C477: Computing for Optimal Decisions
Constrained Optimisation – Algorithms

Panos Parpas
Department of Computing
Imperial College London

www.doc.ic.ac.uk/∼pp500
p.parpas@imperial.ac.uk

Outline

1. Projected Methods
- Reminder: Descent Methods
- Projected Gradient Methods
- Projected Gradient with Linear Constraints

2. Lagrangian Methods
- Lagrangian Methods with Equality Constraints
- Lagrangian Methods with Inequality Constraints

3. Penalty Methods

Additional material:

- Chapter 22 in *An Introduction to Optimization*, Chong & Zhak, Third Edition.
- Chapter 12, 13 in *Linear and Nonlinear Programming*, Luenberger & Ye, Third Edition.

Problem Formulation

$$\begin{aligned} \min\ & f(x) \\ \text{s.t. } & h(x) = 0 \\ & g(x) \leq 0 \end{aligned}$$

- Where $f : \mathbb{R}^n \to \mathbb{R}$, $g : \mathbb{R}^n \to \mathbb{R}^p$, $m \leq n$ and $h : \mathbb{R}^n \to \mathbb{R}^m$
- As before $h_i(x) = 0$, $i = 1, \ldots, m$ are equality constraints
- $g_i(x) \leq 0$, $i = 1, \ldots, p$ are inequality constraints
- The feasible region is $\Omega = \{x \in \mathbb{R}^n \mid h(x) = 0,\ g(x) \leq 0\}$.

Reminder: Descent Methods – Unconstrained

1. Given a point x_k.
2. Transition to the next point,

$$x_{k+1} = x_k + \alpha_k d_k$$

3. where $\alpha_k \in \arg\min f(x_k + \alpha_k d_k)$ (if an exact step-size strategy is used)

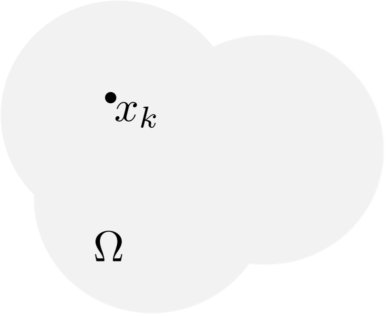
$d_k = -\nabla f(x_k)$ (steepest descent)
$d_k = -\nabla^2 f(x_k)^{-1} \nabla f(x_k)$ (Newton Raphson)

But x is required to stay within some feasible set Ω?

Projection Methods

Basic idea: Project point back into feasible set.

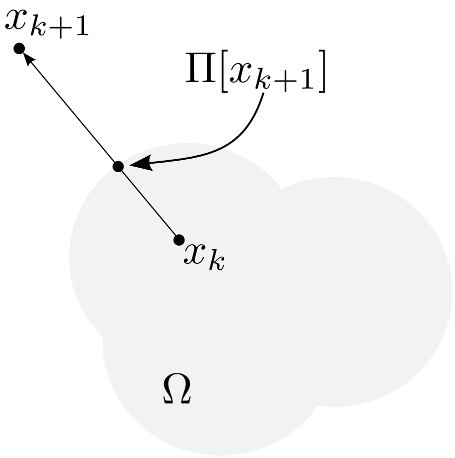
$$x_{k+1} = \begin{cases} x_{k+1} & \text{if } x_{k+1} \in \Omega \\ \Pi[x_{k+1}] & \text{otherwise} \end{cases}$$



Projection Methods

Basic idea: Project point back into feasible set.

$$x_{k+1} = \begin{cases} x_{k+1} & \text{if } x_{k+1} \in \Omega \\ \Pi[x_{k+1}] & \text{otherwise} \end{cases}$$



Projection Methods

Example: Box constraints

Suppose that the constraint set is, $\Omega = \{x \in \mathbb{R}^n \mid l_i \leq x_i \leq u_i\}$
Define $y = \Pi[x]$ as follows,

$$y_i = \begin{cases} u_i & \text{if } x_i > u_i \\ x_i & \text{if } l_i \leq x_i \leq u_i \\ l_i & \text{if } x_i < l_i \end{cases}$$

A concise way to write the above is $y_i = \min\{u_i, \max\{l_i, x_i\}\}$

The point $\Pi[x]$ is called the projection of x into y.
In general the projection operator is defined as,

$$\Pi[x] = \arg\min_{z \in \Omega} \frac{1}{2}\|z - x\|^2$$

Interpretation: $\Pi[x]$ is the closest point in Ω to x

Practical Remarks on Projection Methods

$$\Pi[x] = \arg\min_{z\in\Omega} \frac{1}{2}\|z - x\|^2$$

Projection problem can be as hard as the original problem

Suppose the original problem is:

$$\min \ \frac{1}{2}\|x\|^2$$
$$\text{s.t.} \ \ x \in \Omega.$$

If $0 \notin \Omega$, $\Pi[0]$ is as difficult as the original problem.

Projection not always well defined

If Ω is convex then projection is well defined.
But for some Ω the $\arg\min$ may not be well defined.

Projected Gradient Methods

1. Given a point x_k.
2. Transition to the next point,

$$x_{k+1} = \Pi[x_k - \alpha_k \nabla f(x_k)]$$

3. Where $\alpha_k \in \arg\min f(\Pi[x_k - \alpha_k \nabla f(x_k)])$ (if an exact step-size strategy is used)

Example: Projected Gradient Methods

Consider the problem

$$\begin{aligned} &\min \ \frac{1}{2}x^\top Qx \\ &\text{s.t.} \ \ \|x\|^2 = 1 \end{aligned}$$

where $Q \in \mathbb{R}^{n\times n}$ is a symmetric positive definite matrix. Suppose that a projected gradient method with fixed step size strategy is applied to this problem.

a. Derive a formula for the update equation for the algorithm (i.e. write an explicit formula for x_{k+1} as a function of x_k, Q, and fixed step size α). You may assume that the argument in the projection operator is never zero.

b. Is it possible for the algorithm to not converge to an optimal solution even if the step size $\alpha > 0$ is arbitrarily small?

Example: Projected Gradient Methods

a. Derive a formula for the update equation for the algorithm (i.e. write an explicit formula for x_{k+1} as a function of x_k, Q, and fixed step size α). You may assume that the argument in the projection operator is never zero.

Example: Projected Gradient Methods

b. Is it possible for the algorithm to not converge to an optimal solution even if the step size $\alpha > 0$ is arbitrarily small?

Projected Gradient with Linear Constraints

$$\min \; f(x)$$
$$\text{s.t. } Ax = b$$

Where

- $f : \mathbb{R}^n \to \mathbb{R}$
- $A \in \mathbb{R}^{m \times n}$, $m < n$ and $\text{rank}(A) = m$, $b \in \mathbb{R}^m$.

Derivation of the projection matrix

Suppose that

- x_k is feasible i.e. $Ax_k = b$
- d_k is a descent but not a feasible direction.

Direction will be feasible if,

$$Ax_{x+1} = A(x_k + \alpha_k d_k) = b$$
$$Ax_k + \alpha_k A d_k = b$$

So if,

$$Ad_k = 0$$

then $Ax_{x+1} = b$.

Derivation of the projection matrix

The projection problem is,

$$\min \frac{1}{2}\|d - d_k\|^2$$
$$\text{s.t. } Ad = 0$$

The projection operator is the matrix $P = I - A^\top(AA^\top)^{-1}A$.

Derivation of the projection matrix

The projection problem is,

$$\min \frac{1}{2}\|d - d_k\|^2$$
$$\text{s.t. } Ad = 0$$

The projection operator is the matrix $P = I - A^\top(AA^\top)^{-1}A$.
To see this note that the first order conditions for this problem are,

$$d - d_k + A^\top\lambda = 0$$

Therefore $\lambda = (AA^\top)^{-1}Ad_k$. Substituting this relationship back into the first order condition we obtain that the optimum solution is,

$$d = (I - A^\top(AA^\top)^{-1}A)d_k$$

Properties of the projection matrix

Given a set of linear constraints,

$$Ax = b,$$

with $A \in \mathbb{R}^{m \times n}$, $m < n$ and $\mathrm{rank}(A) = m$, $b \in \mathbb{R}^m$. Then,

$$P = I - A^\top (AA^\top)^{-1} A$$

is called the projection matrix.
Exercise: Show that the following statements are true for the projection matrix defined above.

1. $P^\top P = P$
2. $P^\top = P$

Projected Gradient with Linear Constraints

General Iterative algorithm:

$$x_{k+1} = \Pi[x_k - \alpha_k \nabla f(x_k)]$$

If projection is on the set $\Omega = \{x \in \mathbb{R}^n \mid Ax = b\}$ then,

$$x_{k+1} = x_k - \alpha_k P \nabla f(x_k)$$

where $P = I - A^\top (AA^\top)^{-1} A$, and x_0 was assumed to be in Ω.

Projected Gradient with Linear Constraints

Theorem (Feasibility)

In the projected gradient algorithm with linear constraints, if x_0 is feasible, then $Ax_k = b$, $k \geq 0$.

Proof.

Proof is by induction. Assume that $Ax_k = b$ we show that $Ax_{k+1} = b$. First note that,

$$AP\nabla f(x_k) = A(I - A^\top (AA^\top)^{-1}A)\nabla f(x_k) = (A - A)\nabla f(x_k) = 0.$$

Therefore,

$$Ax_{k+1} = A(x_k - \alpha_k P\nabla f(x_k))) = Ax_k - \alpha_k AP\nabla f(x_k) = b,$$

as required. □

Projected Gradient and Descent Property

$$x_{k+1} = x_k - \alpha_k P \nabla f(x_k)$$

where $P = I - A^\top (AA^\top)^{-1} A$, and x_0 was assumed to be in Ω.
So far we know that if x_0 is feasible then all the iterates x_k will also be feasible.
But is this a descent algorithm?

Projected Gradient and Descent Property

Theorem

If $\{x_k\}$ is the sequence of points generated by the projected gradient algorithm (with the exact step-size strategy). If $P\nabla f(x_k) \neq 0$ then $f(x_{k+1}) < f(x_k)$.

Proof.

Projected Gradient and Convergence

The convergence of the algorithm is based on the previous Theorem and the following result.

Theorem

Let x^* be a feasible point then $P\nabla f(x^*) = 0$ if and only if x^* satisfies the Lagrange condition.

Proof.

Summary Projected Gradient Methods

1. Fast and easy algorithm to implement.
2. All the algorithms (including Newton method) can be used in conjunction with projection.
3. Only possible when Ω is "simple" e.g. a box constraints or linear constraints

Lagrangian Algorithms

Equality Constrained Problem

$$\min \; f(x)$$
$$\text{s.t. } h(x) = 0$$

Where $f : \mathbb{R}^n \to \mathbb{R}$, $h : \mathbb{R}^n \to \mathbb{R}^m$ and $m \leq n$.

First Order Conditions

$$L(x, \lambda) = f(x) + \lambda^\top h(x) \qquad \text{(Lagrangian)}$$
$$\nabla f(x) + \nabla h(x)\lambda = 0 \qquad \text{(First Order Conditions)}$$
$$h(x) = 0$$

Lagrangian Algorithms

Lagrangian Algorithm

$$x_{k+1} = x_k - \alpha_k(\nabla f(x_k) + \nabla h(x_k)\lambda_k)$$
$$\lambda_{k+1} = \lambda_k + \beta_k h(x_k)$$

- **Update equation for** x : same as applying the steepest descent method for minimising $L(x, \lambda)$ over x with no constraints
- **Update equation for** λ : same as applying the steepest descent method for maximising $L(x, \lambda)$ over λ
- Only gradients are used so method is called *first order method.*

The General Case

Equality Constrained Problem

$$
\begin{aligned}
\min\ & f(x) \\
\text{s.t. } & h(x) = 0 \\
& g(x) \leq 0
\end{aligned}
$$

First Order Conditions

$$
\begin{aligned}
& L(x, \lambda) = f(x) + \lambda^\top h(x) + \mu^\top g(x) && \text{(Lagrangian)} \\
& \nabla f(x) + \nabla h(x)\lambda + \nabla g(x)\mu = 0 && \text{(First Order Conditions)} \\
& \mu_i g_i(x) = 0 \\
& \mu \geq 0 \\
& h(x) = 0 \\
& g(x) \leq 0
\end{aligned}
$$

Lagrangian Algorithm – Inequality constraints

Lagrangian Algorithm

$$
\begin{aligned}
x_{k+1} &= x_k - \alpha_k(\nabla f(x_k) + \nabla h(x_k)\lambda_k + \nabla g(x_k)\mu_k) \\
\lambda_{k+1} &= \lambda_k + \beta_k h(x_k) \\
\mu_{k+1} &= P_+[\mu_k + \gamma_k g(x_k)]
\end{aligned}
$$

- P_+ is the projection to the positive part of $\mathbb{R}^p$ applied component wise.
- **Update equation for** x : same as applying the steepest descent method for minimising $L(x, \lambda, \mu)$ with no constraints
- **Update equation for** λ : same as applying the steepest descent method for maximising $L(x, \lambda, \mu)$ over λ
- **Update equation for** μ : same as applying the **projected** steepest descent method for maximising $L(x, \lambda, \mu)$ over μ
- Only gradients are used so method is called *first order method.*

Lagrangian Algorithm Theory

- Can be shown that method converges to a KKT point.
- Complementarity condition also satisfied.
- Rate of convergence is linear (since it is based on steepest descent method)
- No guarantees it will converge to the global minimum or that second order conditions will be satisfied.

Penalty Methods

Basic Idea: Convert the constrained optimisation problem to an unconstrained problem

Original constrained problem:

$$\min_x f(x)$$

$$x \in \Omega$$

Modified unconstrained problem:

$$\min_x f(x) + \gamma P(x)$$

Where:

- γ is a positive scalar called the **penalty parameter.**
- $P : \mathbb{R}^n \to \mathbb{R}$ is called the penalty function. The aim of this function is to penalise points outside Ω

Penalty Methods

Basic Idea: Convert the constrained optimisation problem to an unconstrained problem
Original constrained problem:

$$\min_x f(x)$$
$$x \in \Omega$$

Modified unconstrained problem:

$$\min_x f(x) + \gamma P(x)$$

Where:

- γ is a positive scalar called the **penalty parameter.**
- $P : \mathbb{R}^n \to \mathbb{R}$ is called the penalty function. The aim of this function is to penalise points outside Ω

Penalty Methods

Original constrained problem:

$$\min_x f(x)$$
$$\text{s.t. } g_i(x) \leq 0 \; i = 1, \ldots, p.$$

Penalty Function:

$$P(x) = \sum_{i=1}^{p} g_i^+(x)$$

where

$$g_i^+(x) = \max\{0, g_i(x)\} = \begin{cases} 0 & \text{if } g_i(x) \leq 0 \\ g_i(x) & \text{if } g_i(x) > 0 \end{cases}$$

The penalty function defined above is also called the absolute value penalty function since it is equal to $\sum_{i=1}^{p} |g_i(x)|$

Example

Suppose the feasible region is given by,

$$g_1(x) = x - 2 \leq 0.$$
$$g_2(x) = -(x+1)^3 \leq 0$$

The penalty function is defined as follows,

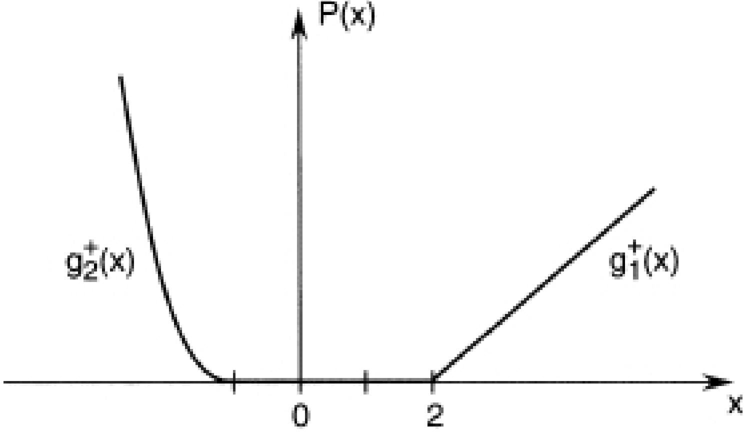
$$g_1^+(x) = \max\{0, g_1(x)\} = \begin{cases} 0 & \text{if } x \leq 2 \\ x - 2 & \text{otherwise} \end{cases}$$

$$g_2^+(x) = \max\{0, g_2(x)\} = \begin{cases} 0 & \text{if } x \geq -1 \\ -(x+1)^3 & \text{otherwise} \end{cases}$$

So,

$$P(x) = g_1^+(x) + g_2^+(x) = \begin{cases} x - 2 & \text{if } x > 2 \\ 0 & \text{if } -1 \leq x \leq 2 \\ -(x+1)^3 & \text{if } x < -1 \end{cases}$$

Example



Penalty Methods

The absolute value penalty function may not be differentiable everywhere (e.g. last example $P(x)$ is not differentiable at $x = 2$). Some differentiable & widely used alternatives are:

- The *Courant-Beltrami penalty function*

$$P(x) = \sum_{i=1}^{p} (g_i^+(x))^2$$

- *Logarithmic Barrier function*

$$P(x) = -\sum_{i=1}^{p} \log(-g_i(x))$$

- *Inverse Barrier function*

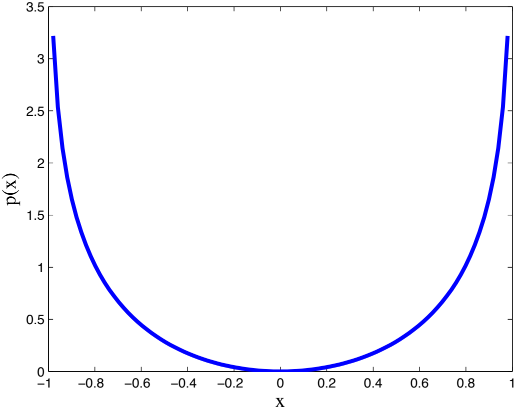
$$P(x) = -\sum_{i=1}^{p} \frac{1}{g_i(x)}$$

For the two barrier functions the convention is to let the penalty parameter γ go to zero

Penalty Methods

The Logarithmic Barrier Function associated with the constraint,

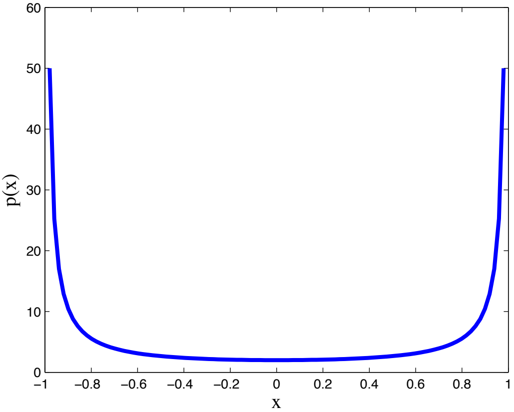
$$-1 \leq x \leq 1$$



Penalty Methods

The Inverse Barrier Function associated with the constraint,

$$-1 \leq x \leq 1$$



Penalty Methods Summary

- Penalty methods convert the problem into an unconstrained problem and use unconstrained algorithms (e.g. Steepest Descent, Newton Method etc..)
- Logarithmic Barrier Methods are very popular for solving convex optimisation problems (these are polynomial time algorithms)
- Convergence results exists that guarantee that these methods will converge to a KKT point as $\gamma \to \infty$ (or zero in the case of barrier penalty functions)
- Because of the penalty parameter problem becomes ill conditioned