x}(0)\end{pmatrix}=0$$

Measurements: The distance the body has fallen at given times

$$h(y;p)=\left(x(t_1),\ldots,x(t_N)\right)^T:Y\times P\to\mathbb{R}^N$$

Abstract formulation

Approach: Let us assume that we have measured a vector of values z. Our goal is to find a set of parameters p so that our model $f(y;p)=0$ predicts a state for which the *predicted* measurements $h(y;p)$ are as close as possible to the *actual* measurements z:

$$\begin{aligned} \min_{y,p} \quad & \frac{1}{2}\|h(y;p)-z\|^2 \\ \text{subject to} \quad & f(y;p)=0 \end{aligned}$$

This problem gives rise to the Lagrangian

$$L(y,p,\lambda)=\frac{1}{2}\|h(y;p)-z\|^2-\lambda^T f(y;p)$$

Abstract formulation

Approach: Let us assume that we have measured a vector of values z. Our goal is to find a set of parameters p so that our model $f(y;p)=0$ predicts a state for which the *predicted* measurements $h(y;p)$ are as close as possible to the *actual* measurements z.

From the Lagrangian

$$L(y,p,\lambda)=\frac{1}{2}\|h(y;p)-z\|^2-\lambda^T f(y;p)$$

we obtain the optimality conditions:

$$f(y;p)=0$$

$$\nabla_y f(y;p)^T\lambda=(h(y;p)-z)\nabla_y h(y;p)$$

$$(h(y;p)-z)\nabla_p h(y;p)-\nabla_p f(y;p)\lambda=0$$

Solving parameter estimation problems

Approach: The optimality conditions

$$
\begin{aligned}
&(h(y;p)-z)\nabla_y h(y;p)-\nabla_y f(y;p)^T\lambda=0\\
&(h(y;p)-z)\nabla_p h(y;p)-\nabla_p f(y;p)\lambda=0\\
&f(y;p)=0
\end{aligned}
$$

can be solved using a Newton method:

$$
H\begin{pmatrix}\delta y_k\\ \delta p_k\\ \delta\lambda_k\end{pmatrix}=-L'(y_k,p_k,\lambda_k)
$$

$$
H=\begin{pmatrix}
\nabla_y h^T\nabla_y h+(h-z)\nabla_y^2 h & \nabla_p h^T\nabla_y h+(h-z)\nabla_{py}^2 h-\nabla_{py}^2 f^T\lambda & \nabla_y f^T\\
\nabla_y h^T\nabla_p h+(h-z)\nabla_{yp}^2 h-\nabla_{yp}^2 f^T\lambda & \nabla_p h^T\nabla_p h+(h-z)\nabla_{pp}^2 h-\nabla_{pp}^2 f^T\lambda & \nabla_p f^T\\
\nabla_y f & \nabla_p f & 0
\end{pmatrix}
$$

Note: The Newton matrix on the left is symmetric, but indefinite.

Practical considerations 1

Remark 1:

If we can assume that the model is correct and that our measurements are reasonably accurate, then we can expect that:

- We should be able to find a parameter p so that the predicted measurements $h(y;p)$ are close to the actual measurements z
- In other words: At the solution of the parameter estimation problem,

$$h(y;p)-z$$

  is small!

Corollary: If we omit all terms from the Newton matrix that are proportional to the "misfit" $(h\text{-}z)$, we are only going to make a small error in the Newton direction.

Practical considerations 2

Remark 2: We can think of the model $f(y;p)=0$ as follows:

- For every parameter value p there is a state y that results
- In other words, for every p, $f(.;p)=0$ defines an implicit function $y=y_p$
- A slight change in parameter values results in a change in state:

$$0=f(y+\delta y;p+\delta p)\approx f(y;p)+\frac{\partial f(y;p)}{\partial y}\delta y+\frac{\partial f(y;p)}{\partial p}\delta p$$
$$=\nabla_y f(y;p)\delta y+\nabla_p f(y;p)\delta p$$

Or, written differently:

$$\nabla_y f(y;p)\delta y=-\nabla_p f(y;p)\delta p$$

Note: For the problem to be well-posed, we must have that $\nabla_y f(y;p)$ is an invertible matrix/operator.

Practical considerations 2

Consequence:

Among the optimality conditions, we had the equation

$$\nabla_y f(y;p)^T \lambda = (h(y;p) - z) \nabla_y h(y;p)$$

This implies

$$\|\lambda\| \leq \|\nabla_y f(y;p)^{-T}\| \; \|\nabla_y h(y;p)\| \; \|h(y;p) - z\|$$

Corollary 1: The Lagrange multiplier is small if we can fit data well, i.e. if $\|h(y;p) - z\|$ is small!

Corollary 2: If we omit all terms from the Newton matrix that contain the Lagrange multiplier, we are only going to make a small error in the Newton direction.

Practical considerations – conclusions

***Practical* approach:** The optimality conditions

$$
\begin{aligned}
&(h(y;p)-z)\nabla_y h(y;p)-\nabla_y f(y;p)^T\lambda = 0\\
&(h(y;p)-z)\nabla_p h(y;p)-\nabla_p f(y;p)\lambda = 0\\
&f(y;p) = 0
\end{aligned}
$$

can be solved using a variant of Newton's method called the *Gauss-Newton method*:

$$
\begin{pmatrix}
\nabla_y h^T \nabla_y h & \nabla_p h^T \nabla_y h & \nabla_y f^T \\
\nabla_y h^T \nabla_p h & \nabla_p h^T \nabla_p h & \nabla_p f^T \\
\nabla_y f & \nabla_p f & 0
\end{pmatrix}
\begin{pmatrix}
\delta y_k \\ \delta p_k \\ \delta \lambda_k
\end{pmatrix}
= -L'(y_k, p_k, \lambda_k)
$$

Note: The Newton matrix on the left is symmetric, but still indefinite.

Practical considerations – conclusions

***Practical* approach:** In many cases, our measurements do not *directly* depend on the parameters p (after all, we need to solve a parameter estimation problem to determine them since they are not directly measurable!). Then

$$\nabla_p h = 0$$

and the Gauss-Newton method reads:

$$\begin{pmatrix} \nabla_y h^T \nabla_y h & 0 & \nabla_y f^T \\ 0 & 0 & \nabla_p f^T \\ \nabla_y f & \nabla_p f & 0 \end{pmatrix} \begin{pmatrix} \delta y_k \\ \delta p_k \\ \delta \lambda_k \end{pmatrix} = -L'(y_k, p_k, \lambda_k)$$

Let us define $A = \nabla_y f,\ J = \nabla_y h,\ Q = \nabla_p f$ then

$$\begin{pmatrix} J^T J & 0 & A^T \\ 0 & 0 & Q^T \\ A & Q & 0 \end{pmatrix} \begin{pmatrix} \delta y_k \\ \delta p_k \\ \delta \lambda_k \end{pmatrix} = -\begin{pmatrix} L_y \\ L_p \\ L_\lambda \end{pmatrix}$$

Practical considerations – conclusions

***Practical* approach:** The linear system that needs to be solved in each step of the Gauss-Newton method reads:

$$\begin{pmatrix} J^T J & \cdot & A^T \\ \cdot & \cdot & Q^T \\ A & Q & \cdot \end{pmatrix} \begin{pmatrix} \delta y_k \\ \delta p_k \\ \delta \lambda_k \end{pmatrix} = - \begin{pmatrix} L_y \\ L_p \\ L_\lambda \end{pmatrix}$$

Recall: We have assumed that the model is locally invertible, i.e. A^{-1} exists.

Consequence: Using a block elimination as discussed in the section on PDE-constrained optimization, this system can be solved in three steps:

$$\underbrace{Q^T A^{-T} J^T J A^{-1} Q}_{S} \; \delta p_k = \ldots$$

$$A \; \delta y_k = \ldots$$

$$A^T \; \delta \lambda_k = \ldots$$

We know how to invert all matrices here. This system is *much* simpler than what we would have gotten for the *full* Newton method.

Part 31

Further Applications II:

Optimization Problems with Optimization Problems as Constraints – Optimal Experimental Design

The parameter estimation problem

We have formulated the parameter estimation problem as follows:
Let us assume that we have measured a vector of values z. Our goal is to find a set of parameters p so that our model $f(y;p)=0$ predicts a state for which the *predicted* measurements $h(y;p)$ are as close as possible to the *actual* measurements z:

$$\min_{y,p} \quad \frac{1}{2}\|h(y;p)-z\|^2$$
$$\text{subject to} \quad f(y;p)=0$$

Observation: For a given set of measurements z, we get a set of parameters p that solve this optimization problem.

Question: How does p depend on z? In other words, if we have a little bit of measurement noise, how does p respond?

The parameter estimation problem

Answer 1:

For a given vector z, p satisfies

$$(h(y;p)-z)\nabla_y h(y;p)-\nabla_y f(y;p)^T\lambda=0$$
$$(h(y;p)-z)\nabla_p h(y;p)-\nabla_p f(y;p)\lambda=0$$
$$f(y;p)=0$$

or in short for $x=(y,p,\lambda)$:

$$L_x(x;z)=0$$

Here, we have made the dependence of L on z explicit:

$$L(x;z)=L(y,p,\lambda;z)=\frac{1}{2}\|h(y;p)-z\|^2-\lambda^T f(y;p)$$

The parameter estimation problem

Answer 2:

For a given set of measurements $z+\delta z$, we will estimate a parameter $p+\delta p$ that satisfies

$$(h(y+\delta y; p+\delta p)-(z+\delta z))\nabla_y h(y+\delta y; p+\delta p)-\ldots=0$$
$$(h(y+\delta y; p+\delta p)-(z+\delta z))\nabla_p h(y+\delta y; p+\delta p)-\ldots=0$$
$$f(y+\delta y; p+\delta p)=0$$

or in short for $x+\delta x=(y+\delta y, p+\delta p, \lambda+\delta\lambda)$:

$$L_x(x+\delta x; z+\delta z)=0$$

The parameter estimation problem

Consequence:

Measurement error δz and resulting parameter errors δp are related to each other as

$$L_x(x+\delta x; z+\delta z)=0$$

If measurement errors are small, then we can do a Taylor expansion:

$$L_x(x;z)+L_{xx}(x;z)\delta x+L_{xz}(x;z)\delta z+\ldots=0$$

Using the optimality conditions for x and neglecting higher order terms we get:

$$L_{xx}(x;z)\delta x=-L_{xz}(x;z)\delta z$$

The parameter estimation problem

Consequence:

Measurement error δz and resulting parameter errors δp are related to each other to first order as

$$L_{xx}(x;z)\,\delta x = -L_{xz}(x;z)\,\delta z$$

The matrix on the left is again the nasty looking Newton matrix. If we make the same assumptions/approximations as before, we get:

$$\begin{pmatrix} J^T J & 0 & A^T \\ 0 & 0 & Q^T \\ A & Q & 0 \end{pmatrix} \begin{pmatrix} \delta y \\ \delta p \\ \delta \lambda \end{pmatrix} = \begin{pmatrix} J^T \delta z \\ 0 \\ 0 \end{pmatrix}$$

where

$$A = \nabla_y f,\ J = \nabla_y h,\ Q = \nabla_p f$$

The parameter estimation problem

Consequence:

Measurement error δz and resulting parameter errors δp are related to each other to first order as

$$L_{xx}(x;z)\,\delta x = -L_{xz}(x;z)\,\delta z$$

After block elimination as usual, we find that

$$Q^T A^{-T} J^T J A^{-1} Q\ \delta p \ = \ -Q^T A^{-T} J^T\ \delta z$$

Or shorter: $S\ \delta p \ = \ -X^T\ \delta z$

Note 1: $S \ = \ -X^T X$

Note 2: The matrices S, X typically depend on x.

Optimal experimental design

Motivation:

We now know that a measurement error δz results in parameter errors δp related to each other by

$$S\,\delta p \;=\; -X^T\,\delta z$$

Ideally, we would want the matrix S to be as "large" as possible so that the errors δp are as "small" as possible!

Observation: All parts of the equation depend on how we measure, e.g. where we measure, what we measure, etc.

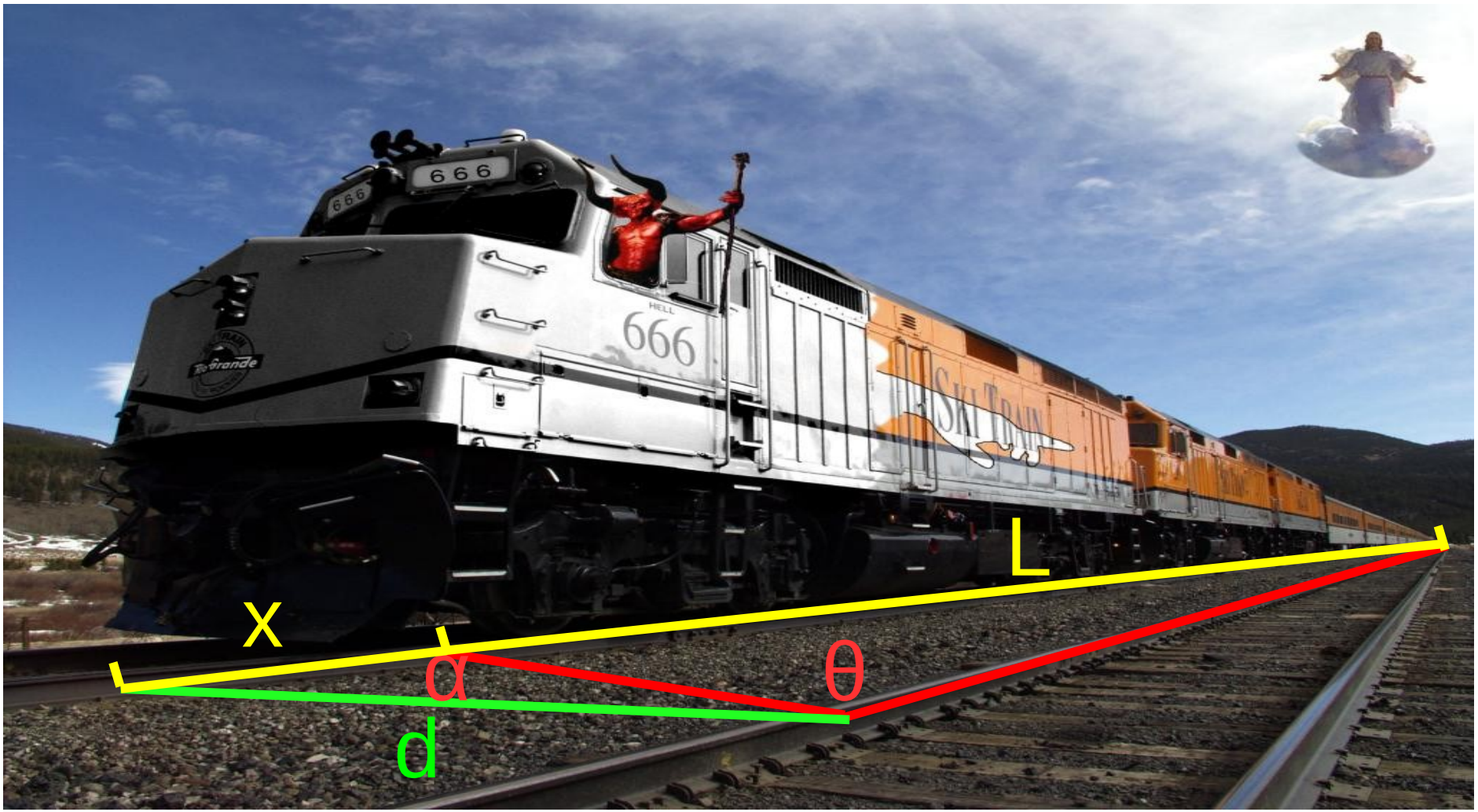
Optimal experimental design – an example

An example!

Question: How long is this train and how far is it away from the closest point to the tracks?

Optimal experimental design – an example

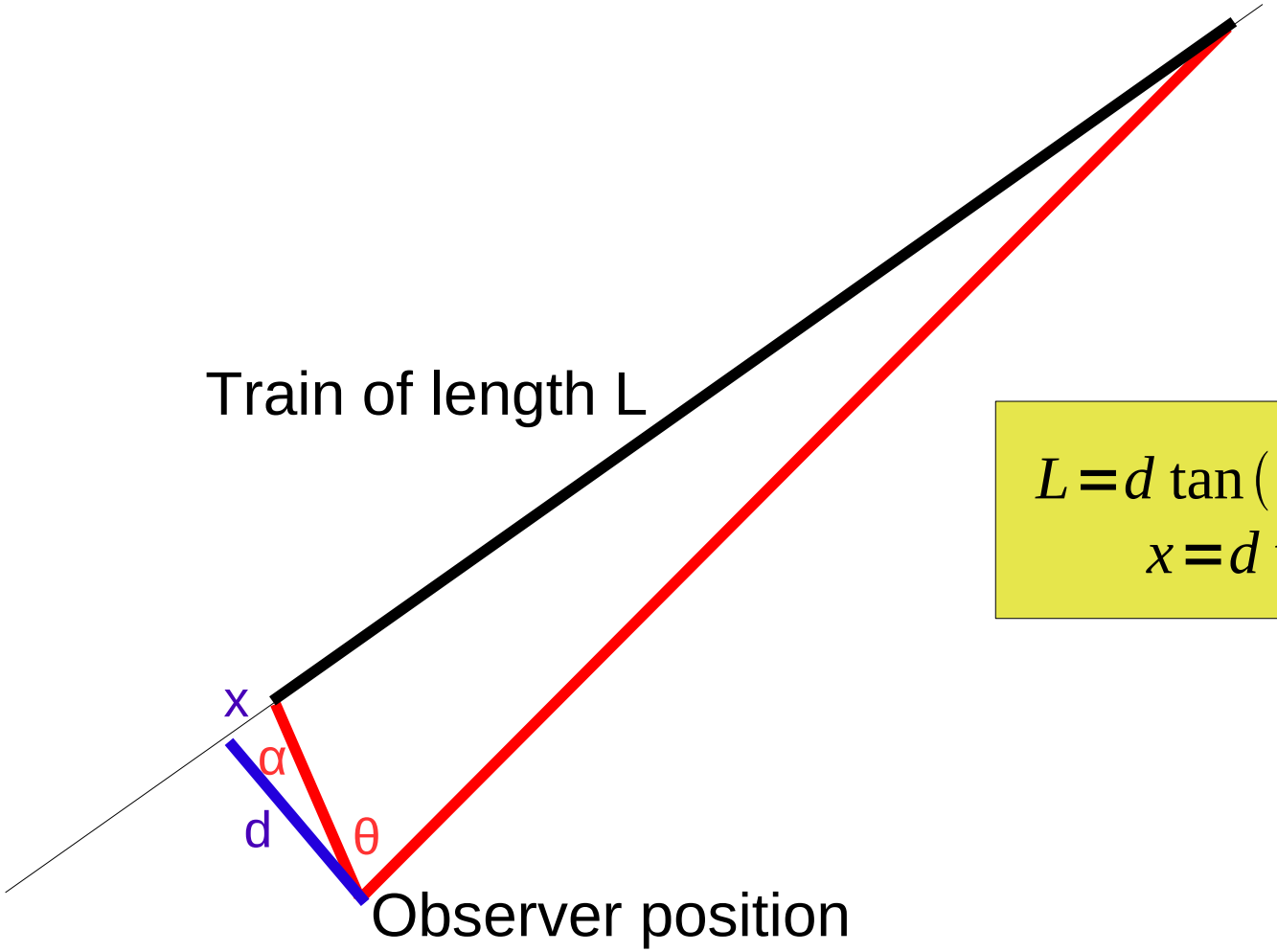
An example!



Answer: Measure the distance to the tracks and the angles!

Optimal experimental design – an example

View from the top:



$$L = d\tan(\alpha+\theta) - x$$
$$x = d\tan\alpha$$

Answer: Measure the distance to the tracks and the angles!

Optimal experimental design – an example

Given a distance d to the tracks, we have the following:

- State and parameter variables are

$$y=\begin{pmatrix}\alpha\\ \theta\end{pmatrix}, \qquad p=\begin{pmatrix}L\\ x\end{pmatrix},$$

- The state equation is given by

$$f(y;p)=\begin{pmatrix}\alpha\\ \theta\end{pmatrix}-\begin{pmatrix}\arctan\left(\frac{x}{d}\right)\\ \arctan\frac{(L+x)}{d}-\arctan\left(\frac{x}{d}\right)\end{pmatrix}=0$$

- The measurement operator is given by

$$h(y;p)=\begin{pmatrix}\alpha\\ \theta\end{pmatrix}$$

Note: The distance d to the tracks is the *design parameter*. We can choose it as we see fit!

Optimal experimental design – an example

After some algebra, we find:

Measurement errors and resulting errors in our parameter estimates are related by

$$S\begin{bmatrix}\delta L\\ \delta x\end{bmatrix}=X\begin{bmatrix}\delta z_{\alpha}\\ \delta z_{\theta}\end{bmatrix}$$

where

$$S=\begin{bmatrix}\dfrac{1}{d^{2}\left(1+\dfrac{(L+x)^{2}}{d^{2}}\right)^{2}} & \dfrac{1}{d^{2}\left(1+\dfrac{(L+x)^{2}}{d^{2}}\right)^{2}}+\dfrac{1}{d^{2}\left(1+\dfrac{(L+x)^{2}}{d^{2}}\right)\left(1+\dfrac{x^{2}}{d^{2}}\right)}\\ \dfrac{1}{d^{2}\left(1+\dfrac{(L+x)^{2}}{d^{2}}\right)^{2}}+\dfrac{1}{d^{2}\left(1+\dfrac{(L+x)^{2}}{d^{2}}\right)\left(1+\dfrac{x^{2}}{d^{2}}\right)} & \dfrac{1}{d^{2}\left(1+\dfrac{x^{2}}{d^{2}}\right)^{2}}+\left(\dfrac{1}{d\left(1+\dfrac{(L+x)^{2}}{d^{2}}\right)}+\dfrac{1}{d\left(1+\dfrac{x^{2}}{d^{2}}\right)}\right)^{2}\end{bmatrix}$$

Optimal experimental design – an example

After some algebra, we find:

Measurement errors and resulting errors in our parameter estimates are related by

$$S\begin{bmatrix}\delta L\\ \delta x\end{bmatrix} = X\begin{bmatrix}\delta z_\alpha\\ \delta z_\theta\end{bmatrix}$$

Note: $S = S(d), \quad \delta p = \begin{bmatrix}\delta L\\ \delta x\end{bmatrix} = \delta p(d), \quad X = X(d)$ as well as $p = p(d)$

Goal: Given measurement error, we want the smallest possible error in parameter estimates. In other words, we have to maximize *S*!

Approach:

- We have to define what it means to make *S* "large"
- We can then choose *d* so as to maximize *S*.

Approach

Definition:

We can now define the optimal experimental design problem as follows for a scalar function $\phi(S): \mathbb{R}^{n_p \times n_p} \to \mathbb{R}$:

$$\max_d \quad \phi(S(y, p; d))$$

where y, q solves the parameter estimation problem:

$$\min_{y,p,\lambda} \quad \frac{1}{2}\|h(y; p; d) - z\|^2$$

$$\text{subject to } f(y; p; d) = 0$$

In other words: We need to solve an optimization problem with a constraint that one of the variables solves another optimization problem!

Formulation

Mathematical formulation:

We can now define the optimal experimental design problem as follows for a scalar function $\phi(S): \mathbb{R}^{n_p \times n_p} \rightarrow \mathbb{R}$:

$$
\begin{aligned}
\max_{d,y,p,\lambda} \quad & \phi(S) \\
\text{subject to} \quad & (h(y;p)-z)\nabla_y h(y;p;d) - \nabla_y f(y;p;d)^T \lambda = 0 \\
& (h(y;p)-z)\nabla_p h(y;p;d) - \nabla_p f(y;p;d)^T \lambda = 0 \\
& f(y;p;d) = 0
\end{aligned}
$$

$$
\begin{aligned}
S &= Q^T A^{-T} J^T J A^{-1} Q \\
A &= \nabla_y f(y;p;d) \\
J &= \nabla_y h(y;p;d) \\
Q &= \nabla_p f(y;p;d)
\end{aligned}
$$

second order optimality conditions are satisfied

Formulation

Mathematical formulation:

Alternatively, we can resolve a few of the constraints already:

$$
\begin{aligned}
\max_{d,y,q,\lambda} \quad & \phi\left([\nabla_p f]^T [\nabla_y f]^{-T} [\nabla_y h]^T [\nabla_y h][\nabla_y f]^{-1}[\nabla_p f]\right) \\
\text{subject to} \quad & (h(y;p)-z)\nabla_y h(y;p;d)-\nabla_y f(y;p;d)^T \lambda = 0 \\
& (h(y;p)-z)\nabla_p h(y;p;d)-\nabla_p f(y;p;d)^T \lambda = 0 \\
& f(y;p;d)=0
\end{aligned}
$$

Note 1: The first three constraints are the first order necessary conditions of the parameter estimation problem.

Note 2: The second order necessary condition could be added by requiring that the projected Hessian be positive semidefinite.

Solution

Solution:

We can solve the optimal experimental design problem

$$\begin{aligned}
\max_{d,y,p,\lambda} \quad & \phi([\nabla_p f]^T[\nabla_y f]^{-T}[\nabla_y h]^T[\nabla_y h][\nabla_y f]^{-1}[\nabla_p f]) \\
\text{subject to} \quad & (h(y;p)-z)\nabla_y h(y;p;d)-\nabla_y f(y;p;d)^T\lambda=0 \\
& (h(y;p)-z)\nabla_p h(y;p;d)-\nabla_p f(y;p;d)^T\lambda=0 \\
& f(y;p;d)=0
\end{aligned}$$

$$[\nabla_p f]^T[\nabla_y f]^{-T}[\nabla_y h]^T[\nabla_y h][\nabla_y f]^{-1}[\nabla_p f] \geqslant 0$$

by introducing a Lagrangian

$$\begin{aligned}
& L(d,y,p,\lambda,\mu_1,\mu_2,\mu_3) \\
= \; & \phi([\nabla_p f]^T[\nabla_y f]^{-T}[\nabla_y h]^T[\nabla_y h][\nabla_y f]^{-1}[\nabla_p f]) \\
& -\mu_1^T\left[(h(y;p)-z)\nabla_y h(y;p;d)-\nabla_y f(y;p;d)^T\lambda\right] \\
& -\mu_2^T\left[(h(y;p)-z)\nabla_p h(y;p;d)-\nabla_p f(y;p;d)^T\lambda\right]-\mu_3^T[f(y;p;d)]
\end{aligned}$$

and then proceeding as usual.

Choices for the objective function

The optimal experimental design problem was defined as follows:

$$\max_d \quad \phi(S(y,p;d))$$

where y, p solves the parameter estimation problem:

$$\min_{y,p,\lambda} \quad \frac{1}{2}\|h(y;p;d)-z\|^2$$

$$\text{subject to } f(y;p;d)=0$$

In other words: Make the matrix S as large as possible so that for a given measurement error δz the resulting parameter errors δp are as small as possible.

Question: How should we choose the function $\phi(S):\mathbb{R}^{n_p\times n_p}\to\mathbb{R}$?

Choices for the objective function

Question: How should we choose the function $\phi(S): \mathbb{R}^{n_p \times n_p} \to \mathbb{R}$?

Choices: Different objective functions are typically denotes by letters in optimal experimental design:

- D-criterion: $\phi(S) = \det S$
- T-criterion: $\phi(S) = \text{trace } S$
- A-criterion: $\phi(S) = -\text{trace } S^{-1}$
- E-criterion: $\phi(S) = \min \text{ eigenvalue(S)}$
- G-criterion, C-criterion, V-criterion, I-criterion, ...

Part 32

Further Applications III:

Optimization Problems with Optimization Problems as Constraints – Multiobjective Optimization

Motivation

Problem: Oftentimes we have several objectives. For example:

- We want to design a plane that is as fuel efficient as possible but also as stable as possible.

- We want to find a trajectory from Earth to Pluto that is as cheap as possible (i.e. we need to lift the least amount of fuel into space) but also takes the least amount of time to get there.

Approach 1: If we have a hard limit on one or the other goal, we can formulate it as a constraint.

Approach 2: We can choose an objective function that is a linear combination of our goals.

Approach 3: We can state limits on how much we want to compromise on one goal for another.

Multiobjective optimization

Example: Let the problem be to find a control q so that

$$F_1(y,q) \rightarrow \min$$
$$F_2(y,q) \rightarrow \min$$

subject to constraints

$$f(y,q) = 0$$
$$g(y,q) \geq 0$$

Approach 1: If we have a hard limit on one or the other goal, we can formulate it as a constraint. For example, F_1 is more important to us, and we are willing to accept any control q so that $F_2(y,q) \leq F^*$. Then:

$$\begin{aligned} \min_{y,q} \quad & F_1(y,q) \\ \text{subject to} \quad & f(y,q) = 0 \\ & g(y,q) \geq 0 \\ & F_2(y,q) \leq F^* \end{aligned}$$

Multiobjective optimization

Approach 2: We can choose an objective function that is a linear combination of our goals. For example:

$$\begin{aligned} \min_{y,q} \quad & s\,F_1(y,q)+(1-s)\,F_2(y,q) \\ \text{subject to} \quad & f(y,q)=0 \\ & g(y,q)\geq 0 \end{aligned}$$

Multiobjective optimization

Approach 3: We can state limits on how much we want to compromise on one goal for another.

Example: We want to find a control q so that F_2 is minimized, but we are only willing to compromise at most 10% of goal F_1.

Let $\hat{y}, \hat{q}$ be the optimizer of F_1:

$$\begin{aligned} \min_{\hat{y},\hat{q}} \quad & F_1(\hat{y},\hat{q}) \\ \text{subject to} \quad & f(\hat{y},\hat{q}) = 0 \\ & g(\hat{y},\hat{q}) \geq 0 \end{aligned}$$

Then find y, q so that

$$\begin{aligned} \min_{y,q} \quad & F_2(y,q) \\ \text{subject to} \quad & f(y,q) = 0 \\ & g(y,q) \geq 0 \\ & F_1(y,q) \leq 1.1\, F_1(\hat{y},\hat{q}) \end{aligned}$$

Multiobjective optimization

Joint formulation:

We can write this problem as one optimization problem. In words, it then reads as follows:

$$
\begin{aligned}
\min_{y,q,\hat{y},\hat{q}} \quad & F_2(y,q) \\
\text{subject to} \quad & f(y,q)=0 \\
& g(y,q)\geq 0 \\
& F_1(y,q)\leq 1.1\,F_1(\hat{y},\hat{q}) \\
& \text{where } \hat{y},\hat{q} \text{ solves} \quad \min_{\hat{y},\hat{q}} \quad F_1(\hat{y},\hat{q}) \\
& \qquad\qquad\qquad\qquad\qquad\qquad \text{subject to} \quad f(\hat{y},\hat{q})=0 \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad g(\hat{y},\hat{q})\geq 0
\end{aligned}
$$

Multiobjective optimization

Joint formulation:

We can write this problem as one optimization problem.

Mathematically, we can write it using the optimality conditions:

$$
\begin{aligned}
\min_{y,q,\hat{y},\hat{q},\hat{\lambda},\hat{\mu}} \quad & F_2(y,q) \\
\text{subject to} \quad & f(y,q)=0 \\
& g(y,q)\geq 0
\end{aligned}
$$

$$
\begin{aligned}
& \nabla_y F_1(\hat{y},\hat{q}) - \hat{\lambda}^T \nabla_y f(\hat{y},\hat{q}) - \hat{\mu}^T \nabla_y g(\hat{y},\hat{q}) = 0 \\
& \nabla_q F_1(\hat{y},\hat{q}) - \hat{\lambda}^T \nabla_q f(\hat{y},\hat{q}) - \hat{\mu}^T \nabla_q g(\hat{y},\hat{q}) = 0 \\
& f(\hat{y},\hat{q})\geq 0 \\
& g(\hat{y},\hat{q})\geq 0
\end{aligned}
$$

$$
F_1(y,q)\leq 1.1\,F_1(\hat{y},\hat{q})
$$

Part 33

Further Applications IV:

Stochastic and Robust Optimization

Motivation

Problem: In many optimization applications, we have material/system parameters that are not known exactly, or external forces that are unpredictable. We want to take this into account when optimizing.

Example 1: We want to design an air plane that is as efficient as possible, but we only know that the viscosity of air at 10km altitude is

$$1.1 \cdot 10^{-4} \frac{\text{ft}^2}{\text{s}} \leq \nu \leq 1.2 \cdot 10^{-4} \frac{\text{ft}^2}{\text{s}}$$

depending on the prevailing temperature.

Example 2: We want to compute the trajectory for a rocket that takes the least amount of fuel. But this trajectory will depend on current wind conditions which we don't know exactly.

Example: We want to optimally produce an oil field but we have only incomplete knowledge of the physical structure of the oil reservoir.

Some preliminaries

Definition: Let p be an uncertain parameter (such as the viscosity, the wind field, the oil reservoir) and let

$$P(p)$$

be a probability density for this parameter.

Let $F(p)$ be a function of p. Then we call

$$E[F] = \int F(p)P(p)\,dp$$

the expectation value of F and

$$\sigma[F] = \sqrt{\int (F(p)-E[F])^2 P(p)\,dp}$$

the standard deviation of F under P.

Some preliminaries

Example 1 (Viscosity): If we know that

$$1.1\cdot 10^{-4}\frac{\text{ft}^2}{\text{s}} \le \nu \le 1.2\cdot 10^{-4}\frac{\text{ft}^2}{\text{s}}$$

then a reasonable choice would be

$$P(\nu) = \begin{cases} 10^5 \dfrac{\text{s}}{\text{ft}^2} & \text{if } 1.1\cdot 10^{-4}\dfrac{\text{ft}^2}{\text{s}} \le \nu \le 1.2\cdot 10^{-4}\dfrac{\text{ft}^2}{\text{s}} \\ 0 & \text{otherwise} \end{cases}$$

Example 2 (Rocket in a wind field): Close to the surface, a reasonable model could be

$$P(\vec{v}) = \frac{1}{C} e^{-\frac{\|\vec{v}\|^2}{(10\,mph)^2}}$$

Some preliminaries

Practical approach: In practice, computing integrals like

$$E[F] = \int F(p)P(p)\,dp$$

is difficult if the number of variables in p is large.

In that case, we can choose a uniformly distributed sample $\{p_i\}$ and approximate

$$E[F] \approx \frac{1}{N}\sum_{i=1}^{N} F(p_i)P(p_i)$$

Alternatively, we can choose samples $\{p_i\}$ based on the probability distribution $P(p)$ and approximate

$$E[F] \approx \frac{1}{N}\sum_{i=1}^{N} F(p_i)$$

Stochastic optimization

The state equation: Let us assume that state variables y, control variables q and system parameters p are related by

$$f(y,q;p)=0$$

and that we have additional constraints of the form

$$g(y,q)\geq 0$$

Deterministic optimization: If we knew that the parameters p then the optimization problem would have the form

$$\begin{aligned} \min_{y,q} \quad & F(y,q) \\ \text{subject to} \quad & f(y,q;p)=0 \\ & g(y,q)\geq 0 \end{aligned}$$

Since we have assumed that we know p this problem can be deterministically solved to find an optimal control q.

Stochastic optimization

Stochastic optimization: In reality, me may not know p exactly but only a probability distribution function $P(p)$. In this case, we can pose several versions of a stochastic problem.

Version 1: Average control – optimize the *average* cost over all possible values p by finding a single control q so that:

$$\begin{aligned} \min_{y_p, q} \quad & E\left[F(y_p, q)\right] \\ \text{subject to} \quad & f(y_p, q; p) = 0 \\ & g(y_p, q) \geq 0 \end{aligned}$$

Practical implementation: Draw samples $\{p_i\}$ from $P(p)$ and solve

$$\begin{aligned} \min_{y_i, q} \quad & \frac{1}{N}\sum_{i=1}^{N} F(y_i, q_i) \\ \text{subject to} \quad & f(y_i, q; p_i) = 0 \qquad i = 1, \ldots, N \\ & g(y_i, q) \geq 0 \qquad i = 1, \ldots, N \end{aligned}$$

Stochastic optimization

Stochastic optimization: In reality, me may not know p exactly but only a probability distribution function $P(p)$. In this case, we can pose several versions of a stochastic problem.

Version 2a: *Risk averse* control – optimize *average cost plus some safety factor* over all possible values p:

$$\begin{aligned} \min_{y_p,q} \quad & E[F(y_p,q)]+\alpha\sigma[F(y_p,q)] \\ \text{subject to} \quad & f(y_p,q;p)=0 \\ & g(y_p,q)\geq 0 \end{aligned}$$

Practical implementation: Draw samples $\{p_i\}$ from $P(p)$ and solve

$$\begin{aligned} \min_{y_i,q} \quad & \frac{1}{N}\sum_{i=1}^{N} F(y_i,q_i)+\frac{\alpha}{\sqrt{N}}\left\{\sum_{i=1}^{N}\left[F(y_i,q_i)-\frac{1}{N}\sum_{j=1}^{N}F(y_j,q_j)\right]^2\right\}^{\frac{1}{2}} \\ \text{subject to} \quad & f(y_i,q_i;p_i)=0 \qquad i=1,\ldots,N \\ & g(y_i,q_i)\geq 0 \qquad i=1,\ldots,N \end{aligned}$$

Stochastic optimization

Stochastic optimization: In reality, me may not know p exactly but only a probability distribution function $P(p)$. In this case, we can pose several versions of a stochastic problem.

Version 2b: *Risky* control – optimize average cost *minus* some safety factor over all possible values p:

$$\begin{aligned} \min_{y_p,q} \quad & E[F(y_p,q)] - \alpha\,\sigma[F(y_p,q)] \\ \text{subject to} \quad & f(y_p,q;p)=0 \\ & g(y_p,q)\geq 0 \end{aligned}$$

Practical implementation: Draw samples $\{p_i\}$ from $P(p)$ and solve

$$\begin{aligned} \min_{y_i,q} \quad & \frac{1}{N}\sum_{i=1}^{N} F(y_i,q_i) - \frac{\alpha}{\sqrt{N}}\left\{\sum_{i=1}^{N}\left[F(y_i,q_i)-\frac{1}{N}\sum_{j=1}^{N}F(y_j,q_j)\right]^2\right\}^{\frac{1}{2}} \\ \text{subject to} \quad & f(y_i,q_i;p_i)=0 \qquad i=1,\ldots,N \\ & g(y_i,q_i)\geq 0 \qquad i=1,\ldots,N \end{aligned}$$

Stochastic optimization

Stochastic optimization: In reality, me may not know p exactly but only a probability distribution function $P(p)$. In this case, we can pose several versions of a stochastic problem.

Version 3: *Robust* control – optimize the *worst case* cost over all possible values p:

$$\begin{aligned} \min_{y_p,q} \quad & \max_{p,P(p)>0} F(y_p,q) \\ \text{subject to} \quad & f(y_p,q;p)=0 \\ & g(y_p,q)\geq 0 \end{aligned}$$

Practical implementation: Draw samples $\{p_i\}$ from $P(p)$ and solve

$$\begin{aligned} \min_{y_i,q} \quad & \max_{1\leq i\leq N} F(y_i,q_i) \\ \text{subject to} \quad & f(y_i,q_i;p_i)=0 \qquad i=1,\ldots,N \\ & g(y_i,q_i)\geq 0 \qquad i=1,\ldots,N \end{aligned}$$

Practical aspects

Which formulation to choose for stochastic optimization:

- If no recourse is possible if the "real" parameters happen to be unfavorable, then optimization must be *robust*.

  - Airfoils must be designed for the least favorable viscosity
  - Available rocket fuel must be designed for the worst possible winds

- If losses are harder to tolerate than wins, then be *risk averse*:

  - Investment strategies for retirement funds

- If we can mitigate unfavorable parameters, then we can choose the *risky* strategy ("gambling"):

  - If we are Warren Buffett
  - Production strategies for oil fields where in the worst case another hole can be drilled

Practical aspects

Stochastic optimization is typically very expensive:

- Integrals can rarely be computed analytically
- Sample sets $\{p_i\}$ must be large enough to provide a good approximation of the integrals
- If the deterministic problem has M constraints of the form

$$f(y,q;p)=0$$
$$g(y,q)\geq 0$$

then the stochastic implementation has MN constraints:

$$f(y_i,q_i;p_i)=0 \qquad i=1,\ldots,N$$
$$g(y_i,q_i)\geq 0 \qquad i=1,\ldots,N$$

- Current research topics therefore are:
  - efficient sample generation
  - model reduction techniques
  - parametric descriptions of constraints (e.g. polynomial chaos)