Constrained Optimization

Constrained Optimization Problem

Recall that the constrained optimization problem is

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{by varying} \quad & x \\ \text{subject to} \quad & g_i(x) \leq 0 \quad i = 1, \ldots, n_g \\ & h_j(x) = 0 \quad j = 1, \ldots, n_h \end{aligned}$$

x^* is now a *constrained minimizer* if

$$f(x^*) \leq f(x) \quad \text{for all} \quad x \in \mathcal{F}$$

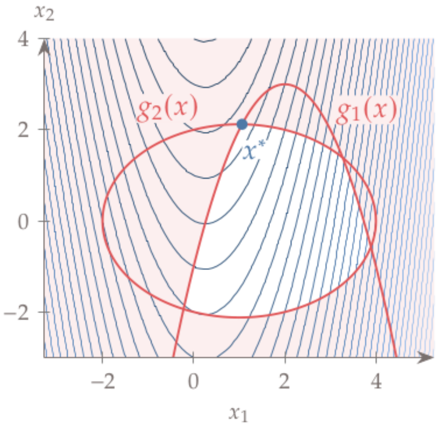
where $\mathcal{F}$ is the feasibility region

$$\mathcal{F} = \{x \mid g_i(x) \leq 0, h_j(x) = 0, i = 1, \ldots, n_g, j = 1, \ldots, n_h\}$$

Thus, to find a constrained minimizer, we have to inspect unconstrained minima of f inside of $\mathcal{F}$ and points along the boundary of $\mathcal{F}$.

COP - Example

$$
\begin{aligned}
\underset{x_1, x_2}{\text{minimize}} \quad & f(x_1, x_2) = x_1^2 - \tfrac{1}{2}x_1 - x_2 - 2 \\
\text{subject to} \quad & g_1(x_1, x_2) = x_1^2 - 4x_1 + x_2 + 1 \leq 0 \\
& g_2(x_1, x_2) = \tfrac{1}{2}x_1^2 + x_2^2 - x_1 - 4 \leq 0
\end{aligned}
$$



Equality Constraints

Let us restrict our problem only to the equality constraints:

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{by varying} \quad & x \\ \text{subject to} \quad & h_j(x) = 0 \quad j = 1, \ldots, n_h \end{aligned}$$

Assume that f and h_j have continuous second derivatives.

Now, we try to imitate the theory from the unconstrained case and characterize minima using gradients.

This time, we must consider the gradient of f and h_j.

Unconstrained Minimizer

Consider the first-order Taylor approximation of f at x

$$f(x + p) \approx f(x) + \nabla f(x)^\top p$$

Unconstrained Minimizer

Consider the first-order Taylor approximation of f at x

$$f(x+p) \approx f(x) + \nabla f(x)^\top p$$

Note that if x^* is an unconstrained minimizer of f, then

$$f(x^*+p) \geq f(x^*)$$

for all p small enough.

Unconstrained Minimizer

Consider the first-order Taylor approximation of f at x

$$f(x+p) \approx f(x) + \nabla f(x)^\top p$$

Note that if x^* is an unconstrained minimizer of f, then

$$f(x^*+p) \geq f(x^*)$$

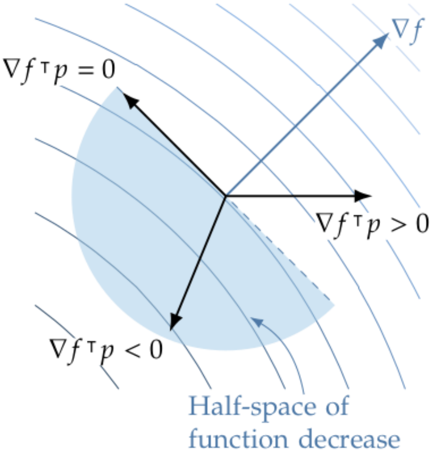
for all p small enough.

Together with the Taylor approximation, we obtain

$$f(x^*) + \nabla f(x^*)^\top p \geq f(x^*)$$

and hence

$$\nabla f(x^*)^\top p \geq 0$$



The hyperplane defined by $\nabla f^\top p = 0$ contains directions p of *zero variation* in f.

In the unconstrained case, x^* is minimizer only if $\nabla f(x^*) = 0$ because otherwise there would be a direction p satisfying $\nabla f(x^*)p < 0$, a *decrease direction*.

Decrease Direction in COP

In COP, p is a decrease direction in $x \in \mathcal{F}$ if $\nabla f(x)^\top p < 0$ and if p is a *feasible direction*!

I.e., point into the feasible region.

Decrease Direction in COP

In COP, p is a decrease direction in $x \in \mathcal{F}$ if $\nabla f(x)^\top p < 0$ and if p is a *feasible direction*!

I.e., point into the feasible region. How do we characterize feasible directions?

Decrease Direction in COP

In COP, p is a decrease direction in $x \in \mathcal{F}$ if $\nabla f(x)^\top p < 0$ and if p is a *feasible direction*!

I.e., point into the feasible region. How do we characterize feasible directions?

Consider Taylor approximation of h_j for all j:

$$h_j(x + p) \approx h_j(x) + \nabla h_j(x)^\top p$$

Decrease Direction in COP

In COP, p is a decrease direction in $x \in \mathcal{F}$ if $\nabla f(x)^\top p < 0$ and if p is a *feasible direction*!

I.e., point into the feasible region. How do we characterize feasible directions?

Consider Taylor approximation of h_j for all j:

$$h_j(x + p) \approx h_j(x) + \nabla h_j(x)^\top p$$

Assuming $x \in \mathcal{F}$, we have $h_j(x) = 0$ for all j and thus

$$h_j(x + p) \approx \nabla h_j(x)^\top p$$

Decrease Direction in COP

In COP, p is a decrease direction in $x \in \mathcal{F}$ if $\nabla f(x)^\top p < 0$ and if p is a *feasible direction*!

I.e., point into the feasible region. How do we characterize feasible directions?

Consider Taylor approximation of h_j for all j:

$$h_j(x + p) \approx h_j(x) + \nabla h_j(x)^\top p$$

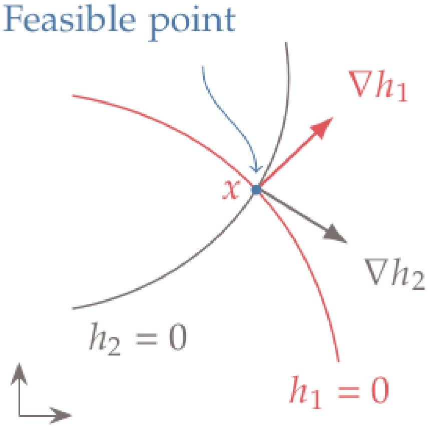
Assuming $x \in \mathcal{F}$, we have $h_j(x) = 0$ for all j and thus

$$h_j(x + p) \approx \nabla h_j(x)^\top p$$

As p is a feasible direction iff $h_j(x + p) = 0$, we obtain that

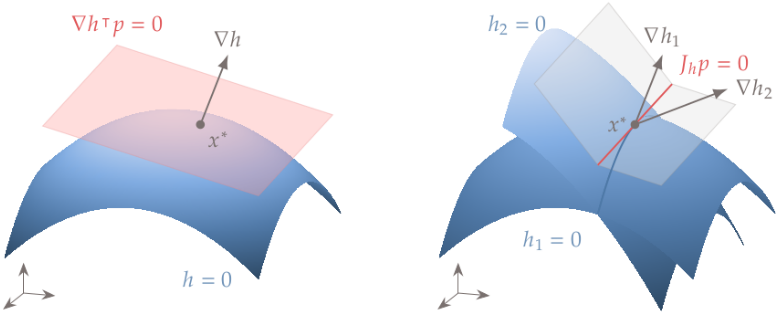
$$p \text{ is a } \textit{feasible direction} \quad \text{iff} \quad \nabla h_j(x)^\top p = 0 \text{ for all } j$$

Feasible Points and Directions



Here, the only feasible direction at x is $p = 0$.

Feasible Points and Directions



Here the feasible directions at x^* point along the red line, i.e.,

$$\nabla h_1(x^*)p = 0 \qquad \nabla h_2(x^*)p = 0$$

Necessary Condition for Constrained Minima

Consider a direction p. Observe that

- If $h_j(x)^\top p \neq 0$, then moving a short step in the direction p violates the constraint $h_j(x) = 0$.

Necessary Condition for Constrained Minima

Consider a direction p. Observe that

- If $h_j(x)^\top p \neq 0$, then moving a short step in the direction p violates the constraint $h_j(x) = 0$.
- If $h_j(x)^\top p = 0$ for all j and
  - $\nabla f(x) p > 0$, then moving a short step in the direction p increases f and stays in $\mathcal{F}$.

Necessary Condition for Constrained Minima

Consider a direction p. Observe that

- If $h_j(x)^\top p \neq 0$, then moving a short step in the direction p violates the constraint $h_j(x) = 0$.
- If $h_j(x)^\top p = 0$ for all j and
  - $\nabla f(x) p > 0$, then moving a short step in the direction p increases f and stays in $\mathcal{F}$.
  - $\nabla f(x) p < 0$, then moving a short step in the direction p decreases f and stays in $\mathcal{F}$.

Necessary Condition for Constrained Minima

Consider a direction p. Observe that

- If $h_j(x)^\top p \neq 0$, then moving a short step in the direction p violates the constraint $h_j(x) = 0$.
- If $h_j(x)^\top p = 0$ for all j and
  - $\nabla f(x)p > 0$, then moving a short step in the direction p increases f and stays in $\mathcal{F}$.
  - $\nabla f(x)p < 0$, then moving a short step in the direction p decreases f and stays in $\mathcal{F}$.
  - $\nabla f(x)p = 0$, then moving a short step in the direction p does not change f and stays $\mathcal{F}$.

Necessary Condition for Constrained Minima

Consider a direction p. Observe that

- If $h_j(x)^\top p \neq 0$, then moving a short step in the direction p violates the constraint $h_j(x) = 0$.
- If $h_j(x)^\top p = 0$ for all j and
  - $\nabla f(x) p > 0$, then moving a short step in the direction p increases f and stays in $\mathcal{F}$.
  - $\nabla f(x) p < 0$, then moving a short step in the direction p decreases f and stays in $\mathcal{F}$.
  - $\nabla f(x) p = 0$, then moving a short step in the direction p does not change f and stays $\mathcal{F}$.

To be a minimizer, x^* must be feasible and every direction satisfying $h_j(x^*)^\top p = 0$ for all j must also satisfy $\nabla f(x^*)^\top p \geq 0$.

Necessary Condition for Constrained Minima

Consider a direction p. Observe that

- If $h_j(x)^\top p \neq 0$, then moving a short step in the direction p violates the constraint $h_j(x) = 0$.
- If $h_j(x)^\top p = 0$ for all j and
  - $\nabla f(x)p > 0$, then moving a short step in the direction p increases f and stays in $\mathcal{F}$.
  - $\nabla f(x)p < 0$, then moving a short step in the direction p decreases f and stays in $\mathcal{F}$.
  - $\nabla f(x)p = 0$, then moving a short step in the direction p does not change f and stays $\mathcal{F}$.

To be a minimizer, x^* must be feasible and every direction satisfying $h_j(x^*)^\top p = 0$ for all j must also satisfy $\nabla f(x^*)^\top p \geq 0$.

Note that if p is a feasible direction, then $-p$ is also, and thus $\nabla f(x^*)^\top(-p) \geq 0$. So finally,

Necessary Condition for Constrained Minima

Consider a direction p. Observe that

- If $h_j(x)^\top p \neq 0$, then moving a short step in the direction p violates the constraint $h_j(x) = 0$.
- If $h_j(x)^\top p = 0$ for all j and
  - $\nabla f(x)p > 0$, then moving a short step in the direction p increases f and stays in $\mathcal{F}$.
  - $\nabla f(x)p < 0$, then moving a short step in the direction p decreases f and stays in $\mathcal{F}$.
  - $\nabla f(x)p = 0$, then moving a short step in the direction p does not change f and stays $\mathcal{F}$.

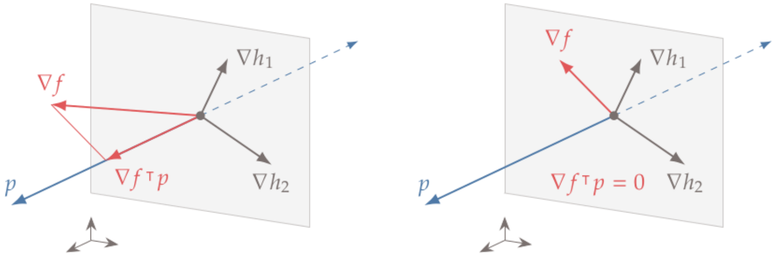
To be a minimizer, x^* must be feasible and every direction satisfying $h_j(x^*)^\top p = 0$ for all j must also satisfy $\nabla f(x^*)^\top p \geq 0$.

Note that if p is a feasible direction, then $-p$ is also, and thus $\nabla f(x^*)^\top(-p) \geq 0$. So finally,

If x^* is a *constrained minimizer*, then

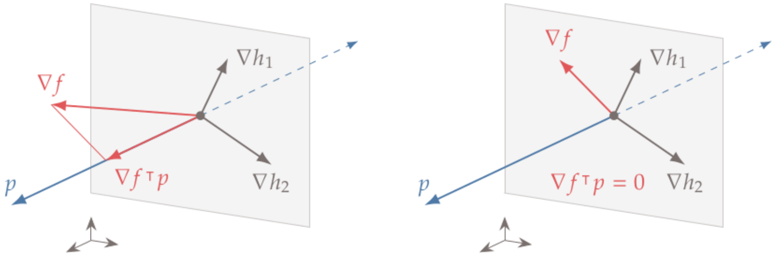
$$\nabla f(x^*)^\top p = 0 \text{ for all } p \text{ satisfying } (\forall j : \nabla h_j(x^*)^\top p = 0)$$

Lagrange Multipliers



Left: f increases along p. **Right:** f does not change along p.

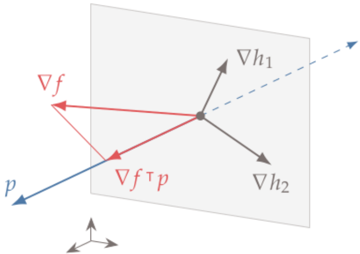
Lagrange Multipliers



Left: f increases along p. **Right:** f does not change along p.

Observe that at an optimum, ∇f lies in the space spanned by the gradients of constraint functions.

Lagrange Multipliers



Left: f increases along p. **Right:** f does not change along p.

Observe that at an optimum, ∇f lies in the space spanned by the gradients of constraint functions.

There are *Lagrange multipliers* λ_1, λ_2 satisfying

$$\nabla f(x^*) = -(\lambda_1 \nabla h_1 + \lambda_2 \nabla h_2)$$

The minus sign is arbitrary for equality constraints but will be significant when dealing with inequality constraints.

Lagrange Multipliers

We know that if x^* is a constrained minimizer, then.

$$\nabla f(x^*)^\top p = 0 \text{ for all } p \text{ satisfying } (\forall j : \nabla h_j(x^*)^\top p = 0)$$

Lagrange Multipliers

We know that if x^* is a constrained minimizer, then.

$$\nabla f(x^*)^\top p = 0 \text{ for all } p \text{ satisfying } (\forall j : \nabla h_j(x^*)^\top p = 0)$$

But then, from the geometry of the problem, we obtain

Theorem 1

Consider the COP with only equality constraints and f and all h_j twice continuously differentiable.
Assume that x^* is a constrained minimizer and that x^* is regular, *which means that $\nabla h_j(x^*)$ are linearly independent.*
Then there are $\lambda_1, \ldots, \lambda_{n_h} \in \mathbb{R}$ satisfying

$$\nabla f(x^*) = -\sum_{j=1}^{n_h} \lambda_j \nabla h_j(x^*)$$

The coefficients $\lambda_1, \ldots, \lambda_{n_h}$ are called *Lagrange multipliers*.

Lagrangian Function

Try to transform the constrained problem into an unconstrained one by moving the constraints $h_j(x) = 0$ into the objective.

Lagrangian Function

Try to transform the constrained problem into an unconstrained one by moving the constraints $h_j(x) = 0$ into the objective.

Consider *Lagrangian function* $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^{n_h} \to \mathbb{R}$ defined by

$$\mathcal{L}(x, \lambda) = f(x) + \lambda^\top h(x) \qquad \text{here} \qquad h(x) = (h_1(x), \ldots, h_{n_h}(x))^\top$$

Lagrangian Function

Try to transform the constrained problem into an unconstrained one by moving the constraints $h_j(x) = 0$ into the objective.

Consider *Lagrangian function* $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^{n_h} \to \mathbb{R}$ defined by

$$\mathcal{L}(x, \lambda) = f(x) + \lambda^\top h(x) \qquad \text{here} \qquad h(x) = (h_1(x), \ldots, h_{n_h}(x))^\top$$

Note that the stationary point of $\mathcal{L}$ gives us the Lagrange multipliers:

$$\nabla_x \mathcal{L} = \nabla f(x) + \sum_{j=1}^{n_h} \lambda_j \nabla h_j(x)$$

$$\nabla_\lambda \mathcal{L} = h(x)$$

Lagrangian Function

Try to transform the constrained problem into an unconstrained one by moving the constraints $h_j(x) = 0$ into the objective.

Consider *Lagrangian function* $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^{n_h} \to \mathbb{R}$ defined by

$$\mathcal{L}(x, \lambda) = f(x) + \lambda^\top h(x) \qquad \text{here} \qquad h(x) = (h_1(x), \ldots, h_{n_h}(x))^\top$$

Note that the stationary point of $\mathcal{L}$ gives us the Lagrange multipliers:

$$\nabla_x \mathcal{L} = \nabla f(x) + \sum_{j=1}^{n_h} \lambda_j \nabla h_j(x)$$

$$\nabla_\lambda \mathcal{L} = h(x)$$

Now putting $\nabla \mathcal{L}(x) = 0$, we obtain precisely the above properties of the constrained minimizer:

$$h(x) = 0 \qquad \text{and} \qquad \nabla f(x) = -\sum_{j=1}^{n_h} \lambda_j \nabla h_j(x)$$

So we can now use methods for searching stationary points. This will lead to the Lagrange-Newton method.

$$\begin{array}{ll} \underset{x_1, x_2}{\text{minimize}} & f(x_1, x_2) = x_1 + 2x_2 \\ \text{subject to} & h(x_1, x_2) = \frac{1}{4}x_1^2 + x_2^2 - 1 = 0 \end{array}$$

The Lagrangian function

$$\mathcal{L}(x_1, x_2, \lambda) = x_1 + 2x_2 + \lambda\left(\frac{1}{4}x_1^2 + x_2^2 - 1\right)$$

$$\begin{array}{ll} \underset{x_1, x_2}{\text{minimize}} & f(x_1, x_2) = x_1 + 2x_2 \\ \text{subject to} & h(x_1, x_2) = \frac{1}{4}x_1^2 + x_2^2 - 1 = 0 \end{array}$$

The Lagrangian function

$$\mathcal{L}(x_1, x_2, \lambda) = x_1 + 2x_2 + \lambda\left(\frac{1}{4}x_1^2 + x_2^2 - 1\right)$$

Differentiating this to get the first-order optimality conditions,

$$\frac{\partial \mathcal{L}}{\partial x_1} = 1 + \frac{1}{2}\lambda x_1 = 0 \qquad \frac{\partial \mathcal{L}}{\partial x_2} = 2 + 2\lambda x_2 = 0$$

$$\frac{\partial \mathcal{L}}{\partial \lambda} = \frac{1}{4}x_1^2 + x_2^2 - 1 = 0.$$

$$\begin{array}{ll} \underset{x_1, x_2}{\text{minimize}} & f(x_1, x_2) = x_1 + 2x_2 \\ \text{subject to} & h(x_1, x_2) = \frac{1}{4}x_1^2 + x_2^2 - 1 = 0 \end{array}$$

The Lagrangian function

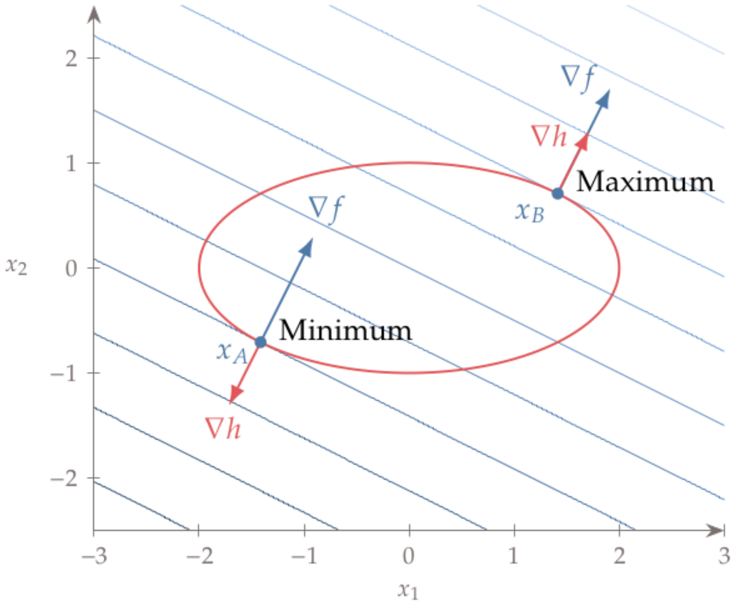
$$\mathcal{L}(x_1, x_2, \lambda) = x_1 + 2x_2 + \lambda\left(\frac{1}{4}x_1^2 + x_2^2 - 1\right)$$

Differentiating this to get the first-order optimality conditions,

$$\frac{\partial \mathcal{L}}{\partial x_1} = 1 + \frac{1}{2}\lambda x_1 = 0 \qquad \frac{\partial \mathcal{L}}{\partial x_2} = 2 + 2\lambda x_2 = 0$$
$$\frac{\partial \mathcal{L}}{\partial \lambda} = \frac{1}{4}x_1^2 + x_2^2 - 1 = 0.$$

Solving these three equations for the three unknowns (x_1, x_2, λ), we obtain two possible solutions:

$$x_A = (x_1, x_2) = (-\sqrt{2}, -\sqrt{2}/2), \quad \lambda_A = \sqrt{2}$$
$$x_B = (x_1, x_2) = (\sqrt{2}, \sqrt{2}/2), \quad \lambda_A = -\sqrt{2}$$


∇f
∇h
Maximum
x_B
∇f
Minimum
x_A
∇h
2
1
0
−1
−2
x_2
−3
−2
−1
0
1
2
3
x_1

Second-Order Sufficient Conditions

As in the unconstrained case, the first-order conditions characterize any "stable" point (minimum, maximum, saddle).

Second-Order Sufficient Conditions

As in the unconstrained case, the first-order conditions characterize any "stable" point (minimum, maximum, saddle).

Consider *Lagrangian Hessian*:

$$H(x, \lambda) = H_f(x) + \sum_{j=1}^{n_h} \lambda_j H_{h_j}(x)$$

Here H_f is the Hessian of f, and each H_{h_j} is the Hessian of h_j.

Note that Lagrangian Hessian is NOT the Hessian of the Lagrangian!

Second-Order Sufficient Conditions

As in the unconstrained case, the first-order conditions characterize any "stable" point (minimum, maximum, saddle).

Consider *Lagrangian Hessian*:

$$H(x, \lambda) = H_f(x) + \sum_{j=1}^{n_h} \lambda_j H_{h_j}(x)$$

Here H_f is the Hessian of f, and each H_{h_j} is the Hessian of h_j.

Note that Lagrangian Hessian is NOT the Hessian of the Lagrangian!

The second-order sufficient conditions are as follows: Assume x^* is regular and feasible. Also, assume that there is λ^* s.t.

$$\nabla f(x^*) = \sum_{j=1}^{n_h} -\lambda_j^* \nabla h_j(x^*)$$

Second-Order Sufficient Conditions

As in the unconstrained case, the first-order conditions characterize any "stable" point (minimum, maximum, saddle).

Consider *Lagrangian Hessian*:

$$H(x, \lambda) = H_f(x) + \sum_{j=1}^{n_h} \lambda_j H_{h_j}(x)$$

Here H_f is the Hessian of f, and each H_{h_j} is the Hessian of h_j.
Note that Lagrangian Hessian is NOT the Hessian of the Lagrangian!

The second-order sufficient conditions are as follows: Assume x^* is regular and feasible. Also, assume that there is λ^* s.t.

$$\nabla f(x^*) = \sum_{j=1}^{n_h} -\lambda_j^* \nabla h_j(x^*)$$

and that

$$p^\top H(x^*, \lambda^*) p > 0 \text{ for all } p \text{ satisfying } (\forall j : \nabla h_j(x^*)^\top p = 0)$$

Then, x^* is a constrained minimizer of f.

Inequality Constraints

Recall that the constrained optimization problem is

$$
\begin{array}{rll}
\text{minimize} & f(x) & \\
\text{by varying} & x & \\
\text{subject to} & g_i(x) \leq 0 & i = 1, \ldots, n_g \\
 & h_j(x) = 0 & j = 1, \ldots, n_h
\end{array}
$$

Inequality Constraints

Recall that the constrained optimization problem is

$$\begin{array}{rll} \text{minimize} & f(x) & \\ \text{by varying} & x & \\ \text{subject to} & g_i(x) \leq 0 & i = 1, \ldots, n_g \\ & h_j(x) = 0 & j = 1, \ldots, n_h \end{array}$$

Lagrange multipliers and the Lagrangian function can be extended to deal with inequality constraints.

The resulting necessary conditions for constrained minima are called Karush-Tucker-Kuhn (KKT) conditions.

In this course, Lagrange methods are considered only for equality-constrained problems. So, we omit further discussion of KKT.

Constrained Optimization

Penalty Methods

Penalty methods

The idea: Transform a constrained problem into an unconstrained one by adding a penalty to the objective function when constraints are violated or close to being violated.

Penalty methods

The idea: Transform a constrained problem into an unconstrained one by adding a penalty to the objective function when constraints are violated or close to being violated.

Assuming an objective function f, the penalized objective is of the form

$$\hat{f}(x) = f(x) + \mu\pi(x)$$

Here, μ is a fixed constant determining how strong the penalty should be, and π is the penalty function.

Penalty methods

The idea: Transform a constrained problem into an unconstrained one by adding a penalty to the objective function when constraints are violated or close to being violated.

Assuming an objective function f, the penalized objective is of the form

$$\hat{f}(x) = f(x) + \mu\pi(x)$$

Here, μ is a fixed constant determining how strong the penalty should be, and π is the penalty function.

Now we may apply the unconstrained optimization methods (e.g., L-BFGS) to $\hat{f}$ and obtain an approximation of a minimizer of f.

Penalty methods

The idea: Transform a constrained problem into an unconstrained one by adding a penalty to the objective function when constraints are violated or close to being violated.

Assuming an objective function f, the penalized objective is of the form

$$\hat{f}(x) = f(x) + \mu\pi(x)$$

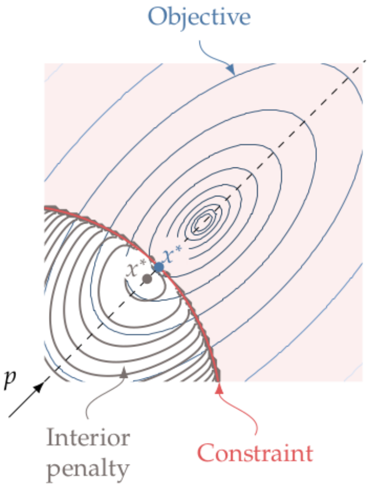
Here, μ is a fixed constant determining how strong the penalty should be, and π is the penalty function.

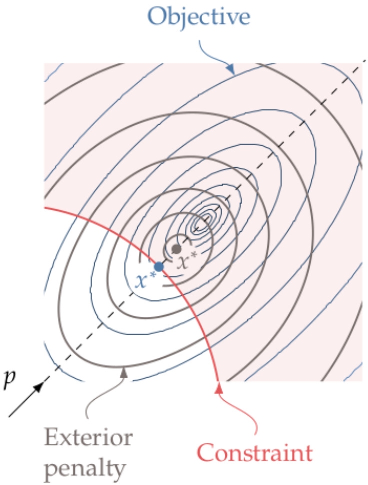
Now we may apply the unconstrained optimization methods (e.g., L-BFGS) to $\hat{f}$ and obtain an approximation of a minimizer of f.

There are two kinds of penalty methods:

- *exterior* - penalizing infeasible x
- *interior* - penalizing x close to being infeasible

Interior vs Exterior Penalty





Exterior Penalty Methods - Quadratic Penalty

Consider equality-constrained problems:

$$\begin{array}{rl} \text{minimize} & f(x) \\ \text{by varying} & x \\ \text{subject to} & h_j(x) = 0 \quad j = 1, \ldots, n_h \end{array}$$

Exterior Penalty Methods - Quadratic Penalty

Consider equality-constrained problems:

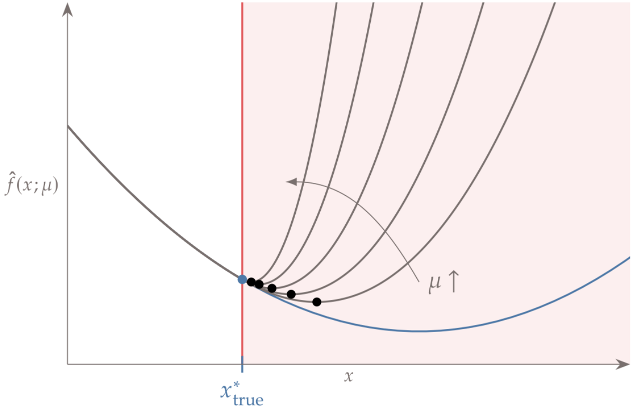
$$\begin{array}{rl} \text{minimize} & f(x) \\ \text{by varying} & x \\ \text{subject to} & h_j(x) = 0 \quad j = 1, \ldots, n_h \end{array}$$

Consider *quadratic penalty*:

$$\hat{f}(x;\mu) = f(x) + \frac{\mu}{2} \sum_{j=1}^{n_h} h_j(x)^2$$

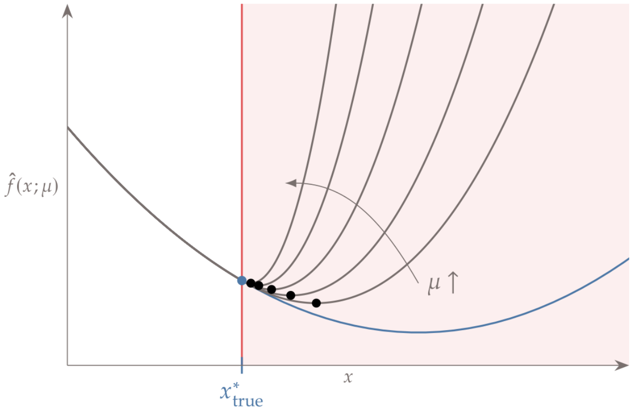
If f is continuously differentiable, $\hat{f}$ is as well (w.r.t. x).

Quadratic Penalty



The true solution would be recovered for $\mu = \infty$.

Quadratic Penalty



The true solution would be recovered for $\mu = \infty$.

However, large μ means large condition number of the Hessian of $\hat{f}$
Intuitively, large curvature of $\hat{f}$, not good for optimization.

Need to choose μ carefully, possibly iteratively.

Algorithm 1 Exterior Penalty Method

Choose starting point x_0
Choose an initial penalty parameter μ_0
Choose a penalty increase factor $\rho > 1$
$k \leftarrow 0$
repeat
    $x_{k+1} \leftarrow x$ minimizing $\hat{f}(x; \mu_k)$
    $\mu_{k+1} \leftarrow \rho \mu_k$
    $k \leftarrow k + 1$
until convergence

Convergence of Quadratic Penalty Method

Theorem 2

Assume that f and all h_j have continuous second derivatives. Suppose that each x_k is the exact global minimizer of $\hat{f}(x;\mu_k)$ and that $\lim_{k\to\infty} \mu_k = \infty$. Then, every limit point x^* of the sequence $\{x_k\}$ solves the constrained optimization problem.

In practice, inexact methods are used to minimize $\hat{f}(x;\mu_k)$

Convergence of Quadratic Penalty Method

Theorem 2

Assume that f and all h_j have continuous second derivatives. Suppose that each x_k is the exact global minimizer of $\hat{f}(x;\mu_k)$ and that $\lim_{k\to\infty}\mu_k = \infty$. Then, every limit point x^* of the sequence $\{x_k\}$ solves the constrained optimization problem.

In practice, inexact methods are used to minimize $\hat{f}(x;\mu_k)$

Let x^* be a limit point of x_k and let λ^* be such that (x^*,λ^*) satisfy the Lagrange conditions for the constrained problem.

Convergence of Quadratic Penalty Method

Theorem 2

Assume that f *and all* h_j *have continuous second derivatives. Suppose that each* x_k *is the exact global minimizer of* $\hat{f}(x;\mu_k)$ *and that* $\lim_{k\to\infty}\mu_k = \infty$*. Then, every limit point* x^* *of the sequence* $\{x_k\}$ *solves the constrained optimization problem.*

In practice, inexact methods are used to minimize $\hat{f}(x;\mu_k)$

Let x^* be a limit point of x_k and let λ^* be such that (x^*,λ^*) satisfy the Lagrange conditions for the constrained problem.

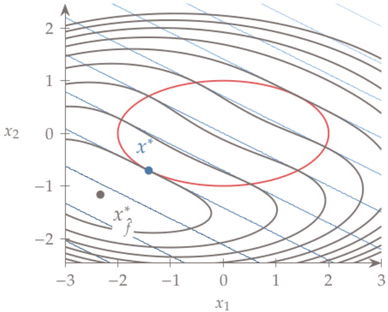
Then, for a subsequence of points x_k, which converges to x^*, we have that

$$\lim_{k\to\infty}\mu_k h_j(x_k) = \lambda_j^*$$

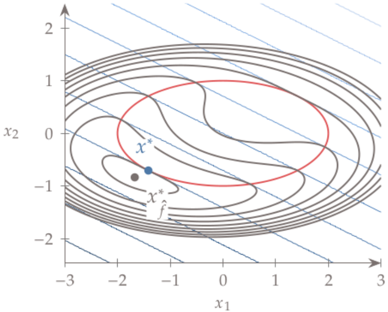
Practical Problems

- Small μ may result in so weak penalty that f unbounded below results in $\hat{f}$ unbounded as well
- As $\mu = \infty$ is impossible, the solution is always slightly infeasible
- Growing curvature of $\hat{f}$ as μ grows makes the Hessian of $\hat{f}$ almost singular

$$\hat{f}(x;\mu) = x_1 + 2x_2 + \frac{\mu}{2}\left(\frac{1}{4}x_1^2 + x_2^2 - 1\right)^2$$

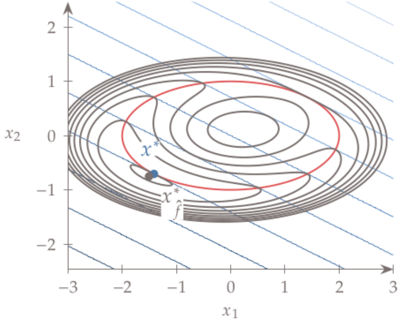


$\mu = 0.5$



$\mu = 3.0$

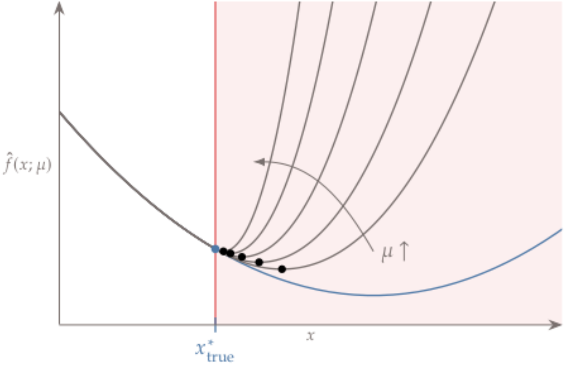
$$\hat{f}(x;\mu) = x_1 + 2x_2 + \frac{\mu}{2}\left(\frac{1}{4}x_1^2 + x_2^2 - 1\right)^2$$



$\mu = 10.0$

Quadratic Penalty for Inequality Constraints

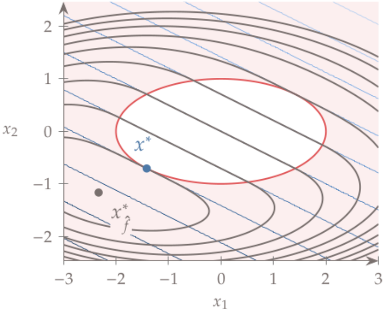
$$\hat{f}(x;\mu) = f(x) + \frac{\mu_h}{2}\sum_{j=1}^{n_h} h_j(x)^2 + \frac{\mu_g}{2}\sum_{i=1}^{n_g} \max\left(0, g_i(x)\right)^2$$



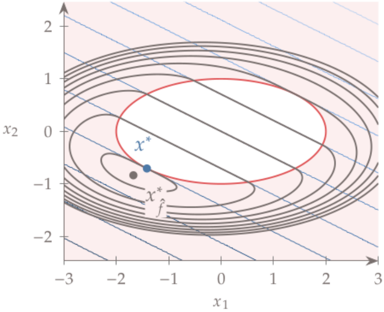
Minimizer approached from the infeasible side.

Example

$$\hat{f}(x;\mu) = x_1 + 2x_2 + \frac{\mu}{2}\max\left(0, \frac{1}{4}x_1^2 + x_2^2 - 1\right)^2$$



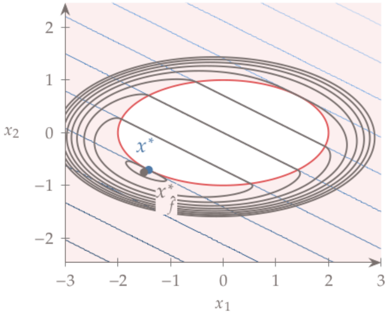
$\mu = 0.5$



$\mu = 3.0$

Example

$$\hat{f}(x;\mu) = x_1 + 2x_2 + \frac{\mu}{2}\max\left(0, \frac{1}{4}x_1^2 + x_2^2 - 1\right)^2$$



$\mu = 10.0$

Augmented Lagrangian (Optional)

We may augment the Lagrangian $\mathcal{L} = f(x) + \sum_{j=1}^{n_h} \lambda_j h_j(x)$ with penalty and optimize the augmented Lagrangian

$$\hat{f}(x; \lambda, \mu) = f(x) + \sum_{j=1}^{n_h} \lambda_j h_j(x) + \frac{\mu}{2} \sum_{j=1}^{n_h} h_j(x)^2$$

Augmented Lagrangian (Optional)

We may augment the Lagrangian $\mathcal{L} = f(x) + \sum_{j=1}^{n_h} \lambda_j h_j(x)$ with penalty and optimize the augmented Lagrangian

$$\hat{f}(x; \lambda, \mu) = f(x) + \sum_{j=1}^{n_h} \lambda_j h_j(x) + \frac{\mu}{2} \sum_{j=1}^{n_h} h_j(x)^2$$

Note the relationship between optimality conditions for $\mathcal{L}$ and $\hat{f}$

$$\nabla_x \hat{f}(x; \lambda, \mu) = \nabla f(x) + \sum_{j=1}^{n_h} (\lambda_j + \mu h_j(x)) \nabla h_j(x) = 0$$

$$\nabla_x \mathcal{L}(x^*, \lambda^*) = \nabla f(x^*) + \sum_{j=1}^{n_h} \lambda_j^* \nabla h_j(x^*) = 0.$$

Augmented Lagrangian (Optional)

We may augment the Lagrangian $\mathcal{L} = f(x) + \sum_{j=1}^{n_h} \lambda_j h_j(x)$ with penalty and optimize the augmented Lagrangian

$$\hat{f}(x; \lambda, \mu) = f(x) + \sum_{j=1}^{n_h} \lambda_j h_j(x) + \frac{\mu}{2} \sum_{j=1}^{n_h} h_j(x)^2$$

Note the relationship between optimality conditions for $\mathcal{L}$ and $\hat{f}$

$$\nabla_x \hat{f}(x; \lambda, \mu) = \nabla f(x) + \sum_{j=1}^{n_h} \left(\lambda_j + \mu h_j(x)\right) \nabla h_j(x) = 0$$

$$\nabla_x \mathcal{L}\left(x^*, \lambda^*\right) = \nabla f\left(x^*\right) + \sum_{j=1}^{n_h} \lambda_j^* \nabla h_j\left(x^*\right) = 0.$$

Comparing these two conditions suggests an approximation:

$$\lambda_j^* \approx \lambda_j + \mu h_j.$$

Augmented Lagrangian Penalty Method (Optional)

Inputs:

- x_0: Starting point
- $\lambda_0 = 0$: Initial Lagrange multiplier
- $\mu_0 > 0$: Initial penalty parameter
- $\rho > 1$: Penalty increase factor

Outputs:

- x^*: Optimal point
- $f(x^*)$: Corresponding function value

Algorithm:

$k = 0$
repeat
    $x_{k+1} \leftarrow x$ minimizing $\hat{f}(x; \lambda_k, \mu_k)$
    $\lambda_{k+1} = \lambda_k + \mu_k h(x_k)$
    $\mu_{k+1} \leftarrow \rho\mu_k$
    $k \leftarrow k + 1$
until convergence

Comparison of Quadratic and Lagrangian Penalty (Optional)

Compare

$$h_j \approx \frac{1}{\mu}\left(\lambda_j^* - \lambda_j\right).$$

with the corresponding approximation of h_j in the quadratic penalty method is

$$h_j \approx \frac{\lambda_j^*}{\mu}$$

Thus, the quadratic penalty relies solely on increasing μ.

Comparison of Quadratic and Lagrangian Penalty (Optional)

Compare

$$h_j \approx \frac{1}{\mu}\left(\lambda_j^* - \lambda_j\right).$$

with the corresponding approximation of h_j in the quadratic penalty method is

$$h_j \approx \frac{\lambda_j^*}{\mu}$$

Thus, the quadratic penalty relies solely on increasing μ.

However, the augmented Lagrangian also controls the numerator via estimating λ_j.

If λ_j is close to λ_j^*, we may obtain a close solution for modest values of μ.

Several variants of the Lagrangian penalty exist for inequality constraints; see Nocedal & Wright.

Interior Penalty Methods

Always seek to maintain feasibility as opposed to the exterior methods.

Instead of adding a penalty only when constraints are violated; add a penalty as the constraint is approached from the feasible region.

Interior Penalty Methods

Always seek to maintain feasibility as opposed to the exterior methods.

Instead of adding a penalty only when constraints are violated; add a penalty as the constraint is approached from the feasible region.

Desirable if the objective function is ill-defined outside the feasible region.

The interior methods are also referred to as *barrier methods* because the penalty function acts as a barrier preventing iterates from leaving the feasible region.

Barrier Methods

Consider inequality-constrained problems:

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{by varying} \quad & x \\ \text{subject to} \quad & g_i(x) \leq 0 \quad i = 1, \ldots, n_g \end{aligned}$$

Barrier Methods

Consider inequality-constrained problems:

$$\begin{array}{rl} \text{minimize} & f(x) \\ \text{by varying} & x \\ \text{subject to} & g_i(x) \leq 0 \quad i = 1, \ldots, n_g \end{array}$$

Minimize the augmented objective function.

$$\hat{f}(x; \mu) = f(x) + \mu\pi(x)$$

Here π is a penalty function.

Barrier Methods

Consider inequality-constrained problems:

$$\begin{array}{rll} \text{minimize} & f(x) & \\ \text{by varying} & x & \\ \text{subject to} & g_i(x) \leq 0 & i = 1, \ldots, n_g \end{array}$$

Minimize the augmented objective function.

$$\hat{f}(x; \mu) = f(x) + \mu\pi(x)$$

Here π is a penalty function.

Inverse barrier

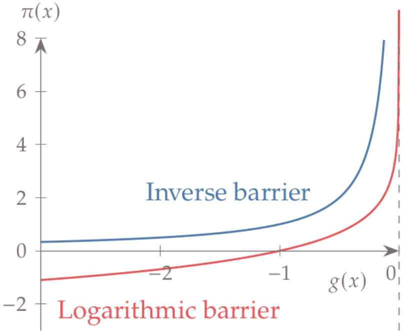
$$\pi(x) = \sum_{i=1}^{n_g} -\frac{1}{g_i(x)}$$

Logarithmic barrier

$$\pi(x) = \sum_{i=1}^{n_g} -\ln(-g_i(x))$$

Algorithms based on these penalties must be prevented from evaluating infeasible points.

Barrier Methods



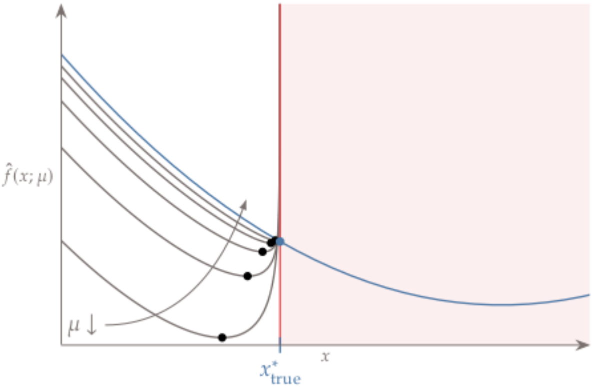
Inverse barrier

$$\pi(x) = \sum_{i=1}^{n_g} -\frac{1}{g_i(x)}$$

Logarithmic barrier

$$\pi(x) = \sum_{i=1}^{n_g} -\ln(-g_i(x))$$

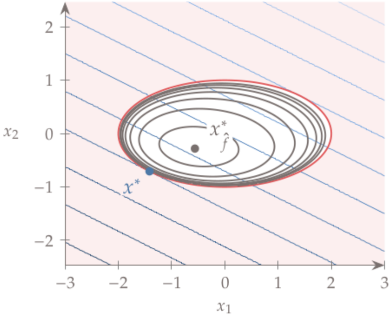
Barrier methods



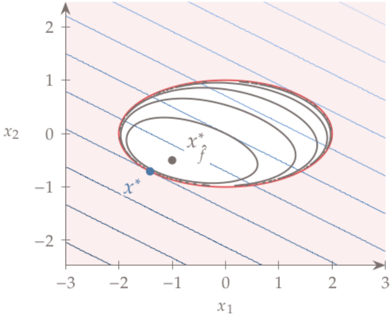
Solve a sequence of unconstrained problems for $\hat{f}$ with $\mu \to 0$.

Example

$$\hat{f}(x;\mu) = x_1 + 2x_2 - \mu \ln\left(-\frac{1}{4}x_1^2 - x_2^2 + 1\right)$$



$\mu = 3.0$

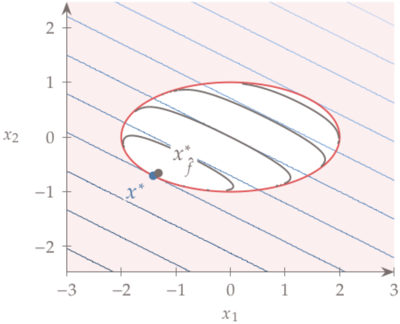


$\mu = 1.0$

As for exterior methods, the Hessian becomes increasingly ill-conditioned as $\mu \to 0$.

Example

$$\hat{f}(x;\mu) = x_1 + 2x_2 - \mu \ln\left(-\frac{1}{4}x_1^2 - x_2^2 + 1\right)$$



$\mu = 0.2$

As for exterior methods, the Hessian becomes increasingly ill-conditioned as $\mu \to 0$.

Comments on Interior Penalty Methods

Interior penalty methods must stay in the feasible region:

- Every unconstrained optimization must start at an initial point feasible for the constrained problem.
- The line search must check for feasibility and backtrack from steps to infeasible points.

Comments on Interior Penalty Methods

Interior penalty methods must stay in the feasible region:

- Every unconstrained optimization must start at an initial point feasible for the constrained problem.
- The line search must check for feasibility and backtrack from steps to infeasible points.

Convergence issues:

- As $\mu \to 0$ solutions of $\hat{f}$ converge to solutions of the constrained problem.
- On the other hand, with $\mu \to 0$ the Hessian of $\hat{f}$ becomes increasingly ill-conditioned.

Various modifications exist to alleviate the problem with ill-conditioned Hessians.

These methods lead to a class of modern *interior point methods*.

Summary of Penalty Methods

Penalty methods penalize approximations that either leave the feasible region (exterior methods), or are close to the border of the feasible region (interior methods).

Penalty methods are simple and easy to implement.

Both exterior and interior methods lead to ill-conditioned Hessians when approaching the correct solutions to the constrained problem.

Constrained Optimization

Sequential Quadratic Programming

Quadratic Programming

The *quadratic optimization problem with equality constraints* is to

$$\begin{array}{rl} \text{minimize} & \frac{1}{2}x^\top Qx + q^\top x \\ \text{by varying} & x \\ \text{subject to} & Ax + b = 0 \end{array}$$

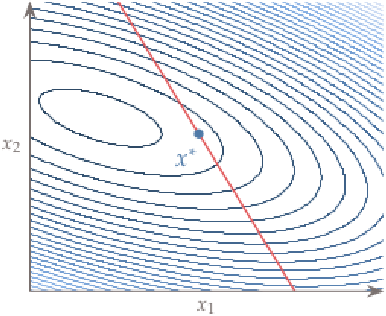
Quadratic Programming

The *quadratic optimization problem with equality constraints* is to

$$\begin{array}{rl} \text{minimize} & \frac{1}{2}x^\top Qx + q^\top x \\ \text{by varying} & x \\ \text{subject to} & Ax + b = 0 \end{array}$$

Here

- Q is a $n \times n$ symmetric matrix. For simplicity assume positive definite.
- A is a $m \times n$ matrix. Assume full rank.



Quadratic Programming

How to solve the quadratic program?

Quadratic Programming

How to solve the quadratic program?

Consider the Lagrangian function

$$L(x, \lambda) = \frac{1}{2}x^\top Qx + q^\top x + \lambda^\top (Ax + b)$$

Quadratic Programming

How to solve the quadratic program?

Consider the Lagrangian function

$$L(x, \lambda) = \frac{1}{2}x^\top Qx + q^\top x + \lambda^\top (Ax + b)$$

and its partial derivatives:

$$\nabla_x L(x) = Qx + q + A^\top \lambda = 0$$
$$\nabla_\lambda L(x) = Ax + b = 0$$

Quadratic Programming

How to solve the quadratic program?

Consider the Lagrangian function

$$L(x, \lambda) = \frac{1}{2} x^\top Q x + q^\top x + \lambda^\top (Ax + b)$$

and its partial derivatives:

$$\begin{aligned} \nabla_x L(x) &= Qx + q + A^\top \lambda = 0 \\ \nabla_\lambda L(x) &= Ax + b = 0 \end{aligned}$$

For Q positive definite, we know that a solution to the above system is a minimizer.

So in order to solve the quadratic program, it suffices to solve the system of linear equations.

Lagrange-Newton

Now consider an arbitrary $f : \mathbb{R}^n \to \mathbb{R}$ and arbitrary constraint functions $h_j : \mathbb{R}^n \to \mathbb{R}$.

Consider the Lagrangian function $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^{n_h} \to \mathbb{R}$ defined by

$$\mathcal{L}(x, \lambda) = f(x) + \lambda^\top h(x) \quad \text{here} \quad h(x) = (h_1(x), \ldots, h_{n_h}(x))^\top$$

Lagrange-Newton

Now consider an arbitrary $f : \mathbb{R}^n \to \mathbb{R}$ and arbitrary constraint functions $h_j : \mathbb{R}^n \to \mathbb{R}$.

Consider the Lagrangian function $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^{n_h} \to \mathbb{R}$ defined by

$$\mathcal{L}(x, \lambda) = f(x) + \lambda^\top h(x) \quad \text{here} \quad h(x) = (h_1(x), \ldots, h_{n_h}(x))^\top$$

We search for the stationary point of $\mathcal{L}$, that is (x^*, λ^*) satisfying

$$\nabla_x \mathcal{L}(x^*, \lambda^*) = \nabla f(x^*) + \sum_{j=1}^{n_h} \lambda_j^* \nabla h_j(x^*) = 0$$

$$\nabla_\lambda \mathcal{L}(x^*, \lambda^*) = h(x^*) = 0$$

These are $n + n_h$ equations in unknowns (x^*, λ^*).

Lagrange-Newton

Now consider an arbitrary $f : \mathbb{R}^n \to \mathbb{R}$ and arbitrary constraint functions $h_j : \mathbb{R}^n \to \mathbb{R}$.

Consider the Lagrangian function $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^{n_h} \to \mathbb{R}$ defined by

$$\mathcal{L}(x, \lambda) = f(x) + \lambda^\top h(x) \quad \text{here} \quad h(x) = (h_1(x), \dots, h_{n_h}(x))^\top$$

We search for the stationary point of $\mathcal{L}$, that is (x^*, λ^*) satisfying

$$\nabla_x \mathcal{L}(x^*, \lambda^*) = \nabla f(x^*) + \sum_{j=1}^{n_h} \lambda_j^* \nabla h_j(x^*) = 0$$

$$\nabla_\lambda \mathcal{L}(x^*, \lambda^*) = h(x^*) = 0$$

These are $n + n_h$ equations in unknowns (x^*, λ^*).

From Lagrange theorem: If x^* is regular and solves the COP, then there exists λ^* such that (x^*, λ^*) solves the system of equations.

Lagrange-Newton

Now consider an arbitrary $f : \mathbb{R}^n \to \mathbb{R}$ and arbitrary constraint functions $h_j : \mathbb{R}^n \to \mathbb{R}$.

Consider the Lagrangian function $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^{n_h} \to \mathbb{R}$ defined by

$$\mathcal{L}(x, \lambda) = f(x) + \lambda^\top h(x) \quad \text{here} \quad h(x) = (h_1(x), \ldots, h_{n_h}(x))^\top$$

We search for the stationary point of $\mathcal{L}$, that is (x^*, λ^*) satisfying

$$\nabla_x \mathcal{L}(x^*, \lambda^*) = \nabla f(x^*) + \sum_{j=1}^{n_h} \lambda_j^* \nabla h_j(x^*) = 0$$

$$\nabla_\lambda \mathcal{L}(x^*, \lambda^*) = h(x^*) = 0$$

These are $n + n_h$ equations in unknowns (x^*, λ^*).

From Lagrange theorem: If x^* is regular and solves the COP, then there exists λ^* such that (x^*, λ^*) solves the system of equations.

We use Newton's method to solve the system of equations.

Lagrange-Newton

Start with some (x_0, λ_0) and compute $(x_1, \lambda_1), \ldots, (x_k, \lambda_k), \ldots$

Lagrange-Newton

Start with some (x_0, λ_0) and compute $(x_1, \lambda_1), \ldots, (x_k, \lambda_k), \ldots$

In every step we compute (x_{k+1}, λ_{k+1}) from (x_k, λ_k) using Newton's step.

Lagrange-Newton

Start with some (x_0, λ_0) and compute $(x_1, \lambda_1), \dots, (x_k, \lambda_k), \dots$

In every step we compute (x_{k+1}, λ_{k+1}) from (x_k, λ_k) using Newton's step.

Consider the gradient of the Lagrangian:

$$
\begin{aligned}
\nabla \mathcal{L}(x_k, \lambda_k) &= (\nabla_x \mathcal{L}(x_k, \lambda_k), \nabla_\lambda \mathcal{L}(x_k, \lambda_k))^\top \\
&= (\nabla f(x_k) + \sum_{j=1}^{n_h} \lambda_{kj} \nabla h_j(x_k), \quad h(x_k))^\top \in \mathbb{R}^{n+n_h}
\end{aligned}
$$

Lagrange-Newton

Start with some (x_0, λ_0) and compute $(x_1, \lambda_1), \ldots, (x_k, \lambda_k), \ldots$

In every step we compute (x_{k+1}, λ_{k+1}) from (x_k, λ_k) using Newton's step.

Consider the gradient of the Lagrangian:

$$\begin{aligned}\nabla \mathcal{L}(x_k, \lambda_k) &= (\nabla_x \mathcal{L}(x_k, \lambda_k), \nabla_\lambda \mathcal{L}(x_k, \lambda_k))^\top \\ &= (\nabla f(x_k) + \sum_{j=1}^{n_h} \lambda_{kj} \nabla h_j(x_k), \quad h(x_k))^\top \in \mathbb{R}^{n+n_h}\end{aligned}$$

and the Hessian matrix of the (complete) Lagrangian

$$\nabla^2 \mathcal{L}(x_k, \lambda_k) \in \mathbb{R}^{n+n_h} \times \mathbb{R}^{n+n_h}$$

We compute this Hessian in the next slide.

Lagrange-Newton

Start with some (x_0, λ_0) and compute $(x_1, \lambda_1), \ldots, (x_k, \lambda_k), \ldots$

In every step we compute (x_{k+1}, λ_{k+1}) from (x_k, λ_k) using Newton's step.

Consider the gradient of the Lagrangian:

$$\begin{aligned}\nabla \mathcal{L}(x_k, \lambda_k) &= (\nabla_x \mathcal{L}(x_k, \lambda_k), \nabla_\lambda \mathcal{L}(x_k, \lambda_k))^\top \\ &= (\nabla f(x_k) + \sum_{j=1}^{n_h} \lambda_{kj} \nabla h_j(x_k), \quad h(x_k))^\top \in \mathbb{R}^{n+n_h}\end{aligned}$$

and the Hessian matrix of the (complete) Lagrangian

$$\nabla^2 \mathcal{L}(x_k, \lambda_k) \in \mathbb{R}^{n+n_h} \times \mathbb{R}^{n+n_h}$$

We compute this Hessian in the next slide.

The Newton's step is then computed by

$$x_{k+1} = x_k + p_k \qquad \lambda_{k+1} = \lambda_k + \mu_k$$

$$(p_k, \mu_k) = -\left(\nabla^2 \mathcal{L}(x_k, \lambda_k)\right)^{-1} \nabla \mathcal{L}(x_k, \lambda_k)$$

Hessian of Lagrangian

Note that

$$\nabla^2 \mathcal{L}(x_k, \lambda_k) = \begin{pmatrix} \nabla_{xx}\mathcal{L}(x_k, \lambda_k) & \nabla_{x\lambda}\mathcal{L}(x_k, \lambda_k) \\ \nabla_{\lambda x}\mathcal{L}(x_k, \lambda_k) & \nabla_{\lambda\lambda}\mathcal{L}(x_k, \lambda_k) \end{pmatrix}$$
$$= \begin{pmatrix} H(x_k, \lambda_k) & \nabla h(x_k) \\ \nabla h(x_k)^\top & 0 \end{pmatrix}$$

Here H is the Lagrangian-Hessian:

$$H(x_k, \lambda_k) = H_f(x_k) + \sum_{j=1}^{n_h} \lambda_{kj} H_{h_j}(x_k)$$

Here H_f is the Hessian of f, and each H_{h_j} is the Hessian of h_j.

$$\nabla h(x_k) = (\nabla h_1(x_k) \cdots \nabla h_{n_h}(x_k))$$

is the matrix of columns $\nabla h_j(x_k)$ for $j = 1, \ldots, n_h$.

Lagrange-Newton for Equality Constraints

Algorithm 2 Lagrange-Newton

Choose starting point x_0
$k \leftarrow 0$
repeat
  Compute $\nabla f(x_k)$, $\nabla h(x_k)$, $h(x_k)$
  Compute $\nabla \mathcal{L}(x_k, \lambda_k)$
  Compute Hessians $H_f(x_k), H_{h_j}(x_k)$ for $j = 1, \ldots, n_h$
  Compute Lagrangian-Hessian $H(x_k, \lambda_k)$
  Compute $\nabla^2 \mathcal{L}(x_k, \lambda_k)$
  Compute $(p_k, \mu_k)^\top = -\left(\nabla^2 \mathcal{L}(x_k, \lambda_k)\right)^{-1} \nabla \mathcal{L}(x_k, \lambda_k)$
  $x_{k+1} \leftarrow x_k + p_k$
  $\lambda_{k+1} \leftarrow \lambda_k + \mu_k$
  $k \leftarrow k + 1$
until convergence

Sequential Quadratic Programming for Inequality Constraints

Introducing inequality constraints brings serious problems.

The main problem is caused by the fact that active constraints behave differently from inactive ones.

Sequential Quadratic Programming for Inequality Constraints

Introducing inequality constraints brings serious problems.

The main problem is caused by the fact that active constraints behave differently from inactive ones.

Roughly speaking, algorithms proceed by searching through possible combinations of active/inactive constraints and solve for each combination as if only equality constraints were present.

This is very closely related to the support enumeration algorithm from game theory.

Sequential Quadratic Programming for Inequality Constraints

Introducing inequality constraints brings serious problems.

The main problem is caused by the fact that active constraints behave differently from inactive ones.

Roughly speaking, algorithms proceed by searching through possible combinations of active/inactive constraints and solve for each combination as if only equality constraints were present.

This is very closely related to the support enumeration algorithm from game theory.

We will consider this type of algorithm only for linear programming (the simplex algorithm).

Summary of Differentiable Optimization

We have considered optimization for differentiable f and h_j's.

We have considered both constrained and unconstrained optimization problems.

Primarily line-search methods: Local search, in every step set a direction and a step length.

The step length should satisfy the strong Wolfe conditions.

Summary of Unconstrained Methods

Consider only f without constraints.

For setting direction we used several methods

- Gradient descent

  Go downhill. Only first-order derivatives needed. Zig-zags.

- Newton's method

  Always minimize the local quadratic approximation of f. Second-order derivatives needed. Better behavior than GD, computationally heavy.

- quasi-Newton (SR1, BFGS, L-BFGS)

  Approximate the quadratic approximation of f. Only first-order derivatives needed. Behaves similarly to Newton's method. Much more computationally efficient.

Summary of Constrained Optimization

Penalty methods, both exterior and interior.

Penalize minimizer approximations out of the feasible region (exterior), or close to the border (interior).

- Exterior

  Penalize minimizer approximations out of the feasible region.

  Quadratic penalty, both for equality and inequality constraints.

- Interior

  Penalize minimizer approximations close to the border (interior).
  Inverse barrier, logarithmic barrier, only for inequality constraints.

Finally, we have considered the Lagrange-Newton method for equality constraints.