CS 6501: Constrained-Aware Generative AI

Lecture Notes 6
Optimization Essentials

Projection, penalties, proximal operators, and duality as reusable primitives

Prof. Ferdinando Fioretto
Department of Computer Science, University of Virginia

Thursday, January 29, 2026

Abstract

Constrained-aware generation repeatedly reduces to a small number of optimization primitives that can be inserted into training-time objectives, architectural design, or inference-time procedures. This lecture reviews four primitives that will recur throughout the course. First, we formalize constraint sets and Euclidean projections, emphasizing why projection is a least-squares problem and how its optimality conditions encode feasibility through a normal cone. Second, we review penalty methods as a mechanism to convert constrained problems into unconstrained ones, clarifying when and why large penalty weights lead to ill-conditioning and how exact penalties avoid this pathology. Third, we introduce proximal operators, which generalize projection to nonsmooth penalties and yield a clean algorithmic template for "one step of model improvement, one step of constraint enforcement." Finally, we review Lagrangian duality and KKT conditions, interpreting dual variables as constraint prices and motivating splitting methods such as ADMM. The intended outcome is that, later in the course, projection, proximal steps, and primal-dual updates can be recognized as the common computational core behind diverse constrained generation methods.

1 Introduction: optimization as a constraint-injection language

In Lecture 1 we framed constrained-aware generation as sampling or optimization under a target distribution that blends a base model with soft and hard constraint factors. If $p_{\boldsymbol{\theta}}(\mathbf{x} \mid c)$ is the base generator and $\mathcal{C}(c)$ is a hard feasibility set, then "injecting constraints" in practice means designing algorithms that repeatedly reconcile two pressures: the model's plausibility prior and the external feasibility requirements.

A useful way to see why the same optimization ideas keep reappearing is to isolate the following design pattern. We maintain an iterate $\mathbf{x}^k$ that lives in an ambient space $\mathcal{X} \subseteq \mathbb{R}^d$. A model step proposes a move that improves plausibility or a smooth objective, and a constraint step then restores feasibility or reduces violation. Projection and proximal operators formalize the constraint step. Penalty methods formalize the choice to never enforce feasibility exactly, but to bias solutions toward feasibility. Duality formalizes the idea that constraints can be priced and enforced by coupling primal variables to multipliers.

Throughout, we focus on convex analysis primitives because they are (i) mathematically crisp, (ii) computationally reusable, and (iii) robust under composition. Even when the underlying constraint set is nonconvex (collision-free robotics trajectories, protein stability constraints, or grammar constraints in text), these primitives still guide the design of relaxations, local approximations, and splitting strategies.

2 Constraint sets and projection

2.1 Constraint sets, distance, and indicator functions

A *constraint set* is a subset $\mathcal{C} \subseteq \mathbb{R}^d$ that encodes hard validity requirements. The simplest convex constraints are affine subspaces, halfspaces, and norm balls, but the same notation applies to combinatorial sets after relaxation.

Two auxiliary functions are useful. The *distance to a set* is

$$\operatorname{dist}(\mathbf{x}, \mathcal{C}) \triangleq \inf_{\mathbf{y} \in \mathcal{C}} \|\mathbf{x} - \mathbf{y}\|_2. \tag{1}$$

The *indicator* of $\mathcal{C}$ is the extended-real-valued function

$$\iota_{\mathcal{C}}(\mathbf{x}) \triangleq \begin{cases} 0, & \mathbf{x} \in \mathcal{C}, \\ +\infty, & \mathbf{x} \notin \mathcal{C}. \end{cases} \tag{2}$$

While $\iota_{\mathcal{C}}$ looks artificial, it is the key bridge from constraint sets to proximal operators in section 4.

2.2 Euclidean projection is a least-squares problem

Definition 1 (Euclidean projection)**.** *For a nonempty closed set $\mathcal{C} \subseteq \mathbb{R}^d$, the Euclidean projection map is*

$$\operatorname{Proj}_{\mathcal{C}}(\mathbf{v}) \triangleq \arg\min_{\mathbf{x} \in \mathcal{C}} \ \frac{1}{2}\|\mathbf{x} - \mathbf{v}\|_2^2. \tag{3}$$

If $\mathcal{C}$ is convex, the minimizer is unique.

> **Key identity: projection is least squares**
>
> Projection is not an additional primitive. It is the solution of a constrained least-squares problem. The quadratic term $\frac{1}{2}\|\mathbf{x} - \mathbf{v}\|_2^2$ is precisely the least-squares objective measuring how much we must change $\mathbf{v}$ to restore feasibility.

The optimality conditions of equation (3) expose the geometry of feasibility. For a closed convex set $\mathcal{C}$, define its *normal cone* at $\mathbf{x} \in \mathcal{C}$ as

$$\mathcal{N}_{\mathcal{C}}(\mathbf{x}) \triangleq \{\mathbf{g} \in \mathbb{R}^d : \langle \mathbf{g}, \mathbf{y} - \mathbf{x} \rangle \leq 0 \text{ for all } \mathbf{y} \in \mathcal{C}\}. \tag{4}$$

Proposition 1 (Optimality condition for projection)**.** *Let $\mathcal{C} \subseteq \mathbb{R}^d$ be closed and convex, and let $\mathbf{x}^\star = \operatorname{Proj}_{\mathcal{C}}(\mathbf{v})$. Then*

$$\mathbf{v} - \mathbf{x}^\star \in \mathcal{N}_{\mathcal{C}}(\mathbf{x}^\star). \tag{5}$$

Equivalently, $\langle \mathbf{v} - \mathbf{x}^\star, \mathbf{y} - \mathbf{x}^\star \rangle \leq 0$ for all $\mathbf{y} \in \mathcal{C}$.

The vector $\mathbf{v} - \mathbf{x}^\star$ is the shortest correction needed to return to the feasible region. The normal-cone condition equation (5) says that this correction points orthogonally outward from the feasible set. This fact is often the right interpretation for "constraint forces" inserted into iterative generation.

2.3 Examples of projections

Affine subspace. Let $\mathcal{C} = \{\mathbf{x} : A\mathbf{x} = \mathbf{b}\}$ with $A \in \mathbb{R}^{m \times d}$ full row rank. Projection solves $\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{x} - \mathbf{v}\|_2^2$ s.t. $A\mathbf{x} = \mathbf{b}$. The KKT conditions yield

$$\mathbf{x}^\star = \mathbf{v} - A^\top \boldsymbol{\lambda}^\star, \qquad AA^\top \boldsymbol{\lambda}^\star = A\mathbf{v} - \mathbf{b}. \tag{6}$$

Thus projection reduces to solving an $m \times m$ linear system when $m \ll d$.

Box constraints. If $\mathcal{C} = [\ell_1, u_1] \times \cdots \times [\ell_d, u_d]$, then projection is coordinatewise clipping: $\big(\mathrm{Proj}_{\mathcal{C}}(\mathbf{v})\big)_i = \min\{u_i, \max\{\ell_i, v_i\}\}$.

ℓ_2 ball. For $\mathcal{C} = \{\mathbf{x} : \|\mathbf{x}\|_2 \le r\}$,

$$\mathrm{Proj}_{\mathcal{C}}(\mathbf{v}) = \begin{cases} \mathbf{v}, & \|\mathbf{v}\|_2 \le r, \\ \frac{r}{\|\mathbf{v}\|_2}\mathbf{v}, & \|\mathbf{v}\|_2 > r. \end{cases} \tag{7}$$

Probability simplex. Let $\Delta^d = \{\mathbf{x} \in \mathbb{R}^d : \mathbf{x} \ge 0,\ \sum_i x_i = 1\}$. The projection onto Δ^d has a closed form in terms of a threshold τ: $\big(\mathrm{Proj}_{\Delta^d}(\mathbf{v})\big)_i = \max\{v_i - \tau, 0\}$ with τ chosen so that the entries sum to 1. Computationally, τ is found by sorting v_i and locating the active set. This projection appears whenever a generative procedure updates a relaxed categorical distribution and then restores normalization and nonnegativity.

2.4 Projected gradient as a two-step template

Consider the constrained smooth minimization problem

$$\min_{\mathbf{x} \in \mathcal{C}} \ f(\mathbf{x}), \tag{8}$$

where f is differentiable with L-Lipschitz gradient. The simplest algorithm is *projected gradient descent*:

$$\mathbf{x}^{k+1} = \mathrm{Proj}_{\mathcal{C}}\left(\mathbf{x}^k - \eta \nabla f(\mathbf{x}^k)\right). \tag{9}$$

The form of equation (9) is the template we will repeatedly reuse in constrained generation: first a model-driven update, then a feasibility restoration.

Remark 1 (Nonconvex constraints)**.** *If $\mathcal{C}$ is nonconvex,* $\mathrm{Proj}_{\mathcal{C}}$ *may be set-valued and difficult to compute. In practice, one uses tractable relaxations (convex outer approximations), local projections, or "projection" defined implicitly through a solver. The algebraic structure of equation (9) remains useful even when* $\mathrm{Proj}_{\mathcal{C}}$ *is approximate.*

3 Penalty methods

Penalty methods enforce constraints by adding violation terms to the objective. They are the optimization analogue of turning a hard constraint factor $\mathbf{1}\{\mathbf{x} \in \mathcal{C}\}$ into a soft potential $\exp(-\lambda\phi(\mathbf{x}))$.

3.1 From constrained to unconstrained objectives

Consider the generic constrained problem

$$\min_{\mathbf{x} \in \mathbb{R}^d} \ f(\mathbf{x}) \quad \text{s.t.} \quad h(\mathbf{x}) = \mathbf{0}, \quad g_i(\mathbf{x}) \le 0 \ (i = 1, \ldots, m). \tag{10}$$

A quadratic-penalty objective is

$$\min_{\mathbf{x} \in \mathbb{R}^d} \ f(\mathbf{x}) + \frac{\rho}{2}\|h(\mathbf{x})\|_2^2 + \frac{\rho}{2}\sum_{i=1}^{m}[g_i(\mathbf{x})]_+^2, \tag{11}$$

where $[t]_+ = \max\{t, 0\}$. As $\rho \to \infty$, minimizers of equation (11) approach feasibility under suitable regularity assumptions. The drawback is numerical: large ρ makes the objective ill-conditioned, so gradient-based optimization becomes slow and unstable.

3.2 Exact penalties

Exact penalties aim to achieve feasibility at a finite penalty weight. For equality constraints, the ℓ_1 penalty

$$\min_{\mathbf{x}\in\mathbb{R}^d} \; f(\mathbf{x}) + \rho \, \|h(\mathbf{x})\|_1 \tag{12}$$

is *exact* under standard constraint qualifications: for sufficiently large ρ, minimizers of equation (12) are solutions of equation (10). A similar statement holds for inequalities using $\sum_i [g_i(\mathbf{x})]_+$. Exactness is a key reason why nonsmooth penalties and proximal operators are central in modern constrained ML pipelines.

3.3 Example: penalty shaping in generative inference

In constrained generation, one often introduces a violation function $\nu(\mathbf{x}, c)$ and samples from $\pi(\mathbf{x} \mid c) \propto p_{\boldsymbol{\theta}}(\mathbf{x} \mid c) \exp(-\lambda \nu(\mathbf{x}, c))$. When ν measures constraint violations (grammar errors, physical constraint residuals, collision indicators smoothed into a barrier), this is precisely a penalty method in probabilistic form. The same pathology appears: very large λ enforces feasibility but can destabilize inference or lead to poor mixing, while moderate λ yields higher diversity but lower validity. The rest of the course can be read as algorithmic responses to this tradeoff.

4 Proximal operators and nonsmooth penalties

4.1 Definition and first properties

Definition 2 (Proximal operator)**.** *Let $g : \mathbb{R}^d \to (-\infty, +\infty]$ be a proper, closed, convex function, and let $\gamma > 0$. The proximal operator of g is*

$$\operatorname{prox}_{\gamma g}(\mathbf{v}) \triangleq \arg\min_{\mathbf{x}\in\mathbb{R}^d} \left\{ g(\mathbf{x}) + \frac{1}{2\gamma} \|\mathbf{x} - \mathbf{v}\|_2^2 \right\}. \tag{13}$$

Proposition 2 (Projection is a proximal operator)**.** *For a nonempty closed convex set $\mathcal{C}$,*

$$\operatorname{prox}_{\gamma \iota_{\mathcal{C}}}(\mathbf{v}) = \operatorname{Proj}_{\mathcal{C}}(\mathbf{v}) \quad \textit{for all } \gamma > 0. \tag{14}$$

Thus, proximal operators unify hard constraints ($g = \iota_{\mathcal{C}}$) and soft, possibly nonsmooth penalties (g a regularizer). A prox step can be read as "move toward lower penalty, but stay close to the proposal $\mathbf{v}$."

4.2 Canonical proximal maps

ℓ_1 regularization (soft-thresholding). For $g(\mathbf{x}) = \|\mathbf{x}\|_1$, the proximal operator is coordinatewise soft-thresholding:

$$\big(\operatorname{prox}_{\gamma\|\cdot\|_1}(\mathbf{v})\big)_i = \operatorname{sign}(v_i) \, \max\{|v_i| - \gamma, 0\}. \tag{15}$$

This is the workhorse for sparse structure and exact penalties.

Group sparsity. For $g(\mathbf{x}) = \sum_{g\in\mathcal{G}} \|\mathbf{x}_g\|_2$ (a group $\ell_{2,1}$ norm), the prox is blockwise shrinkage. This matters when constraints or penalties are naturally grouped, as in structured editing of sequences (edit only a subset of blocks).

Hinge-type penalties. Penalties such as $g(t) = [t]_+$ and $g(t) = [t]_+^2$ appear for inequalities. Their proximal maps are 1D closed forms and enable efficient handling of large numbers of simple constraints.

4.3 Forward-backward splitting and proximal gradient

A common optimization form is

$$\min_{\mathbf{x}\in\mathbb{R}^d} \; F(\mathbf{x}) \triangleq f(\mathbf{x}) + g(\mathbf{x}), \tag{16}$$

where f is differentiable with L-Lipschitz gradient and g is convex but possibly nonsmooth. The *proximal gradient* (forward-backward) iteration is

$$\mathbf{x}^{k+1} = \mathrm{prox}_{\eta g}\left(\mathbf{x}^k - \eta\nabla f(\mathbf{x}^k)\right), \qquad 0 < \eta \le \frac{1}{L}. \tag{17}$$

If $g = \iota_{\mathcal{C}}$, equation (17) reduces to projected gradient descent equation (9).

Acceleration (optional). FISTA is an accelerated variant of proximal gradient that achieves the optimal first-order rate $O(1/k^2)$ for convex objectives (Beck and Teboulle, 2009). The important message for this course is not the exact update, but that acceleration is available whenever we can express constraint enforcement as a prox step.

5 Duality and KKT conditions

Duality formalizes constraint enforcement through multipliers rather than penalties. This is the conceptual basis for Lagrangian relaxation in constrained decoding, primal-dual control, and splitting methods like ADMM.

5.1 Lagrangian and dual function

Consider the convex program

$$\min_{\mathbf{x}} \; f(\mathbf{x}) \quad \text{s.t.} \quad A\mathbf{x} = \mathbf{b}, \;\; g_i(\mathbf{x}) \le 0. \tag{18}$$

Its Lagrangian is

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) = f(\mathbf{x}) + \boldsymbol{\lambda}^\top(A\mathbf{x} - \mathbf{b}) + \sum_i \nu_i g_i(\mathbf{x}), \qquad \boldsymbol{\nu} \ge 0. \tag{19}$$

The dual function is $d(\boldsymbol{\lambda}, \boldsymbol{\nu}) = \inf_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\nu})$. Maximizing d yields the dual problem.

Proposition 3 (Weak duality)**.** *For any primal-feasible* $\mathbf{x}$ *and dual-feasible* $(\boldsymbol{\lambda}, \boldsymbol{\nu})$*,* $d(\boldsymbol{\lambda}, \boldsymbol{\nu}) \le f(\mathbf{x})$*. In particular, the optimal dual value lower-bounds the optimal primal value.*

5.2 Strong duality and Slater

When equation (18) is convex and satisfies Slater's condition (strict feasibility for inequalities), strong duality holds and the duality gap is zero (Boyd and Vandenberghe, 2004). In that regime, primal and dual solutions satisfy KKT conditions.

KKT as "physics" of constrained opt

At a primal-dual optimum $(\mathbf{x}^\star, \boldsymbol{\lambda}^\star, \boldsymbol{\nu}^\star)$, KKT conditions state that (i) the primal point is feasible, (ii) the dual multipliers are feasible ($\boldsymbol{\nu}^\star \ge 0$), (iii) stationarity balances objective gradient and constraint gradients, and (iv) complementary slackness ensures multipliers activate exactly on tight constraints. This is the cleanest formal meaning of "constraints exert pressure."

Algorithm 1 ADMM (scaled form)

Initialize $\mathbf{x}^0, \mathbf{z}^0, \mathbf{u}^0$ and choose $\rho > 0$.
for $k = 0, 1, 2, \dots$ **do**
  $\mathbf{x}^{k+1} \leftarrow \arg\min_{\mathbf{x}} \; f(\mathbf{x}) + \frac{\rho}{2}\|A\mathbf{x} + B\mathbf{z}^k - \mathbf{c} + \mathbf{u}^k\|_2^2$
  $\mathbf{z}^{k+1} \leftarrow \arg\min_{\mathbf{z}} \; g(\mathbf{z}) + \frac{\rho}{2}\|A\mathbf{x}^{k+1} + B\mathbf{z} - \mathbf{c} + \mathbf{u}^k\|_2^2$
  $\mathbf{u}^{k+1} \leftarrow \mathbf{u}^k + (A\mathbf{x}^{k+1} + B\mathbf{z}^{k+1} - \mathbf{c})$
end for

5.3 Example: dual view of projection onto affine constraints

The projection problem onto $\{\mathbf{x} : A\mathbf{x} = \mathbf{b}\}$ is

$$\min_{\mathbf{x}} \; \frac{1}{2}\|\mathbf{x} - \mathbf{v}\|_2^2 \quad \text{s.t.} \quad A\mathbf{x} = \mathbf{b}. \tag{20}$$

Its Lagrangian is $\frac{1}{2}\|\mathbf{x} - \mathbf{v}\|_2^2 + \boldsymbol{\lambda}^\top(A\mathbf{x} - \mathbf{b})$. Minimizing over $\mathbf{x}$ yields $\mathbf{x} = \mathbf{v} - A^\top\boldsymbol{\lambda}$ and the dual problem

$$\max_{\boldsymbol{\lambda}} \; -\frac{1}{2}\|A^\top\boldsymbol{\lambda}\|_2^2 - \boldsymbol{\lambda}^\top(A\mathbf{v} - \mathbf{b}), \tag{21}$$

whose optimality condition is exactly the linear system in equation (6). This example is worth remembering because many "projection layers" reduce to solving a small KKT system.

6 ADMM as a reusable splitting mechanism

ADMM is a prototypical algorithm that alternates between two easy subproblems and updates a dual variable. It is often the right abstraction for "prior step plus constraint step" when the constraint couples two representations of the object.

Consider

$$\min_{\mathbf{x},\mathbf{z}} \; f(\mathbf{x}) + g(\mathbf{z}) \quad \text{s.t.} \quad A\mathbf{x} + B\mathbf{z} = \mathbf{c}. \tag{22}$$

The augmented Lagrangian adds a quadratic penalty to the constraint:

$$\mathcal{L}_\rho(\mathbf{x}, \mathbf{z}, \mathbf{u}) = f(\mathbf{x}) + g(\mathbf{z}) + \mathbf{u}^\top(A\mathbf{x} + B\mathbf{z} - \mathbf{c}) + \frac{\rho}{2}\|A\mathbf{x} + B\mathbf{z} - \mathbf{c}\|_2^2. \tag{23}$$

ADMM alternates minimization over $\mathbf{x}$ and $\mathbf{z}$ and then updates $\mathbf{u}$. When $A = I$, $B = -I$, and $\mathbf{c} = 0$, the $\mathbf{z}$-update becomes a proximal step for g.

ADMM is central in distributed optimization (Boyd et al., 2011), but it is also a conceptual blueprint for constrained inference procedures: separate a "model variable" from a "constraint variable," alternate updates, and use a dual variable to keep them consistent.

7 Summary and outlook

The purpose of this lecture is to make projection, penalties, proximal steps, and duality feel like a compact constraint toolkit. Projection is a least-squares correction map whose optimality conditions encode geometry through normal cones. Penalty methods are soft constraint injection and exhibit a stability-feasibility tradeoff. Proximal operators unify hard sets and nonsmooth penalties, yielding clean splitting algorithms such as proximal gradient and FISTA. Duality adds the complementary viewpoint that constraints can be enforced by prices, leading naturally to primal-dual methods and ADMM.

In subsequent lectures, these primitives will reappear in disguised form: guidance in diffusion can be read as a penalty shaping term, projection-based diffusion and constrained decoding are projected or proximal steps on structured sets, and differentiable optimization layers are KKT systems differentiated implicitly.

References

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

N. Parikh and S. Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3):123–231, 2014.

S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

P. L. Combettes and J.-C. Pesquet. Proximal splitting methods in signal processing. In H. Bauschke, R. Burachik, P. Combettes, V. Elser, D. Luke, and H. Wolkowicz, editors, *Fixed-Point Algorithms for Inverse Problems in Science and Engineering*, pages 185–212. Springer, 2011.

J.-J. Moreau. Proximité et dualité dans un espace hilbertien. *Bulletin de la Société Mathématique de France*, 93:273–299, 1965.

R. T. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.