CSCI 0410: AI Foundations — **Spring 2026**

Professor Serena Booth[1] — **Continuous Optimization Unit**

Lagrangian Methods

Consider the following constrained optimization problem:

$$\begin{aligned} \max_{x} \quad & 2x_1 + x_2 \\ \text{s.t.} \quad & x_1^2 + x_2^2 \leq 1 \end{aligned}$$

The constraint $x_1^2 + x_2^2 \leq 1$ constrains all possible solutions to lie within a circle of radius 1 centered at the origin. The optimal solution is the point in that feasible space that maximizes $2x_1 + x_2$.

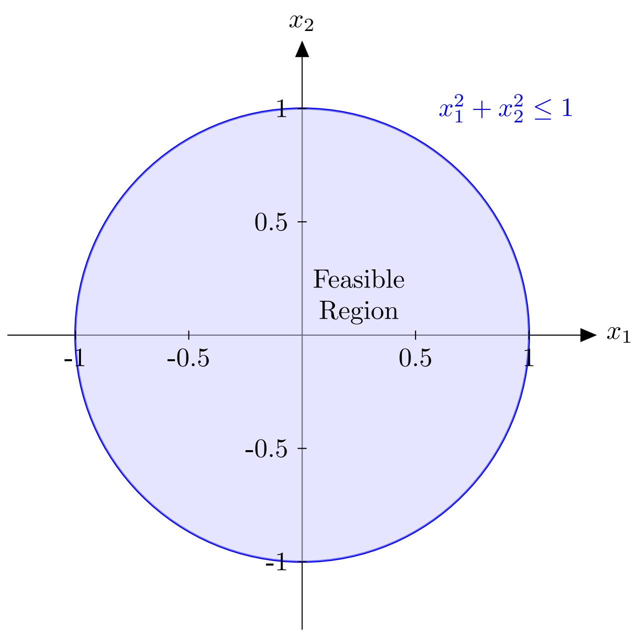

Figure 1: The feasible region given by $x_1^2 + x_2^2 \leq 1$

What if we didn't have to solve this as a constrained optimization problem? What if we could solve an unconstrained problem? We have a few tools for solving unconstrained optimization problems (e.g., closed-form solutions and gradient descent). The key idea is to add a **penalty term** to our objective. This penalty term should penalize our solution when a constraint is **violated**.

Consider the framing presented in equation 1:

$$\max_{x} \quad 2x_1 + x_2 - \text{penalty}(x_1, x_2) \tag{1}$$

How would like this penalty function to work? When our constraint is violated (i.e., $x_1^2 + x_2^2 > 1$), then penalty should return some large value to reduce our objective value and make that solution less favorable

[1] This notes were compiled in conjunction with with Professors Eric Ewing and Amy Greenwald.

to our optimizer. The size of that penalty term should ideally make it so that any solution outside of the region is penalized so greatly that it looks worse than any solution in the feasible region.

What if our solution is in the feasible region? Then we shouldn't penalize our solution and should return 0.

How can we create a penalty function in practice? We can rearrange our constraint as follows:

$$x_1^2 + x_2^2 - 1 \leq 0$$

If the $x_1^2 + x_2^2 - 1$ is ever positive, we'd like to penalize our model. Let's introduce a multiplier $\lambda \leq 0$ that we can multiply this quantity by:

$$\lambda \cdot (x_1^2 + x_2^2 - 1)$$

Note: in this case we are restricting λ to be non-positive because we'd only like to penalize our objective when $x_1^2 + x_2^2 - 1$ is positive, and penalizing our objective means adding negative values.

The Lagrangian is:

$$\mathcal{L}(x_1, x_2, \lambda) = 2x_1 + x_2 - \lambda \cdot (x_1^2 + x_2^2 - 1)$$

To find the optimal solution, we compute the gradient of the Lagrangian:

$$\nabla \mathcal{L} = \left[\frac{\partial \mathcal{L}}{\partial x_1}, \frac{\partial \mathcal{L}}{\partial x_2}, \frac{\partial \mathcal{L}}{\partial \lambda} \right]^T = [2 - 2\lambda x_1, 1 - 2\lambda x_2, -(x_1^2 + x_2^2 - 1)]^T$$

Setting $\nabla \mathcal{L} = \vec{0}$, we obtain the following system of equations:

$$2 - 2\lambda x_1 = 0 \tag{2}$$

$$1 - 2\lambda x_2 = 0 \tag{3}$$

$$x_1^2 + x_2^2 - 1 = 0 \tag{4}$$

From equation (2):

$$x_1 = \frac{1}{\lambda}$$

From equation (3):

$$x_2 = \frac{1}{2\lambda}$$

Substituting into equation (4):

$$\left(\frac{1}{\lambda}\right)^2 + \left(\frac{1}{2\lambda}\right)^2 = 1$$

$$\frac{1}{\lambda^2} + \frac{1}{4\lambda^2} = 1$$

$$\frac{4+1}{4\lambda^2} = 1$$

$$\frac{5}{4\lambda^2} = 1$$

$$\lambda^2 = \frac{5}{4}$$

$$\lambda = \pm\frac{\sqrt{5}}{2}$$

Since we require $\lambda \geq 0$ for the constraint to be active (by the KKT conditions), we take $\lambda = \frac{\sqrt{5}}{2}$.

Therefore, the optimal solution is:

$$\begin{aligned}
x_1^* &= \frac{1}{\lambda} = \frac{2}{\sqrt{5}} = \frac{2\sqrt{5}}{5} \\
x_2^* &= \frac{1}{2\lambda} = \frac{1}{\sqrt{5}} = \frac{\sqrt{5}}{5} \\
\lambda^* &= \frac{\sqrt{5}}{2}
\end{aligned}$$

We can verify this solution satisfies our constraint:

$$x_1^{*2} + x_2^{*2} = \frac{4}{5} + \frac{1}{5} = 1$$

The optimal objective value is:

$$2x_1^* + x_2^* = \frac{4\sqrt{5}}{5} + \frac{\sqrt{5}}{5} = \frac{5\sqrt{5}}{5} = \sqrt{5} \approx 2.236$$

KKT Conditions:

1. **Stationarity:** $\nabla f(\boldsymbol{x}^*) + \sum_{i=1}^{m} \lambda_i^* \nabla g_i(\boldsymbol{x}^*) + \sum_{j=1}^{p} \mu_j^* \nabla h_j(\boldsymbol{x}^*) = \mathbf{0}$
2. **Primal feasibility:**

$$g_i(\boldsymbol{x}^*) \leq 0, \quad i = 1, \ldots, m \tag{5}$$
$$h_j(\boldsymbol{x}^*) = 0, \quad j = 1, \ldots, p \tag{6}$$

3. **Dual feasibility:** $\lambda_i^* \geq 0, \quad i = 1, \ldots, m$
4. **Complementary slackness:** $\lambda_i^* g_i(\boldsymbol{x}^*) = 0, \quad i = 1, \ldots, m$

Interpretation of the KKT Conditions

Stationarity Condition: This generalizes the unconstrained optimality condition $\nabla f(\boldsymbol{x}^*) = \mathbf{0}$. At the optimum, the gradient of the objective function must be a linear combination of the constraint gradients. Geometrically, this means that $\nabla f(\boldsymbol{x}^*)$ lies in the cone spanned by the active constraint gradients.

Feasibility

The solution must be feasible for both the primal and dual variables. Condition 2 simply states that the solutions must obey the constraints laid out by the problem. Condition 3 is enforcing that the dual variables also meet the appropriate constraints (i.e., non-negative for multipliers corresponding to less-than-or-equal-to constraints)

Complementary Slackness: This condition captures the intuitive idea that:

- If constraint i is inactive (the constraint is not actively constraining the solution, $g_i(\boldsymbol{x}^*) < 0$), then its multiplier must be zero ($\lambda_i^* = 0$)
- If the multiplier is positive ($\lambda_i^* > 0$), then the constraint must be active ($g_i(\boldsymbol{x}^*) = 0$)

The following was not covered in lecture and will not be covered on the midterm (Spring 2026), but is shared for the interested reader to dig into the question of how we perform projections in projected gradient descent.

Projection onto Hyperplanes Let's first consider the case of projecting onto a hyperplane. That is, let's find the projection $\boldsymbol{y}^*$ of $\boldsymbol{x}$ onto the hyperplane $\boldsymbol{a}^T\boldsymbol{y} = b$.

The closest point $\boldsymbol{y}^*$ to $\boldsymbol{x}$ on the hyperplane $\boldsymbol{a}^T\boldsymbol{y} = b$ satisfies two conditions. First of all, the point $\boldsymbol{y}^*$ must be feasible, i.e., $\boldsymbol{y}^*$ must indeed lie on the hyperplane $\boldsymbol{a}^T\boldsymbol{y} = b$. That is, it must hold that $\boldsymbol{a}^T\boldsymbol{y}^* = b$.

Second, for $\boldsymbol{x}$ to be the closest point to the hyperplane, $\boldsymbol{y}^* - \boldsymbol{x}$ must be normal to the hyperplane $\boldsymbol{a}^T\boldsymbol{y}^* = b$. But recall that hyperplanes are defined by their normals, in this case $\boldsymbol{a}$. In other words, $\boldsymbol{a}$ and $\boldsymbol{y}^* - x$ must share a direction, i.e., $\boldsymbol{y}^* - \boldsymbol{x} = \lambda\boldsymbol{a}$, for some scalar $\lambda \in R$.

It follows from the second requirement that $\boldsymbol{y}^* = \boldsymbol{x} + \lambda\boldsymbol{a}$. Substituting this value into the equation of the hyperplane, and solving for λ yields (and assuming $\boldsymbol{a} \neq 0$):

$$\begin{aligned} \boldsymbol{a}^T(\boldsymbol{x} + \lambda\boldsymbol{a}) &= b \\ \boldsymbol{a}^T\boldsymbol{x} + \lambda\|\boldsymbol{a}\|_2^2 &= b \\ \lambda &= \frac{b - \boldsymbol{a}^T\boldsymbol{x}}{\|\boldsymbol{a}\|_2^2} \end{aligned}$$

Therefore,

$$\boldsymbol{y}^* = \boldsymbol{x} + \frac{b - \boldsymbol{a}^T\boldsymbol{x}}{\|\boldsymbol{a}\|_2^2}\boldsymbol{a} = \boldsymbol{x} - \frac{\boldsymbol{a}^T\boldsymbol{x} - b}{\|\boldsymbol{a}\|_2^2}\boldsymbol{a}$$

Projection onto Half-Spaces Next, knowing how to project onto a hyperplane $\boldsymbol{a}^T\boldsymbol{x} = b$, we consider the case of projecting onto a half-space. The intuition in this case is simple: if $\boldsymbol{x}$ already satisfies the constraint (i.e., if $\boldsymbol{a}^T\boldsymbol{x} \leq b$), then there is no need to project—$\boldsymbol{x}$ is its own projection. Otherwise, we simply project onto the boundary hyperplane $\boldsymbol{a}^T\boldsymbol{x} = b$: i.e.,

$$\mathrm{proj}_{\{\boldsymbol{y}:\boldsymbol{a}^T\boldsymbol{y}\leq b\}}(\boldsymbol{x}) = \begin{cases} \boldsymbol{x} & \text{if } \boldsymbol{a}^T\boldsymbol{x} \leq b \\ \boldsymbol{x} - \frac{\boldsymbol{a}^T\boldsymbol{x} - b}{\|\boldsymbol{a}\|_2^2}\boldsymbol{a} & \text{if } \boldsymbol{a}^T\boldsymbol{x} > b \end{cases}$$

Projection onto a Convex Polytope Projecting onto a feasible region in the case of linear constraints requires projecting onto the intersection of multiple half-spaces. More generally, to project a point $\boldsymbol{x}$ onto the intersection of two convex sets C and D, we can use the method of **alternating projections**, which initializes $\boldsymbol{y}^{(0)} = \boldsymbol{x}$, and then alternates between projecting onto C and projecting onto D:

$$\boldsymbol{y}^{(t+1)} = \mathrm{proj}_D(\mathrm{proj}_C(\boldsymbol{y}^{(t)}))$$

Under certain assumptions (e.g., when C and D are closed and convex, and their intersection is non-empty), this method converges to $\mathrm{proj}_{C\cap D}(\boldsymbol{x})$. Moreover, this method extends naturally to the intersection of more than two sets, simply by cycling through the sets.

Projections for Common Constraint Sets In some common cases, the projections have a closed form.

Box Constraints: For $\mathcal{X} = \{\boldsymbol{x} : \boldsymbol{\ell} \leq \boldsymbol{x} \leq \boldsymbol{u}\}$,

$$[\mathrm{proj}_{\mathcal{X}}(\boldsymbol{x})]_i = \begin{cases} \ell_i & \text{if } x_i < \ell_i \\ x_i & \text{if } \ell_i \leq x_i \leq u_i \\ u_i & \text{if } x_i > u_i \end{cases}$$

As boxes are axis-aligned, the projection for each coordinate i is simply the nearest boundary.

ℓ_2-Norm Ball Constraints: For $\mathcal{X} = \{\boldsymbol{x} : \|\boldsymbol{x}\|_2 \leq r\}$,

$$\text{proj}_{\mathcal{X}}(\boldsymbol{x}) = \begin{cases} \boldsymbol{x} & \text{if } \|\boldsymbol{x}\|_2 \leq r \\ r\frac{\boldsymbol{x}}{\|\boldsymbol{x}\|_2} & \text{if } \|\boldsymbol{x}\|_2 > r \end{cases}$$

This projection is simply $\boldsymbol{x}$, if $\boldsymbol{x}$ is already inside the ball; otherwise, it is in the same direction as $\boldsymbol{x}$, but is scaled so that its norm is r.