Chapter 4: computation, complexity, conic programming

EDOUARD PAUWELS

Statistics and optimization in high dimensions

M2RI, Toulouse 3 Paul Sabatier

Different estimators

$\mathbb{X} \in \mathbb{R}^{n \times d}$, $Y \in \mathbb{R}^n$ (random).

$$
\begin{aligned}
&\hat{\theta} \in \arg\min_{\theta \in \mathbb{R}^d} \|\mathbb{X}\theta - Y\|_2^2 \\
&\hat{\theta} \in \arg\min_{\theta \in \mathbb{R}^d} \|\mathbb{X}\theta - Y\|_2^2, \qquad \text{s.t.} \qquad \|\theta\|_1 \leq 1 \\
&\hat{\theta} \in \arg\min_{\theta \in \mathbb{R}^d} \|\mathbb{X}\theta - Y\|_2^2, \qquad \text{s.t.} \qquad \|\theta\|_0 \leq k \\
&\hat{\theta} \in \arg\min_{\theta \in \mathbb{R}^d} \|\mathbb{X}\theta - Y\|_2^2 + \lambda \|\theta\|_0 \\
&\hat{\theta} \in \arg\min_{\theta \in \mathbb{R}^d} \|\mathbb{X}\theta - Y\|_2^2 + \lambda \|\theta\|_1 \\
&\hat{\theta} \in \arg\min_{\theta \in \mathbb{R}^d} \|\theta\|_1, \qquad \text{s.t.} \qquad \mathbb{X}\theta = Y.
\end{aligned}
$$

- The first one can be computed in $O(n^2 d)$ operations.
- How about the other ones?
- How to deal with large values of n and d.

What does it mean to compute?

Estimators involving the ℓ_1 norm can be computed in polynomial time. Computing estimators involving the ℓ_0 norm is NP-hard.

Plan for today: meaning of this.

- Basics of computational complexity theory.
- Recap on convex geometry in finite dimension.
- Conic programming and interior point methods.

(Partial) picture of theory and practice of numerical computing in the late 90's.

Source, mostly Ben-Tal and Nemirowski "Modern Convex Opitmization" 2001 and Alexander Schrijver "Theory of linear and integer programming" 1986.

Plan

1. Computational complexity
2. Recap on convexity
3. Conic programming
4. Computational complexity of our statistical estimators

Alphabet, words, size

Alphabet: a finite set Σ (usually $\Sigma = \{0, 1\}$), which elements are called "letters".

Words: ordered finite sequence of elements in Σ. The set of words is denoted by Σ^*.

Size: for a word (or a string), the number of its components. The zero length word is the empty word $\emptyset$.

Example: binary encoding of natural numbers, if $\alpha = p/q$ (where p and q are relatively prime integers), $c = (c_1, \ldots, c_n)$ is a rational vector and $A = (a_{ij})_{i=1\ldots m, j=1\ldots n}$ a rational matrix, we have

$$\mathrm{size}(\alpha) = 1 + \lceil \log_2(p) \rceil + \lceil \log_2(q) \rceil$$

$$\mathrm{size}(c) = n + \sum_{i=1}^{n} \mathrm{size}(c_i)$$

$$\mathrm{size}(A) = nm + \sum_{i=1}^{m} \sum_{j=1}^{n} \mathrm{size}(a_{ij})$$

Size of linear inequalities, or equalities are defined in a similar way. We essentially ignore multiplicative constants.

Problems

A (search) problem: a subset $\Pi \subset \Sigma^* \times \Sigma^*$, meta-mathematical problem:

Given $z \in \Sigma^*$, find $y \in \Sigma^*$ such that $(z, y) \in \Pi$ or decide that there exists no such y.

Example: given a matrix $A \in \mathbb{Q}^{m \times n}$ and a vector $b \in \mathbb{Q}^m$, find $x \in \mathbb{Q}^n$ such that $Ax \leq b$.

Decision problem: a problem which output is either 0 or 1.

Example: given A and b, is there an x such that $Ax \leq b$?

Decision problem: the set $\mathcal{L} \subset \Sigma^*$ of 1 instances.

Algorithm

Algorithm: a finite list of instruction to solve a problem.

Turing machine: thought experiment object which formalizes the notion of algorithm. Idealized computer.

Church-Turing thesis computable functions of natural numbers are precisely the ones which can be computed by a Turing machine.

Turing equivalent system: a formal system which can compute exactly the same functions as a Turing machine.

Examples: recursive functions, lambda calculus, circuits, ... and most programming languages (eventually idealized).

Algorithm: a computer program, *i.e.* a finite list of symbols from a finite alphabet.

Given input Σ^*, an algorithm A for problem Π determines y such that $(z, y) \in \Pi$, or stops without output if there is no such y.

Runing time: Number of elementary operations during the execution of an algorithm. Formally, the runing time function of an algorithm $f : \mathbb{N} \mapsto \mathbb{N}$ can be given by

$$f(\sigma) = \max_{\mathrm{size}(z) \leq \sigma} (\text{running time of } A \text{ for input } z).$$

Polynomial time algorithm An algorithm A for problem Π which time function is upper bounded by a polynomial. In this case Π is *polynomially solvable*.

The class $\mathcal{P}$: The class of polynomially solvable decision problems.

Computation over $\mathbb{Q}$

Euclidean algorithm is polynomial time (Gabriel Lamé 1844). Unique representation quotients in $\mathbb{Q}$. Addition, multiplication are also polynomial time.

Complexity over $\mathbb{Q}$: Number of elementary arithmetic operations.

In practice: Most numerical softwares perform finite precision arithmetic over $\mathbb{Q}$.

The class $\mathcal{NP}$

Non deterministic Polynomial time: Decisions problems which have a polynomial size proof.

$\mathcal{L} \subset \Sigma^* \in \mathcal{NP}$: there exists $\mathcal{L}' \subset \Sigma^* \times \Sigma^*$, $\mathcal{L}' \in \mathcal{P}$ and a polynomial ϕ such that

$$z \in \mathcal{L} \quad \Leftrightarrow \quad \exists y \in \Sigma^*, (z, y) \in \mathcal{L}' \text{ and } \mathrm{size}(y) \leq \phi(\mathrm{size}(z)).$$

such an y is called a certificate.

Brute force search: for any $\Pi \in \mathcal{NP}$ there is a polynomial ψ such that the solution for input z can be found in time at most $2^{\psi(\mathrm{size}(z))}$.

The class $\mathcal{NP}$: examples

Traveling salesman: Given pairwise distances between n cities (in $\mathbb{Q}$):

Given $d \in \mathbb{Q}$, decide if there is a path visiting all the cities of total length at most d.

Linear inequalities:

Given $A \in \mathbb{Q}^{n \times d}$ and $b \in \mathbb{Q}^n$, decide if $Ax \leq b$ has a solution over $\mathbb{Q}^n$.

Schiver's book chapter 10: if feasible, there is a solution which size is polynomially bounded by the size of A and b.

Karp reduction

$\mathcal{L} \in \Sigma^*$ is *Karp* reducible to $\mathcal{L}' \subset \Sigma^*$ if there exists a polynomial time algorithm such that, for any input string $z \in \Sigma^*$, A delivers a string x such that

$$z \in \mathcal{L} \quad \Leftrightarrow \quad x \in \mathcal{L}'$$

Notation: $\mathcal{L} \leq \mathcal{L}'$, an algorithm for solving $\mathcal{L}'$ would provide an algorithm for solving $\mathcal{L}$ with an added computational cost at most polynomial.

Karp reduction: example

$\mathcal{L}$: boolean formula satisfiability problems (SAT).
$\mathcal{L}'$: satisfiability problems of boolean formula in 3 conjunctive normal form (3-SAT)
$\mathcal{L} \leq \mathcal{L}'$.

Proof sketch: For any boolean formula there is a formula

- over linearly more variable.
- which size is at most linear in the size of the original formula
- in conjunctive normal form.
- which preserves satisfiability.

For example using Tseytin transformation. We obtain a formula of the form

$$(a \vee b \vee c \vee d) \wedge (\bar{a} \vee e \vee f \vee \bar{g} \vee d) \ldots$$

Any disjunction can be reduced to a conjunction of disjunctions of size at most 3 by adding variables:

$$\begin{aligned} & q \vee r \vee s \vee t \vee u \\ \Leftrightarrow \quad & (q \vee r \vee a) \wedge (\bar{a} \vee s \vee b) \wedge (\bar{b} \vee t \vee u). \end{aligned}$$

Karp reduction and the class $\mathcal{NP}$

if $\mathcal{L}'$ belongs to $\mathcal{NP}$ and $\mathcal{L} \leq \mathcal{L}'$, then $\mathcal{L}$ also belongs to $\mathcal{NP}$ (exercise).

$\mathcal{NP}$ **hardness:** $\mathcal{L}$ is $\mathcal{NP}$-*hard*, if each problem in $\mathcal{NP}$ is reducible to $\mathcal{L}$.

$\mathcal{NP}$ **completeness:** If furhtermore $\mathcal{L} \in \mathcal{NP}$, then $\mathcal{L}$ is called $\mathcal{NP}$-complete.

- brute force exponential time algorithm for problems in $\mathcal{NP}$.
- a polynomial time algorithm for one $\mathcal{NP}$ complete problem would provide a proof that $\mathcal{P} = \mathcal{NP}$.
- widely believed to be false.

$\mathcal{NP}$ hardness is a worst case theoretical notion

- $\mathcal{NP}$-complete problems considered hard: believed that no polynomial time algorithm exists **on all instances**.
- Karp reduction: some instances are hard, not necessarily all of them.
- No notion of constant or exponent, problems in $\mathcal{P}$ may still be intractable in practice.

Cook's Theorem

Boolean satisfiability (SAT): first problem proved to be $\mathcal{NP}$-complete by Cook in 1971.

Idea of the proof. SAT is clearly in $\mathcal{NP}$. The problem is $\mathcal{NP}$-hard: a polynomial time verifier implemented on a Turing machine can be shown to be equivalent to a boolean formula (technical bulk of the proof).

Consequence: As SAT$\leq$3-SAT and 3-SAT$\in \mathcal{NP}$, 3-SAT is also $\mathcal{NP}$-complete.

Sparse regression is $\mathcal{NP}$-hard

Theorem

Given $A \in \mathbb{Q}^{m \times n}$, $b \in \mathbb{Q}^m$, decide if there exist $x \in \mathbb{Q}^n$ such that $Ax = b$ and $\|x\|_0 \leq m/3$. This problem is $\mathcal{NP}$-hard.

Proof from Natarajan (1995) "Sparse approximate solutions to linear systems".

Same result replacing $Ax = b$ by $\|Ax - b\|_2^2 \leq 1/2$.

This is what is meant by "computing $\min_{\|x\|_0 \leq k} \|Ax - b\|_2^2$ is $\mathcal{NP}$-hard".

Computation over the reals

- $a \in \mathbb{R}$ is computable if there is a terminating algorithm A such that $\forall \epsilon \in \mathbb{Q}$, $\epsilon > 0$, $|A(\epsilon) - a| \leq \epsilon$. Computable numbers are only denumerable.
- BSS machine from Blum, Shub and Smale. Exact computation over real numbers. Leads to a notion of "algebraic complexity".
- Oracle complexity: an orcale performs real operations (and more), given a precision threshold $\epsilon > 0$, the complexity is the number of call to the oracle to reach precision ϵ. Used in optimization.

Plan

1. Computational complexity

2. Recap on convexity

3. Conic programming

4. Computational complexity of our statistical estimators

Definition

- $\mathcal{X} \subset \mathbb{R}^d$ is convex if for any $x, y \in \mathcal{X}$, $\alpha \in [0,1]$, $\alpha x + (1-\alpha)y \in \mathcal{X}$.
- $f : \mathbb{R}^d \to \mathbb{R}$ is convex if its epigraph is convex in $\mathbb{R}^{d+1}$.

$$\mathrm{epi}(f) = \left\{(x, z) \in \mathbb{R}^{d+1},\ z \geq f(x)\right\}$$

- Equivalently, for any $x, y \in \mathbb{R}^d$, and any $\alpha \in [0,1]$,

$$f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha) f(y).$$

Lemma

For any convex set $\mathcal{X} \subset \mathbb{R}^d$ we have

- *The closure of $\mathcal{X}$ is convex.*
- *The interior of $\mathcal{X}$ is convex.*
- *For any $u \in \mathrm{int}(\mathcal{X})$ and $v \in \mathrm{cl}(\mathcal{X})$, $[u, v) \subset \mathrm{int}(\mathcal{X})$.*
- *If the interior of $\mathcal{X}$ is non empty, then $\mathrm{cl}(\mathcal{X}) = \mathrm{cl}(\mathrm{int}(\mathcal{X}))$.*
- *The interior of $\mathcal{X}$ is empty if and only if it is contained in a lower dimensional affine subspace.*

Characterization of convex functions

Theorem

Let $f : \mathbb{R}^d \to \mathbb{R}$*:*

1. *If* f *is continuously differentiable, then* f *is convex if and only if or any* $x, y \in \mathbb{R}^d$*,* $f(y) \geq f(x) + \nabla f(x)^T (y - x)$*.*
2. *If* f *is continuously differentiable, then* f *is convex if and only if or any* $x, y \in \mathbb{R}^d$*,* $(\nabla f(x) - \nabla f(y))^T (y - x) \geq 0$*.*
3. *If* f *is twice continuously differentiable, then* f *is convex if and only if or any* $x \in \mathbb{R}^d$*,* $\nabla^2 f(x)$ *is positive semidefinite.*

Start with dimension 1.
$f : \mathbb{R}^d \mapsto \mathbb{R}$ is convex if and only if for any $x, y \in \mathbb{R}^d$, the function $g_{xy} : t \mapsto f(x + t(y - x))$ is convex.

Corollary (Fermat rule)

Let $f : \mathbb{R}^d \to \mathbb{R}$ *be convex continuously differentiable, then the following are equivalent*

- x *is a global minimizer of* f*.*
- $\nabla f(x) = 0$*.*

Example: $\hat{\theta}^{LS} \in \arg\min_{\theta \in \mathbb{R}^d} \|\mathbb{X}\theta - y\|_2^2$

Separating hyperplane

Theorem (Separating hyperplane)

- *Let $\mathcal{X}, \mathcal{Y} \subset \mathbb{R}^d$ be two disjoint closed convex sets, then there exists a vector $v \in \mathbb{R}^d$, $v \neq 0$ and a number $c \in \mathbb{R}$ such that $x^T v > c$ for all $x \in \mathcal{X}$ and $y^T v < c$ for all $y \in \mathcal{Y}$.*
- *Let $\mathcal{X}, \mathcal{Y} \subset \mathbb{R}^d$ be two disjoint convex sets, then there exists a vector $v \in \mathbb{R}^d$, $v \neq 0$ and a number $c \in \mathbb{R}$ such that $x^T v \geq c$ for all $x \in \mathcal{X}$ and $y^T v \leq c$ for all $y \in \mathcal{Y}$.*

Supporting hyperplane

Theorem (Supporting hyperplane)

- *Let $\mathcal{X} \subset \mathbb{R}^d$ be a convex sets such that $0 \notin \mathcal{X}$, then there exists a vector $v \in \mathbb{R}^d$, $v \neq 0$ such that $v^T x \geq 0$, for all $x \in \mathcal{X}$.*
- *Let $\mathcal{X} \subset \mathbb{R}^d$ be a convex set such that 0 is on the boundary of $\mathcal{X}$, then there exists a vector $v \in \mathbb{R}^d$, $v \neq 0$ such that $v^T x \geq 0$, for all $x \in \mathcal{X}$.*

Definition

x is an extreme point of the convex set $\mathcal{X} \subset \mathbb{R}^d$, if for any $x_1, x_2 \in \mathcal{X}$, $x = (x_1 + x_2)/2$ implies that $x_1 = x_2 = x$.

Lemma

Let $c \in \mathbb{R}^d$, $c \neq 0$ and $\mathcal{X}$ be a convex and compact set. Then $\min_{x \in \mathcal{X}} c^T x$ is attained then the optimum is attained at an extreme point $\bar{x} \in \mathcal{X}$.

Theorem (Krein Millman)

Let $\mathcal{X}$ be a compact convex set, then $\mathcal{X} \subset \mathbb{R}^d$ is the convex hull of its extreme points.

Definition

A polyhedra is a set $\mathcal{X} \subset \mathbb{R}^d$ such that there exists $A \in \mathbb{R}^{m \times d}$, $b \in \mathbb{R}^m$ such that $\mathcal{X} = x \in \mathbb{R}^d,\ Ax \leq b$. This is a canonical form representation.

Add variables $s \in \mathbb{R}^m$, set x_+ and x_- the entry-wise positive and negative part of x, $\mathcal{X} = \{(x_+, x_-, s) \in \mathbb{R}^{2n+m},\ s = b - A(x_+ - x_-),\ s \geq 0, x_+ \geq 0, x_- \geq 0\}$.

$\mathcal{X} = \{x \in \mathbb{R}^d, Ax = b, x \geq 0\}$ for a matrix A and a vector b which is called standard form.

Lemma

Let $\mathcal{X} = \left\{x \in \mathbb{R}^d, Ax = b, x \geq 0\right\}$ be non empty. Then $\mathcal{X}$ has at least one and at most a finite number of extreme points. We have the following equivalence

- *x is an extreme point of $\mathcal{X}$*
- *the columns of A corresponding to non zero entries of x are independent.*

Example: The ℓ_1 ball in $\mathbb{R}^d$ is a polytope which has $2d$ extreme points. Linear fuctions attain their optimum at these extreme points.

Plan

Definition

$\mathcal{K} \subset \mathbb{R}^d$ is a cone if it satisfies for any $x \in \mathcal{K}$ and $\alpha \geq 0$, $\alpha x \in \mathcal{K}$.

Conic programs: Given a closed convex cone $\mathcal{K}$, for any $A \in \mathbb{R}^{m \times d}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^d$,

$$p^* = \inf_{x \in \mathbb{R}^d} \quad c^T x \qquad \text{s.t.} \qquad Ax = b,\ x \in \mathcal{K}. \tag{P}$$

Conic hierarchy:

- $\mathcal{K} = \mathbb{R}^d_+$, linear programs (LP).
- $\mathcal{K} = \left\{(x, t) \in \mathbb{R}^{d+1},\ \|x\|_2 \leq t\right\}$, second order cone programs (SOCP).
- $\mathcal{K}$ the cone of positive semidefinite matrices, semidefinite programs (SDP).
  $C \in \mathbb{R}^{d \times d}$, $\mathcal{A}\colon \mathbb{R}^{d \times d} \to \mathbb{R}^m$ linear, $b \in \mathbb{R}^m$

$$\min_{X \in \mathbb{R}^{d \times d}} \operatorname{tr}(C^T X) \quad \text{s.t.} \quad \mathcal{A}(X) = b,\ X^T = X,\ X \succcurlyeq 0.$$

Exercise: An LP can be expressed as a SOCP which can be expressed as an SDP (Schur complement argument).

Dual cone

Definition

Let $\mathcal{K} \subset \mathbb{R}^d$ be a convex cone, the dual cone of $\mathcal{K}$ is denoted by

$$\mathcal{K}^* = \left\{ y \in \mathbb{R}^d,\, x^T y \geq 0,\, \forall x \in \mathcal{K} \right\}$$

If $\mathcal{K} = \mathcal{K}^*$, we say that $\mathcal{K}$ is self dual

Conic duality

Primal program: closed convex cone $\mathcal{K}$, $A \in \mathbb{R}^{m \times d}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^d$,

$$p^* = \inf_{x \in \mathbb{R}^d} \quad c^T x \qquad \text{s.t.} \qquad Ax = b,\, x \in \mathcal{K}. \tag{P}$$

Lagrangian: for any $x \in \mathbb{R}^d$, $\mu \in \mathbb{R}^d$, $\nu \in \mathbb{R}^m$,

$$\mathcal{L}(x, \mu) = c^T x + \mu^T (b - Ax) \tag{1}$$

Dual problem: obtained by minimizing the Lagrangian over $\mathcal{K}$.

$$d^* = \sup b^T \mu \qquad \text{s.t.} \qquad c - A^T \mu \in \mathcal{K}^*. \tag{D}$$

Theorem

- *It holds that* $d^* \leq p^*$.
- *If* $\mathrm{rank}(A) = m$, *there is* $\bar{x}$ *such that* $A\bar{x} = b$ *and* $\bar{x} \in \mathrm{int}(\mathcal{K})$ *and* $p^* > -\infty$, *then* $p^* = d^*$ *and the dual problem has a solution.*
- *In this case,* x *is primal optimal if and only if it is primal feasible and there exists a dual feasible* μ *such that*

$$x^T (c - A^T \mu) = 0 \qquad \text{or} \qquad x^T c = b^T \mu.$$

This notion will be important to develop algorithmic ideas to solve the optimization problems which we have seen.

Definition

A function $f : \mathbb{R}^d \mapsto \mathbb{R}$ is μ strongly convex, if $f - \frac{\mu}{2}\|\cdot\|$ is convex. The following provide sufficient conditions:

- If f is differentiable, $f(y) \geq f(x) + (y-x)^T \nabla f(x) + \frac{\mu}{2}\|y-x\|_2^2$, for all x, y.
- If f is differentiable, $(\nabla f(x) - \nabla f(y))^T (y - x) \geq \mu \|y-x\|_2^2$ for all x, y.
- If f is twice differentiable, the matrix $\nabla^2 f(x) - \mu I$ is positive semidefinite for all x.

Exercise: Prove that the function $f : x \mapsto -\log(1 - \|x\|^2)$ is strongly convex (when restricted to the unit Euclidean ball).

Newton's method

Strongly convex function $f : \mathbb{R}^d \mapsto \mathbb{R}$, $\alpha > 0$: choose x_0 and iterate for $k \in \mathbb{N}$,

$$x_{k+1} = x_k - \alpha \left(\nabla^2 f(x_k)\right)^{-1} \nabla f(x_k). \tag{2}$$

Where α is a positive stepsize, determined algorithmically.

Theorem (Local quadratic convergence for Newton's method)

Let f be μ-strongly convex, twice continuoulsy differentiable, with L-Lipschitz Hessian (operator norm) and $\bar{x}$ be the (unique) minimum of f. Newton's method with unit step size satisfy, for all $k \in \mathbb{N}$,

$$\frac{L}{2\mu^2}\|\nabla f(x_k)\|_2 \leq \left(\frac{L}{2\mu^2}\|\nabla f(x_0)\|_2\right)^{2^k},$$

In particular, if $\|\nabla f(x_0)\|_2 < \frac{L}{2\mu^2}$, we have quadratic convergence.

Given $a \in \mathbb{R}^d$, $b \in \mathbb{R}$, and $f : \mathbb{R}^d \mapsto \mathbb{R}$, convex differentiable

$$f^* = \min_{x \in \mathbb{R}^d} f(x) \qquad \text{s.t.} \qquad \|x\|_2 \leq 1,\ a^T x \leq b \tag{3}$$

Barrier method: For any $t > 0$,

$$f_t^* = \min_{x \in \mathbb{R}^d} tf(x) - \log(1 - \|x\|_2^2) - \log(b - a^T x) \tag{4}$$

Central path: (4) is strongly convex, its minimum x_t is attained and is unique, the central path is the map $t \mapsto x_t$.

Lemma

For any $t > 0$, we have $f(x_t) \leq f^* + \frac{2}{t}$.

Polynomial time LP (and QP) solver

Theorem (Khachiyan,Karmarkar)

Given inputs $A \in \mathbb{Q}^{n\times d}$, $b \in \mathbb{Q}^n$ and $c \in \mathbb{Q}^d$ consider the problem of computing

$$\rho = \inf_{x\in\mathbb{Q}^d} c^T x \quad \text{s.t. } Ax \leq b. \tag{5}$$

This problem is in $\mathcal{P}$.

- if the infimum is not attained there polynomial time certificates for this can be found in polynomial time.
- the optimum is attained at one of the finitely many vertices of the polyhedra.
- Only polynomialy many candidate optimal values for ρ.
- Ellipsoid method (for Khachiyan's algorithm) or interior point methods (for Karmarkar's algorithm) converge exponentially fast to ρ.
- Carefully controling the magnitude of accumulated errors allong the local search path and the degree of approximation required to dicriminate between any two candidate optimal values.

Convex quadratic objectives over linear constraint can also be solved in polynomial time over $\mathbb{Q}$.

Plan

Consequence for our statistical estimators

Conclusion:

- All the estimators involving the ℓ_1 norm can be computed in polynomial time given data in $\mathbb{Q}$.
- It is largely accepted that there is no efficient algorithm to compute all the possible large scale instances of the estimators involving the ℓ_0 pseudo norm.